Global Optimization of Hybrid Systems

by

Cha Kun Lee

Submitted to the Department of Chemical Engineering
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Chemical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2006

Author .....
Department of Chemical Engineering
July 12, 2006

Certified by .....
Paul I. Barton
Professor
Thesis Supervisor

Accepted by .....
William M. Deen
Chairman, Department Committee on Graduate Students

Global Optimization of Hybrid Systems

by

Cha Kun Lee

Submitted to the Department of Chemical Engineering
on July 12, 2006, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Chemical Engineering

Abstract

Systems that exhibit both discrete state and continuous state dynamics are called hybrid systems. In most nontrivial cases, these two aspects of system behavior interact to such a significant extent that they cannot be decoupled effectively by any kind of abstraction and must be analyzed simultaneously. Hybrid system models are important in many areas of science and engineering, including flip-flops and latching relays, manufacturing systems, air-traffic management systems, controller synthesis, switched systems, chemical process systems, signaling and decision making mechanisms in (biological) cells, robotic systems, safety interlock systems, and embedded systems.

The primary focus of this thesis is to explore deterministic methods for the global optimization of hybrid systems. While the areas of modeling, simulation and sensitivity analysis of hybrid systems have received much attention, there are many challenging difficulties associated with the optimization of such systems. The contents of this thesis represent the first steps toward deterministic global optimization of hybrid systems in the continuous time domain. There are various reasons for wanting to solve optimization problems globally. In particular, there are many applications which demand that the global solution be found, for example, formal safety verification problems and parameter estimation problems. In the former case, a suboptimal local solution could falsely indicate that all safety specifications are met, leading to disastrous consequences if, in actuality, a global solution exists which provides a counter example that violates some safety specification. In the latter case, a suboptimal local solution could falsely indicate that a proposed model structure did not match experimental data in a statistically significant manner, leading to the false rejection of a valid model structure. In addition, for many optimization problems in engineering, the presence of nonlinear equality constraints makes the optimization problem nonconvex such that local optimization methods can often fail to produce a single feasible point, even though the problem is indeed feasible.

The control parameterization framework is employed for the solution of the optimization problem with continuous time hybrid systems embedded. A major difficulty of such a framework lies in the fact that the mode sequence of the embedded hybrid

system changes in the space of the optimization decision variables for most nontrivial problems. This makes the resulting optimization problem nonsmooth because the parametric sensitivities of the hybrid system do not exist everywhere, thus invalidating efficient gradient based optimization solvers. In this thesis, the general optimization problem is decomposed into three subproblems, and tackled individually: (a) when the mode sequence is fixed, and the transition times are fixed; (b) when the mode sequence is allowed to vary, and the transition times are fixed; and (c) when the mode sequence is fixed, and the transition times are allowed to vary. Because even these subproblems are nontrivial to solve, this thesis focuses on hybrid systems with linear time varying ordinary differential equations describing the continuous dynamics, and proposes various methods to exploit the linear structure. However, in the course of solving the last subproblem, a convexity theory for general, nonlinear hybrid systems is developed, which can be easily extended for general, nonlinear hybrid systems.

Subproblem (a) is the easiest to solve. A convexity theory is presented that allows convex relaxations of general, nonconvex Bolza type functions to be constructed for the optimization problem. This allows a deterministic branch-and-bound framework to be employed for the global optimization of the subproblem. Subproblems (b) and (c) are much more difficult to solve, and require the exploitation of structure. For subproblem (b), a hybrid superstructure is proposed that enables the linear structure to be retained. A branch-and-cut framework with a novel dynamic bounds tightening heuristic is proposed, and it is shown that the generation of cuts from dynamic bounds tightening can have a dramatic impact on the solution of the problem. For subproblem (c), a time transformation is employed to transform the problem into one with fixed transition times, but nonlinear dynamics. A convexity theory is developed for constructing convex relaxations of general, nonconvex Bolza type functions with the nonlinear hybrid system embedded, along with the development of bounding methods, based on the theory of differential inequalities. A novel bounding technique that exploits the time transformation is also introduced, which can provide much tighter bounds than that furnished utilizing differential inequalities.

Thesis Supervisor: Paul I. Barton
Title: Professor

Acknowledgments

First and foremost, I would like to thank my advisor, Paul Barton, for inspiration, help, and guidance, for without him, this thesis would not have been possible. I would also like to thank my thesis committee, Klavs Jensen and Gregory McRae, for their helpful comments and encouragement over the duration of my doctoral studies.

As Newton famously said, "If I have seen further, it is by standing on the shoulder of giants." I am indebted to my good friend, Adam Singer, who has helped me so much in my work. It is his work that forms the basis for my work, and I owe him many thanks. I would also like to thank the many outstanding researchers that I have had the utmost pleasure of working with in the Process Systems Engineering Laboratory at MIT: Bambang Adiwijaya, Ingrid Berkelmans, Binita Bhattacharjee, Benoit Chachuat, Jerry Clabaugh, David Collins, Edward Gatzke, Panayiotis Lemonidis, Alexander Mitsos, Bahy Noureldin, Derya Ozyurt, Ignacio Palou-Rivera, Patricio Ramirez, Gunther Reisig, Ajay Selot, Kirill Titievsky, John Tolsma, Katharina Wilkins, Mehmet Yunt. I would also like to thank my many friends from outside the PSEL, who have been so important to me in so many ways: Hiroyo Kawai, I-Chieh Chou, Lino Gonzalez, Ivy Lee, Yinthai Chan, Tseh-Hwan Yong, YongHwee Chua, Jujin An, Jijon Sit, Paige Koh.

Adam, I'll miss all the basketball sessions, and watching the Superbowl/NBA finals together. John, thank you for writing DAEPACK, and treating me to pizza and guiding me home that infamous night. Binita, thanks for the encouragement and companionship. Benoit, thanks for being my badminton partner. Someday, we'll have to play table tennis, although you'll always crush me in squash. Derya, thank you for being such a great and wonderful guy, I'll always remember Trondheim and San Franciso. Alexander, thank you for being so helpful, reliable and dependable. It has really been a pleasure. Ajay, thank you for putting up with all of my ramblings and always having a willing ear. And, of course, the food. Panos, thank you for all the soccer. And remember, "Pull it Down!". I-Chieh, thank you for all the beautiful memories. And Hiroyo, arigato for being Hirochan, yesterday, today and tomorrow.

Finally, I would like to thank my family. Words cannot adequately express the gratitude I owe them, for they have always been there for me, without fail, whenever I have needed them. They remain the most important part of my life, and always will.

To My Family

Contents

List of Figures

List of Tables

Chapter 1

Hybrid Systems

Systems that exhibit both discrete state and continuous state dynamics are called *hybrid* systems. Commonly, a hybrid system is denoted as a discrete/continuous system. In most nontrivial cases, these two aspects of system behavior interact to such a significant extent that they cannot be decoupled effectively by any kind of abstraction and must be analyzed simultaneously. The partitioning into discrete and continuous states is most often a modeling convenience or an effective tool to make a problem tractable, e.g., to avoid modeling phenomena on wildly differing time scales in which fast, often nonlinear dynamics are simplified by replacing them with discrete transitions (it is also worth recalling that the continuum hypothesis itself is only an approximation for the behavior of large collections of discrete particles). On the other hand, sometimes it may be impossible to avoid discrete phenomena by introducing more sophisticated models. An example of this is a tank initially full of liquid that at some later time becomes empty. As soon as the amount of liquid in the vessel becomes zero the intensive properties (temperature, composition) of the liquid become undefined. These intensive variables either have to be removed from the model, or assigned dummy values. In either case, a discrete change to the model is unavoidable.

Hybrid system models are important in many areas of science and engineering, including flip-flops and latching relays [136], manufacturing systems [38], air-traffic management systems [131], controller synthesis [132, 25], switched autonomous sys-

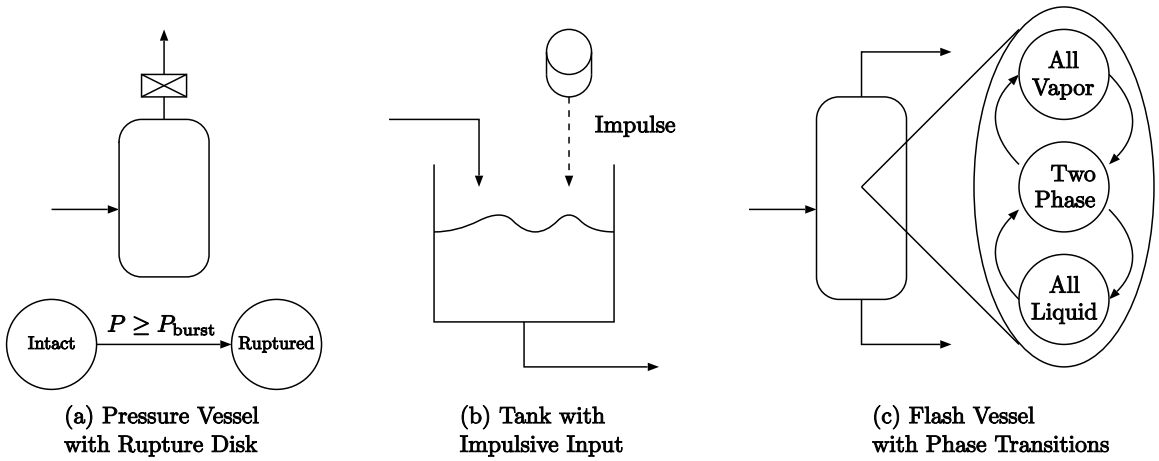


Figure 1-1: Physical systems modeled as hybrid systems.

tems [138], chemical process systems [18, 46], signaling and decision making mechanisms in (biological) cells, robotic systems, safety interlock systems, embedded systems, etc. Figure 1-1 shows some simple examples commonly found in chemical engineering. It is thus not surprising that a lot of work has been done in the areas of modeling [4, 13, 30, 105] and simulation [40, 18, 28] of hybrid systems to date, such that the modeling and simulation of hybrid systems has achieved a high degree of robustness. What is perhaps surprising is that the theory for sensitivity analysis of hybrid systems has only recently been established in [63], extending and generalizing the work first done in [114]. In close relation, it is very desirable to utilize this knowledge and technology to develop a theory for the numerical optimization of hybrid systems [89, 62, 12, 19]. This area remains very exciting, challenging and largely unexplored. Thus, the primary focus of this thesis is to explore deterministic methods for the global optimization of hybrid systems.

An extensive taxonomy of models has been proposed in recent years for the description of hybrid systems (see e.g., [30]). Indeed, it is difficult to make progress without appealing to some form of model formulation. Generally, hybrid systems may be defined on either a continuous or discrete time domain. For example, discrete time formulations have recently been employed in the design of model predictive controllers for systems embedding interlock logic and qualitative descriptions [24] and for numerical optimization [89]. A continuous time formulation assumes that the

continuous state of the hybrid system is differentiable almost everywhere on the time interval of interest. In this thesis we will restrict our attention to continuous time formulations, while noting that most of the observations made have their analogy in the (simpler to analyze) discrete time case, and that continuous time systems are most often approximated by discrete time systems for numerical solution.

A hybrid system can be described by a (collection of) discrete state subsystem(s), a (collection of) continuous state subsystem(s), and the possible interactions between these subsystems. The continuous time formulation admits a (potentially heterogeneous) variety of embedded differential subsystems including ordinary differential equations (ODEs), differential-algebraic equations (DAEs), partial differential equations (PDEs), and even multi-domain integro partial differential-algebraic equations. Again, we will limit ourselves to ODEs and DAEs, while noting that hybrid systems embedded with PDEs represent an unexplored and potentially rich field of study. Similarly, the discrete state subsystems may be heterogeneous and conform to a variety of formalisms, such as finite state machines, Petri nets, sequential logic systems, etc.

This chapter is organized as follows: Section 1.1 will review the concepts behind the modeling of hybrid systems, and present the modeling framework which we will adopt for the sequel. In Section 1.2, we discuss issues concerning the robust simulation of hybrid systems, while in Section 1.3 we present a concise summary of the theory and equations that have been developed for the sensitivity analysis of hybrid systems. Finally, Section 1.4 will describe the early work done and some of the challenges that lie ahead in the numerical optimization of hybrid systems, which sets the stage for the rest of the chapters of this thesis.

1.1 Modeling

The pioneering work on modeling of hybrid systems were the papers of Witsenhausen [136] and Fahrland [53]. In recent years, several mathematical formalisms have been proposed to model hybrid systems. They include hybrid automata [4, 90], hybrid Petri

nets [45], the general abstract dynamical model [30], the state-transition network representation [12], the bond graph representation [101], etc. From the point of view of mathematical and numerical analysis, we have found the *hybrid automaton* representation most useful [4, 13, 62]. In this section, we will present, based on the concept of hybrid automata, a clear and intuitive modeling framework that is amenable for the analysis of hybrid systems.

Roughly speaking, the evolution of a hybrid system through time consists of the following: starting with some fixed time and initial condition for the discrete and continuous state, the continuous state of the hybrid system evolves according to the differential equations attributed to the initial discrete state of the hybrid system. At some point in time, a transition may occur. If a transition occurs, the discrete state of the hybrid system switches to another (not necessarily different) state, the continuous state of the hybrid system is reset to some point in its Euclidean space, following which it then evolves according to the differential equations attributed to the new discrete state of the hybrid system after the transition. At some point in time, a transition may occur. And the cycle repeats indefinitely.

The above description seems deceptively simple to model. However, in our efforts to develop a modeling framework for the evolution of the hybrid system described above, we have encountered the following (interconnected) issues and questions:

1. What is a transition, and how does one define the semantics of a transition?

2. How does one define a deterministic evolution of a hybrid system for a given initial time and condition?

3. Are instantaneous, and/or multiple, transitions allowed?

4. How does one resolve the issue of modeling reversible discontinuities?

In particular, we will be illustrating these issues contrasted against the modeling frameworks of [136, 90, 30, 5, 63]. While these references may have addressed some of these issues, none of them have addressed all of these issues completely in a satis-

factory manner. We will now lay the skeleton of our proposed modeling framework, based on that in [63].

We shall call the continuous time axis the *time horizon*, which is split into contiguous time intervals called *epochs*. The discrete and continuous subsystems only interact via instantaneous discrete *transitions* at distinct points in time called *events*. Similar to [90], we will define a *hybrid time trajectory* T_τ as a sequence of epochs $\{I_i\}$ such that each epoch is a closed time interval $I_i = [\sigma_i, \tau_i] \subset \mathbb{R}$, $\sigma_{i+1} = \tau_i$ and $\tau_i \leq \tau_{i+1}$ for all $i = 1, 2, 3, \ldots$ with the initial time $t_0 = \sigma_1$. For the epoch $I_i = [\sigma_i, \tau_i]$, the system evolves continuously by allowing time to pass if $\sigma_i < \tau_i$, and it evolves discretely by making an instantaneous transition if $\sigma_i = \tau_i$. Loosely speaking, the evolution of the hybrid system over the time horizon will be called the *execution* of the hybrid system; this term will defined rigorously in Section 1.1.6. For the sequel, we will only consider finite sequences T_τ terminating with epoch I_{n_e} where n_e is the total number of epochs, and the final time $t_f = \tau_{n_e}$.

The hybrid system can be viewed as a directed graph whose vertices represent the continuous state subsystems, called *modes*, and whose edges represent the *transitions* between the modes. We introduce the following elements:

1. An finite index set M for the modes, $M = \{1, 2, \ldots, n_m\}$, where n_m is the total number of modes in the system. The corresponding sequence of modes for T_τ is called the *hybrid mode trajectory* and is denoted by $T_\mu = \{m_i\}, m_i \in M$. Note that m_i denotes the *mode* of the system in *epoch* I_i, and hence can be represented by the pair (m, I_i), where $m \in M$, if desired. However, we will use m_i for notational simplicity.

2. A set of variables, $V^{(m)}$, for each mode $m \in M$. The dependent variables that we are concerned with are the state variables $\mathbf{x}^{(m)}(\mathbf{p}, t) \in \mathbb{R}^{n_x^{(m)}}$. The time invariant parameters $\mathbf{p} \in \mathbb{R}^{n_p}$ and time, $t \in \mathbb{R}$ are the independent variables. Clearly $V^{(m)} = \{\mathbf{x}^{(m)}, \mathbf{p}, t\}$. Also, for $t \in I_i$, the real value of the continuous state is given by $\mathbf{x}^{(m_i)}(\mathbf{p}, t)$.

3. A finite set of equations for each mode $m \in M$. The state of the hybrid system

evolves according to the *dynamics* of the system, which are represented by ODEs or DAEs given by

$$\mathbf{f}^{(m)}\left(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}, t\right) = \mathbf{0} \tag{1.1}$$

where $\mathbf{f}^{(m)} : \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}^{n_x^{(m)}}$.

4. A set of initial conditions for the hybrid system, for any initial mode $m_1 \in M$,

$$\mathbf{T}^{(m_1,0)}\left(\dot{\mathbf{x}}^{(m_1)}, \mathbf{x}^{(m_1)}, \mathbf{p}, t_0\right) = \mathbf{0}.$$

where $\mathbf{T}^{(m_1,0)} : \mathbb{R}^{n_x^{(m_1)}} \times \mathbb{R}^{n_x^{(m_1)}} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}^{r^{(m_1)}}$, where $r^{(m_1)}$ is the dynamic degrees of freedom for the DAE system $\mathbf{f}^{(m_1)} = \mathbf{0}$. We will assume that the set of initial conditions specified by $\mathbf{T}^{(m_1,0)}$ is consistent with the differential equations specified by $\mathbf{f}^{(m_1)} = \mathbf{0}$ for all $m_1 \in M$. See Section 1.2 for a discussion on consistent initialization.

Henceforth, we shall use the superscript (m) to refer to any mode in M, while (m_i) refers to the active mode in epoch I_i. We will also make the following assumption concerning the dynamics of the hybrid system in each mode given by the differential equations (1.1): for any $m^* \in M$, and any $\mathbf{p}^* \in P$, we assume that a solution $\mathbf{x}^{(m^*)}$ exists, is unique, and is continuous (though it may be nonsmooth, e.g., when $\mathbf{f}^{(m^*)}$ is discontinuous) for the initial value problem (IVP) given by (1.1), any arbitrary consistent initial condition $\mathbf{x}_0(\sigma) \in \mathbb{R}^{n_x^{(m)}}$, for any time interval $[\sigma, \tau] \subset [t_0, t_f]$ (see Section 1.2 for a discussion on consistent initial conditions for DAE systems). Obviously, this assumption implies that the continuous states of the hybrid system are also bounded on $[t_0, t_f]$, provided that the solution of the hybrid system exists on $[t_0, t_f]$ (see Section 1.1.3 for examples where the solution of the hybrid system may not exist on the time domain of interest).

1.1.1 Definition of a Transition

There are two broad classes of transitions that are possible: a switch (impulse) occurs when the transition ends in a different (the same) mode. Informally, we can think of

switching as a change in the functional form of the embedded differential equations, while impulses are jumps that cause discontinuities in the state variables with no change in the equations. However, it is important to note that with an embedded DAE, a switch or even nonsmoothness in the controls may cause both discontinuities and even Dirac functions to appear in the state variable trajectories. This issue will be discussed further in Section 1.2. It is clear that the state of the discrete subsystems, described uniquely by the mode m_i, changes only at switches (autonomous or controlled), which can be as simple as the deletion of one equation and its replacement with a new equation, or as complex as the deletion and/or insertion of a number of active *agents* each described by their own individual system of differential equations, e.g., vehicles in a traffic management system. The latter phenomenon is also seen in variable structure systems [52, 105].

Branicky et al. [30] further classified transitions into two types: *autonomous* transitions, which occur naturally without a choice, and *controlled* transitions, which occur in response to a control command. Controlled changes can also be effected by introducing control variables $\mathbf{u}$, where $\mathbf{u}$ belongs to some appropriate function space, and treating the controlled transitions as autonomous ones whose transition conditions (see below) are expressed as a function of $\mathbf{u}$.

We shall introduce the following definitions. Consider any transition from mode m_i to mode m_{i+1}. We shall call mode m_{i+1} a *successor* of mode m_i, which we call the *predecessor*. For impulses, both the predecessor and successor modes are the same. We say that the transition is *enabled* when the transition can be made and the transition is *taken* when the transition actually occurs at an event. In [4, 5], if a transition is enabled, the transition must be taken before an *exception* occurs, if it exists. This concept incorporates a random element in the timing of the transition, as transitions can, but not necessarily must, be taken when enabled. This implies that the exact timings of the transitions are not known, which makes deterministic simulation impossible. Note that this nondeterministic behavior can be incorporated in the framework of Branicky et al. [30] by introducing said transitions as controlled transitions. In our framework, which is a special case of the more general frameworks

described above, all transitions have enabling and exception conditions that coincide at exactly the point where the transition is to be taken, i.e., the transition is taken instantaneously once it becomes enabled. We shall call this the *transition condition*. We shall have more to say about the determinism of transitions below. Note that in essence, we have formulated the question "When is a transition taken from mode m_i to m_{i+1}?", to which the answer is "A transition is taken when the transition condition becomes true."

To define a transition from mode m_i, we need to define uniquely a time, called the *transition time*, at which the transition is to be taken, i.e., at which the transition condition becomes true. We also need to define uniquely a successor mode, as well as a unique initial condition for the continuous state in the successor mode. The modeling frameworks of [136, 30, 90] (which all consider ODEs as the continuous dynamics) do this by defining a subset of the Euclidean state space of the continuous state of the hybrid system, $G^{(m_i)} \subset \mathbb{R}^{n_x^{(m_i)}}$. This set is called the *departure set* in [136], the (autonomous) *jump set* in [30], and is formed by the *guard* conditions in [90]. Whenever the continuous state of the hybrid system in mode m_i enters the set $G^{(m_i)}$, the transition condition is satisfied, and a transition is taken. In order to define a unique earliest time, they make the assumption that $G^{(m_i)}$ is a closed set. After defining this set, the rest is relatively easy. In [136], the successor mode is defined by the departure set, and an identity mapping sets the initial value of the continuous state in the successor mode to be equal to the value of the continuous state in the predecessor mode at the transition time. In [30], the (autonomous) *jump transition map* determines the successor mode, as well as the initial value of the continuous state in the successor mode. In [90], the *set of edges* determines the successor mode, and the *reset map* determines the initial conditions of the continuous state in the successor mode.

What these modeling frameworks have done is to say effectively that the transition condition becomes true when the continuous state of the hybrid system in mode m_i enters the closed set $G^{(m_i)}$. From the assumption of continuity of $\mathbf{x}^{(m_i)}$, there will thus be a unique earliest transition time that can be determined. This makes

the hybrid system deterministic in the sense that the transition time is always well defined. However, for the following reasons, this definition of a transition is not entirely satisfactory:

1. How does one characterize the set $G^{(m_i)}$, especially for the case where there may be several pending transitions from mode m_i? The type of characterization used could potentially raise problems for the simulation of hybrid systems, in which autonomous events have to be detected in strict time order. Also, what happens in the boundaries between two pending transitions? In addition, each mode could have many competing, pending transitions to other modes, and whose transition conditions could also depend on the system parameters, $\mathbf{p}$. Visualizing these pending transitions becomes complicated with a single, lumped set $G^{(m_i)}$.

2. The treatment of reversible discontinuities is problematic. See Section 1.1.5 below for a detailed discussion.

Note that the modeling frameworks in [5] and [63] allow the use of strict inequalities while defining (elements of) a transition condition. This implies that no unique, earliest transition time can be determined for the transition. To mitigate the above issues, we propose the following definition of a transition, based on a modification of the framework in [63]. While this does not treat the difficulties with reversible discontinuities, it does explicitly characterize the closed set $G^{(m_i)}$.

Each mode $m \in M$ has associated with it a finite index set of transitions emanating from it, $J^{(m)} = \{1, 2, \ldots, n_\tau^{(m)}\}$, where $n_\tau^{(m)}$ is the total number of transitions with predecessor mode m. Note that there may be several transitions to the same successor mode, there could be impulsive transition(s) back to the predecessor mode or $J^{(m)}$ could be the empty set ($n_\tau^{(m)} = 0$). We introduce a mapping $S^{(m)} : J^{(m)} \to M$ such that $S^{(m)}(j)$ represents the successor mode corresponding to the jth transition, i.e., $S^{(m)}$ keeps track of the successor modes for the set $J^{(m)}$. Each transition $j \in J^{(m)}$ has associated with it:

1. A logical, or Boolean, transition condition, which is represented by the following mapping,

$$L_j^{(m)}(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}, t),$$

where $L_j^{(m)} : \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \{\texttt{TRUE}, \texttt{FALSE}\}$. This condition is formed by logical operators (AND,OR) connecting a finite number of atomic propositions (relational expressions) composed of valid real functions and the relational operators $\{\leq, \geq\}$. We will assume further that the real functions are continuous on $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_p} \times \mathbb{R}$. See [107] for a discussion motivating the use of logical expressions. Note that we have removed the use of the NOT operator and strict relational operators $\{<, >\}$ compared to the framework in [63], see below for a discussion.

2. A system of *transition functions* that maps the final values of the variables in the predecessor mode $m = m_i$ to the initial values in the successor mode $S^{(m)}(j) = m_{i+1}$ at time $\tau_i = \sigma_{i+1}$ where the transition is made between epochs I_i and I_{i+1}:

$$\mathbf{T}_j^{(m)} \Big(\dot{\mathbf{x}}^{(m)}(\mathbf{p}, \tau_i), \mathbf{x}^{(m)}(\mathbf{p}, \tau_i), \dot{\mathbf{x}}^{(S^{(m)}(j))}(\mathbf{p}, \sigma_{i+1}), \mathbf{x}^{(S^{(m)}(j))}(\mathbf{p}, \sigma_{i+1}), \mathbf{p}, \tau_i \Big) = \mathbf{0}, \quad (1.2)$$

where $\mathbf{T}_j^{(m)} : \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(S^{(m)}(j))}} \times \mathbb{R}^{n_x^{(S^{(m)}(j))}} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}^{r^{(S^{(m)}(j))}}$, where $r^{(S^{(m)}(j))}$ is the dynamic degrees of freedom for the DAE system $\mathbf{f}^{(S^{(m)}(j))} = \mathbf{0}$. We will assume that the set of initial conditions for the successor mode specified by $\mathbf{T}_j^{(m)}$ is consistent with the differential equations specified by $\mathbf{f}^{(S^{(m)}(j))} = \mathbf{0}$. See Section 1.2 for a discussion on consistent initialization.

Note that in order to define a unique, earliest time for the transition, the mapping $L_j^{(m)}(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}^*, t)$ associated with each pending transition $j \in J^{(m)}$ of mode m must define a closed set in its domain, $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}$, for which the condition becomes TRUE, for each fixed $\mathbf{p}^* \in P$. The use of only regular or weak inequalities, the assumption of continuity of the real functions comprising the atomic propositions, and

the use of only the logical operators (`AND`,`OR`) ensures that this will always be true. To illustrate this, note that an atomic proposition composed of continuous real functions and the relational operators $\{\leq, \geq\}$ will define a closed set in $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}$ by construction. In addition, the `AND` and `OR` operators can be viewed as intersection and union operators. Since any finite intersection and/or union of closed sets is also closed, the logical condition must define a closed set in its domain. This is formalized in the following propositions. Note that in this framework, it is also possible to include atomic propositions formed by strict inequalities, operated on by `NOT`. However, it is always possible to convert such propositions to ones involving only regular inequalities.

Proposition 1.1. *For any $\mathbf{p}^* \in P$, an atomic proposition composed of continuous real functions on the domain $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}$ and the relational operators $\{\leq, \geq\}$ defines a closed set in $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}$ for which the proposition becomes* `TRUE`.

Proof. Without loss of generality, let the atomic proposition be given by

$$\{g(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}^*, t) \leq 0\} \Longleftrightarrow \texttt{TRUE}$$

where $g : \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_p} \times \mathbb{R} \to \mathbb{R}$ is continuous. Clearly, it suffices to show that the following set is closed,

$$G \equiv \{(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, t) \in \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R} \mid g(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}^*, t) \leq 0\}.$$

Assume, for contradiction, that the set G is not closed. Then, there exists a limit point of G that is not in G. Let such a point be $(\underline{\dot{\mathbf{x}}}^{(m)}, \underline{\mathbf{x}}^{(m)}, \underline{t})$. By assumption,

$$g(\underline{\dot{\mathbf{x}}}^{(m)}, \underline{\mathbf{x}}^{(m)}, \underline{t}) = \varepsilon > 0.$$

Since it is a limit point, every neighborhood of the point contains a point

$$(\dot{\mathbf{x}}^{(m)\dagger}, \mathbf{x}^{(m)\dagger}, t^\dagger) \neq (\underline{\dot{\mathbf{x}}}^{(m)}, \underline{\mathbf{x}}^{(m)}, \underline{t})$$

such that $(\dot{\mathbf{x}}^{(m)\dagger}, \mathbf{x}^{(m)\dagger}, t^\dagger) \in G$. However, by continuity of g, there exists some $\delta > 0$ such that

$$|g(\dot{\mathbf{x}}^{(m)\dagger}, \mathbf{x}^{(m)\dagger}, \mathbf{p}^*, t^\dagger) - g(\underline{\dot{\mathbf{x}}}^{(m)}, \underline{\mathbf{x}}^{(m)}, \mathbf{p}^*, \underline{t})| \leq \varepsilon/2$$

for all $(\dot{\mathbf{x}}^{(m)\dagger}, \mathbf{x}^{(m)\dagger}, t^\dagger)$ such that

$$|(\dot{\mathbf{x}}^{(m)\dagger}, \mathbf{x}^{(m)\dagger}, t^\dagger) - (\underline{\dot{\mathbf{x}}}^{(m)}, \underline{\mathbf{x}}^{(m)}, \underline{t})| \leq \delta.$$

Clearly, this is a contradiction. Hence, every limit point of G must be in G, and it is a closed set. □

Proposition 1.2. *The transition condition $L_j^{(m)}(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}^*, t)$ associated with each pending transition $j \in J^{(m)}$ of mode m defines a closed set in its domain, $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}$, for which the condition becomes* `TRUE`*, for all fixed $\mathbf{p}^* \in P$.*

Proof. From Proposition 1.1, each atomic proposition defines a closed set in $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times \mathbb{R}$ for which the proposition becomes `TRUE`. The transition condition $L_j^{(m)}$ is composed of the logical operators `AND` and `OR`, which are the intersection and union operators respectively, connecting a finite number of atomic propositions. Since the intersection and union of a finite collection of closed sets is also closed [116, Theorem 2.24], we have the desired result. □

Hence, under this framework, the transition condition for the pending transition $j \in J^{(m)}$ in mode m is simply given by $L_j^{(m)}(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}, t)$. For example, in the pressure vessel shown in Figure 1-1(a), we have $M = \{1, 2\}$ where mode 1 denotes the `Intact` mode and mode 2 the `Ruptured` mode. We have $J^{(1)} = \{1\}$, $S^{(1)}(1) = 2$ and $J^{(2)} = \emptyset$, with the transition condition $L_1^{(1)} := (P \geq P_{\text{burst}})$, and the transition function $\mathbf{T}_1^{(1)} = \mathbf{x}^{(1)}(\mathbf{p}, \tau_i) - \mathbf{x}^{(2)}(\mathbf{p}, \sigma_{i+1})$. This is an example of a switching transition.

As another example, consider the buffer tank shown in Figure 1-1(b) where we have $M = \{1\}$, $J^{(1)} = \{1\}$, $S^{(1)}(1) = 1$ and $L_1^{(1)} := (t \geq t_m) \wedge (t \leq t_m)$. Here, t_m is a known time event where we add material to the tank and the transition function is given by $\mathbf{T}_1^{(1)} = \mathbf{x}^{(1)}(\mathbf{p}, \tau_1) + \Delta\mathbf{x} - \mathbf{x}^{(1)}(\mathbf{p}, \sigma_2)$, where $\Delta\mathbf{x}$ is the amount of material added, and $t_m = \tau_1 = \sigma_2$. This is an example of an impulsive transition. Note that

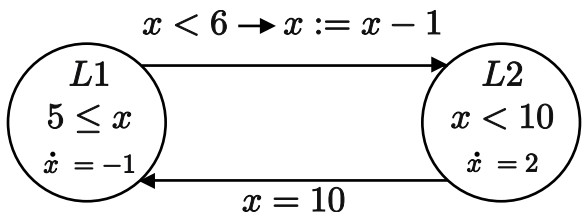


Figure 1-2: Graphical representation of linear hybrid system with nondeterministic transition.

although $\tau_1 = \sigma_2$, this does not imply that $\Delta\mathbf{x} = \mathbf{0}$, because the transition function between the two epochs I_1 and I_2 is implicit and the state variables are allowed to take multiple values at the same instant in time provided the epoch is incremented (this also allows the phenomena of multiple instantaneous transitions to occur). In other words, the notion of an *active epoch* allows the state variables to take multiple values at the same point in time. Note that this is also the case in the framework of [90], whereas in [136], multiple instantaneous transitions are forbidden by requiring that some minimum duration be spent in each mode (this is achieved by assuming some positive distance between the departure and *arrival* sets, see [136] for the details and definitions).

1.1.2 Determinism and Competing Transitions

Thus far, we have effectively constrained our modeling framework to only consider hybrid systems with deterministic transition times, i.e., when a transition is to be taken, the timing of the transition can be determined uniquely. This allows one to simulate such hybrid systems. As mentioned above, in the framework of [4, 5], transitions take on a stochastic or nondeterministic nature in the sense that a transition can be taken at any time the *enabling* conditions are satisfied, and before the continuous state leaves the *invariant* set. For example, consider the first example of Alur et al. [4], where a linear hybrid system is considered (see Figure 1-2 reproduced from [4]).

The conditions immediately below the mode labels $L1$ and $L2$ are the *invariant* conditions, which are the complement of the exception conditions (recall that a transition must be taken before an exception occurs). In mode $L1$, the value of x decreases

at a constant rate of 1. The transition from $L1$ to $L2$ may be taken at any time after the value of x has fallen below 6 (the *enabling* conditions are on top of the transition arrows in Figure 1), and it must be taken before the value of x falls below 5 (the invariant condition). When the transition is taken, the value of x is instantaneously decreased by 1. In other words, the state can remain in mode 1 (stay invariant) as long as $5 \leq x$. If x falls below 5, the exception occurs, and the transition *must be taken*. However, no guidance is provided as to when the transition is taken, since it can be taken any time when $5 \leq x < 6$. This illustrates the nondeterministic nature of the transitions allowed in the framework of [4, 5].

It is unreasonable to expect a simulator to be able to handle this kind of nondeterministic transition phenomena, since it does not know when to take the transition. However, it might be possible to incorporate such behavior (in a limited way) when we consider optimization problems with hybrid systems embedded. Consider the following hybrid system,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} + 1, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x \leq 5 + p), \\ T_1^{(1)} = x(\sigma_{i+1}) - x(\tau_i) + 1, \end{cases}$$

$$\text{Mode 2}: \begin{cases} f^{(2)} = \dot{x} - 2, \quad J^{(2)} = \{1\}, \\ S^{(2)}(1) = 1, \qquad L_1^{(2)} := (x \geq 10), \\ T_1^{(2)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

where $p \in [0, 1]$. It is clear that a given value of p corresponds to a stochastic transition from mode 1 to mode 2. Given an optimization problem, we can include p as an optimization decision variable, and let the optimizer select the best value of p that would minimize the objective, i.e., the optimizer searches over all possible timings for the stochastic transition. The disadvantage to this approach, is that we will have to introduce another variable, p_i, for every possible transition from mode 1 to 2, since there is no reason that the transition will be taken at exactly the same value of p for each transition. Hence, this approach only works if we know, a priori,

the number (or an upper bound for this number) of stochastic transitions that are taken (or are allowed).

We will now describe another form of nondeterminism in hybrid systems, even when the transition times are deterministic. In our framework, for any predecessor mode, there are $n_\tau^{(m)}$ pending transitions where $n_\tau^{(m)}$ might be greater than 1. The event time which determines the correct transition from that mode is governed by which of the transition conditions, $L_j^{(m)}$, becomes true first. For *deterministic* models, in which the successor mode for any transition can be uniquely determined with a unique transition time, and which we are concerned with, there is usually only one transition that satisfies the above said condition, i.e., there is never a case where there exists more than one transition condition becoming true at the same (earliest) time. For situations in which that is true, a set of *precedence relations* or rules have to be stipulated that uniquely determines the successor mode for all possible combinations of multiple transition conditions becoming true at the same time, in order to define a deterministic execution of the hybrid system. If such a precedence rule set is not defined, or if a successor cannot be uniquely determined, we shall call such a hybrid system *nondeterministic* (even though all transition times can be uniquely determined). It can also be seen that the set $J^{(m)}$ may not be uniquely defined as transitions with the same transition function to the same successor mode could be grouped together by means of the Boolean operator `OR`. To remove this ambiguity, we shall always set $J^{(m)}$ as the *minimal transition set*, where $T_j^{(m)} \neq T_l^{(m)},\ \forall j, l \in J^{(m)}$ such that $j \neq l,$ for all $m \in M$.

Consider any mode $m \in M$. Let the power set of $J^{(m)}$ (the set of all subsets of $J^{(m)}$) be represented by $\mathcal{P}(J^{(m)})$. Then, the precedence relation function, $K^{(m)} : \mathcal{P}(J^{(m)}) \to J^{(m)}$ will define a set of precedence relations for mode m. Obviously, the precedence relation function can map the empty set to any index in $J^{(m)}$ because it will not be needed. Also, it is obvious that the precedence relation function will map singleton sets to their singleton. A very simple precedence relation function would be $K^{(m)} : Z \mapsto \inf Z$, i.e., the set $J^{(m)}$ is arranged in descending order of priority, with transition 1 having the highest priority, followed by transition 2, etc.

Consider now Example 1 from [136]. A second order system, $\mathbf{x} = (x_1, x_2)$, with two latching relays is considered. The hybrid system is in mode 1 when both relays are open, mode 2 with only the first relay closed, mode 3 with only the second relay closed, and mode 4 with both relays closed. It is assumed that the first relay closes when $x_1 \geq 0$, and the second relay when $x_2 \geq 0$, so that the conditions for the closure of one relay are independent of the position of the other relay.

From mode one, Witsenhausen defines the following transition set (defined in [136]):

$$\mathcal{T}_{12} = \{\mathbf{x} \mid x_1 \geq 0, x_2 < 0\}$$
$$\mathcal{T}_{13} = \{\mathbf{x} \mid x_1 < 0, x_2 \geq 0\}$$
$$\mathcal{T}_{14} = \{\mathbf{x} \mid x_1 \geq 0, x_2 \geq 0\}$$

where $\mathcal{T}_{ij}$ denotes the transition set from mode i to mode j. The departure set is defined as the union of these transition sets,

$$\mathcal{T}_1^- = \bigcup_{j \neq i} \mathcal{T}_{1j} = \{\mathbf{x} \mid (x_1 \geq 0) \vee (x_2 \geq 0)\}.$$

Note that the departure set is closed, although the individual transition sets that make up this set are not necessarily closed. The reason for this is that Witsenhausen makes the assumption that for any 3 distinct indexes i, j, k in M, the sets $\mathcal{T}_{ij}$ and $\mathcal{T}_{ik}$ are disjoint in $\mathbb{R}^{n_x^{(i)}}$. And the reason that he makes this assumption is because no unique evolution of the state could be defined when the conditions for transition to two or more different modes are fulfilled at the same time, i.e., the hybrid system becomes nondeterministic. For modes 2 and 3, the following transition sets are defined,

$$\mathcal{T}_{24} = \{\mathbf{x} \mid x_2 \geq 0\}$$
$$\mathcal{T}_{34} = \{\mathbf{x} \mid x_1 \geq 0\}.$$

Note that in the framework of Witsenhausen [136], the use of strict inequalities to

define transition conditions is allowed, as long as the departure set is guaranteed to be closed. Furthermore, the need for strict inequalities arises from the need to specify that the individual transition sets are disjoint. In addition, a burden is placed on the modeler to (a) verify that the departure set is indeed closed, and (b) verify that the transition sets are disjoint. The use of precedence relations as proposed mitigates all of the above issues. In the proposed framework, we would have the following transitions for the hybrid system,

$$\text{Mode 1}: \begin{cases} J^{(1)} = \{1,2,3\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x_1 \geq 0) \wedge (x_2 \leq 0), \\ S^{(1)}(2) = 3, \qquad L_2^{(1)} := (x_1 \leq 0) \wedge (x_2 \geq 0), \\ S^{(1)}(3) = 4, \qquad L_3^{(1)} := (x_1 \geq 0) \wedge (x_2 \geq 0), \end{cases}$$

$$\text{Mode 2}: \begin{cases} J^{(2)} = \{1\}, \\ S^{(2)}(1) = 4, \qquad L_1^{(2)} := (x_2 \geq 0), \end{cases}$$

$$\text{Mode 3}: \begin{cases} J^{(3)} = \{1\}, \\ S^{(3)}(1) = 4, \qquad L_1^{(3)} := (x_1 \geq 0), \end{cases}$$

and for $m \in M$, the precedence relation function $K^{(m)} : Z \mapsto \sup Z$.

Another area where establishing a precedence relation is extremely helpful is when considering optimization problems with hybrid systems embedded, especially when the optimization decision variables are incorporated into the transition conditions. For example, consider the following hybrid system from [17],

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} - 1, \quad J^{(1)} = \{1,2\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x - p \geq 0), \\ S^{(1)}(2) = 3, \qquad L_2^{(1)} := (x + p > 6), \\ T_1^{(1)} = T_2^{(1)} = x(p, \sigma_{i+1}) - x(p, \tau_i), \end{cases} \tag{1.3}$$

$$\text{Mode 2}: f^{(2)} = \dot{x} + 1, \quad J^{(2)} = \emptyset, \tag{1.4}$$

$$\text{Mode } 3 : f^{(3)} = \dot{x} - 2, \quad J^{(3)} = \emptyset. \tag{1.5}$$

Suppose we start with $x(p, 0) = 0$ in mode 1, and wish to end at $t_f = 4$, with $p \in [2, 4]$. There are two pending transitions from mode 1, and it is easy to see that for values of $p \leq 3$, transition 1 to mode 2 will be taken, whereas for $p > 3$, transition 2 to mode 3 will be taken (see also Figure 1-12(a)). Note that the critical value of p at which T_μ changes from $1, 2$ to $1, 3$ is $p = 3$. At this value, the discontinuity functions (defined in Section 1.2) for both pending transitions will cross zero at the same time $t = 3$. If the transition condition $L_2^{(1)}$ is redefined as $x + p \geq 6$, both transitions would become true at the same time, making the system nondeterministic. In [17], the strict inequality is used in (1.3) for the transition condition between mode 1 and 3, so that the transition to mode 2 should be taken at $p = 3$, much in the spirit of Witsenhausen [136] as discussed in the previous example above. However, the difficulty in this is that for values of $p < 3$, the transition to mode 3 should be taken, and the use of the strict inequality then poses the difficulty of determining a unique transition time for which the transition condition becomes true.

Note that the specification of a precedence relation function resolves this situation in a much more satisfactory manner. We simply set $L_2^{(1)}$ as $x + p \geq 6$, and the precedence relation function $K^{(m)} : Z \mapsto \inf Z$ for $m \in M$. This stipulates that the transition to mode 2 is given priority over that to mode 3, and defines a deterministic execution of the hybrid system for any $p \in [2, 4]$. This simple example will also highlight interesting issues for the parametric sensitivity analysis and optimization of hybrid systems which will be presented later in this Chapter.

Clearly, for general optimization problems, it is also possible to have more than 2 transitions whose timings converge at some critical value of the parameters, as long as one uniquely determined transition is allowed to be taken at that value. Again, this situation is handled easily by specifying the appropriate precedence relation function.

1.1.3 Zeno Phenomena

As mentioned previously, we will only consider deterministic hybrid systems with finite sequences T_τ terminating with epoch I_{n_e}. However, we note that the phenomena of having an infinite number of transitions occurring in a finite amount of time has also been studied extensively. In this case, T_τ is an infinite sequence. Sometimes called "chattering" behavior, this phenomena of having infinitely fast mode switches is also called *sliding* in variable structure systems and relay control systems [133, 92]. The term *Zeno* was used to describe a physical model of a bouncing ball in which the ball covers a finite distance in a finite time but with an infinite number of transitions [62]. Zeno phenomena arise due to modeling abstractions, since it is clear that no physical system in reality can be Zeno (after all, time never stops). For an overview on the theory and applications of Zeno phenomena, see [79], where the term *Zeno hybrid automata* was formally characterized.

For example, consider the following hybrid system,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} - 1, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x \geq 1), \\ T_1^{(1)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

$$\text{Mode 2}: \begin{cases} f^{(2)} = \dot{x} + 1, \quad J^{(2)} = \{1\}, \\ S^{(2)}(1) = 1, \qquad L_1^{(2)} := (x \leq 1), \\ T_1^{(2)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

where $t \in [0, 2]$, and $x(0) = 0$ with initial mode 1. For $t \in [0, 1]$, the value of x will be given by $x(t) = t$. At $t = 1$, the transition condition to mode 2 becomes true, and so a transition is taken to mode 2. Since we have continuity of the state variable as the transition function, the value of x in mode 2 at time $t = 1$ for epoch 2 is given by $x(\sigma_2 = 1) = 1$. Since we allow instantaneous transitions within our framework, the transition condition for the transition from mode 2 to mode 1 becomes true at $\sigma_2 = 1$, and so an instantaneous transition is taken back to mode 1. And the cycle repeats, because the transition condition for the transition from mode 1 to mode 2 becomes

true at $\sigma_3 = 1$. In essence, the hybrid system becomes "stuck" at the point $t = 1$, and will exhibit an infinite number of transitions at $t = 1$. The transitions between the discrete modes of the hybrid system dominate, and the continuous state is not allowed to evolve past the point $t = 1$. Hence, for this Zeno system, the solution of the hybrid system is not well defined for $t > 1$.

As mentioned previously, physical systems are clearly not Zeno in the sense that time does not get "stuck." Thus, when modeling physical systems, the onus is on the modeler to ensure that the hybrid system model does not exhibit Zeno behavior. Still, special care has to be taken, especially when reversible discontinuities are to be modeled, see Section 1.1.5 for a discussion.

One possible, practical way to prevent the solution of the hybrid system from getting "stuck" at a critical point in time is to impose a constraint on the evolution of the hybrid system that there must be a minimum duration to be spent in each epoch before a transition can be taken. While this does not prevent chattering between modes, it does ensure that there will never be an infinite number of transitions. The disadvantages of implementing such an approach is the following: (a) it automatically precludes instantaneous transitions from happening; and (b) no matter what value of the minimum duration is chosen, there could be a legitimate transition that should be taken to another mode before this duration is exceeded. This could potentially alter the execution of the hybrid system drastically.

1.1.4 Tangential Events and Transversality

The aim of this section is to discuss what we mean by a "well-behaved" execution of a hybrid system. Roughly speaking, an execution of a hybrid system is well-behaved when, given a small perturbation in the system parameters $\mathbf{p}$, the hybrid mode trajectory does not change, while the hybrid time trajectory and the trajectory of the continuous state variables change smoothly with respect to the perturbation. Thus, if a particular reference execution of a hybrid system is well-behaved, then one can expect to be able to predict the execution of the system within some small neighborhood or region of the reference system parameters. We shall give a more

precise definition of a well-behaved execution later in this section, after we have highlighted the many issues surrounding such a characterization.

For an idea of why such a characterization is important, consider the many efforts in the literature to define well-behaved executions. In [136], necessary optimality conditions for the optimal control problem posed are only developed for well-behaved solutions of the hybrid system (see the reference for the conditions for a well-behaved solution and a statement of the optimal control problem). In [30], assumptions are made on the form of the general unified model of the hybrid system for the hybrid control problem. In [90], conditions are developed to ensure continuity of the hybrid system (see the reference for the definition of a continuous hybrid system) with respect to the initial conditions of the hybrid system. A common observation that can be made is if the execution of a hybrid system is not well-behaved, then problems will arise with its analysis. Another common thread is the presence, in some form or other, of a *transversality* condition that is sufficient to make the execution well-behaved. Roughly speaking, in the words of Witsenhausen [136], an event is transversal if the impact of the trajectory (of the continuous state variables) on the transition surface (defined by the boundary of the set $G^{(m_i)}$ described in Section 1.1.1) is nontangential. In [124], functions (in C^∞) are introduced which describe *switching manifolds* of the hybrid automaton (analogous to the discontinuity functions in our proposed framework which will be introduced below), and the transversality condition is imposed on these manifold functions: an event is transversal if the manifold function that triggers the transition changes sign at the event, and its first derivative does not vanish there. In [90, Theorem III.2], sufficient conditions, based on a condition on the Lie derivative of the function describing the invariant set (in the spirit of Tavernini [124]), are proposed to ensure continuity of the hybrid system. Effectively, all of these conditions described above can be seen as variants of transversality conditions.

We shall now discuss how transversality conditions can be developed for our proposed modeling framework, as well as the difficulties in doing so. As will be seen shortly, we will abandon the idea of solely developing transversality conditions; in-

stead, we will characterize a well-behaved execution of a hybrid system as an execution for which the parametric sensitivities exist and are unique (for a discussion of parametric sensitivities, see Section 1.3). The reasons for doing this will become apparent, and can be summarized by the following: (a) transversality is just one of many conditions which need to be satisified, and is somewhat cumbersome to analyze within the proposed modeling framework, and (b) conditions for the existence and uniqueness of parametric sensitivities of hybrid systems have been developed in [63], and are a more intuitive and elegant way to characterize well-behaved executions.

Due to the flexibility of specifying transition conditions using atomic logical propositions connected by the `AND` and/or `OR` operators in our proposed modeling framework, transversality conditions are difficult to define and visualize without some additional work. First, we shall transform the atomic logical propositions into functions. Consider the transition condition $L_j^{(m)}$, for $j \in J^{(m)}, m \in M$. The relational atoms that make up the condition can be rearranged to:

$$g_{j,l}^{(m)}\left(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{p}, t\right) \leq 0,\ l = 1, \ldots, n_j^{(m)} \tag{1.6}$$

in order to define atomic *discontinuity functions*, where $n_j^{(m)}$ is the total number of separate relational atoms making up the transition condition. Viewed this way, the transition condition becomes true whenever the atomic discontinuity functions are satisfied according to the logic of the transition condition. For example, consider the following transition condition,

$$L_1^{(1)} := (x_1 \geq 0) \vee (x_2 \leq 0).$$

The atomic discontinuity functions are then given by

$$g_{1,1}^{(1)} = -x_1, \quad g_{1,2}^{(1)} = x_2.$$

The transition condition thus becomes true when either one of the discontinuity functions becomes nonpositive. Roughly speaking, each atomic logical proposition changes

its value when its associated discontinuity function crosses the zero axis (the time axis). Henceforth, we will assume that all atomic discontinuity functions, $g_{j,l}^{(m)}$, are continuous on $\mathbb{R}^{n_x^{(m)}} \times \mathbb{R}^{n_x^{(m)}} \times P \times [t_0, t_f]$ for all $m \in M, j \in J^{(m)}, l \in \{1, \ldots, n_j^{(m)}\}$.

Now, we are in a position to describe what we mean by a tangential and transversal event for a transition condition with a single atomic logical proposition. Consider any arbitrary $\mathbf{p}^* \in P$, epoch I_i, $i \in \{1, \ldots, n_e\}$ and current mode $m_i \in M$. In this case, we have $J^{(m_i)} = \{1\}$ and $n_1^{(m_i)} = 1$, with $g_{1,1}^{(m_i)}$ as the discontinuity function. Consider now a transition that is taken to mode $S^{(m_i)}(1)$ at time $t^* \geq \sigma_i$. There are only two possibilities, either

$$g_{1,1}^{(m_i)}(\dot{\mathbf{x}}^{(m_i)}(\mathbf{p}^*, t^*), \mathbf{x}^{(m_i)}(\mathbf{p}^*, t^*), \mathbf{p}^*, t^*) < 0 \tag{1.7}$$

or

$$g_{1,1}^{(m_i)}(\dot{\mathbf{x}}^{(m_i)}(\mathbf{p}^*, t^*), \mathbf{x}^{(m_i)}(\mathbf{p}^*, t^*), \mathbf{p}^*, t^*) = 0. \tag{1.8}$$

Condition (1.7) can be true only when $t^* = \sigma_i$, i.e., there is an instantaneous transition for the epoch I_i. This can happen, for example, when an impulsive transition function maps the final value of $\mathbf{x}^{(m_{i-1})}(\mathbf{p}^*, \tau_{i-1})$ in epoch I_{i-1} such that the initial conditions for epoch I_i, $\mathbf{x}^{(m_i)}(\mathbf{p}^*, \sigma_i)$, automatically satisfies the transition condition (when considering the closed set $G^{(m_i)}$ described by the transition condition, the transition function maps the final value of $\mathbf{x}^{(m_{i-1})}(\mathbf{p}^*, \tau_{i-1})$ in epoch I_{i-1} to a point in the interior of $G^{(m_i)}$). For the case where (1.7) is true, we will consider it a *transversal* (instantaneous) event.

Consider now the case where (1.8) is true. If $t^* = \sigma_i$, we have an instantaneous event, and we will consider it a *tangential* (instantaneous) event (typically, such events correspond to critical values of $\mathbf{p}^*$ such that the mode sequence of the hybrid system changes qualitatively in a neighborhood around $\mathbf{p}^*$). Note that if $t^* = t_f$, we have a tangential instantaneous event at the final time. If $\sigma_i < t^* < t_f$, then we will consider the event *transversal* if the following condition is satisfied,

$$\exists \varepsilon > 0 \text{ s.t. } g_{1,1}^{(m_i)}(\dot{\mathbf{x}}^{(m_i)}(\mathbf{p}^*, \alpha), \mathbf{x}^{(m_i)}(\mathbf{p}^*, \alpha), \mathbf{p}^*, \alpha) < 0, \ \forall \alpha \in (t^*, t^* + \varepsilon) \tag{1.9}$$

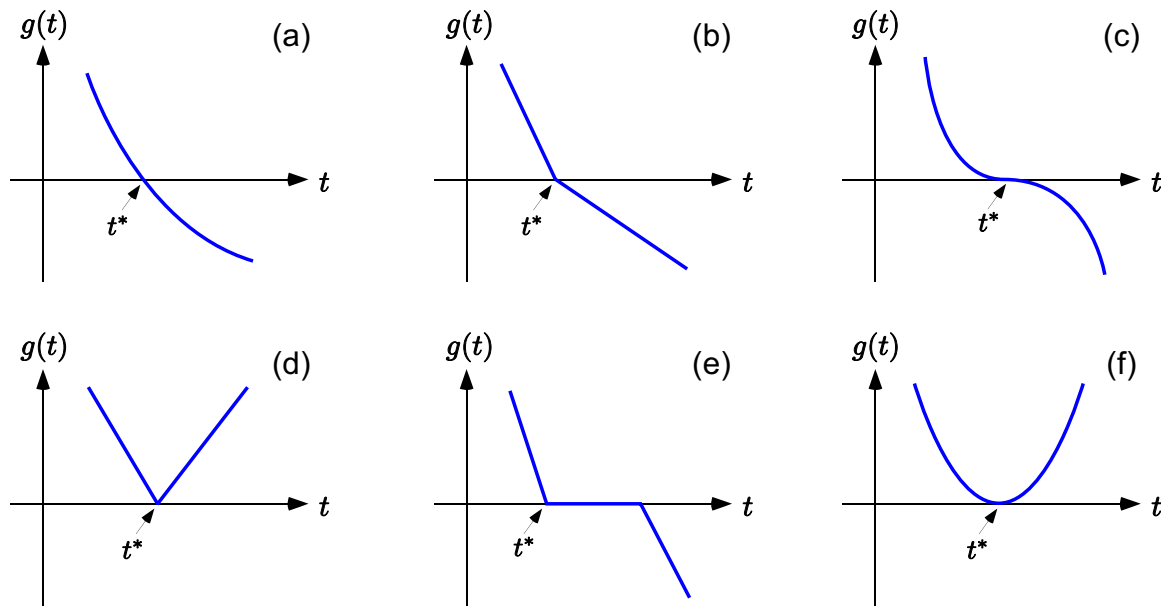


Figure 1-3: Types of discontinuity functions.

otherwise, the event is *tangential*. In order to use the above condition, we have to define the quantities $\dot{\mathbf{x}}^{(m_i)}(\mathbf{p}^*, \alpha)$ and $\mathbf{x}^{(m_i)}(\mathbf{p}^*, \alpha)$, $\forall \alpha \in (t^*, t^* + \varepsilon)$. To do this, we will use the concept of *discontinuity locking*, which will be described in detail in Section 1.2. The idea is to "lock" the dynamics of the current mode, m_i, up to the time $t^* + \varepsilon$, and effectively ignore any pending transitions. By assumption, if $t^* < t_f$, there will exist some ε such that $t^* + \varepsilon < t_f$, where the solution of the ODE system in mode m_i exists and is unique, and thus $\dot{\mathbf{x}}^{(m_i)}(\mathbf{p}^*, \alpha)$ and $\mathbf{x}^{(m_i)}(\mathbf{p}^*, \alpha)$ are uniquely defined for all $\alpha \in (t^*, t^* + \varepsilon)$.

Figure 1-3 shows some examples of discontinuity functions for the case where $t^* > \sigma_i$. Note that for all cases, the event occurs at the time t^* as indicated. According to the definition above, cases (a), (b) and (c) in Figure 1-3 are transversal events, while cases (d), (e) and (f) are tangential events. What is perhaps controversial is the classification of case (c) as a transversal event, even though the trajectory of the discontinuity function is tangent to the zero time axis at the point of the event. This is because such an event can still possibly lead to a well behaved execution of the hybrid system, as a small perturbation of the system parameters will not lead to a qualitatively different mode sequence.

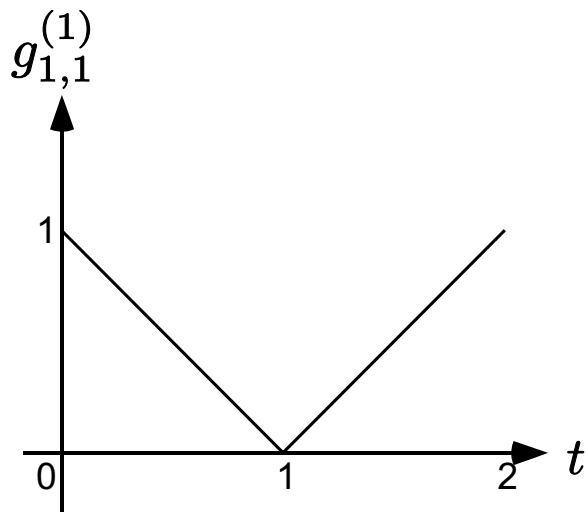


Figure 1-4: Example of a nonsmooth discontinuity function.

Note that even though the discontinuity function is continuous (indeed, even if we assume that the discontinuity function is continuously differentiable on its domain), there is no guarantee that it will be smooth, because there are no guarantees that the continuous state of the hybrid system will be smooth in time. For example, consider the following ODE hybrid system with piecewise constant right hand sides,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \begin{cases} \dot{x} + 1 \text{ if } t < 1 \\ \dot{x} - 1 \text{ if } t \geq 1 \end{cases}, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x \leq 0), \\ T_1^{(1)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$
$$\text{Mode 2}: f^{(2)} = \dot{x}, \quad J^{(2)} = \emptyset.$$

where $t \in [0, 2]$, and $x(0) = 1$ with initial mode 1. Then, the discontinuity function $g_{1,1}^{(1)} = x$, and is shown in Figure 1-4, which corresponds to case (d) of Figure 1-3.

Of course, one way to mitigate this problem is to treat the points of discontinuities of the dynamics as time events. For the above example, one can construct the

following equivalent hybrid system,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} + 1, \quad J^{(1)} = \{1, 2\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (t \geq 1), \\ S^{(1)}(2) = 3, \qquad L_2^{(1)} := (x \leq 0), \\ T_1^{(1)} = x(\sigma_{i+1}) - x(\tau_i), \\ T_2^{(1)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

$$\text{Mode 2}: \begin{cases} f^{(2)} = \dot{x} - 1, \quad J^{(2)} = \{1\}, \\ S^{(2)}(1) = 3, \qquad L_1^{(2)} := (x \leq 0), \\ T_1^{(2)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

$$\text{Mode 3}: f^{(3)} = \dot{x}, \quad J^{(3)} = \emptyset.$$

where $t \in [0, 2]$, and $x(0) = 1$ with initial mode 1, and the precedence relation $K^{(1)} : Z \mapsto \inf Z$. Note that at $t = 1$, there will be a transition from Mode 1 to Mode 2 because of the precedence relation, followed by an instantaneous transition from Mode 2 to Mode 3. If the precedence relation $K^{(1)} : Z \mapsto \sup Z$ was used instead, then there will be a transition from Mode 1 to Mode 3 at $t = 1$ instead. In either case, there will be a qualitative change in the mode sequence if there is a small perturbation in the initial conditions of the hybrid system, which implies that the execution of the hybrid system is not well behaved. Thus, reformulating the hybrid system into one where the discontinuity functions are smooth will not make the execution of the hybrid system well behaved, because the original points of nonsmoothness of the discontinuity functions will often correspond to points at which competing transitions of the reformulated hybrid system coincide.

Nevertheless, it will be useful to examine situations where one can be sure that the discontinuity functions are smooth with respect to time. In these cases, we will replace (1.9) with the following condition:

$$\dot{g}_{1,1}^{(m_i)}(\ddot{\mathbf{x}}^{(m_i)}, \dot{\mathbf{x}}^{(m_i)}, \mathbf{x}^{(m_i)}, \mathbf{p}^*, t^*) < 0, \tag{1.10}$$

in the spirit of Tavernini [124]. Note that (1.10) is not equivalent to (1.9). To see this, consider Figure 1-3. According to (1.10), case (a) is transversal, while cases (c) and (f) are tangential. In other words, (1.10) is a stronger transversality condition. On the other hand, it seems easier to implement as an additional atomic proposition to be added to the current transition condition via the `AND` operator. However, we meet with a technical difficulty here. Notice the form of (1.10) involves the use of a strict inequality. To add the condition within our proposed modeling framework, we have to add the logical condition corresponding to the following condition instead,

$$\dot{g}_{1,1}^{(m_i)}(\ddot{\mathbf{x}}^{(m_i)}, \dot{\mathbf{x}}^{(m_i)}, \mathbf{x}^{(m_i)}, \mathbf{p}^*, t^*) \leq 0. \tag{1.11}$$

Of course, the moment we do this, cases (a), (c) and (f) in Figure 1-3 become transversal. What this means is that although one can verify whether an event is transversal, one cannot simply add a condition that will ensure that an event is transversal. However, we will see where the addition of such a condition will become useful for modeling reversible discontinuities in the next section.

In addition, a word of caution has to be said about enforcing transversality through the addition of (1.11) as another atomic logical proposition that has to be satisfied. Since the modeling framework allows instantaneous transitions, adding (1.11) to a transition condition may wrongly prevent a transversal instantaneous transition from occurring. In addition, the situation becomes more complicated to analyze when considering transition conditions involving multiple atomic logical propositions. One can easily come up with examples for atomic logical propositions joined together by the `AND` operator. If (1.11) were to be added for each atomic logical proposition, there is no guarantee that the original logic will be preserved, because any atomic discontinuity functions which satisfy (1.7) should not have to satisfy (1.11) at the same time.

Hence, while one can verify whether transversality holds for a particular execution of the hybrid system, it is very difficult to impose conditions, a priori, on a hybrid system such that transversality will always hold. Thus, it is argued that it would be

better to characterize a well behaved execution of a hybrid system as one in which the parametric sensitivities of the hybrid system exist and are unique (see Section 1.3). Note that this automatically encompasses the "continuous" dependence on the initial conditions of the hybrid system as described in [90], since additional parameters can be introduced to serve as the initial conditions of the hybrid system.

Note that sufficient conditions for the existence and uniqueness of the parametric sensitivities of hybrid systems have been developed in [63]. Two of the key assumptions that are made are smoothness in the neighborhood of the transition times, and that for any transition, only one relational expression (atomic logical proposition) activates. Note that transversality conditions would fall under the category of the first assumption. Indeed, for an event whose discontinuity function is tangent to the time axis (case (f) for Figure 1-3), there can be a nonsmoothness in the event time if the discontinuity function depends on the system parameters, see e.g., the ODE example (1.21) - (1.22) presented later.

The second key assumption that only one relational expression activates is important, especially in our proposed modeling framework where transitions can have multiple atomic logical propositions linked together by `AND` and `OR`. To see this, consider the following hybrid system,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} - 1, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x^2 \leq p) \vee (x \geq p), \\ T_1^{(1)} = x(p, \sigma_{i+1}) - x(p, \tau_i), \end{cases}$$
$$\text{Mode 2}: f^{(2)} = \dot{x}, \quad J^{(2)} = \emptyset.$$

where $p \in [-0.5, 0.5]$, $t \in [0, 2]$, and $x(0) = -1$ with initial mode 1. Note that for this hybrid system, the mode sequence is 1, 2 for all $p \in [-0.5, 0.5]$, and that the atomic proposition $x \geq p$ is transversal according to condition (1.9) for any $p \in [-0.5, 0.5]$. Since the transition condition involves an `OR`, one might expect that the system would be well behaved for all values of $p \in [-0.5, 0.5]$. However, this is not the case.

Note that for $p < 0$, the atomic logical proposition $(x \geq p)$ is satisfied first, and

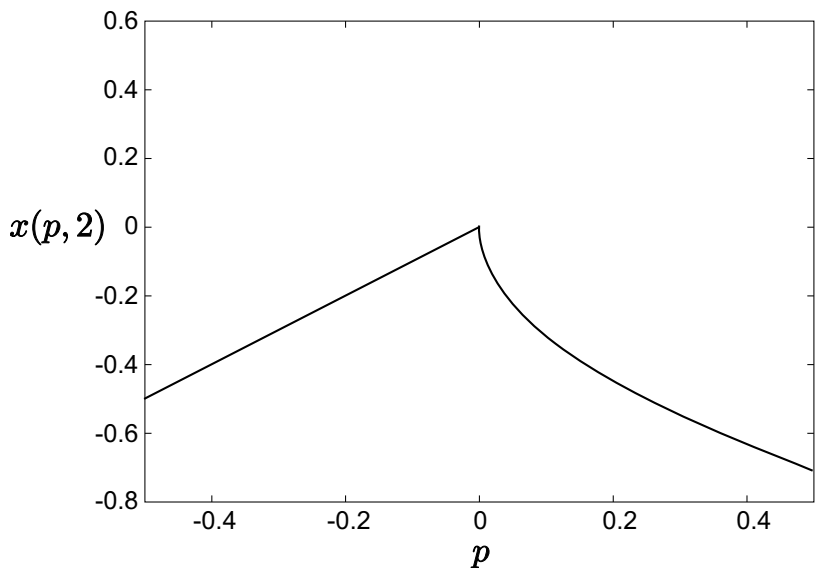


Figure 1-5: Plot of $x(p, 2)$ against p.

is transversal, while for $p > 0$, the atomic logical proposition $(x^2 \leq p)$ is satisfied first, and is also transversal. At $p = 0$, both atomic logical propositions are satisfied at the same time, however, only $(x \geq p)$ is transversal. Figure 1-5 shows the plot of $x(p, 2)$ against p, and it can be seen that there is a point of nonsmoothness at $p = 0$. Obviously, the parametric sensitivities of the hybrid system do not exist at that critical point. This example does not contradict the sufficient conditions proposed in [63] because at the critical point of $p = 0$, there is not just one relational expression that becomes true.

The example above highlights the difficulties involved in deciding on transversality conditions for transition conditions involving multiple logical propositions. However, even when transversality is satisfied for all atomic logical propositions, the parametric sensitivities may not exist, as the following example will show. Consider the following hybrid system,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} + 1, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x \leq p) \vee (x \leq 0), \\ T_1^{(1)} = x(p, \sigma_{i+1}) - x(p, \tau_i), \end{cases}$$

$$\text{Mode 2}: f^{(2)} = \dot{x}, \quad J^{(2)} = \emptyset.$$

where $p \in [-0.5, 0.5]$, $t \in [0, 2]$, and $x(0) = 1$ with initial mode 1. Note that both atomic propositions are transversal according to condition (1.9) or (1.10) or (1.11) for any $p \in [-0.5, 0.5]$. Also, note that the mode sequence is 1, 2 for all $p \in [-0.5, 0.5]$. The value of $x(p, 2)$ is given by the following equation,

$$x(p,2) = \begin{cases} 0 & \text{if } -0.5 \leq p < 0, \\ p & \text{if } 0 \leq p \leq 0.5. \end{cases}$$

Clearly, there is a point of nonsmoothness at $p = 0$ where the parametric sensitivities do not exist. Of course, this behavior is not solely restricted to transition conditions involving the `OR` operator. For example, consider the following hybrid system,

$$\text{Mode 1} : \begin{cases} f^{(1)} = \dot{x} + 1, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (t \geq 1) \wedge (x \leq 0), \\ T_1^{(1)} = x(p, \sigma_{i+1}) - x(p, \tau_i), \end{cases}$$
$$\text{Mode 2} : f^{(2)} = \dot{x} - 1, \quad J^{(2)} = \emptyset.$$

where $p \in [0.5, 1.5]$, $t \in [0, 2]$, and $x(0) = p$ with initial mode 1. Again, note that both atomic propositions are transversal according to condition (1.9) or (1.10) or (1.11) for any $p \in [0.5, 1.5]$. Also, note that the mode sequence is 1, 2 for all $p \in [0.5, 1.5]$. The value of $x(p, 2)$ is given by the following equation,

$$x(p,2) = \begin{cases} p & \text{if } 0.5 \leq p < 1, \\ 2 - p & \text{if } 1 \leq p \leq 1.5. \end{cases}$$

There is clearly a point of nonsmoothness at $p = 1$ where the parametric sensitivities do not exist. Again, this is because at $p = 1$, both atomic logical propositions become true at the same time, which violates the assumption made in [63]. These examples illustrate that, within our proposed modeling framework, transversality is not sufficient to guarantee the existence of the parametric sensitivities, and thus a well behaved execution of the hybrid system.

Another important point to note about the theory developed in [63] is that it does not apply for transversal instantaneous transitions that satisfy (1.7), because it is assumed that (1.8) holds at all events. On the other hand, tangential instantaneous transitions will most likely violate the key smoothness assumption made. Thus, within our modeling framework, instantaneous transitions will, in general, not lead to well behaved executions of hybrid systems.

Finally, we note that even though an execution of a hybrid system may not be well behaved, it does not mean that the solution of the hybrid system does not exist. For example, consider the last example with the value of $p = 0$. The execution of the hybrid system exists and is unique for that value of $p = 0$. It merely means that the solution of the hybrid system does not change smoothly with a small perturbation in the value of p.

1.1.5 Modeling Reversible Discontinuities

Reversible discontinuities occur when we have switching behavior between two modes, A and B, where the transition condition for the transition between mode A and B is the negation of the transition condition for the transition between mode B and A. This is commonly represented by the `IF .. THEN .. ELSE .. END` structure in modeling languages for describing the dynamic system, e.g., the `EQUATION` section of an ABACUSS II [41] input file, or the res0.f input file using DAEPACK [130].

In this section, we will highlight some of the difficulties with modeling reversible discontinuities within the proposed modeling framework. Note that the same difficulties exist within the frameworks of [136, 30, 90]. Indeed, within the framework of [136], reversible discontinuities cannot be modeled because it is assumed that the arrival and departure sets are disjoint; we shall see shortly that this assumption can never be satisfied.

There are many instances where it is useful to model reversible discontinuities in physical processes. Reversible discontinuities occur naturally when modeling physico-chemical mechanisms. This has been discussed in detail in [15, 18]. Here, we shall provide a simple example of a physico-chemical discontinuity: that of a tank with a

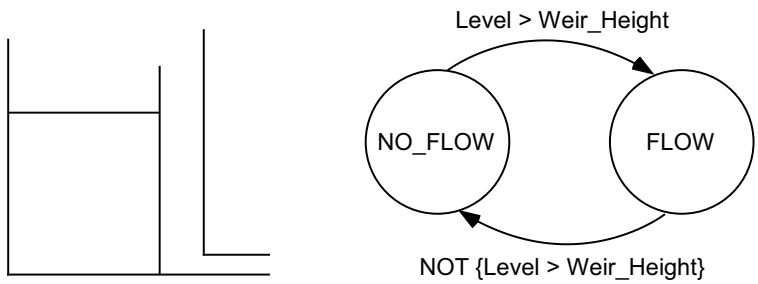


Figure 1-6: Tank with a weir.

weir. Consider a vessel containing an overflow weir that regulates the flow of liquid from it. During normal operation, this device will maintain a relatively constant holdup of material, but if, for any reason, the level of liquid in the vessel drops to or below the height of the weir, flow from the vessel will cease until the level rises above that of the weir again. This can be modeled conveniently as a hybrid system with two modes, `FLOW` and `NO_FLOW` respectively, with the transition condition between the two modes expressed as a function of the level of liquid in the vessel. Figure 1-6 depicts a typical tank with a weir, as well as the associated hybrid automaton model. Clearly, the transition condition for one transition is the negation of the condition for the other.

Another example of reversible discontinuities arises in the modeling of min and max operations. Consider the following dynamic system,

$$\dot{x}_1 = \min(x_1, x_2),$$
$$\dot{x}_2 = x_1.$$

Consider the term $z = \min(x_1, x_2)$. Clearly, we must have

$$z = \begin{cases} x_1 & \text{if } x_1 < x_2 \ , \\ x_2 & \text{if } x_1 \geq x_2 \ . \end{cases}$$

This can also be expressed as the following logical structure,

$$\text{if } (x_1 < x_2) \text{ then } z = x_1 \text{ else } z = x_2 \text{ endif.}$$

The equivalent hybrid system for the dynamic system above would be the following,

$$\text{Mode 1}: \begin{cases} f_1^{(1)} = \dot{x}_1 - x_1, \; f_2^{(1)} = \dot{x}_2 - x_1 \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (x_1 < x_2), \\ T_1^{(1)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

$$\text{Mode 2}: \begin{cases} f_1^{(2)} = \dot{x}_1 - x_2, \; f_2^{(2)} = \dot{x}_2 - x_1 \quad J^{(2)} = \{1\}, \\ S^{(2)}(1) = 1, \qquad L_1^{(2)} := (x_1 \geq x_2), \\ T_1^{(2)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$

where the transition condition $L_1^{(1)}$ is the negation of the transition condition $L_1^{(2)}$. Immediately, we see the problem. Due to the fact that one condition is the negation of the other, one of the conditions will not define a closed set. In this example, a strict inequality is used to define $L_1^{(1)}$ which is not allowed within our proposed modeling framework, as no unique, earliest time can be determined when the condition first becomes true, and thus, when the transition is to be made.

Of course, one way to eliminate this problem is to force the transition condition to define a closed set, by replacing the strict inequality with a weak inequality, i.e., replace $L_1^{(1)}$ with the following,

$$L_1^{(1)} := (x_1 \leq x_2).$$

This way, the form of the hybrid system satisfies the conditions of the proposed modeling framework. However, there is again a major problem with this model of a reversible discontinuity: the system is inherently Zeno. To see this, assume that we are in Mode 1, and epoch I_i, and a transition is made at time t^* to Mode 2. At the transition time, we have

$$x_1(t^*) = x_2(t^*).$$

Since we have state continuity, we have $x_1(\sigma_{i+1}) = x_2(\sigma_{i+1}) = x_1(t^*)$. This satisfies the transition condition $L_1^{(2)}$, which means an instantaneous transition is taken back to Mode 1. And the cycle repeats.

Note that this Zeno behavior will persist for any kind of discontinuity function shown in Figure 1-3, even the transversal case (a). This behavior is inherent for the model proposed for reversible discontinuities above, and will be the default behavior for any of the modeling frameworks proposed to date. It is outside the scope of this thesis to devise a theoretical modeling framework that can handle this issue satisfactorily; that is left for future work.

Instead, we will propose a fix for modeling reversible discontinuities by adding a transversality condition, (1.11), to the transition conditions. Of course, this assumes that the original discontinuity functions derived from the reversible discontinuity are smooth in time. For the example above, this means replacing $L_1^{(1)}$ and $L_1^{(2)}$ with the following,

$$L_1^{(1)} := (x_1 \leq x_2) \wedge (\dot{x}_1 - \dot{x}_2 \leq 0),$$
$$L_1^{(2)} := (x_1 \geq x_2) \wedge (\dot{x}_2 - \dot{x}_1 \leq 0).$$

It is clear that the addition of such a transversality condition will prevent Zeno behavior for discontinuity functions which were transversal such as case (a) in Figure 1-3, as discussed in the previous section. However, Zeno behavior will still occur for tangential discontinuity functions such as cases (c) and (f) in Figure 1-3. What this means is that, within the current modeling framework, there is no effective way to model reversible discontinuities without preventing Zeno behavior (theoretically) for tangential events, even if transversality conditions are employed. It appears that a totally new definition of a transition has to be devised in order to establish a theoretical modeling framework that prevents Zeno behavior for reversible discontinuities.

Finally, we note that in practice, the simulation of reversible discontinuities does not pose problems for a vast majority of physical systems. This is because of the practicalities of algorithms for event detection and reinitialization after every event, and the mechanisms devised to prevent discontinuity sticking (see Section 1.2 for a discussion on the simulation of hybrid systems). Roughly speaking, after reinitialization, the value of the active discontinuity function is guaranteed to be negative, which

means that an instantaneous transition will not be taken back into the predecessor mode for reversible discontinuities. On the other hand, chattering behavior might still be observed, where the simulator takes very small steps while switching between two modes. It has been our experience that when this occurs, it is very likely that a modeling error has been made, and such behavior can usually be remedied by fixing the model appropriately, or, (presumably) the rapid switching actually reflects the behavior of the physical system.

1.1.6 A Hybrid Automaton Model

We will now summarize the hybrid automaton model. A *hybrid automaton* is given by $\mathcal{H} = (M, \mathcal{V}, \mathcal{F}, \mathcal{T}^0, \mathcal{J}, \mathcal{L}, \mathcal{T}, \mathcal{S}, \mathcal{K})$ where

- M is the index set of modes.
- $\mathcal{V}$ is a mapping that maps the index set of modes to a finite set of variables, $\mathcal{V}(m) = V^{(m)}$ for any $m \in M$.
- $\mathcal{F}$ is a mapping that maps the index set of modes to a system of differential equations, $\mathcal{F}(m) = \mathbf{f}^{(m)}$ for any $m \in M$.
- $\mathcal{T}^0$ is a mapping that maps the index set of modes to a system of consistent initial conditions, $\mathcal{T}^0(m) = \mathbf{T}^{(m,0)}$ for any $m \in M$.
- $\mathcal{J}$ is a mapping that maps the index set of modes to a finite set of pending transitions, $\mathcal{J}(m) = J^{(m)}$ for any $m \in M$.
- $\mathcal{L}$ is a mapping that maps the index set of modes and the index set of pending transitions to a logical transition condition, $\mathcal{L}(m, j) = L_j^{(m)}$ for any $m \in M$, $j \in J^{(m)}$.
- $\mathcal{T}$ is a mapping that maps the index set of modes and the index set of pending transitions to system of transition functions, $\mathcal{T}(m, j) = \mathbf{T}_j^{(m)}$ for any $m \in M$, $j \in J^{(m)}$.

- $\mathcal{S}$ is a mapping that maps the index set of modes to a mapping of successor modes, $\mathcal{S}(m) = S^{(m)}$ for any $m \in M$.

- $\mathcal{K}$ is a mapping that maps the index set of modes to a precedence relation function, $\mathcal{K}(m) = K^{(m)}$ for any $m \in M$.

A finite, deterministic, non-Zeno *execution* of a hybrid automaton $\mathcal{H}$ is given by $\mathcal{E}(\mathcal{H}) = (T_\mu, T_\tau, \mathbf{p})$, such that the following conditions are satisfied,

1. T_μ and T_τ are finite sequences.

2. $\mathcal{T}^0(m_1)$ provides a consistent set of initial conditions for mode m_1.

3. For the transition between mode m_i in epoch I_i to mode m_{i+1} in epoch I_{i+1} for any $i \in \{1, \ldots, n_e - 1\}$,

   (a) If $\tau_i > \sigma_i$, there does not exist a time $t^* \in [\sigma_i, \tau_i)$, such that any pending transition condition $\mathcal{L}(m_i, j)$ is `TRUE` at t^*, for any $j \in \mathcal{J}(m_i)$.

   (b) At τ_i, at least one pending transition condition $\mathcal{L}(m_i, j)$ is `TRUE`, where $j \in \mathcal{J}(m_i)$.

   (c) The successor mode $m_{i+1} = \mathcal{S}(m_i)(\mathcal{K}(m_i)(Z))$ where Z is a set that contains the indices of transition conditions which are `TRUE` at τ_i.

   (d) $\mathcal{T}(m_i, \mathcal{K}(m_i)(Z))$ provides a consistent set of initial conditions for mode m_{i+1}, where Z is a set that contains the indices of the transition conditions which are `TRUE` at τ_i.

4. For each mode m_i in epoch I_i, for any $i \in \{1, \ldots, n_e\}$, the values of $\mathbf{x}^{(m)}(\mathbf{p}, t)$ are given by the solution of the IVP whose dynamics are given by $\mathcal{F}(m_i)$ with initial conditions $\mathbf{x}^{(m_i)}(\mathbf{p}, \sigma_i)$ given by $\mathcal{T}^0(m_1)$ for $i = 1$, and $\mathcal{T}(m_i, \mathcal{K}(m_i)(Z))$ for $i = 2, \ldots, n_e$, where Z is a set that contains the indices of the transition conditions which are `TRUE` at τ_i.

Note that condition 1 automatically makes the execution of the hybrid system finite and non-Zeno. By specifying T_μ, T_τ and $\mathbf{p}$, the initial mode, initial time, final time

Table 1.1: Comparison of modeling frameworks: Thesis refers to this thesis, "sens" refers to parametric sensitivities, and "trans" refers to transversality.

Feature	Thesis	[136]	[90]	[62]	[30]	[5]
Deterministic formulation	yes	yes	yes	yes	yes	no
Can determine a unique transition time	yes	yes	yes	no	yes	no
Can determine a unique successor for competing transitions	yes	no	no	no	no	no
Can model reversible discontinuities	partially	no	no	no	no	no
Use of logical propositions in transition conditions	yes	no	no	yes	no	no
Characterization of well-behaved executions	sens	trans	trans	no	trans	no

and system parameters are specified. Hence, the solution of the hybrid system can be uniquely determined, because our proposed modeling framework is deterministic as discussed in the preceding sections.

A *well behaved* execution $\mathcal{E}(\mathcal{H})$ is defined as a finite, deterministic, non-Zeno execution of a hybrid system for which the parametric sensitivities of the hybrid system exist and are unique. Table 1.1 summarizes the features of various modeling frameworks in the literature against the many modeling issues raised in the preceding sections.

The time evolution, or execution, of a hybrid system may then be viewed in the following manner. Starting from consistent initial conditions in some specified initial mode, the continuous state evolves according to the relevant differential equations until an event occurs, at which point the discrete subsystem influences the continuous subsystem, for example, if this is a switching event, the system switches to a new mode, and the continuous state evolves in the new mode until another such event, etc. One important facet of hybrid system behavior is thus the sequence of modes that is visited during a particular execution, encapsulated in the hybrid mode trajectory, T_μ, which has a one to one correspondence with the hybrid time trajectory T_τ. T_μ is characterized by specific parameter values, the initial mode and initial conditions (an individual mode may be visited many times along a time trajectory).

As a practical example of a hybrid system, consider the hybrid dynamic model of a pressure vessel located in a chemical plant [19]. The tank may be supplied with

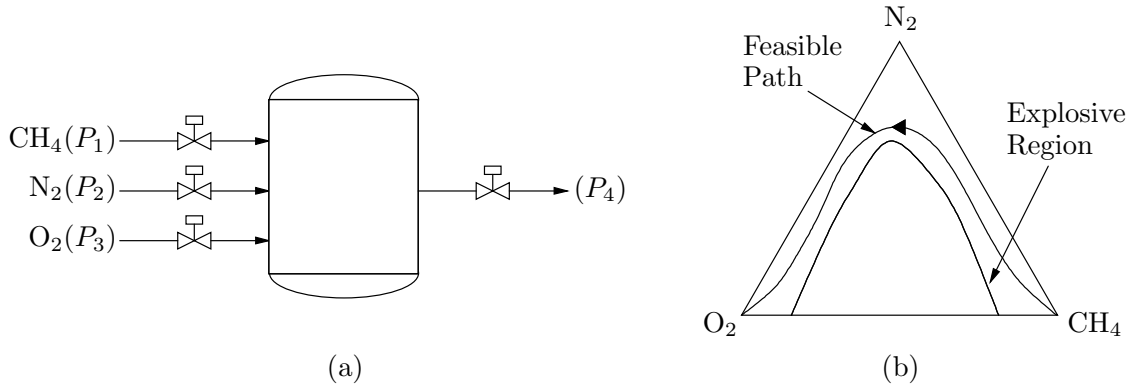


Figure 1-7: Schematic of pressure vessel: (a) Process flowsheet (b) Mole fraction space.

oxygen, nitrogen and/or methane via three separate lines, and gas mixtures may be withdrawn from the vessel through a fourth line. The flow in each line is regulated by an open/close non-return valve as shown in Figure 1-7. Discontinuities in the flow through the lines appear because the non-return valves are modeled using three distinct modes: zero flow, laminar/turbulent and choked flow regimes. It is evident that the equations describing the flow/pressure relationships will differ in each mode.

While the modeling framework presented above is particularly suitable for the analysis and understanding of the interactions between the continuous and discrete subsystems, it poses a practical problem for the design of simulation software, because many applications require a combinatorial number of modes for their description. For example, suppose that we wish to create a plant model with 100 instances of this valve model. Since each valve could independently be in any flow regime at any point in time, this would imply 3^{100} modes for the plant model. Enumeration of these modes by a software system or algorithm is clearly not practical. This problem has been noted as early as [72].

Indeed, a lot of ingenuity in designing software and numerical algorithms for the analysis of hybrid systems goes into avoiding enumeration of the modes while still retaining a flexible modeling framework for the user. It should be noted that the most convenient representation is usually dictated by the class of application, and thus many different modeling languages have been proposed over the years (see [104]

and [40] for early reviews, and [69], [8], [6], [9], and [10]). Our efforts in the Process Systems Engineering Laboratory (PSEL) are embodied in the software systems ABACUSS II [41] and JACOBIAN^{TM}[87] (for chemical engineering applications), and DAEPACK [130, 128] (for general user supplied FORTRAN code), where each instance of the flow-pressure equation, or each `IF` statement in a FORTRAN code, records its own mode (e.g., zero flow, laminar/turbulent or choked). Subsequently, when, for example, a function evaluation is required, the simulation executive can query the set of equations currently active as those contributed by each of these individual modes; this avoids exhaustive enumeration of all possible modes.

An implementation of this methodology is illustrated in Figure 1-8. Assuming isothermal conditions, a hybrid dynamic model of the pressure vessel in Figure 1-7 is represented by the equations shown, where $I = \{1, 2, 3\}$ is the index set for the chemical species present, $K = \{1, 2, 3, 4\}$ is the index set for the valves, the constant V is the volume of the tank, $y_{i \in I}(t)$ are the mole fractions in the vessel, $y_{k \in K, i \in I}(t)$ are the mole fractions in the flow through each line, $P_{k \in K}(t)$ are the known supply and discharge pressures, and the controls $u_{k \in K}(t) \in \{0, 1\}$ are known time profiles for the on/off signals to the respective valves. We have 5 submodels in the overall model, where submodels 1 to 4 represent the four non-return valves with 3 modes in each submodel, and submodel 5 represents the system of equations (dynamic mass balances and the ideal gas equation of state) that are invariant to the system.

It is worth noting that we do not have a direct transition from the mode `Zero Flow` to the mode `Choked Flow`. Instead, if the valve signal is on, and the pressure drop across the valve is high enough, we would have a transition from mode `Zero Flow` to mode `Laminar/Turbulent`, followed by an instantaneous transition from mode `Laminar/Turbulent` to mode `Choked Flow`. This is an example where an instantaneous epoch would appear (multiple transitions at a given time). A possible alternative way to model the system is to include an explicit transition between the said modes with a transition condition of $(u_k \geq 0.5) \wedge (P_{out}/P_{in} \leq 0.53)$ where P_{in} and P_{out} are the respective inlet and outlet valve pressures, and specify a precedence rule that states that whenever transitions to mode `Laminar/Turbulent` and

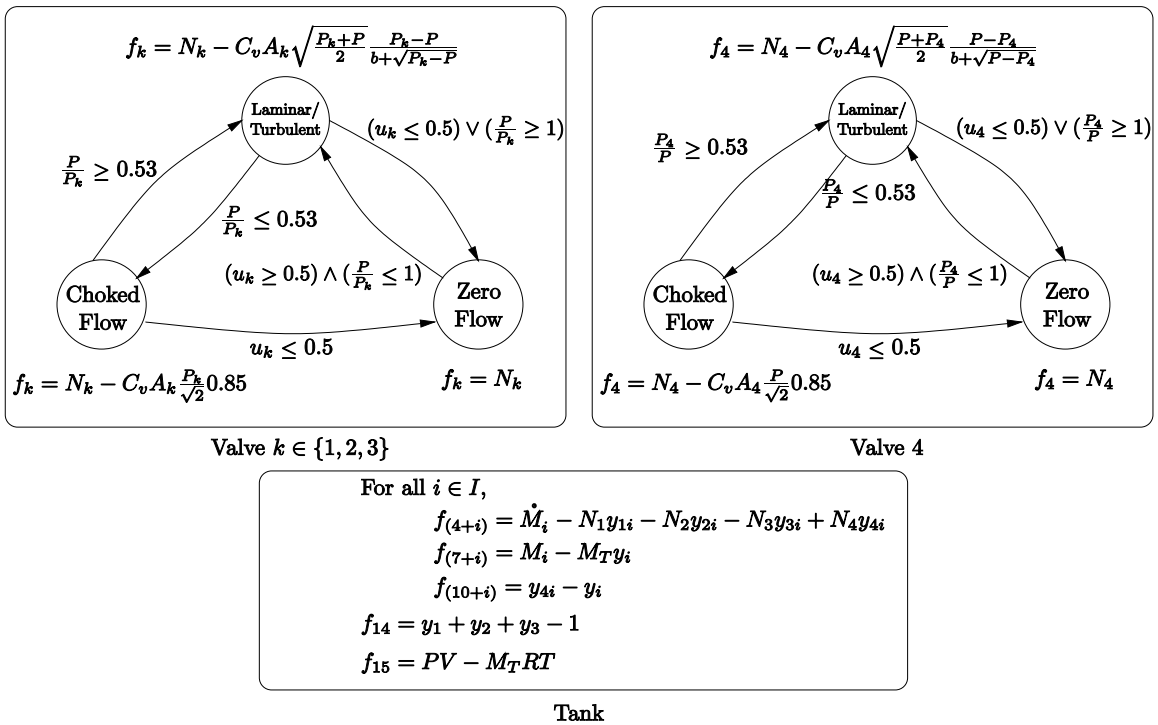


Figure 1-8: Hybrid dynamic model of pressure vessel.

`Choked Flow` are active at the same time, the transition to mode `Choked Flow` is always taken. Also, note that we have reversible discontinuities while modeling the transitions between the `Choked Flow` mode and the `Laminar/Turbulent` mode. As discussed in the previous section, we would add the relevant transversality conditions for these reversible discontinuities, which have not been shown in Figure 1-8 due to space constraints.

It can easily be shown that this methodology of tracking the modes of the individual components can be expressed in the hybrid automaton framework, and vice versa. Clearly, the only difference is in the way that the discrete state of the hybrid system is described and partitioned. In the hybrid automaton framework, there is a single index enumerating all possible discrete states of the system, whilst in practice, each component of the system has an associated index tracking the active mode of the component. Consider the pressure tank example. We have $M = \{1, 2, \ldots, 81\}$, whereas in practice, we have three modes for each valve, $V_k \in N = \{1, 2, 3\}, \ \forall\, k \in K$. It is thus trivial to obtain a bijective mapping, $f : N^4 \rightarrow M$ that maps the system in

practice to the hybrid automaton framework. It is clear that transitions are effected in the same way in both forms with a one to one correspondence between them. Therefore, we will use the hybrid automaton framework as the basis for the analysis of hybrid systems because it has a cleaner and simpler structure.

1.2 Simulation

The importance of the applications of simulation has motivated the development of many software packages (see [102] for a recent review of the packages). The great majority of them have been built from an understanding of the simulation of purely continuous systems. In fact, continuous systems can be seen as a special case of hybrid systems where there are no discrete transitions. Numerical methods for solving systems of ODEs and DAEs for a purely continuous system (see [11] for an overview) are well established and a number of robust codes are widely available in the public domain, e.g., RKSUITE [31], ODEPACK [74], VODE [33], DASSL [108]. Today, any simulator worth its salt must be able to pass the test of simulating a purely continuous system robustly (and with flying colors) before even attempting hybrid systems. In general purpose simulators, implicit linear multi-step methods, particularly Gear's BDF method [65], and implicit Runge-Kutta methods based on collocation, e.g., Radau methods [11], are favored because they can robustly solve a very broad range of problems that the user may pose.

The incorporation of discrete dynamics into continuous system simulation started with [39]. Subsequently, in the early 1990's, there emerged a growing interest in the modeling and simulation of large-scale hybrid systems [15, 52, 7] as the limitations of continuous system modeling methodology became more apparent. It is now widely accepted that all but the most trivial engineering models of dynamic systems contain discontinuities.

We will concentrate the remaining discussion in this section on the numerical treatment of transitions as these are the additional complications introduced by moving from continuous system simulation to hybrid system simulation. Events can be

either *time events* or *state events*. Note that this classification of events is an alternative form of classification, as opposed to controlled or autonomous transitions. Time events occur at a specified future time that is known when the event is scheduled, and present few problems for simulation. Thus, time events can either be controlled or autonomous, provided that the time of transition is known beforehand. The numerical integration procedure is simply asked to step exactly to the time event. On the other hand, state events are the mechanism whereby the state of the continuous subsystem influences the discrete subsystem. A state event occurs (and the discrete state potentially changes) when some condition on the continuous state is satisfied (e.g., a negative pressure drop across a non-return valve forces the valve to close). Thus the timing of state events is a function of the solution of the differential equations, transition conditions and transition functions governing the current and previous modes visited along T_μ. Indeed, in a particular mode, a number of different events may be pending, each implying a switch to a different mode (e.g., transition to the choked flow or no flow regime while in the laminar/turbulent flow regime).

The relational atoms that make up a currently pending transition condition are rearranged to form discontinuity functions according to (1.6) as described in Section 1.1.4. As stated above, the next (and correct) event is defined as the earliest time at which one of the currently pending transition conditions becomes true. During the course of a simulation, the actual mode switching that occurs depends on which transition condition is satisfied first, which in turn depends on the parameters and/or initial conditions. Once the system is in one of these new modes, it may evolve in a radically different way from that if it had switched to another of the pending modes. Thus, both T_τ and T_μ can be extremely sensitive to parameters (as mentioned before, we can treat the initial conditions as parameters by adding auxiliary parameters to represent these conditions).

In their seminal paper, Hay and Griffin [72] present a thoughtful treatment of the subject, in which they propose the use of discontinuity functions to track and detect discontinuities (leading to the method of "discontinuity locking") and recognize the role and effects of error in the numerical integrator among other issues. Many methods

have been proposed to deal with state event location and detection (see [107] for a detailed discussion). A description of how hybrid simulators work in general follows:

1. The model is compiled and validated. With modern DAE solvers such as DASSL [108] there is no need to determine computational causality, which was necessary in earlier simulators, particularly those based on the CSSL (Continuous System Simulation Language) standard [123].

2. The current mode is set to the initial mode.

3. A number of structural[1] diagnoses may be applied to the DAEs in the current mode. First, the DAEs may be checked for structural consistency [106]. If this check fails, the corrector iteration of any implicit integration method will be singular, so it is impossible to proceed. DAEs are characterized by various indices, see [32] and [36] (all indices are equivalent in the linear time invariant (LTI) case). The differentiation index is the commonly used index for general, nonlinear DAEs. Explicit and implicit ODEs are differentiation index 0 DAEs. Standard integration codes can usually handle differentiation index ≤ 1 DAEs. However, if the differentiation index ≥ 2, specialized codes only applicable to special equation structures are necessary. On the other hand, differentiation index ≥ 2 systems can be reduced to differentiation index ≤ 1 systems via a process of repeatedly differentiating subsets of the DAEs. The resulting index ≤ 1 system may then be solved using standard codes. For example, a structural algorithm for this purpose is described by [95]. Alternatively, a structural check for differentiation index ≤ 1 may be made, and if this check fails, the user is asked to reformulate his or her model (e.g., Pantelides' [106] algorithm can be applied; if it performs zero iterations, the structural check is passed). It should be noted that none of these structural analyses bear any connection to the actual differentiation index [111], even in the LTI case. However, it has been our experience in the PSEL that these structural checks almost al-

[1]Here, structural analysis is used to refer to algorithms that operate on the incidence matrices of the relevant equations and variables.

ways catch any problems with the formulation of physical models, and provide adequate information for a numerically well-behaved reformulation. Moreover, these structural checks are practical to implement and apply in large-scale modeling environments.

4. A consistent initialization calculation [106], usually obtained by solving a system of nonlinear algebraic equations (NLEs), is performed to determine a set of consistent initial values for the continuous state variables. The number of additional equations required (in addition to the DAEs and possibly their first and higher order time derivatives) is determined by the dynamic degrees of freedom, $r^{(m_i)}$, of the DAE ($= n_x^{(m_i)}$ for an ODE). For many (but not all) index 1 DAEs, first and higher order time derivatives of the DAEs do not constrain the initial values of the continuous state variables. These additional equations either come from the initial condition (in the initial mode) or from the relevant transition function (in all subsequent modes). A structural analysis [49] may be used to permute this NLE to block lower triangular (BLT) form, hence presenting a sequence of smaller subproblems, which yields a more robust solution procedure.

5. The continuous state variables are integrated forward from the consistent initial values according to the DAEs that govern the current mode until the earliest event, either time or state, occurs. For sparse large-scale systems, the corrector matrix employed in an implicit integration method may also permute to a BLT form. This can be exploited at the level of the DAE, the NLEs comprising the corrector iteration, or the linear solve at each corrector iteration. In DSL48S/DSL48E/DSL48SE [57, 129] this is exploited at the level of the linear solve via use of MA48 [48]. This approach yields large computational savings without the error control complications inherent to applying it at the DAE level.

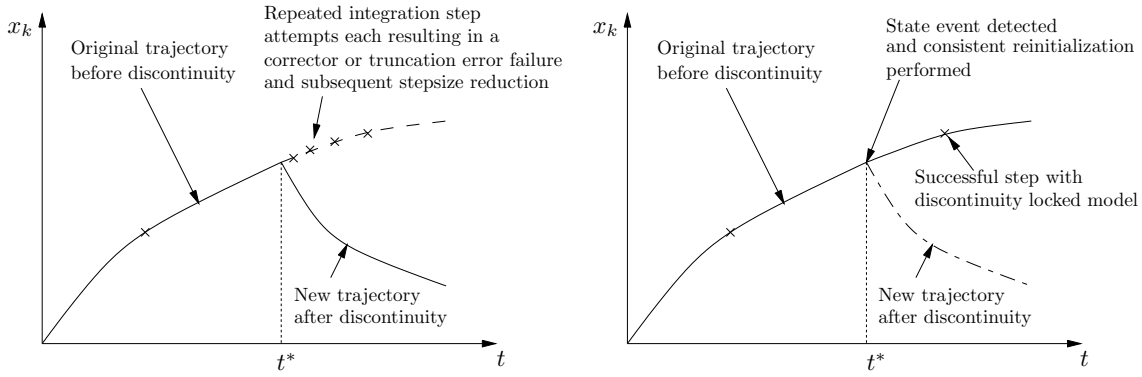
6. State events must be dealt with carefully since their timing is not known a priori. At an event, either the simulation has reached its termination condition, or a transition to the successor mode occurs. In the latter case, update the

current mode and go to Step 3.

1.2.1 State Event Location

State events pose particular problems for simulation. Time evolution in a mode is approximated by numerical solution of an IVP in the relevant differential equations. This numerical procedure in turn implies some form of time stepping, and there is no reason that the time steps chosen by (for example) a variable step size variable order method will coincide with the points in time at which the state conditions first become satisfied. On the other hand, due to the sensitivity mentioned above, it is extremely important to locate the state events in strict time order and implement the correct mode changes (i.e., events must not be missed by stepping over them completely). Similarly, just flipping the equations when they are evaluated at a point at which a state condition is satisfied (e.g., an `IF` statement in a residual evaluator code), and thus presenting the IVP solver with a discontinuous vector field, can cause severe inefficiency, and even simulation failures or incorrect sequences to be generated [39, 72], because this nonsmoothness violates the theoretical assumptions on which IVP solvers are founded.

These difficulties can be overcome via the notion of *discontinuity locking*. The idea is to 'lock' the function evaluator for the IVP solver so that the functional form of the equations evaluated cannot change while a time step is being taken, thus presenting a smooth vector field. Once a successful time step has been taken, it is then necessary to determine if event(s) have occurred during the time step just taken, and if so, to backtrack to the earliest event in order to implement the requisite transition. This is illustrated in Figure 1-9. Most modern algorithms do this by searching for zero crossings in the discontinuity functions. Note that according to our proposed modeling framework in the previous section, algorithms should search, instead, for *roots* of the discontinuity function, or points at which the discontinuity function touches zero. Thus, it is extremely important that the discontinuity functions be known with high accuracy over the entire step. One way to guarantee this is to introduce a *discontinuity*



(a) Without Discontinuity Locking

(b) With Discontinuity Locking

Figure 1-9: Discontinuity locking, $\times$ denotes a time mesh point.

variable, $z_{j,l}^{(m_i)}$, for each discontinuity function, and append the algebraic equations:

$$z_{j,l}^{(m_i)} = g_{j,l}^{(m_i)}\left(\dot{\mathbf{x}}^{(m_i)}, \mathbf{x}^{(m_i)}, \mathbf{p}, t\right), \quad l = 1, \ldots, n_j^{(m_i)}, j \in J^{(m_i)}, \tag{1.12}$$

to the differential equations describing the current mode. The fact that these equations are explicit in $z_{j,l}^{(m_i)}$ can be exploited by a modern DAE code, so that, although there may be many discontinuity functions (indeed $> n_x^{(m_i)}$), the computational cost per step hardly increases [107]. On the other hand, the need to control the integration error in the discontinuity functions in addition to the states may increase the number of steps taken. However, this is the unavoidable price of locating the zero crossings accurately and thus getting the correct sequence of events.

The more reliable algorithms for locating zero crossings in the discontinuity functions search for roots of the interpolation polynomials for the discontinuity variables extracted from the IVP solver. Again, there are a number of ways this search can be performed, with different degrees of reliability. In [107], a one dimensional interval-Newton method is applied to the interpolating polynomials that *guarantees* all roots will be found in strict time order. However, if applied naïvely, this approach can be extremely expensive because an interval-Newton search has to be applied for each discontinuity function at each step. To mitigate this cost, a root exclusion test (also based on interval arithmetic) is employed before applying the interval-Newton

method. This provides an extremely cheap test of nonexistence of a root in the current step. Since most discontinuity functions in most steps do not touch or cross the zero time axis, this is an effective way of avoiding the expensive interval-Newton search unless it is really needed.

All of the above discussion assumes that it is possible to extend the solution of the embedded differential system beyond the event time. Classical theory has shown that sometimes, solutions cannot be extended uniquely beyond a finite limit in time (e.g., a lack of Lipschitz continuity for ODEs, or impasse points for DAEs [110]). Sometimes, events are employed to switch the vector field at these limits and thus continue simulation. However, the solution does not extend uniquely past the limit, so the aforementioned state event location algorithms do not apply. A practical solution is to move the event slightly to the left of the limit, but even this can have a dramatic effect on the error control and step size as the integrator attempts to locate a point beyond the event. An opportunity exists to develop state event location algorithms that can better deal with these "limit events."

1.2.2 Consistent Reinitialization

Another, still somewhat controversial issue, is the *consistent reinitialization* of the continuous state at mode switching events [34, 91, 68, 16, 112]. In the absence of an explicit specification of transition functions (1.2) by the user, we desire to obtain consistent initial values for the successor mode, $(\mathbf{x}^{(m_{i+1})}(\mathbf{p}, \sigma_{i+1}),\ \dot{\mathbf{x}}^{(m_{i+1})}(\mathbf{p}, \sigma_{i+1}))$, in order to restart numerical integration immediately following the event. In other words, are there "natural" transition functions that must hold unless purposely overridden by a user specification? The simplest case for this is that of an ODE embedded system described by the same variables (but different vector fields) in each mode. In this situation, a jump in the continuous state has an unambiguous interpretation as an impulsive forcing of one or more of the state variables, and can therefore be considered as a separate and distinct hybrid phenomenon to switching. Hence, it is not enough to pick arbitrary initial conditions for the successor mode; the said initial conditions must be defined in terms of the final state of the predecessor mode,

$\mathbf{x}^{(m_i)}(\mathbf{p}, \tau_i)$. In this case, in the absence of impulsive forcing, the "natural" transition function to use would be *state continuity*, which is to assume that the state variables remain unchanged at the mode switching, i.e.,

$$\mathbf{x}^{(m_i)}(\mathbf{p}, \sigma_{i+1}) - \mathbf{x}^{(m_i)}(\mathbf{p}, \tau_i) = \mathbf{0}.$$

The number of transition functions needed (in addition to the embedded DAE and possibly its first and higher order derivatives) is determined by $r^{(m_{i+1})}$, the dynamic degrees of freedom of the DAE in the successor mode. For example, impulsive forcing of an ODE is modeled with the transition function:

$$\mathbf{x}^{(m_{i+1})}(\mathbf{p}, \sigma_{i+1}) - \mathbf{x}^{(m_i)}(\mathbf{p}, \tau_i) - \Delta\mathbf{x} = \mathbf{0},$$

where $\Delta\mathbf{x} \neq \mathbf{0}$ is the desired increment of the state variables and $m_{i+1} = m_i$ because the mode remains unchanged (although the epoch increments from I_i to I_{i+1}). As mentioned above, in the absence of the explicit specification of impulsive forcing, it is natural to assume state continuity at a mode switching. In general, it appears that current software for modeling hybrid systems provides weak support for the specification of transition functions, with most software implicitly assuming some natural transition functions such as state continuity. The notion of state continuity in a subset of the continuous state variables can be extended to certain index 1 DAEs [91], although all other state variables may jump at a mode switching.

However, for more general DAE systems, this requires the modeler to recognize the potential exceptions that might arise and intervene with the explicit specification of transition functions when the assumption of state continuity in the absence of impulsive forcing is wrong. For example, consider a DAE embedded hybrid system. Even a LTI index 1 DAE subject to step changes in the forcing functions (which can be interpreted as the simplest form of mode switch) may exhibit jumps in *all* the continuous state variables, as shown by the following LTI DAE in a predecessor

mode:

$$\begin{bmatrix} 1 & 2 \\ 0 & 0 \end{bmatrix} \dot{\mathbf{x}} + \begin{bmatrix} 0 & 0 \\ 1 & 1 \end{bmatrix} \mathbf{x} = \begin{bmatrix} 0 \\ f(t) \end{bmatrix}, \tag{1.13}$$

which is index 1 and has $r^{(m)} = 1$. How should the state be transferred if a step change in $f(t)$ is implemented at an event? Assuming state continuity for x_1 implies a jump in x_2, and vice versa. In fact, neither continuity assumption is correct. An analysis of the canonical form of this DAE [16] shows that the linear combination of the state variables $x_1 + 2x_2$ should be treated as continuous, so that a step change in $f(t)$ will cause both states to jump. Thus it is incorrect to associate jumps in the state exclusively with impulsive forcing if the index is 1 or greater. In general, it is possible to obtain the $r^{(m_{i+1})}$ additional natural transition functions needed for LTI DAEs of arbitrary index and dimension by appealing to the canonical form. However, as stated in [16], this is hampered in practice for large-scale systems by the inherent density of the matrices required to transform a matrix pencil to generalized upper triangular form.

A method has recently been developed to compute consistent initial values of the continuous state variables in the successor mode for hybrid systems described by a collection of (uniquely solvable) LTI DAEs,

$$\mathbf{A}^{(m)}\dot{\mathbf{x}} + \mathbf{B}^{(m)}\mathbf{x} = \bar{\mathbf{f}}^{(m)}(\mathbf{p}, t), \quad m \in M,$$

in which the number of variables, n_x, does not change between modes. This is done by solving the linear system:

$$\mathbf{T}_{2\nu^{(m_{i+1})}}(\mathbf{A}^{(m_{i+1})}, \mathbf{B}^{(m_{i+1})})\mathbf{y} = \mathbf{z}, \tag{1.14}$$

characterized by $\mathbf{A}^{(m_{i+1})}, \mathbf{B}^{(m_{i+1})}$, derivatives of $\bar{\mathbf{f}}^{(m_{i+1})}$ at transition time σ_{i+1}, $\mathbf{x}(\mathbf{p}, \tau_i)$ and $\nu^{(m_{i+1})}$, which is the index of the DAE in the successor, provided that $\nu^{(m)}$ can be calculated explicitly for all required modes $m \in M$ [112][Theorem II.2]. Here,

$\mathbf{T}_\alpha(\mathbf{A},\mathbf{B})$ is the $\alpha \cdot n_x \times \alpha \cdot n_x$ matrix defined by

$$\mathbf{T}_\alpha(\mathbf{A},\mathbf{B}) = \begin{pmatrix} \mathbf{A} & & & \\ \mathbf{B} & \mathbf{A} & & \\ & \ddots & \ddots & \\ & & \mathbf{B} & \mathbf{A} \end{pmatrix}.$$

When $\mathbf{y} = [\mathbf{y}_1, ..., \mathbf{y}_{2\nu^{(m_{i+1})}}]^{\mathrm{T}}$, $\mathbf{y}_\beta \in \mathbb{R}^{n_x}$ $\forall \beta \in \{1, ..., 2\nu^{(m_{i+1})}\}$, and $\mathbf{z} = [\mathbf{0}, ..., \mathbf{0}, \mathbf{A}^{(m_i+1)}\mathbf{x}(\mathbf{p}, \tau_i), \bar{\mathbf{f}}^{(m_{i+1})}(\mathbf{p}, \sigma_{i+1}), \bar{\mathbf{f}}'^{(m_{i+1})}(\mathbf{p}, \sigma_{i+1}), \ldots, \bar{\mathbf{f}}^{(\nu^{(m_{i+1})}-1)(m_{i+1})}(\mathbf{p}, \sigma_{i+1})]^{\mathrm{T}}$, then $\mathbf{y}_{\nu^{(m_{i+1})}} = \mathbf{x}(\mathbf{p}, \sigma_{i+1})$. As pointed out by [112], this problem of obtaining a family of mappings,

$$\mathbf{x}(\mathbf{p}, \tau_i) \to \mathbf{x}(\mathbf{p}, \sigma_{i+1}),$$

has been the subject of much research. The importance of the work by Reißig et al. [112] is that they provide a justification of this choice to satisfy the above equation which is elementary and physically reasonable compared to previous justifications, and provide a practical way of implementing the solution, (1.14), using sparse LU factorization.

For example, consider again the problem described by (1.13). Let $M = \{1, 2\}$, $\mathbf{A}^{(1)} = \mathbf{A}^{(2)} = \mathbf{A}$, $\mathbf{B}^{(1)} = \mathbf{B}^{(2)} = \mathbf{B}$, $f^{(1)}(t) = t$ and $f^{(2)}(t) = t + 1$. Suppose we start with $m_1 = 1$, and we have a time event, τ_1, at which we switch from mode 1 to mode 2. Applying Theorem II.2 from [112], we have

$$\mathbf{T}_2(\mathbf{A},\mathbf{B})\mathbf{y} = \begin{bmatrix} 1 & 2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 1 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \end{bmatrix} = \begin{bmatrix} x_1(\tau_1) + 2x_2(\tau_1) \\ 0 \\ 0 \\ \sigma_2 + 1 \end{bmatrix} = \mathbf{z},$$

from which it is clear that the consistent initial values are given by:

$$\begin{bmatrix} 1 & 2 \end{bmatrix} \mathbf{y}_1 = x_1(\sigma_2) + 2x_2(\sigma_2) = x_1(\tau_1) + 2x_2(\tau_1), \tag{1.15}$$

$$\begin{bmatrix} 1 & 1 \end{bmatrix} \mathbf{y}_1 = x_1(\sigma_2) + x_2(\sigma_2) = \sigma_2 + 1. \tag{1.16}$$

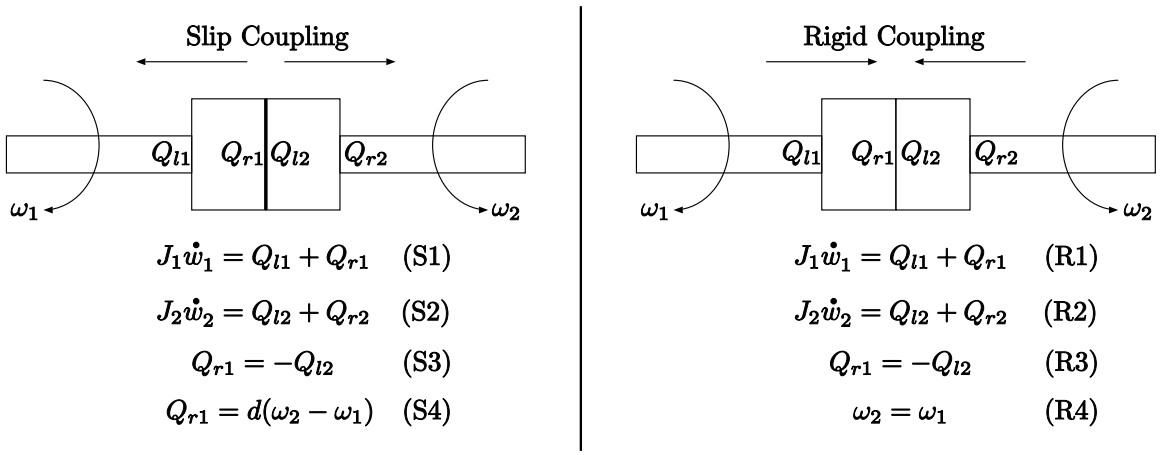


Figure 1-10: Schematic of two rotating masses.

Note that (1.15) is exactly the natural transition function as described above, and (1.16) would have been satisfied by solving a consistent initialization problem with (1.15) to satisfy the dynamic degrees of freedom.

A nice physical example to illustrate the utility of (1.14) is a simple model of two rotating masses that may be switched between a slip coupling (mode 1), described by equations (S1 - S4), and a rigid coupling (mode 2) described by equations (R1 - R4) [94] (see Figure 1-10). ω_1 and ω_2 represent the angular velocities of the two bodies, J_1, J_2 and d are parameters, and the torques Q_{l1} and Q_{r2} are known functions of time. When connected by a slip coupling the model is index 1 with $r^{(1)} = 2$, and when connected with a rigid coupling the model is index 2 with $r^{(2)} = 1$. From physical considerations, one can argue that the transition functions needed for the switch 'rigid' to 'slip' should be continuity of the two angular velocities, whereas for the switch 'slip' to 'rigid' the transition function needed should be conservation of angular momentum. Applying Theorem II.2 in [112], we will obtain consistent initial values for both switching transitions, as shown below.

Consider $T_\mu = 1, 2, 1$ with two separate time events $\tau_1 < \tau_2$ respectively. At τ_1,

the successor is an index 2 DAE ($\nu^{(2)} = 2$, assuming that $J_1 + J_2 \neq 0$ [94]) with

$$\mathbf{A}^{(2)} = \begin{bmatrix} J_1 & 0 & 0 & 0 \\ 0 & J_2 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{bmatrix}, \ \mathbf{B}^{(2)} = \begin{bmatrix} 0 & 0 & 0 & -1 \\ 0 & 0 & -1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & -1 & 0 & 0 \end{bmatrix}, \ \bar{\mathbf{f}}^{(2)} = \begin{bmatrix} Q_{l1}(t) \\ Q_{r2}(t) \\ 0 \\ 0 \end{bmatrix},$$

where $\mathbf{x} = [\omega_1(t) \ \ \omega_2(t) \ \ Q_{l2}(t) \ \ Q_{r1}(t)]^{\mathrm{T}}$. Constructing the relevant form of Equation (1.14), we get

$$\mathbf{T}_4(\mathbf{A}^{(2)}, \mathbf{B}^{(2)})\mathbf{y} = \begin{bmatrix} \mathbf{A}^{(2)} & & & \\ \mathbf{B}^{(2)} & \mathbf{A}^{(2)} & & \\ & \mathbf{B}^{(2)} & \mathbf{A}^{(2)} & \\ & & \mathbf{B}^{(2)} & \mathbf{A}^{(2)} \end{bmatrix} \begin{bmatrix} \mathbf{y}_1 \\ \mathbf{y}_2 \\ \mathbf{y}_3 \\ \mathbf{y}_4 \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{A}^{(2)}\mathbf{x}(\tau_1) \\ \bar{\mathbf{f}}^{(2)}(\sigma_2) \\ \bar{\mathbf{f}}'^{(2)}(\sigma_2) \end{bmatrix} = \mathbf{z}.$$

Denoting $\mathbf{y}_1 = [y_{11} \ y_{12} \ y_{13} \ y_{14}]^{\mathrm{T}}$ and keeping in mind that $\mathbf{y}_2 = \mathbf{x}(\sigma_2)$, we obtain, after some algebra,

$$\begin{aligned} y_{13} + y_{14} &= 0, \\ -y_{13} + J_2\omega_2(\sigma_2) &= J_2\omega_2(\tau_1), \\ -y_{14} + J_1\omega_1(\sigma_2) &= J_1\omega_1(\tau_1), \\ \omega_1(\sigma_2) - \omega_2(\sigma_2) &= 0, \end{aligned}$$

from which it is easy to see that the solution of these equations will satisfy

$$J_1\omega_1(\tau_1) + J_2\omega_2(\tau_1) = (J_1 + J_2)\omega_1(\sigma_2),$$

which is exactly the mathematical statement of conservation of angular momentum. Note that since we have a high index DAE in this mode, we may have to reformulate the problem before the simulation can proceed (e.g., [95]). Similarly, applying the same analysis to the second transition at τ_2, where we switch back to the slip coupling

mode, we can show that solving (1.14) will give us the same consistent initial values as the natural transition functions, i.e., continuity of ω_1 and ω_2.

It appears that such natural transition functions also exist for many linear time varying (LTV) cases and even nonlinear DAEs, but derivation of the relevant continuity conditions appears very difficult except in very simple cases [34, 16]. On the other hand, if the natural transition functions of a system of equations do not conform to physical expectations, this suggests that the model should be reformulated.

As another example, consider a variable structure hybrid system described by a collection of agents, each described by the same ODE. Deletion of an agent presents no problems, but insertion of a new agent implies the need to specify explicitly the initial conditions for the new agent (possibly in terms of the current state of the other currently active agents). In this situation, it appears impossible to use a mathematical analysis to reveal natural transition functions. Instead, the transition functions must be explicitly stated as part of the model formulation. An interesting area for further research is thus the development of rigorous methods to obtain and express the transition functions for LTV, nonlinear and variable structure hybrid systems.

Before we end this section, we will briefly discuss the phenomena of *discontinuity sticking* (for a more detailed discussion, see [107]). The consistent initialization calculation for the successor mode is carried out based on the values of the continuous state variables, $\mathbf{x}^{m_i}(\mathbf{p}, \tau_i)$. If the event time does not occur at the mesh points of the DAE solver (note that this does not apply for instantaneous transitions), and the values of $\mathbf{x}^{m_i}(\mathbf{p}, \tau_i)$ have been located by interpolation, the converged initialization calculation may indicate that the state event detected has actually not quite been triggered. This situation occurs because the BDF method [65] provides no guarantees for the consistency of differential and algebraic variables between mesh points. This numerical phenomenon is termed discontinuity sticking.

The consistent event location phase in [107] determines the consistent state event time t_l^* at which consistency between the differential and algebraic variables is retained, and consequently eliminates discontinuity sticking problems. The consistent event location problem is formulated as a system of nonlinear equations, within which

the discontinuity function that triggers the transition is set to $-\varepsilon_g$, where ε_g is a small positive tolerance. This mitigates the effect of discontinuity sticking. Note that this also prevents Zeno behavior for reversible discontinuities in practice.

1.3 Sensitivity Analysis

Parametric sensitivity analysis is concerned with the sensitivity of the model prediction to infinitesimal perturbations in parameters appearing in the model and/or initial conditions, and is important in many engineering and scientific applications. The information contained in the parametric sensitivity trajectories is useful for model reduction, control system design, parameter estimation, process sensitivity studies, experimental design and numerical optimal control. For example, the control parameterization approach for the numerical solution of optimal control problems [126] can require sensitivity information with respect to hundreds of parameters. The theory for systems with continuous dynamics is well established [61], while a closely related perturbation analysis theory has been developed for discrete event dynamic systems [76].

The traditional method to compute the sensitivity trajectories of stiff ODEs or DAEs has been to handle the combined ODE/DAE and sensitivity system using a staggered direct scheme in which the linear systems for the sensitivity corrector steps are solved directly after convergence of the nonlinear corrector step [37]. Maly and Petzold [93] proposed a simultaneous corrector method that substantially reduced the cost of parametric sensitivity analysis relative to earlier efforts, following which Feehery et al. [57] developed and demonstrated a staggered corrector method (DSL48S) for solving stiff ODES (or DAEs) and sensitivities that was shown to have a number of advantages over that of the simultaneous corrector algorithm described in [93].

In this section we consider an extension to this classical sensitivity theory that defines the parametric sensitivity trajectories of hybrid systems represented by the hybrid automaton framework described in Section 1.1. Rozenvasser [114] first presented the general sensitivity equations, with respect to a parameter, for discontin-

uous systems of ODEs. Galán et al. [63] further extended these results to include DAEs, generalized the discrete aspects of the system model, and presented, for the first time, existence and uniqueness theorems for the sensitivity functions of hybrid systems. Hiskens and Pai [75] present what is in essence an extension of Rozenvasser's approach to a class of hybrid models.

Again, we will be focusing on the effect of transitions on the sensitivity analysis of hybrid systems. This is closely tied to detecting and locating state events correctly. It is crucial that the correct state event is located, or the related sensitivity trajectories will be complete nonsense. For example, as will be seen later, the parametric sensitivities will often jump at transitions, and if the numerical integration is not stopped and the jump computed explicitly, the computed sensitivity trajectories will generally be incorrect. This point is illustrated in detail by [129], in which the detection of hidden discontinuities and parametric sensitivities is handled rigorously and robustly by the software library DAEPACK with minimal user intervention.

1.3.1 Calculation of Sensitivity Trajectories

Let the set of continuous state variables be partitioned into:

$$\mathbf{x}^{(m)} = \begin{bmatrix} \mathbf{v}^{(m)} \\ \mathbf{y}^{(m)} \end{bmatrix},$$

where $\mathbf{v}^{(m)}$ are the differential state variables and $\mathbf{y}^{(m)}$ are the algebraic state variables. The DAE of the current mode, $\mathbf{f}^{(m_i)}$, is augmented with the discontinuity functions associated with the mode's pending transition conditions, $L_j^{(m_i)}, j \in J^{(m_i)}$, given by

$$\mathbf{F}^{(m_i)}\left(\dot{\mathbf{v}}^{(m_i)}, \mathbf{v}^{(m_i)}, \mathbf{y}^{(m_i)}, \mathbf{z}^{(m_i)}, \mathbf{p}, t\right) = \begin{pmatrix} \mathbf{f}^{(m_i)} \\ \mathbf{z}^{(m_i)} - \mathbf{g}^{(m_i)} \end{pmatrix},$$

where $\mathbf{g}^{(m_i)}$ denotes the vector for the discontinuity functions of all transition conditions in the current mode and $\mathbf{z}^{(m_i)}$ is the vector of *discontinuity variables*. The dependencies of the right hand side are omitted for readability and can be inferred

from (1.1), (1.6) and (1.12). Augmenting the original DAE with these additional explicit equations (referred to as *discontinuity equations*) places the discontinuity functions under integration error control. The additional variables added for the discontinuity equations, $\mathbf{z}^{(m_i)}$, are algebraic variables and may be appended to the original algebraic variable vector:

$$\mathbf{w}^{(m_i)} \equiv \begin{pmatrix} \mathbf{y}^{(m_i)} \\ \mathbf{z}^{(m_i)} \end{pmatrix}.$$

For the sequel, we will only consider the case where

$$\operatorname{rank}\begin{pmatrix} \dfrac{\partial \mathbf{F}^{(m_i)}}{\partial \dot{\mathbf{v}}^{(m_i)}} & \dfrac{\partial \mathbf{F}^{(m_i)}}{\partial \mathbf{w}^{(m_i)}} \end{pmatrix} = n_v^{(m_i)} + n_w^{(m_i)},$$

for all $m_i \in M$ and t, where $n_v^{(m_i)}$ and $n_w^{(m_i)}$ are the number of elements of $\mathbf{v}^{(m_i)}$ and $\mathbf{w}^{(m_i)}$ respectively. This is sufficient for a differentiation index less than or equal to 1. Although the size of the model increases, very little additional computational effort is required to integrate the system because it block decomposes.

The augmented DAE and sensitivity equations form an $(n_v^{(m_i)} + n_w^{(m_i)})(n_p + 1)$ system given by

$$\begin{aligned} \mathbf{F}^{(m_i)}\left(\dot{\mathbf{v}}^{(m_i)}, \mathbf{v}^{(m_i)}, \mathbf{w}^{(m_i)}, \mathbf{p}, t\right) &= \mathbf{0}, \\ \frac{\partial \mathbf{F}^{(m_i)}}{\partial \dot{\mathbf{v}}^{(m_i)}} \dot{\mathbf{S}}_{\mathbf{v}}^{(m_i)} + \frac{\partial \mathbf{F}^{(m_i)}}{\partial \mathbf{v}^{(m_i)}} \mathbf{S}_{\mathbf{v}}^{(m_i)} + \frac{\partial \mathbf{F}^{(m_i)}}{\partial \mathbf{w}^{(m_i)}} \mathbf{S}_{\mathbf{w}}^{(m_i)} &= -\frac{\partial \mathbf{F}^{(m_i)}}{\partial \mathbf{p}}, \end{aligned} \tag{1.17}$$

where $\mathbf{S}_{\mathbf{v}}^{(m_i)} \equiv \frac{\partial \mathbf{v}^{(m_i)}}{\partial \mathbf{p}}$, $\mathbf{S}_{\mathbf{w}}^{(m_i)} \equiv \frac{\partial \mathbf{w}^{(m_i)}}{\partial \mathbf{p}}$ and $\dot{\mathbf{S}}_{\mathbf{v}}^{(m_i)} = \frac{\partial \mathbf{S}_{\mathbf{v}}^{(m_i)}}{\partial t} = \frac{\partial \dot{\mathbf{v}}^{(m_i)}}{\partial \mathbf{p}}$. Although this system may be quite large, the algorithms described in [93] and [57] can be used to exploit the special structure for efficient solution. In between mode switching events, the sensitivity trajectories $\frac{\partial \mathbf{v}}{\partial \mathbf{p}}$ and $\frac{\partial \mathbf{w}}{\partial \mathbf{p}}$ are given by (1.17), which are derived via differentiation of the DAE with respect to the parameters $\mathbf{p}$. Initial conditions for the sensitivities in the initial mode are determined via differentiation of the initial conditions with respect to the parameters. The sensitivities will then evolve according to (1.17) until the first event. At this event, we have a system of transition functions in

the form of (1.2). Consider the general case where we have a transition from predecessor m_i in epoch I_i to successor m_{i+1} in epoch I_{i+1}. We will assume smoothness in the neighborhood of the transition time, and that only one atomic logical proposition becomes true at that moment. Let $\tilde{\jmath} \in J^{(m_i)}$ be the transition that is taken at the event. Differentiation of the transition functions with respect to the parameters and some rearrangement yields:

$$\begin{bmatrix} \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \dot{\mathbf{v}}^{(m_{i+1})}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{v}^{(m_{i+1})}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{w}^{(m_{i+1})}} \\ \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \dot{\mathbf{v}}^{(m_{i+1})}} & \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \mathbf{v}^{(m_{i+1})}} & \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \mathbf{w}^{(m_{i+1})}} \end{bmatrix} \begin{bmatrix} \frac{\partial \dot{\mathbf{v}}^{(m_{i+1})}}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{v}^{(m_{i+1})}}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{w}^{(m_{i+1})}}{\partial \mathbf{p}} \end{bmatrix} =$$

$$- \begin{bmatrix} \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \dot{\mathbf{v}}^{(m_{i+1})}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{v}^{(m_{i+1})}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{w}^{(m_{i+1})}} \\ \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \dot{\mathbf{v}}^{(m_{i+1})}} & \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \mathbf{v}^{(m_{i+1})}} & \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \mathbf{w}^{(m_{i+1})}} \end{bmatrix} \begin{bmatrix} \frac{\partial \dot{\mathbf{v}}^{(m_{i+1})}}{\partial \sigma_{i+1}} \\ \frac{\partial \mathbf{v}^{(m_{i+1})}}{\partial \sigma_{i+1}} \\ \frac{\partial \mathbf{w}^{(m_{i+1})}}{\partial \sigma_{i+1}} \end{bmatrix} \frac{\partial \sigma_{i+1}}{\partial \mathbf{p}}$$

$$- \begin{bmatrix} \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \dot{\mathbf{v}}^{(m_i)}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{v}^{(m_i)}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{w}^{(m_i)}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \mathbf{p}} & \frac{\partial \mathbf{T}_{\tilde{\jmath}}^{(m_i)}}{\partial \sigma_{i+1}} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \mathbf{p}} & \frac{\partial \mathbf{F}^{(m_{i+1})}}{\partial \sigma_{i+1}} \end{bmatrix} \begin{bmatrix} \frac{\partial \dot{\mathbf{v}}^{(m_i)}}{\partial \mathbf{p}} + \frac{\partial \dot{\mathbf{v}}^{(m_i)}}{\partial \tau_i} \frac{\partial \tau_i}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{v}^{(m_i)}}{\partial \mathbf{p}} + \frac{\partial \mathbf{v}^{(m_i)}}{\partial \tau_i} \frac{\partial \tau_i}{\partial \mathbf{p}} \\ \frac{\partial \mathbf{w}^{(m_i)}}{\partial \mathbf{p}} + \frac{\partial \mathbf{w}^{(m_i)}}{\partial \tau_i} \frac{\partial \tau_i}{\partial \mathbf{p}} \\ \mathbf{I} \\ \frac{\partial \sigma_{i+1}}{\partial \mathbf{p}} \end{bmatrix} \tag{1.18}$$

where $\frac{\partial \sigma_{i+1}}{\partial \mathbf{p}} = \frac{\partial \tau_i}{\partial \mathbf{p}}$ (since $\tau_i = \sigma_{i+1}$) represents the sensitivity of the event time with respect to the parameters. The jump in sensitivities can then be computed by solving the above linear equation, which provides transition functions for the sensitivities that are implied by the transition functions for the states. However, in order to compute the initial conditions for the sensitivities in the new mode, it is necessary to know $\frac{\partial \tau_i}{\partial \mathbf{p}}$, which can be obtained by differentiating the discontinuity function defining the event time, (1.6), with respect to the parameters $\mathbf{p}$:

$$\frac{\partial g_{\tilde{\jmath},\tilde{l}}^{(m_i)}}{\partial \dot{\mathbf{v}}^{(m_i)}} \left(\dot{\mathbf{S}}_{\mathbf{v}}^{(m_i)} + \ddot{\mathbf{v}}^{(m_i)} \frac{\partial \tau_i}{\partial \mathbf{p}} \right) + \frac{\partial g_{\tilde{\jmath},\tilde{l}}^{(m_i)}}{\partial \mathbf{v}^{(m_i)}} \left(\mathbf{S}_{\mathbf{v}}^{(m_i)} + \dot{\mathbf{v}}^{(m_i)} \frac{\partial \tau_i}{\partial \mathbf{p}} \right) +$$
$$\frac{\partial g_{\tilde{\jmath},\tilde{l}}^{(m_i)}}{\partial \mathbf{w}^{(m_i)}} \left(\mathbf{S}_{\mathbf{w}}^{(m_i)} + \dot{\mathbf{w}}^{(m_i)} \frac{\partial \tau_i}{\partial \mathbf{p}} \right) + \frac{\partial g_{\tilde{\jmath},\tilde{l}}^{(m_i)}}{\partial \mathbf{p}} + \frac{\partial g_{\tilde{\jmath},\tilde{l}}^{(m_i)}}{\partial \tau_i} \frac{\partial \tau_i}{\partial \mathbf{p}} = \mathbf{0}, \tag{1.19}$$

where $\tilde{\jmath} \in J^{(m_i)}$, $\tilde{l} \in \{1, \ldots, n_{\tilde{\jmath}}^{(m_i)}\}$, and $g_{\tilde{\jmath},\tilde{l}}^{(m_i)}$ is the discontinuity function that triggers the event. Provided these linear equations can be solved for a unique $\frac{\partial \tau_i}{\partial \mathbf{p}}$, initial conditions for the sensitivities in the successor mode can be determined via (1.18) (the required elements of $\dot{\mathbf{w}}^{(m_i)}$ and $\ddot{\mathbf{v}}^{(m_i)}$ can be computed from the first order derivatives of the DAE). The sensitivities will then evolve according to (1.17) for the successor mode until the next event. This process repeats until the end of the simulation, defining a unique sensitivity trajectory. Note that (1.18) and (1.19) are evaluated at a fixed value of the parameters, $\mathbf{p} \in P$, i.e., the parametric sensitivities are evaluated for a nominal value of $\mathbf{p}$.

1.3.2 Examples of ODE Hybrid Systems

Consider an ODE embedded hybrid system in which the set of continuous variables does not change between modes. For a single parameter p, the sensitivity trajectories $\frac{\partial \mathbf{x}}{\partial p}$ between events are governed by the following differential equations, which are derived via (1.17):

$$\frac{\partial}{\partial t}\left(\frac{\partial \mathbf{x}}{\partial p}\right) = \frac{\partial \mathbf{f}^{(m_i)}}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial p} + \frac{\partial \mathbf{f}^{(m_i)}}{\partial p}.$$

Now, consider an event between epochs I_i and I_{i+1} where state continuity is employed as the transition function:

$$\mathbf{x}(p, \sigma_{i+1}) = \mathbf{x}(p, \tau_i).$$

Differentiation of this transition function with respect to the parameter and some rearrangement yields, via (1.18):

$$\mathbf{s}(p, \sigma_{i+1}) = \mathbf{s}(p, \tau_i) + \left(\dot{\mathbf{x}}(p, \tau_i) - \dot{\mathbf{x}}(p, \sigma_{i+1})\right) \left.\frac{\mathrm{d}\tau_i}{\mathrm{d}p}\right|_{p,t}, \tag{1.20}$$

where $\mathbf{s} \equiv \frac{\partial \mathbf{x}}{\partial p}$. Equation (1.20) is instructive because it reveals the qualitative behavior of the sensitivities at an event. This equation indicates that the sensitivities will jump at an event provided two conditions are *both* satisfied: a) the vector field is discontinuous (which is often the case with a switch) and b) the event time is sensitive to the parameter. Otherwise, the sensitivities will be continuous at the event.

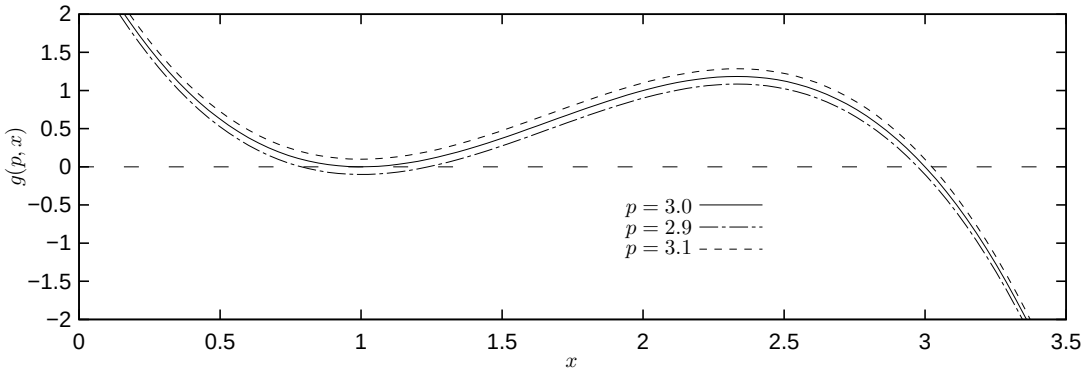
Condition b) will be satisfied when there is a state event whose timing is sensitive to the parameter value (which is usually the case), or there is a time event and the parameter is the time of the event itself ($\left.\frac{\mathrm{d}\tau_i}{\mathrm{d}p}\right|_{p,t} = 1$). Thus, for simulations during which sequences of state events occur, qualitatively, one can expect piecewise continuous sensitivity trajectories containing jumps coinciding with state events, *even if* it is an ODE embedded hybrid system with state continuity employed as the transition function(s).

In [63], a detailed theory is developed governing sufficient conditions for the existence and uniqueness of these (discontinuous) sensitivity trajectories of the hybrid automaton. From these results, it appears that the set of parameter values for which sensitivity trajectories exist and are unique will usually be dense in the parameter space. In particular, the sensitivities cease to exist or be unique for the critical parameter values at which the sequence of events along the solution trajectory changes qualitatively. Consider the following example with two modes [63]:

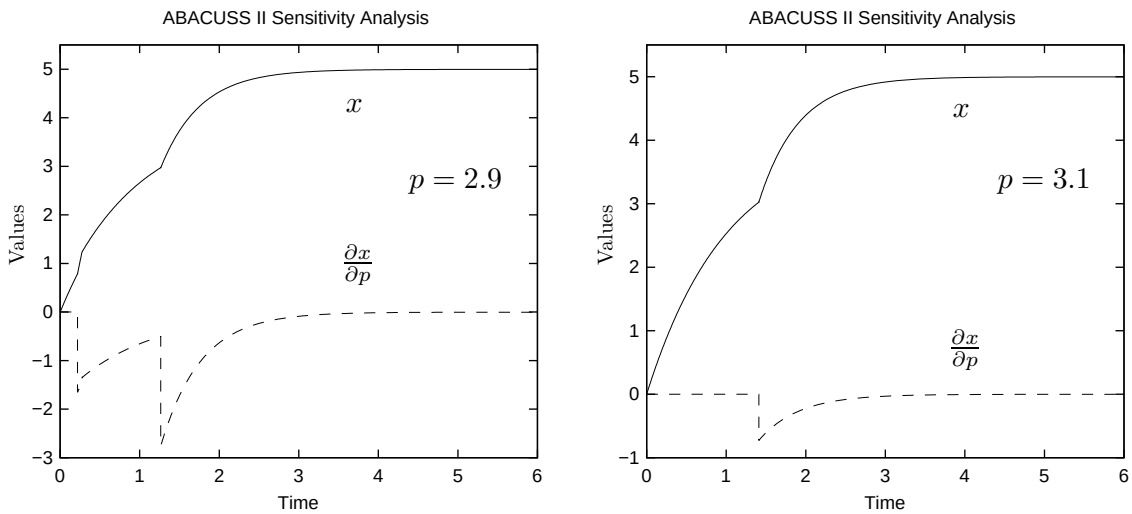
$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} + x - 4, \\ J^{(1)} = \{1\}\ ,\ S^{(1)}(1) = 2, \\ L_1^{(1)} := (-x^3 + 5x^2 - 7x + p \leq 0) \wedge (-3x^2\dot{x} + 10x\dot{x} - 7\dot{x} \leq 0), \\ T_1^{(1)} = x(p, \sigma_{i+1}) - x(p, \tau_i). \end{cases} \tag{1.21}$$

$$\text{Mode 2}: \begin{cases} f^{(2)} = \dot{x} + 2x - 10, \\ J^{(2)} = \{1\}\ ,\ S^{(2)}(1) = 1, \\ L_1^{(2)} := (-x^3 + 5x^2 - 7x + p \geq 0) \wedge (-3x^2\dot{x} + 10x\dot{x} - 7\dot{x} \geq 0), \\ T_1^{(2)} = x(p, \sigma_{i+1}) - x(p, \tau_i). \end{cases} \tag{1.22}$$

Note the addition of the transversality conditions to the transition conditions derived from the reversible discontinuity. The initial mode is 1 with initial condition $T^{(1,0)} = x(0) = 0$. The (discontinuous) sensitivity trajectories exist for all $p \in [2, 4]$, except for $p = 3$. Given a sufficiently long simulation time, for $2 \leq p < 3$ there is a sequence of 3 state events, corresponding to $T_\mu = 1, 2, 1, 2$ and for $3 < p \leq 4$ there is just 1 state event corresponding to $T_\mu = 1, 2$. $p = 3$ is the critical value at which a qualitative



(a) Discontinuity Function

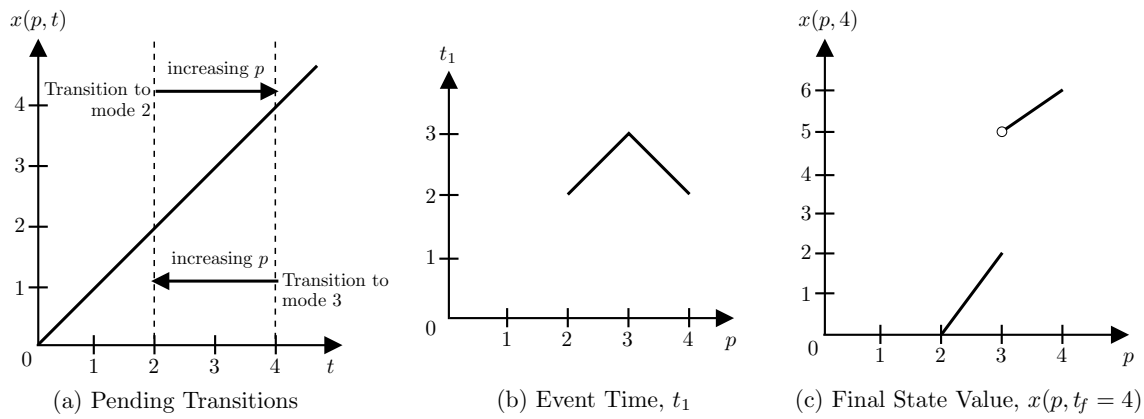


(b) State and Sensitivity Trajectories

Figure 1-11: Sensitivity analysis for ODE example, (1.21)–(1.22). Note $x(p,\cdot)$ is monotonically increasing.

change from 3 events to 1 event occurs. For this parameter value, it is not possible to determine $\left.\frac{\mathrm{d}\tau_1}{\mathrm{d}p}\right|_{p,t}$ using (1.19) at the first event. The state and sensitivity trajectories obtained using ABACUSS II for $p = 2.9$ and $p = 3.1$ are shown in Figure 1-11, where the qualitative change in the sensitivity trajectories is clear.

As another example, consider the hybrid system described in (1.3) to (1.5) with $L_2^{(1)}$ as $x + p \geq 6$, and the precedence relation function $K^{(m)} : Z \mapsto \inf Z$ for $m \in M$. Again, suppose we start with $x(p, 0) = 0$ in mode 1 and wish to end at $t_f = 4$. In addition, the parameter set of interest is $p \in [2, 4]$. The timings of the pending transitions change in the parameter space, and eventually cross at $p = 3$ (see Figure



(a) Pending Transitions (b) Event Time, t_1 (c) Final State Value, $x(p, t_f = 4)$

Figure 1-12: Graphical representation for ODE example, (1.3)–(1.5)

1-12(a)). For $2 \leq p \leq 3$, $T_\mu = 1, 2$, while for $3 < p \leq 4$, $T_\mu = 1, 3$. In other words, $p = 3$ is the critical value at which a qualitative change in the sequence of modes occurs (although there remains a single transition). At this parameter value, the value of $\left.\frac{\mathrm{d}\tau_1}{\mathrm{d}p}\right|_{p,t}$ is not defined (see Figure 1-12(b)), and the sensitivity trajectory does not exist. This situation is excluded in [63] by the assumption of smoothness of transition conditions in a neighborhood of the transition time. An interesting unresolved question is what degree of smoothness is necessary for existence of the sensitivity trajectories.

As mentioned above, it is possible to develop efficient numerical algorithms for the simultaneous computation of the state and sensitivity trajectories of a hybrid automaton. These algorithms are able to exploit the inherent similarities between the state and sensitivity equations in an extremely efficient fashion [57, 63, 129]. An interesting point here is that state event location *must* be performed as described above [129]. With many legacy model codes containing `IF-THEN-ELSE` statements, `MIN/MAX` functions, look-up tables, etc., it is common practice to rely on the error control mechanism of the numerical integrator to deal with these hidden discontinuities. This often works for simulation, even if it is somewhat inefficient. On the other hand, for sensitivity analysis, if the event is not located explicitly and the jump (1.19) not computed, the resulting sensitivity trajectories will not be qualitatively correct. An automated code analysis technique that identifies hidden discontinuities for correct

sensitivity analysis is described in [129].

This theory also has profound implications for the optimization of hybrid systems, and the numerical approximation of hybrid optimal control problems as finite dimensional optimization problems, which will be introduced in Section 1.4. In the case of a small number of functions dependent on the state at fixed times, and a large number of parameters, there exists another possibly more efficient method for calculating derivative information for hybrid systems, the method of adjoints [115]. The development of an existence and uniqueness theory, and efficient numerical algorithms for hybrid adjoints present potentially interesting and rewarding areas for future research.

1.4 Optimization

Most real physical systems experience transitions whose timing and sequence cannot be determined a priori, e.g., the autonomous transitions (which become state events) corresponding to the changes in the flow regime as described in the non-return valve model in Figure 1-8. Clearly, changes in the controls to the system will result in a huge (combinatorial) number of possible hybrid mode trajectories, T_μ, and an infinite dimensional problem with the hybrid time trajectories, T_τ, in the continuous time domain. The search for the optimal control profile that maximizes a given objective function is further complicated by the fact that transitions which occur in hybrid systems often cause nonsmoothness and/or discontinuities in the continuous state variables.

In this thesis, we will only concern ourselves with the open loop optimal control problem, since a large number of engineering tasks can be formulated as such problems. In an open loop optimal control problem, we search a priori for the control profiles and/or continuous parameters for a dynamic system that will optimize a given performance measure over a finite time interval. We note that there is active research in the area of closed loop optimal control for hybrid systems to obtain the optimal feedback control law. In his pioneering work, Bellman generalized the

Hamilton-Jacobi theory to include multi-stage systems and combinatorial problems and he called this theory *dynamic programming* [23]. We will have more to say about dynamic programming in Chapter 3, where we explore the problem of determining the optimal mode sequence. Recent work for hybrid systems include [29] and [30], where the optimal control framework results in a generalized set of Bellman-like equations and several algorithms, among these a boundary-value method and generalized value iterations, are proposed, and [73], where an extended version of Bellman's inequality was discretized to compute a lower bound on the optimal cost function, using linear programming to derive an approximation of the optimal control feedback law. However, dealing with the Hamilton-Jacobi-Bellman equations makes the numerical solution of such problems very difficult in practice, especially for large-scale problems.

While continuous time optimal control problems have received extensive theoretical and numerical treatment in the literature (e.g., [35] and [126]), optimal control of hybrid systems has remained a very difficult problem to solve. The treatments for the continuous case extend, in principle, to problems in which there are discontinuities due to bounds on the control profiles, or sequences of inequality path constraint activations and/or deactivations along the optimal trajectory. Furthermore, [35][pages 106-108] state necessary optimality conditions for an ODE embedded hybrid system with a known sequence of modes, T_μ. However, this classical theory provides no guidance as to how to determine the optimal sequence of modes.

Optimal control of hybrid systems where the continuous state evolves according to difference equations is presented in [89] where the inherent nonsmoothness is noted and a heuristic algorithm is developed. [62] present a quite general mathematical formulation for open loop optimal control and parameter optimization of continuous time hybrid systems based on the hybrid automaton (see Section 1.4.1). They also employ the results for existence and uniqueness of parametric sensitivities mentioned in Section 1.3 to classify when parameter optimization problems with hybrid systems embedded will be smooth or nonsmooth. In [12], a mathematical formulation for the dynamic optimization of hybrid systems described by state-transition networks is presented. The infinite dimensional dynamic optimization problem is solved using

a complete (i.e., control and state) discretization approach, resulting in a large-scale mixed integer nonlinear programming (MINLP) problem. The major difficulty with this formulation is in the size of the problem that can be solved practically. This is compounded by the fact that integer optimization variables are introduced to represent all possible sequences of modes. Although the original authors were not aware of this, the introduction of binary variables eliminates the aforementioned nonsmoothness. Once the binary variables are fixed, defining a mode sequence T_μ, the resulting nonlinear programming (NLP) Master problem in control parameterization (see Section 1.4.2) is smooth. However, with a possibly combinatorial number of modes, a possibly combinatorial number of binary variables may be required in the worst case. Furthermore, the standard methods employed for the solution of MINLPs by [12] rely on the assumption that the participating functions and constraints are convex. If these conditions are not satisfied, standard MINLP algorithms will most likely converge to arbitrary suboptimal points [60]. Deterministic global optimization algorithms for MINLPs have begun to emerge in recent years [117, 2, 81]. However, the size of problem that can be solved is still quite small, so that only very small hybrid optimal control problems can be solved via a complete discretization approach.

In the rest of this section, we present a systematic approach to tackling the optimization problem, building on the concepts and results discussed in the preceding sections. In particular, we focus on the implications of the sensitivity analysis of hybrid systems, since many optimization methods are gradient based.

1.4.1 Optimization Formulation

In addition to the equations describing the hybrid system presented in Section 1.1, with the controls $\mathbf{u}(t)$ introduced as extra arguments, we may impose the following two classes of constraints:

1. Path constraints. These must be satisfied along the entire hybrid time trajectory in a particular mode:

(a) Inequality path constraints:

$$\mathbf{h}^{(m)}\left(\dot{\mathbf{x}}^{(m)}, \mathbf{x}^{(m)}, \mathbf{u}, \mathbf{p}, t\right) \leq \mathbf{0}.$$

(b) Equality path constraints. These can be treated as another equation of the DAE [56] effectively reducing the number of independent controls in that mode.

Note that the path constraints enforced may be completely different from one mode to the next, so that certain path constraints may only hold over some epochs in T_τ.

2. Point constraints. These must be satisfied only at specific times:

(a) Inequality point constraints:

$$\mathbf{c}_r^{(m)}\left(\dot{\mathbf{x}}^{(m)}(t_r), \mathbf{x}^{(m)}(t_r), \mathbf{u}(t_r), \mathbf{p}, t_r\right) \leq \mathbf{0}, r \in \{1, \ldots, n_r^{(m)}\}.$$

(b) Equality point constraints, which include the initial and final conditions:

$$\mathbf{c}_s^{(m)}\left(\dot{\mathbf{x}}^{(m)}(t_s), \mathbf{x}^{(m)}(t_s), \mathbf{u}(t_s), \mathbf{p}, t_s\right) = \mathbf{0}, s \in \{1, \ldots, n_s^{(m)}\}.$$

Again, the point constraints enforced may differ from one mode to the next.

The compact expression of an objective function is complicated by the possibility of different sets of continuous variables characterizing each mode in variable structure models, and the fact that the optimal sequence of modes, and thus the active mode at the final time, are not known a priori. One solution to this problem [19] is to introduce a dummy terminal mode, labeled n_m+1, to occupy the dummy instantaneous terminal epoch, $I_{n_e+1} = [\sigma_{n_e+1}, t_f]$ where $\tau_{n_e} = \sigma_{n_e+1} = t_f$, and require a terminal transition with the appropriate transition function from every other mode to this terminal mode

at the final time, t_f. This allows the formulation of a Mayer type objective function:

$$\min_{\mathbf{u}(t),\mathbf{p},t_f} \phi\left(\dot{\mathbf{x}}^{(m_{n_m+1})}(t_f), \mathbf{x}^{(m_{n_m+1})}(t_f), \mathbf{u}(t_f), \mathbf{p}, t_f\right).$$

This can be made equivalent to Lagrange or Bolza type objective functions through the introduction of additional state variables. Furthermore, terms related to the cost of transitions or the values of states and/or controls at the final time in each mode can be incorporated by introducing additional state variables whose values are modified by the transition functions.

1.4.2 Problem Classification

Here, we shall categorize the optimization problem with hybrid systems embedded into different categories according to the existence and uniqueness of the sensitivity trajectories of the hybrid systems, as we will be focusing on the application of deterministic, gradient based optimization techniques to solve these problems. Incidentally, the classification of problems presented here forms the basis for the organization of this thesis.

To start off, consider the design of a safe changeover operation for a pressure vessel in a chemical process as shown in Figure 1-7 [19]. We wish to move safely in minimum time from an initial state in which methane is flowing through the system to a final state in which oxygen is flowing. The objective function can be expressed as:

$$\min_{\mathbf{u}(t),t_f} t_f.$$

The model equations are summarized in Figure 1-8, with all transition functions (except the initial conditions, $\mathbf{T}^{(0)}$) enforcing continuity of the molar holdups $M_{i\in\{1,2,3\}}$. The point and path constraints define the changeover policy required:

$$y_3(0) = 0, \qquad y_1(0) = 1, \qquad P(0) = P_0, \tag{1.23}$$

$$y_3(t_f) \geq 0.999, \tag{1.24}$$

where (1.23) describes the initial conditions and (1.24) describes the final target composition requirement. The key safety consideration is to avoid the formation of an explosive mixture in the vessel at any time during the changeover. If a curve bounding the explosive region of $O_2/N_2/CH_4$ mixtures in composition space is constructed (see Figure 1-7(b)), this curve can be used to formulate an inequality path constraint:

$$h(y_1, y_3) \leq 0$$

that must hold throughout T_μ. Other path constraints, such as the equipment limits, e.g., design pressure, can be easily incorporated.

An operating procedure would be defined by the control profiles, $u_{k \in K}(t)$, which specify the sequence of openings/closings of the four valves along a particular T_μ. Since these are open/close valves, the signals are binary variables, and the problem as stated above might be termed a mixed-integer optimal control (MIOC) problem. This factor further complicates the variational analysis of such problems, which motivates the study of parameter optimization of hybrid systems, in which the set of decision variables is reduced to time invariant parameters $\mathbf{p}$, making the problem a finite dimensional optimization problem with a hybrid system embedded, rather than an infinite dimensional optimal control problem.

The most widely used methods for numerical optimal control of large-scale and/or highly constrained problems rely on approximation of the infinite dimensional problem by a finite dimensional parameter optimization problem, either via discretization of the controls or discretization of both controls and states [126]. For example, *control parameterization* [126] approximates the controls with a finite set of basis functions, e.g., Lagrange polynomials on finite elements (see [134] for a discussion), and thus the controls in the above formulation become dependent variables $\mathbf{u}(\mathbf{p}, t)$. As an example, consider the one-dimensional control $u(t)$ over the time horizon $t \in [t_0, t_f]$. Suppose that we wish to parameterize the control over N_u finite elements (stages). Let each stage be denoted by the contiguous time intervals, $[\theta_i, \omega_i]$ for $i = 1, \ldots, N_u$ such that $\omega_i > \theta_i$ for $i = 1, \ldots, N_u$, $\theta_1 = t_0$, $\omega_{N_u} = t_f$, and $\theta_{i+1} = \omega_i$ for $i = 2, \ldots, N_u - 1$. Then,

over the stage $k \in \{1, \ldots, N_u\}$, the control is given by

$$u^{(k)}(t) = \sum_{i=1}^{n_o} p_{ik} \phi_i^{(n_o)}(\nu^{(k)}(t)), \; t \in [\theta_i, \omega_i],$$

where p_{ik} represent the real valued control parameters to be determined, $\nu^{(k)}(t)$ represents normalized time over stage k given by

$$\nu^{(k)}(t) = \frac{t - \theta_k}{\omega_k - \theta_k},$$

the superscripts on u and ν indicate the stage, n_o represents the order of the Lagrangian polynomial approximation, and the Lagrangian polynomials of order n_o, $\phi_i^{(n_o)}, i = 1, \ldots, n_o$, are defined in the standard manner,

$$\phi_i^{(n_o)}(\nu) = 1 \text{ if } n_o = 1,$$
$$\phi_i^{(n_o)}(\nu) = \prod_{\substack{j=1 \\ j \neq i}}^{n_o} \frac{\nu - \nu_j}{\nu_i - \nu_j} \text{ if } n_o \geq 2,$$

where $\nu_i, i = 1, \ldots, n_o$ are the set of normalized time points used for the construction of the approximating polynomial. Note that, as discussed in [134], the choice of the set of normalized time points does not affect the solution obtained from control parameterization. However, a judicious choice of the set of normalized time points may be useful in enforcing bounds on the control, e.g., choosing $\nu_1 = 0$ and $\nu_2 = 1$ for piecewise constant ($n_o = 1$) and piecewise linear ($n_o = 2$) approximations.

As an example, consider the one-dimensional control $u(t)$ over the time horizon $t \in [0, 2]$. To implement a piecewise constant control parameterization over two equal finite elements, we simple set

$$u(\mathbf{p}, t) = \begin{cases} p_1 & \text{for } 0 \leq t \leq 1 \\ p_2 & \text{for } 1 \leq t \leq 2 \end{cases} .$$

Note that the value of $u(\mathbf{p}, t)$ can take on two different values at $t = 1$. The notion

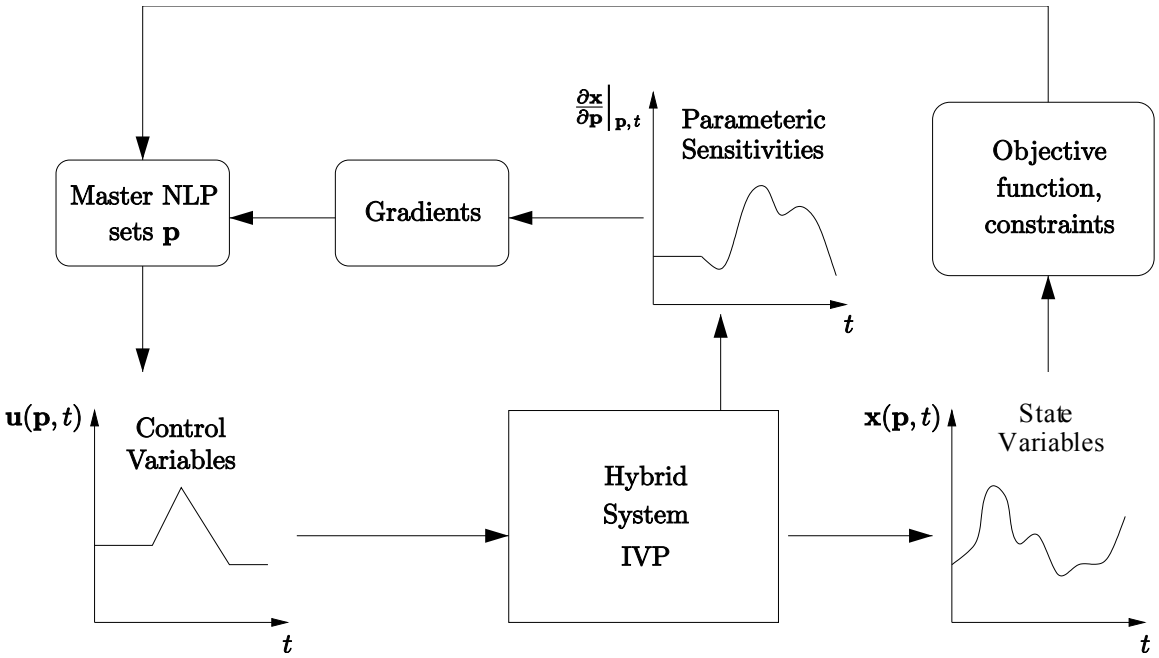


Figure 1-13: Control parameterization.

of the current stage of control parameterization allows this to happen, similar to the way the notion of the current epoch allows the continuous state of the hybrid system can take on multiple values at the boundaries of the epochs.

For numerical solution of the resulting parameter optimization problem, this discretization yields a decomposition into two subproblems, as shown in Figure 1-13:

1. An initial value (IVP) subproblem in which the hybrid system model is solved for given values of $\mathbf{p}$ using the simulation technology described in Section 1.2.

2. An NLP Master problem that searches in the finite parameter space using function and constraint information furnished by the IVP subproblem. If the Master NLP is to be solved using an efficient gradient-based search, gradient information must be extracted in some manner from the embedded hybrid dynamic system. Although there are a number of ways of doing this, probably the most efficient method at present is to compute the parametric sensitivities described in Section 1.3, and then apply the chain rule to the dependent functions to use these sensitivities to deliver the gradients.

Clearly, the utility of such a method hinges on the existence and uniqueness of

the parametric sensitivities of the hybrid system. The discontinuous nature of the hybrid models might suggest that gradient based methods are inappropriate for these problems. Sufficient conditions for the existence and uniqueness of these sensitivities are proved in [63], and provide an important classification of problems for which these gradient based methods can be applied [62]. Surprisingly, these results indicate that the sensitivity trajectories of a hybrid system will usually exist almost everywhere in the parameter space. This enables us to distinguish the following classes of problems, at least provided that there are no inequality path constraints:

1. Problems where the sequence of modes, T_μ, does not change in the parameter space of interest, *and* the parametric sensitivities do not jump at events. For example, this occurs in ODE embedded systems whenever the state is continuous and the vector field is continuous at any transition. This class of problems can be solved by a gradient based method.

2. Problems where T_μ does not change in the parameter space of interest, but the parametric sensitivities exhibit discontinuities at some events. Galán and Barton [62] present a theorem in which sufficient conditions for the smoothness of the objective function of the Master NLP problem for control parameterization are derived. In this case, this class of problems can be solved by a gradient based method.

3. Problems where T_μ changes in the parameter space of interest. We have found that the critical parameter values at which the sequence of events changes typically correspond to points of discontinuity or nondifferentiability in the objective function [62]. Hence, conventional gradient based optimization methods will fail due to the inherent nonsmoothness of the objective function. Unfortunately, these are also the most interesting and complex class of problems.

Roughly speaking, methods to solve problems which fall into categories 1 and 2 are presented in Chapter 2, while problems which fall into category 3 have to be decomposed into smaller subproblems, and our efforts to tackle these challenging problems

are presented in Chapters 3 and 4. To deal with the problem of including inequality path constraints, several numerical methods have been proposed [35, 88, 126, 55], but none is completely satisfactory. There is scope for further improving this area of dynamic optimization problems; however, this will not be addressed in this thesis.

Note that the classification of the problems presented above is a rough rule of thumb to describe most problems which we have encountered in practice. As discussed in Section 1.1.4, it is possible to construct problems in which the mode sequence does not change anywhere in the parameter space of interest, but in which the parametric sensitivities do not exist at critical points. On the other hand, it is also possible to construct problems in which the mode sequence changes in the parameter space of interest, but for which the parametric sensitivities exist for all parameter values. For example, consider the following trivial hybrid system,

$$\text{Mode 1}: \begin{cases} f^{(1)} = \dot{x} - p, \quad J^{(1)} = \{1\}, \\ S^{(1)}(1) = 2, \qquad L_1^{(1)} := (t \geq p), \\ T_1^{(1)} = x(\sigma_{i+1}) - x(\tau_i), \end{cases}$$
$$\text{Mode 2}: f^{(2)} = \dot{x} - p, \quad J^{(2)} = \emptyset.$$

where $p \in [0.5, 1.5]$, $t \in [0, 1]$, and $x(0) = 0$ with initial mode 1. Note that the sequence of modes is $T_\mu = 1, 2$ for $0.5 \leq p \leq 1$ and $T_\mu = 1$ for $1 < p \leq 1.5$. However, one can verify that the parametric sensitivities exist and are unique for all $p \in [0.5, 1.5]$. Note that this is a pathological example because the transition has no effect on the behavior of the continuous state of the hybrid system. Having said this, the above classification of problems is still a very useful guide to think about optimization problems with hybrid systems embedded, because most nontrivial optimization problems will fall within the classification.

The optimization problem for the tank changeover example falls into the third class of problems as T_μ clearly changes in the parameter space of interest. This has motivated Barton et al. [19] to solve the problem using a direct search stochastic procedure that is relatively insensitive to nonsmoothness and nonconvexity of the

Master NLP. We note that with this approach, the optimization procedure implicitly infers, from the existence of the path constraint the need for a N_2 purge before introducing O_2. The purpose of this example is to demonstrate that given a hybrid dynamic model, and the point and path constraints representing safety and operational goals/constraints, a hybrid dynamic optimization procedure can, in principle, design the changeover policy *automatically*.

Consider next the following optimization problem:

$$\min_{p} \; x(p, t_f),$$

where $x(p, t_f)$ is given by the solution of the hybrid system described by Equations (1.3) - (1.5), $x(p, 0) = 0$ in mode 1, $t_f = 4$, and $p \in [2, 4]$. In Section 1.3, we have seen how T_μ changes in the parameter space. From Figure 1-12(c), it is clear that the objective function is nonsmooth (and discontinuous) at the critical point $p = 3$, and this is another example of a problem that falls into the last class of problems described above.

Chapter 2

Fixed Mode Sequence and Transition Times

In this chapter, we will discuss how to solve optimization problems with embedded LTV ODE hybrid systems, and whose transitions are known a priori, i.e., the hybrid mode and time trajectories T_μ and T_τ are fixed (see Section 2.3 for a formal definition of the hybrid systems considered). This class of problems corresponds to categories 1 and 2 in the classification presented in Section 1.4.2, which are smooth in the control parameterization framework, because T_μ is fixed. The application of control parameterization to this class of *multi-stage* problems and the use of local gradient based algorithms have previously been studied [100, 134]. The material described in this chapter makes the following contributions:

1. The computation of parametric sensitivities is treated rigorously using the existence and uniqueness theorems presented in [63]. The correct computation of the parametric sensitivities, especially at the transitions, is crucial to the application of efficient gradient based algorithms;

2. A method for constructing convex relaxations of a Bolza type objective function with a linear hybrid system embedded is presented. This is a natural and direct extension of the theory described in [120]. A deterministic algorithm to find the *global* solution is then possible when branch-and-bound (BB) methods [96, 117]

are applied.

For related work on linear switched systems with quadratic objective functions, see [137], where the sequence of modes is fixed, the timings of the transitions are parameterized, and the gradients to the local NLP solver are obtained by solving the HJB equations using the dynamic programming approach.

This chapter is organized as follows. First, in Section 2.1, we introduce deterministic methods for global optimization of regular optimization problems, which form the foundation for algorithms for global dynamic optimization with hybrid systems embedded. These concepts will be revisited again and again throughout the rest of this thesis. In Section 2.2, we will explain the rationale behind only focussing on LTV systems instead of general nonlinear systems. The LTV hybrid system of interest is defined in Section 2.3 using the modeling framework presented in Chapter 1. This is followed by formulation of the optimization problem in Section 2.4. Section 2.5 contains a discussion on the solution strategy employed, calculation of the parametric sensitivities at transitions, and the existence of a minimum. A simple example of how practical solvers behave when given a nonsmooth problem is also shown here. Convex relaxations of the Bolza type objective function are constructed in Section 2.6, and the branch-and-bound (BB) method utilizing these underestimators is shown to be infinitely convergent when the implied state bounds presented in Section 2.7 are employed. Finally, Section 2.8 contains two illustrative examples where the proposed methods are applied.

2.1 Deterministic Global Optimization

Global optimization methods can be divided into two main categories: stochastic methods and deterministic methods. Deterministic methods utilize gradient based optimization algorithms to guarantee a global solution to the problem within some user specified ε tolerance, with a finite number of iterations. In essence, deterministic methods provide a *certificate* of optimality, i.e., they can guarantee that the global solution value of the problem cannot be better than ε of the objective function value

at the feasible point that has been obtained on termination. To illustrate, given the optimization problem:

$$\min_{\mathbf{x} \in D} f(\mathbf{x})$$

where $D \subset \mathbb{R}^n$ is a nonempty, compact set and $f : D \to \mathbb{R}$ is a continuous function, a deterministic global optimization algorithm furnishes a point $\mathbf{x}^* \in D$ such that

$$f(\mathbf{x}^*) \leq f^* + \varepsilon$$

in a finite number of iterations, where $f^* \equiv \min_{\mathbf{x} \in D} f(\mathbf{x})$.

This property makes deterministic global optimization methods particularly well suited for applications which demand that the global solution be found, e.g., formal safety verification problems and parameter estimation problems. In the former case, a suboptimal local solution could falsely indicate that all safety specifications are met, leading to disastrous consequences if, in actuality, a global solution exists which provides a counter example that violates some safety specification. In the latter case, a suboptimal local solution could falsely indicate that a proposed model structure did not match experimental data in a statistically significant manner, leading to the false rejection of a valid model structure [122]. In addition, for certain engineering problems, the optimization problem is so horribly nonconvex (this is especially true of nonlinear equality constraints) that local optimization methods can often fail to even produce a single feasible point, even though the problem is clearly feasible.

While stochastic methods, such as random direct search methods, simulated annealing, genetic algorithms, etc. are generally much easier to implement than deterministic methods, they are unable to provide a guarantee of global optimality within a finite number of iterations (while many of these methods have provisions and heuristics for avoiding getting "stuck" at local minima, in the best case they can only guarantee global optimality as the number of iterations approaches infinity and they have no way of measuring how close they are to the global solution after a finite number of iterations). Thus, for problems in which a lower bound on the optimal solution value is not known a priori, it is impossible to know whether a so-

lution is "good enough" solely using a stochastic method and without appealing, in some form, to the theory of deterministic global optimization methods. This thesis focuses solely on the deterministic approach, because one of the goals of this thesis is to lay the foundations for the theory and algorithms needed to solve formal safety verification problems with hybrid systems embedded. While stochastic optimization methods have previously been applied for the optimization of hybrid systems [19], the work in this thesis represents the first attempt at deterministic global optimization of hybrid systems.

Many modern, general methods for deterministic global optimization in Euclidean spaces rely on the notion of a *convex relaxation* of a nonconvex function [96, 1]. This is a convex function which underestimates a nonconvex function on the set of interest, i.e., a convex relaxation of a function f on a convex set C is a convex function $u : C \to \mathbb{R}$ such that $u(\mathbf{x}) \leq f(\mathbf{x}), \forall \mathbf{x} \in C$. The convex programs that result from convex relaxation of all nonconvex objective and constraint functions in a nonconvex program can (in principle) be solved to guaranteed global optimality, which, for example, can be used to generate rigorous lower bounds on the solution value of the nonconvex problem for a BB algorithm [77]. In BB, the feasible set is first relaxed and subsequently split into partitions (*branching*) over which rigorous lower and upper bounds on the solution value of the nonconvex problem can be determined (*bounding*). If the lower bound on a partition of the feasible space is greater than the current best upper bound, or if the lower bounding convex problem is infeasible on that partition, the partition is removed from the search space since the minimum can never be attained there (*fathomed*).

It should be noted that BB algorithms involving real valued decision variables do not in general terminate finitely. On the other hand, ε optimality can be achieved in a finite number of iterations. In general, BB algorithms exhibit exponential running time with the number of optimization variables, i.e., in the worst case the running times for these algorithms grow exponentially with the number of optimization variables. However, it is worth noting the BB algorithm can be naively viewed as a multi-start algorithm employing local optimization methods to generate many dif-

ferent feasible solutions, while also providing a rigorous lower bound on the global solution. If the BB algorithm is taking too many iterations, the user can terminate after a specified iteration limit to obtain the current best solution (best upper bound) and a rigorous estimate of how far this best solution is from the global solution (gap between the best upper bound and the current lower bound).

Before we introduce two well established algorithms for solving nonconvex NLPs and nonconvex MINLPs (BB and nonconvex Outer Approximation (OA) respectively), we will briefly comment on the process of obtaining convex relaxations for nonconvex functionals on Euclidean spaces. This process of obtaining rigorous convex relaxations is key to applying deterministic global optimization. There are two main results which enable this to be done for regular nonconvex problems: McCormick's composition theorem [96] and αBB and derivatives [1, 2].

Theorem 2.1 (McCormick's Composition Theorem)**.** *Let $X \subset \mathbb{R}^n$ be a nonempty convex set. Consider the function $T \circ t$ where $t : X \to \mathbb{R}$ is continuous, and let $X \subset \{\mathbf{x} : t(\mathbf{x}) \in [a, b]\}$. Suppose that a convex function u^t and a concave function o^t satisfying*

$$u^t(\mathbf{x}) \leq t(\mathbf{x}) \leq o^t(\mathbf{x}), \quad \forall \mathbf{x} \in X$$

are known. Let $u^{\mathcal{T}}$ be a convex relaxation of T on $[a, b]$, let $o^{\mathcal{T}}$ be a concave relaxation of T on $[a, b]$, let $z^{\min}$ be a point at which $u^{\mathcal{T}}$ attains its infimum on $[a, b]$, and let $z^{\max}$ be a point at which $o^{\mathcal{T}}$ attains its supremum on $[a, b]$. If the above conditions are satisfied, then

$$u^{\mathcal{T} \circ t}(\mathbf{x}) = u^{\mathcal{T}} \left[mid\{u^t(\mathbf{x}), o^t(\mathbf{x}), z^{\min}\} \right]$$

is a convex relaxation of $T \circ t$ on X, and

$$o^{\mathcal{T} \circ t}(\mathbf{x}) = o^{\mathcal{T}} \left[mid\{u^t(\mathbf{x}), o^t(\mathbf{x}), z^{\max}\} \right]$$

is a concave relaxation of $T \circ t$ on X, where the mid function selects the middle value of three scalars.

Note that Theorem 2.1 is not the original formulation in [96], but the modified for-

mulation by Barton [20] which fixes a number of bugs (the theorem is stated without proof, found in [20]). Due to the presence of the mid function, the constructed convex relaxations from applying Theorem 2.1 are not guaranteed to be smooth. However, it is possible to generate smooth relaxations from the McCormick relaxations by linearizing (i.e., constructing supporting hyperplanes) the nonsmooth relaxations at user specified points, and utilizing the linearized relaxations instead.

The methods of αBB and its derivatives guarantee that the constructed convex relaxations are twice-continuously differentiable. However, we have found in practice that the McCormick relaxations tend to produce tighter relaxations. The measure of "tightness" of a convex relaxation can be quantified by the maximum distance between the convex relaxation and the original function in the set on which the relaxation is constructed. The smaller this distance is, the tighter the convex relaxation. When using the BB algorithm, one would like to utilize the tightest convex relaxations possible, as this usually accelerates the convergence of the BB algorithm.

In [96], a factorization scheme was developed to deal with functions defined as compositions of finite sequences of elementary operations. In general, this factorization scheme will generate nonsmooth convex relaxations. Tolsma and Barton [128] shows how a smooth convex relaxation of the factorable function can be constructed through the introduction of extra constraints and variables. Gatzke et al. [64] demonstrated how the aforementioned methods can be combined and automated.

2.1.1 Branch-and-Bound Algorithm

Consider the following NLP:

$$\begin{aligned} &\min_{\mathbf{p}\in P} f(\mathbf{p}) \\ &\text{s.t. } \mathbf{g}(\mathbf{p}) \leq \mathbf{0} \end{aligned} \tag{o-NLP(P)}$$

where $P \subset \mathbb{R}^{n_p}$ is a nonempty, compact, convex set, $f : P \to \mathbb{R}$ and $\mathbf{g} : P \to \mathbb{R}^{n_g}$ are continuous on P. We will assume that a convex relaxation of o-NLP(P) can be

constructed, and is given by the following NLP:

$$\begin{aligned} &\min_{\mathbf{p}\in P} u(\mathbf{p}; P) \\ &\text{s.t. } \mathbf{h}(\mathbf{p}; P) \leq \mathbf{0} \end{aligned} \tag{c-NLP(P)}$$

where u is a convex relaxation of f on P, and $\mathbf{h}$ is a convex relaxation of $\mathbf{g}$ on P. We also assume that given a convex NLP, c-NLP(P), we have a NLP solver, (CNLPS), that terminates finitely with the following output:

1. Return $+\infty$ if c-NLP(P) is infeasible.
2. Return objective function value $u(\mathbf{p}^*; P)$ and a global solution $\mathbf{p}^* \in P$ if c-NLP(P) is feasible.

Without loss of generality, let the set P be represented by known bounds on $\mathbf{p}$, i.e., $P = [\mathbf{p}^L, \mathbf{p}^U]$. We will now define what we mean by a partition of a set.

Definition 2.2. Call R_P a *partition* of P when

$$R_P = \{P_k \mid \bigcup_k P_k = P,\ \text{int}(P_k) \cap \text{int}(P_{j\neq k}) = \emptyset\}.$$

The following is a spatial BB algorithm, taken from [20], for solving o-NLP(P) with input $\varepsilon > 0$ as the convergence tolerance:

Algorithm 2.3.

1. (***Initialization***) $P_0 := P$, $I = \{P_0\}$, $LBD_0 := -\infty$, $UBD := +\infty$, $k = 1$.
2. (***Termination Test***) Delete from I all nodes P_i with $LBD_i \geq UBD$. Set

$$LBD := \min_{P_i \in I} LBD_i.$$

   If $UBD - LBD \leq \varepsilon$ or $I = \emptyset$ terminate. If $UBD = +\infty$, then o-NLP(P) is infeasible. Otherwise, UBD is an ε-optimal estimate for the solution value and $\mathbf{p}^*$ is a feasible point at which UBD is attained.

3. (***Node Selection***) Select and delete a node P_i from I according to a node selection heuristic.

4. (***Lower Bounding***) Solve c-NLP(P_i) using CNLPS. Then,

   (a) If CNLPS returns $+\infty$, set $LBD_i := +\infty$.

   (b) Else, set LBD_i to be the optimal solution value and set $\hat{\mathbf{p}}$ to be an optimal solution.

   If $\hat{\mathbf{p}}$ is feasible for o-NLP(P) and $f(\hat{\mathbf{p}}) < UBD$ then set

$$UBD := f(\hat{\mathbf{p}}), \mathbf{p}^* := \hat{\mathbf{p}}.$$

5. (***Fathoming***) If $LBD_i = +\infty$ or $LBD_i \geq UBD$ then goto 2.

6. (***Optional Upper Bounding***) Solve o-NLP(P_i) locally using CNLPS. If a feasible point is located, let $\hat{\mathbf{p}}$ be the point and if $f(\hat{\mathbf{p}}) < UBD$ then set

$$UBD := f(\hat{\mathbf{p}}), \mathbf{p}^* := \hat{\mathbf{p}}.$$

7. (***Branching***) Partition the set P_i into sets P_k and P_{k+1} according to some partitioning rule. Set LBD_k, $LBD_{k+1} := LBD_i$. Add nodes P_k and P_{k+1} to I. Set $k = k + 2$. Goto 2.

A BB algorithm is at least infinitely convergent if the selection operation is bound improving and the bounding operation is consistent [77, Theorem IV.3]. Since fathoming of a particular partition of the parameter search space occurs only when its lower bound is greater than the best current upper bound, or the lower bounding problem on that partition is infeasible, the selection operation in Step 3 is bound improving by definition. Hence, in order for Algorithm 2.3 to be infinitely convergent, the convex relaxations in c-NLP(P_i) have to become tighter as the optimization parameter set P_i becomes smaller upon branching, and they have to converge to the original functions f and $\mathbf{g}$ in the limit as P_i shrinks to degeneracy (this implies that

the bounding operation is consistent). This is true for both McCormick's method for constructing convex relaxations and αBB. This ensures that for feasible o-NLP(P), the incumbent lower bound, LBD, will eventually approach the upper bound, UBD, as the number of iterations increases. If o-NLP(P) is feasible, then Algorithm 2.3 terminates finitely with $(\mathbf{p}^*, UBD)$, where the global solution value of o-NLP(P) is bounded by $UBD - \varepsilon$ and UBD. If o-NLP(P) is infeasible, then Algorithm 2.3 terminates finitely with an indication that o-NLP(P) is infeasible. A simple way to see this is to consider a problem for which o-NLP(P) is feasible. Suppose that P is partitioned into P_1 and P_2 such that o-NLP(P_1) is feasible, but for which o-NLP(P_2) is infeasible. Then, in order for Algorithm 2.3 to terminate finitely with the solution when applied to o-NLP(P), it will have to terminate finitely with an indication of infeasibility when applied to o-NLP(P_2).

Algorithm 2.3 follows closely the work of [54, 96, 77]. It is also possible to add pre-processing and post-processing steps to accelerate convergence using a branch-and-reduce algorithm [117].

2.1.2 Nonconvex Outer Approximation Algorithm

Consider the following MINLP:

$$\begin{aligned} &\min_{\mathbf{p}\in P, \mathbf{y}\in Y^b} f(\mathbf{p},\mathbf{y}) \\ &\text{s.t. } \mathbf{g}(\mathbf{p},\mathbf{y}) \leq \mathbf{0} \end{aligned} \qquad \text{(o-MINLP}(P, Y^b))$$

where $P \subset \mathbb{R}^{n_p}$ is a nonempty, compact, convex set, $Y^b = \{0,1\}^{n_y}$, $Y = [0,1]^{n_y}$, $f : P \times Y \to \mathbb{R}$ and $\mathbf{g} : P \times Y \to \mathbb{R}^{n_g}$. We will assume that a convex relaxation of o-MINLP(P, Y^b) can be constructed, and is given by the following MINLP:

$$\begin{aligned} &\min_{\mathbf{p}\in P, \mathbf{y}\in Y^b} u(\mathbf{p},\mathbf{y}; P, Y) \\ &\text{s.t. } \mathbf{h}(\mathbf{p},\mathbf{y}; P, Y) \leq \mathbf{0} \end{aligned} \qquad \text{(c-MINLP}(P, Y))$$

where u is a convex relaxation of f on $P \times Y$, and $\mathbf{h}$ is a convex relaxation of $\mathbf{g}$ on $P \times Y$.

Outer approximation as a decomposition approach has been employed very successfully for convex MINLPs [50, 59] (those MINLPs in which the participating functions are convex, e.g., c-MINLP(P, Y)). The extension to nonconvex MINLPs hinges on the ability to construct convex relaxations of the objective function and constraints to form the lower bounding convex c-MINLP(P, Y) [82, 83]. Note that any general mixed-integer problem can be reformulated into mixed-binary problem, hence we will only consider problems where $Y^b = \{0, 1\}^{n_y}$. By relaxing Y^b into $Y = [0, 1]^{n_y}$, c-MINLP(P, Y) can be constructed using the standard methods for convex relaxations described above. In the context of the OA algorithm for the global solution of nonconvex MINLPs described in [83], we introduce the *Primal Problem* as a nonconvex NLP with $\mathbf{y}^*$ fixed in o-MINLP(P, Y^b):

$$\begin{aligned} &\min_{\mathbf{p} \in P} f(\mathbf{p}, \mathbf{y}^*) \\ &\text{s.t. } \mathbf{g}(\mathbf{p}, \mathbf{y}^*) \leq \mathbf{0}, \end{aligned} \tag{NLP($\mathbf{y}^*$)}$$

and the corresponding *Primal Bounding Problem* as a convex NLP with $\mathbf{y}^*$ fixed in c-MINLP(P, Y):

$$\begin{aligned} &\min_{\mathbf{p} \in P} u(\mathbf{p}, \mathbf{y}^*; P, Y) \\ &\text{s.t. } \mathbf{h}(\mathbf{p}, \mathbf{y}^*; P, Y) \leq \mathbf{0}. \end{aligned} \tag{NLPB($\mathbf{y}^*$)}$$

The OA algorithm solves for the global solution by alternating finitely between the primal problem, the primal bounding problem, and relaxations of the Master problem (which is not described here, see [83] for a description), as shown in Figure 2-1. Because the solution of the relaxed master problem provides a non-decreasing sequence of rigorous lower bounds, and the solution of the primal problem provides a sequence of upper bounds, the algorithm terminates finitely either when the lower

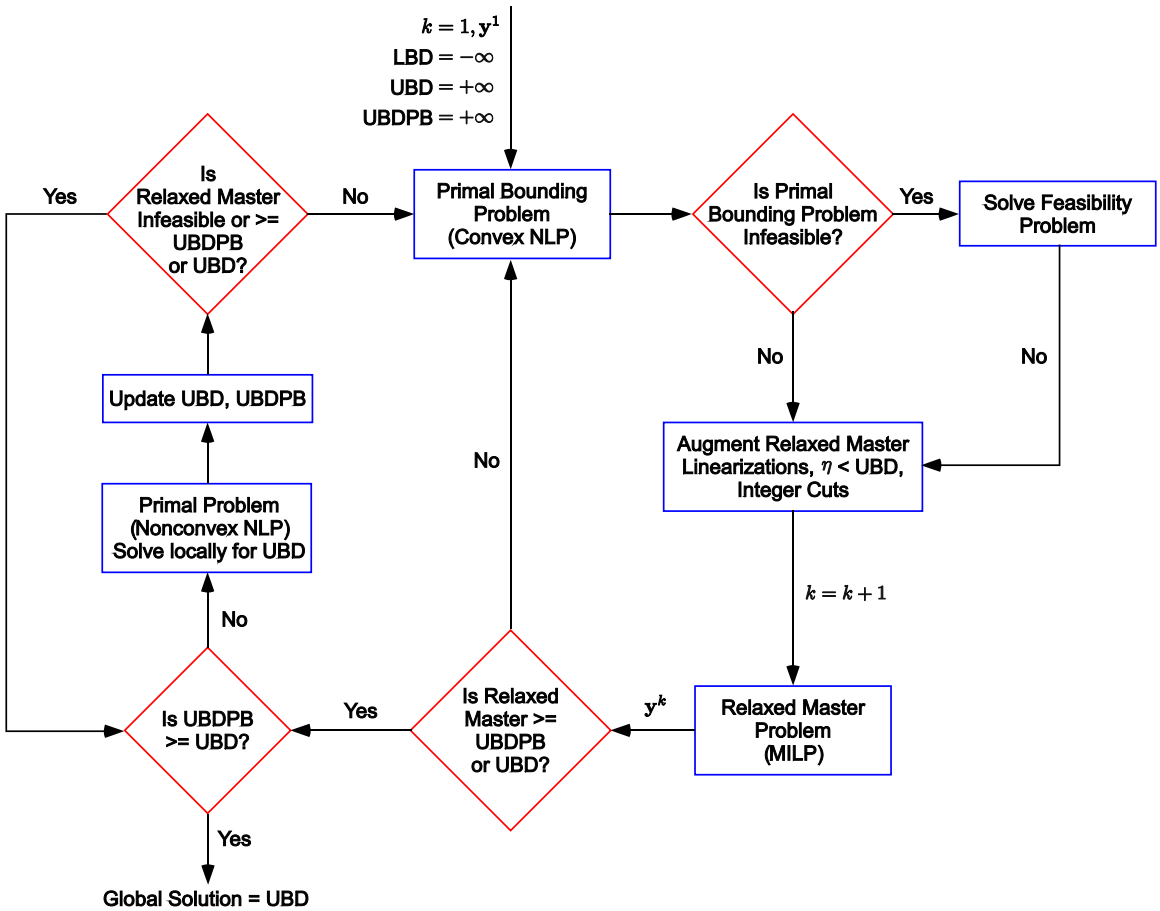


Figure 2-1: Outer approximation for nonconvex MINLPs.

bound crosses the best upper bound, or the relaxed Master problem becomes infeasible. The primal bounding problem provides a valid and tighter lower bound to the primal problem for each binary realization, $\mathbf{y}^k$, then that provided by the current relaxed Master problem. Hence if the solution to the primal bounding problem is greater than the current best upper bound, its corresponding primal problem need not be solved for iteration k. This is important, since the convex primal bounding problem is usually the least expensive to solve, followed by the relaxed Master problem, while the nonconvex primal problem, which requires global deterministic methods to solve, is usually the most expensive, unless a special structure is exhibited once the binary variables are fixed.

We note that the OA algorithm will terminate finitely, even when certain binary realizations of o-MINLP(P, Y^b) may not be feasible. This is because the nonconvex

OA algorithm terminates with a finite number of major iterations (in the worst possible case, all possible combinations of $\mathbf{y}$ are exhaustively generated by the relaxed master problem) [83, Corollary 3.3.1]. The solution of the relaxed master problem (MILP) is accomplished in finite time [103], as is the solution of each primal bounding problem (convex NLP) by the assumption on CNLPS. Finally, Algorithm 2.3 terminates in finite time for each primal problem as discussed above. This implies that the global solution to o-MINLP(P, Y^b) is obtained within ε optimality [83, Corollary 3.3.1], or the OA algorithm will terminate with an indication that o-MINLP(P, Y^b) is infeasible.

2.2 Linear vs. Nonlinear Dynamics

For the rest of this chapter, we will focus on solving problems with LTV hybrid systems embedded. There are a number of reasons for examining linear hybrid systems instead of the general nonlinear case:

1. Linear systems are easier to analyze, and often provide much insight to strategies for solving the general, nonlinear problem.

2. The structure of linear systems lends itself to specialized and tailored algorithms which exploit said structure, leading to efficient solution strategies.

3. One cannot hope to solve problems with general, nonlinear hybrid systems embedded if problems with linear hybrid systems cannot be solved.

Hence, the focus throughout this thesis will be on linear hybrid systems, since it represents fundamental work for the deterministic, global solution of dynamic optimization problems with linear hybrid systems embedded. It is noted, however, that in the course of developing the theory and algorithms in Chapter 4, the same theory and algorithms can be easily extended to handle nonlinear hybrid systems. For more on this, see Chapter 4.

2.3 The Linear Hybrid System

Here, we shall define the linear hybrid system of interest, based on the modeling framework presented in Section 1.1. We say that a transition is *explicit* when its timing, predecessor and successor modes are known a priori.

Definition 2.4. The LTV ODE hybrid system of interest is defined by the following:

1. A fixed $T_\mu = m_1, \ldots, m_{n_e}$, with $m_i \in M$, and a fixed T_τ with fixed initial and final time, σ_1 and τ_{n_e}, and explicit transition times, $\tau_1, \ldots, \tau_{n_e-1}$.

2. An invariant structure system where the number of real valued state and control variables are constant in each mode, $V = \{\mathbf{x}, \mathbf{u}, \mathbf{p}, t\}$, where $\mathbf{p} \in P \subset \mathbb{R}^{n_p}$, $\mathbf{x}(\mathbf{p}, t) \in \mathbb{R}^{n_x}$ and $\mathbf{u}(\mathbf{p}, t) \in U \subset \mathbb{R}^{n_u}$ for all $(\mathbf{p}, t) \in P \times I_i$, $i = 1, \ldots, n_e$.

3. The parameterization of the bounded real valued controls

$$\mathbf{u}(\mathbf{p}, t) = \mathbf{S}(t)\mathbf{p} + \mathbf{v}(t), \tag{2.1}$$
$$\mathbf{u}^L(t) \leq \mathbf{u}(\mathbf{p}, t) \leq \mathbf{u}^U(t), \quad \forall t \in [\sigma_1, \tau_{n_e}],$$

   where $\mathbf{u}^L(t)$ and $\mathbf{u}^U(t)$ are known lower and upper bounds on the controls $\mathbf{u}(\mathbf{p}, t)$ that define the set U, $\mathbf{S}(t)$ and $\mathbf{v}(t)$ are piecewise continuous on $[\sigma_1, \tau_{n_e}]$ and defined at any point of discontinuity.

4. The LTV ODE system in each mode $m \in M$, which is described by

$$\dot{\mathbf{x}}(\mathbf{p}, t) = \mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{p}, t) + \tilde{\mathbf{B}}^{(m)}(t)\mathbf{u}(\mathbf{p}, t) + \mathbf{C}^{(m)}(t)\mathbf{p} + \tilde{\mathbf{q}}^{(m)}(t),$$

   where $\mathbf{A}^{(m)}(t)$ is continuous on $[\sigma_1, \tau_{n_e}]$; $\tilde{\mathbf{B}}^{(m)}(t)$, $\mathbf{C}^{(m)}(t)$ and $\tilde{\mathbf{q}}^{(m)}(t)$ are piecewise continuous on $[\sigma_1, \tau_{n_e}]$, and defined at any point of discontinuity, for all $m \in M$. After control parameterization (substitute Equation (2.1)), we have

$$\dot{\mathbf{x}}(\mathbf{p}, t) = \mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{p}, t) + \mathbf{B}^{(m)}(t)\mathbf{p} + \mathbf{q}^{(m)}(t), \tag{2.2}$$

where $\mathbf{B}^{(m)}(t) \equiv \tilde{\mathbf{B}}^{(m)}(t)\mathbf{S}(t) + \mathbf{C}^{(m)}(t)$ and $\mathbf{q}^{(m)}(t) \equiv \tilde{\mathbf{B}}^{(m)}(t)\mathbf{v}(t) + \tilde{\mathbf{q}}^{(m)}(t)$ are piecewise continuous on $[\sigma_1, \tau_{n_e}]$, and defined at any point of discontinuity, for all $m \in M$.

5. The transition conditions for the transitions between epochs I_i and I_{i+1}, for all $i = 1, \ldots, n_e - 1$, which are trivial since all events are explicit time events:

$$L^{(m_i)} := (t \geq \tau_i),$$

indicating the transition from mode m_i in epoch I_i to mode m_{i+1} in epoch I_{i+1} at time τ_i.

6. The collection of transition functions, which is given by the following equation,

$$\mathbf{x}(\mathbf{p}, \sigma_{i+1}) = \mathbf{D}_i\mathbf{x}(\mathbf{p}, \tau_i) + \mathbf{E}_i\mathbf{p} + \mathbf{k}_i, \ \forall i = 1, \ldots, n_e - 1, \tag{2.3}$$

for the transition from mode m_i in epoch I_i to mode m_{i+1} in epoch I_{i+1}.

7. A given initial condition for mode m_1,

$$\mathbf{x}(\mathbf{p}, \sigma_1) = \mathbf{E}_0\mathbf{p} + \mathbf{k}_0. \tag{2.4}$$

A solution, $\mathbf{x}(\mathbf{p}, t)$, $t \in I_i$, $i = 1, \ldots, n_e$, will exist and be unique for all $\mathbf{p} \in P$, at least in the weak or extended sense (this follows from [42], see e.g., Theorem 4.4). Note that the control parameterization in (2.1) can be used to approximate the controls with piecewise Lagrange polynomials of arbitrary order, which includes piecewise constant and piecewise linear controls. The advantage of using Lagrange polynomials as the basis functions lies in the straightforward translation of the natural bounds on the controls, $\mathbf{u}$, to bounds on the parameters, $\mathbf{p}$, since the coefficients of the Lagrange polynomials correspond to values of the controls at specific points in time. In most cases, this results in the Euclidean parameter space P being a hyper-rectangle, ensuring its compactness.

Definition 2.5. Let P be a nonempty compact convex subset of $\mathbb{R}^{n_p}$. We define the following sets for all $i = 1, \ldots, n_e$:

$$\mathcal{X}^{(i)}(t; P) \equiv \big\{\mathbf{x}(\mathbf{p}, t) \mid \mathbf{p} \in P\big\},\ \forall t \in I_i, \qquad \mathcal{X}^{(i)}(P) \equiv \bigcup_{t \in I_i} \mathcal{X}^{(i)}(t; P),$$

$$\dot{\mathcal{X}}^{(i)}(t; P) \equiv \big\{\dot{\mathbf{x}}(\mathbf{p}, t) \mid \mathbf{p} \in P\big\},\ \forall t \in I_i, \qquad \dot{\mathcal{X}}^{(i)}(P) \equiv \bigcup_{t \in I_i} \dot{\mathcal{X}}^{(i)}(t; P).$$

Before we proceed, for convenience, we will reproduce the definition of a stationary simple discontinuity in an integrable function [120, Definition 2.1]:

Definition 2.6. Let $Z \subset \mathbb{R}^d$ and $f : [t_0, t_f] \times Z \to \mathbb{R}$ be an integrable function. Then, f is said to have a finite number of stationary simple discontinuities in t if the following conditions hold:

1. For each $\underline{t}$ fixed in $[t_0, t_f]$, $f(\underline{t}, \mathbf{z})$ is continuous on Z.

2. For each $\underline{z}$ fixed in Z, $f(t, \underline{\mathbf{z}})$ possesses at most a finite number of simple discontinuities. Additionally, $f(t, \underline{\mathbf{z}})$ must be defined at any point of discontinuity.

2.4 Problem Formulation

Problem 2.7. Consider the following problem:

$$\min_{\mathbf{p} \in P}\ F(\mathbf{p}) \equiv \sum_{i=1}^{n_e} \left\{ \sum_{j=1}^{n_{\phi i}} \phi_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}, \hat{t}_{ij}), \mathbf{x}(\mathbf{p}, \hat{t}_{ij}), \mathbf{p}\Big) + \int_{\sigma_i}^{\tau_i} f_i\big(\dot{\mathbf{x}}(\mathbf{p}, t), \mathbf{x}(\mathbf{p}, t), \mathbf{p}, t\big)\, \mathrm{d}t \right\},$$

subject to the following point and isoperimetric constraints,

$$\mathbf{G}(\mathbf{p}) \equiv \sum_{i=1}^{n_e} \left\{ \sum_{j=1}^{n_{\eta i}} \boldsymbol{\eta}_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}, \check{t}_{ij}), \mathbf{x}(\mathbf{p}, \check{t}_{ij}), \mathbf{p}\Big) + \int_{\sigma_i}^{\tau_i} \mathbf{g}_i\big(\dot{\mathbf{x}}(\mathbf{p}, t), \mathbf{x}(\mathbf{p}, t), \mathbf{p}, t\big)\, \mathrm{d}t \right\} \ \leq\ \mathbf{0},$$

where $\mathbf{x}(\mathbf{p}, t)$ is given by the solution of the embedded LTV ODE hybrid system (Definitions 2.4 and 2.5); $f_i : \dot{\mathcal{X}}^{(i)}(P) \times \mathcal{X}^{(i)}(P) \times P \times I_i \to \mathbb{R}$ and $\mathbf{g}_i : \dot{\mathcal{X}}^{(i)}(P) \times \mathcal{X}^{(i)}(P) \times P \times I_i \to \mathbb{R}^{n_c}$ are piecewise continuous for all $i = 1, \ldots, n_e$, where only a finite

number of stationary simple discontinuities are allowed; $n_{\phi i}$ is an arbitrary number of point objectives in epoch I_i, $\hat{t}_{ij} \in I_i$ and $\phi_{ij} : \dot{\mathcal{X}}^{(i)}(\hat{t}_{ij}; P) \times \mathcal{X}^{(i)}(\hat{t}_{ij}; P) \times P \to \mathbb{R}$ is continuous for all $j = 1, \ldots, n_{\phi i}$ and $i = 1, \ldots, n_e$; and $n_{\eta i}$ is an arbitrary number of point functions in epoch I_i, $\check{t}_{ij} \in I_i$ and $\boldsymbol{\eta}_{ij} : \dot{\mathcal{X}}^{(i)}(\check{t}_{ij}; P) \times \mathcal{X}^{(i)}(\check{t}_{ij}; P) \times P \to \mathbb{R}^{n_c}$ is continuous for all $j = 1, \ldots, n_{\eta i}$ and $i = 1, \ldots, n_e$. Additionally, we require that the set $G = \{\mathbf{p} \in P \mid \mathbf{G}(\mathbf{p}) \leq \mathbf{0}\}$ is nonempty.

In general, testing feasibility for the set G for an arbitrary dynamic optimization problem is a non-trivial problem. However, for most well-posed engineering problems, the existence of a feasible solution is often known a priori, so the assumption is not a strong one. This assumption allows one to write min in the problem statement, as is the convention for optimization problems, instead of inf. It is interesting to note that given some optimality tolerance, the application of Algorithm 2.3 guarantees that either a solution will be found, or the algorithm will terminate with an indication that the problem is infeasible. In that case, the requirement that $G \neq \emptyset$ can be relaxed.

2.5 Solution Strategy

The BB algorithm presented in Algorithm 2.3 will be employed to obtain a global solution, within ε tolerance, of Problem 2.7 with a finite number of iterations. The hybrid system described in Definition 2.4 is embedded in the optimization problem, reducing the otherwise infinite dimensional search space (containing $\mathbf{x} \in (\hat{C}^1[\sigma_i, \tau_i])^{n_x},\ i = 1, \ldots, n_e$) to a finite dimensional one on the parameter set P. The upper bounding problem is solved using any local gradient based method, utilizing the parametric sensitivities of the hybrid system.

Theorem 2.8. *The parametric sensitivities of the LTV ODE hybrid system (Definition 2.4) exist (at least in the weak sense),* $\mathbf{S}(\mathbf{p}, t) \equiv \left.\frac{\partial \mathbf{x}}{\partial \mathbf{p}}\right|_{\mathbf{p},t} \in \mathbb{R}^{n_x \times n_p}$, $t \in I_i$,

$i = 1, \ldots, n_e$ for all $\mathbf{p} \in P$, and are given by the solution of the following equations:

$$\dot{\mathbf{S}} = \mathbf{A}^{(m_i)}(t)\mathbf{S} + \mathbf{B}^{(m_i)}(t), \ \forall t \in (\sigma_i, \tau_i], \ i = 1, \ldots, n_e, \tag{2.5}$$

$$\mathbf{S}(\mathbf{p}, \sigma_0) = \mathbf{E}_0, \tag{2.6}$$

$$\mathbf{S}(\mathbf{p}, \sigma_{i+1}) = \mathbf{D}_i\mathbf{S}(\mathbf{p}, \tau_i) + \mathbf{E}_i, \quad i = 1, \ldots, n_e - 1. \tag{2.7}$$

Proof. Consider an arbitrary epoch $i \in \{1, \ldots, n_e\}$, where $\sigma_i < \tau_i$. We have a finite number of discontinuities (in time) in $\mathbf{B}^{(m_i)}(t)$ and $\mathbf{q}^{(m_i)}(t)$. Let there be k such discontinuities in (σ_i, τ_i) each found at points $t = \lambda_j$ where $\lambda_j \in (\sigma_i, \tau_i)$ for $j \in \{1, \ldots, k\}$. Construct a sequence of *sub-epochs* $[\theta_1, \lambda_1], [\theta_2, \lambda_2], \ldots, [\theta_{k+1}, \lambda_{k+1}]$ where $\theta_1 = \sigma_i$, $\lambda_{k+1} = \tau_i$; $\theta_j < \lambda_j$ for $j = 1, \ldots, k+1$ and $\lambda_j = \theta_{j+1}$ for $j = 1, \ldots, k$. Extend the functions $\mathbf{B}^{(m_i)}(t)$ and $\mathbf{q}^{(m_i)}(t)$ to be continuous on $[\theta_j, \lambda_j]$ for all $j = 1, \ldots, k+1$:

$$\mathbf{B}^{(m_i)}(\theta_j) \equiv \lim_{t \to \theta_j^+} \mathbf{B}^{(m_i)}(t); \quad \mathbf{q}^{(m_i)}(\theta_j) \equiv \lim_{t \to \theta_j^+} \mathbf{q}^{(m_i)}(t),$$

$$\mathbf{B}^{(m_i)}(\lambda_j) \equiv \lim_{t \to \lambda_j^-} \mathbf{B}^{(m_i)}(t); \quad \mathbf{q}^{(m_i)}(\lambda_j) \equiv \lim_{t \to \lambda_j^-} \mathbf{q}^{(m_i)}(t).$$

At the transitions between sub-epochs, impose state continuity as the system of transition functions. In this way, we have defined a hybrid system within the chosen epoch. The form of the LTV ODE system in each sub-epoch is given by (2.2). Let $\mathbf{F}^{(m_i)} = \dot{\mathbf{x}}$, and consider an arbitrary sub-epoch, $[\theta_j, \lambda_j]$. It is clear that the partial derivatives $\frac{\partial \mathbf{F}^{(m_i)}}{\partial \mathbf{x}} = \mathbf{A}^{(m_i)}$ and $\frac{\partial \mathbf{F}^{(m_i)}}{\partial \mathbf{p}} = \mathbf{B}^{(m_i)}$ exist and are continuous. Without loss of generality, assume that $P \subset \mathbb{R}^{n_p}$ is a closed hyper-rectangle. It follows that we can construct an extended, bounded, open set $P^o \subset \mathbb{R}^{n_p}$ such that $P \subset P^o$ by subtracting and adding some small $\varepsilon > 0$ to the bounds on P. At the transition λ_j, consider the following system of equations,

$$\mathbf{h}\big(\mathbf{x}(\mathbf{p}, \lambda_j), \dot{\mathbf{x}}(\mathbf{p}, \lambda_j), \mathbf{x}(\mathbf{p}, \theta_{j+1}), \dot{\mathbf{x}}(\mathbf{p}, \theta_{j+1}); \mathbf{p}\big) = \mathbf{0},$$

where $\mathbf{h} : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P^o \to \mathbb{R}^{4n_x}$ is described by

$$\begin{bmatrix} \mathbf{x}(\mathbf{p}, \lambda_j) - \mathbf{x}(\mathbf{p}, \theta_j) - \int_{\theta_j}^{\lambda_j} \mathbf{A}^{(m_i)}(t)\mathbf{x}(\mathbf{p}, t) + \mathbf{B}^{(m_i)}(t)\mathbf{p} + \mathbf{q}^{(m_i)}(t)\ \mathrm{d}t \\ \dot{\mathbf{x}}(\mathbf{p}, \lambda_j) - \mathbf{A}^{(m_i)}(\lambda_j)\mathbf{x}(\mathbf{p}, \lambda_j) - \mathbf{B}^{(m_i)}(\lambda_j)\mathbf{p} - \mathbf{q}^{(m_i)}(\lambda_j) \\ \mathbf{x}(\mathbf{p}, \lambda_j) - \mathbf{x}(\mathbf{p}, \theta_{j+1}) \\ \dot{\mathbf{x}}(\mathbf{p}, \theta_{j+1}) - \mathbf{A}^{(m_i)}(\theta_{j+1})\mathbf{x}(\mathbf{p}, \theta_{j+1}) - \mathbf{B}^{(m_i)}(\theta_{j+1})\mathbf{p} - \mathbf{q}^{(m_i)}(\theta_{j+1}) \end{bmatrix}.$$

It is clear that the set $E = \mathbb{R}^{4n_x} \times P^o$ is an open set such that $E \subset \mathbb{R}^{4n_x+n_p}$. Consider the following $4n_x \times 4n_x$ submatrix of the Jacobian matrix of $\mathbf{h}$ corresponding to the variables $\mathbf{x}(\mathbf{p}, \lambda_j), \dot{\mathbf{x}}(\mathbf{p}, \lambda_j), \mathbf{x}(\mathbf{p}, \theta_{j+1})$, and $\dot{\mathbf{x}}(\mathbf{p}, \theta_{j+1})$:

$$\mathbf{J} = \begin{bmatrix} \mathbf{I} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ -\mathbf{A}^{(m_i)}(\lambda_j) & \mathbf{I} & \mathbf{0} & \mathbf{0} \\ \mathbf{I} & \mathbf{0} & -\mathbf{I} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & -\mathbf{A}^{(m_i)}(\theta_{j+1}) & \mathbf{I} \end{bmatrix}.$$

$\mathbf{J}$ is clearly invertible. We can then apply [63, Theorem 1] to obtain the existence result. Since the transitions between the sub-epochs do not depend on $\mathbf{p}$, and we have state continuity, it follows from [63, Eq. (57)] that the parametric sensitivities are continuous across the sub-epochs. Equation (2.5) then follows from [63, Eq. (48)]. For the parametric sensitivities at the epoch boundaries, (2.7) follows from a direct application of [63, Eq. (55)]. To complete the proof, we only have to consider the case where we have an instantaneous epoch (i.e., $\sigma_i = \tau_i$). In that case, [63, Theorem 1] is trivially satisfied and the parametric sensitivities will be given by (2.7). □

Note that the sensitivities can be nonsmooth in time due to the presence of the piecewise continuous term $\mathbf{B}^{(m_i)}(t)$, but are continuous internal to an epoch. At the transitions, the sensitivities of the predecessor and successor modes are related by (2.7). Hence, if $\mathbf{D}_i = \mathbf{I}$ and $\mathbf{E}_i = \mathbf{0}$ (of which state continuity is a common case with $\mathbf{k}_i = \mathbf{0}$ in (2.3)), the sensitivities are continuous across the transitions. In addition, in order to use a gradient based algorithm, we require the following sufficient conditions

on the smoothness of the objective function:

Theorem 2.9. *Let $P^o \supset P$, $\mathcal{X}^{(i)o}(\hat{t}_{ij}) \supset \mathcal{X}^{(i)}(\hat{t}_{ij}; P^o)$, $\mathcal{X}^{(i)o} \supset \mathcal{X}^{(i)}(P^o)$, $\dot{\mathcal{X}}^{(i)o}(\hat{t}_{ij}) \supset \dot{\mathcal{X}}^{(i)}(\hat{t}_{ij}; P^o)$ and $\dot{\mathcal{X}}^{(i)o} \supset \dot{\mathcal{X}}^{(i)}(P^o)$ be open subsets of $\mathbb{R}^{n_p}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$ and $\mathbb{R}^{n_x}$ respectively, for all $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$. If the following conditions are satisfied, then the objective function $F(\mathbf{p})$ in Problem 2.7 is continuously differentiable on P^o.*

1. *$\frac{\partial \phi_{ij}}{\partial \dot{\mathbf{x}}}$, $\frac{\partial \phi_{ij}}{\partial \mathbf{x}}$ and $\frac{\partial \phi_{ij}}{\partial \mathbf{p}}$ exist, and are continuous on $\dot{\mathcal{X}}^{(i)o}(\hat{t}_{ij}) \times \mathcal{X}^{(i)o}(\hat{t}_{ij}) \times P^o$ for all $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$.*

2. *$\frac{\partial f_i}{\partial \dot{\mathbf{x}}}$, $\frac{\partial f_i}{\partial \mathbf{x}}$ and $\frac{\partial f_i}{\partial \mathbf{p}}$ are piecewise continuous on $\dot{\mathcal{X}}^{(i)o} \times \mathcal{X}^{(i)o} \times P^o \times I_i$ for all $i = 1, \ldots, n_e$ where only a finite number of stationary simple discontinuities are allowed.*

Proof. Consider any arbitrary epoch I_i, and any arbitrary $k \in \{1, \ldots, n_p\}$. First, consider the point objectives. Taking the partial derivative with respect to p_k and applying the chain rule, we have

$$\sum_{j=1}^{n_{\phi i}} \frac{\partial \phi_{ij}}{\partial p_k} = \sum_{j=1}^{n_{\phi i}} \left\{ \frac{\partial \phi_{ij}}{\partial \dot{\mathbf{x}}} \frac{\partial \dot{\mathbf{x}}}{\partial p_k} + \frac{\partial \phi_{ij}}{\partial \mathbf{x}} \frac{\partial \mathbf{x}}{\partial p_k} + \frac{\partial \phi_{ij}}{\partial p_k} \right\},$$

which clearly exists and is continuous on P by Theorem 2.8 and condition 1. Next, consider the integral objectives. We have a finite number of discontinuities (in time) in f_i, $\mathbf{B}^{(m)}$, $\mathbf{q}^{(m)}$, the parametric sensitivities (2.5) and the partial derivatives in condition 2. Let there be l such discontinuities each found at points $t = \lambda_j$ where $\lambda_j \in (\sigma_i, \tau_i)$ for $j \in \{1, \ldots, l\}$ and $\sigma_i < \tau_i$, since there is nothing to prove if the epoch is instantaneous. Construct a sequence of sub-epochs $[\theta_1, \lambda_1], [\theta_2, \lambda_2], \ldots, [\theta_{l+1}, \lambda_{l+1}]$ where $\theta_1 = \sigma_i$, $\lambda_{l+1} = \tau_i$; $\theta_j < \lambda_j$ for $j = 1, \ldots, l+1$ and $\lambda_j = \theta_{j+1}$ for $j = 1, \ldots, l$.

Partition the integral into the following:

$$F_i(\mathbf{p}) = \int_{\sigma_i}^{\tau_i} f_i\big(\dot{\mathbf{x}}(\mathbf{p},t), \mathbf{x}(\mathbf{p},t), \mathbf{p}, t\big)\,\mathrm{d}t = \sum_{j=1}^{l+1} F_{ij}(\mathbf{p})$$
$$= \sum_{j=1}^{l+1} \int_{\theta_j}^{\lambda_j} f_i\big(\dot{\mathbf{x}}(\mathbf{p},t), \mathbf{x}(\mathbf{p},t), \mathbf{p}, t\big)\,\mathrm{d}t. \quad (2.8)$$

Now, choose any arbitrary $j \in \{1, \ldots, l+1\}$. Extend f_i, $\frac{\partial f_i}{\partial \dot{\mathbf{x}}}$, $\frac{\partial f_i}{\partial \mathbf{x}}$ and $\frac{\partial f_i}{\partial \mathbf{p}}$ to be continuous on $\dot{\mathcal{X}}^{(i)o} \times \mathcal{X}^{(i)o} \times P^o \times [\theta_j, \lambda_j]$, and $\frac{\partial \dot{\mathbf{x}}}{\partial \mathbf{p}}$, $\frac{\partial \mathbf{x}}{\partial \mathbf{p}}$, $\dot{\mathbf{x}}$ and $\mathbf{x}$ to be continuous on $P^o \times [\theta_j, \lambda_j]$. At most, these functions are discontinuous at their endpoints in time. Removing these discontinuities does not alter the value of the integral because the endpoints comprise a set of measure zero. Applying the chain rule, we have

$$\frac{\partial f_i}{\partial p_k} = \frac{\partial f_i}{\partial \dot{\mathbf{x}}}\frac{\partial \dot{\mathbf{x}}}{\partial p_k} + \frac{\partial f_i}{\partial \mathbf{x}}\frac{\partial \mathbf{x}}{\partial p_k} + \frac{\partial f_i}{\partial p_k},$$

which is continuous on $\dot{\mathcal{X}}^{(i)o} \times \mathcal{X}^{(i)o} \times P^o \times [\theta_j, \lambda_j]$. These continuity conditions enable us to differentiate under the integral sign [43, Page 308] to obtain

$$\frac{\partial F_{ij}}{\partial p_k} = \int_{\theta_j}^{\lambda_j} \frac{\partial f_i}{\partial p_k}\,\mathrm{d}t.$$

We can then apply [120, Proposition 2.1] to yield $\frac{\partial F_{ij}}{\partial p_k}$ continuous on P^o. Since j was arbitrary, $\frac{\partial F_i}{\partial p_k}$ is continuous on P^o as the sum of continuous functions is continuous. Since i was arbitrary, $\frac{\partial F}{\partial p_k}$ is continuous on P^o. Since k was arbitrary, $\frac{\partial F}{\partial p_k}$ is continuous for all $k \in \{1, \ldots, n_p\}$ and it follows that F is continuously differentiable on P^o. □

The next theorem shows that there exists a global minimum to Problem 2.7.

Theorem 2.10. *If the sufficient conditions in Theorem 2.9 are satisfied, then a minimum exists for Problem 2.7.*

Proof. From Theorem 2.9, F is continuous on P. The existence of the minimum then follows from [120, Corollary 2.1]. □

To generate rigorous lower bounds in a BB algorithm such as Algorithm 2.3, we

need convex relaxations for the Bolza type objective $F(\mathbf{p})$, which will be constructed as an extension of the theory developed in [120] in the following section. Once we have obtained the convex relaxation for the objective, we can solve the resulting convex underestimating problem globally to obtain a lower bound on the solution using any suitable gradient based algorithm which is also subject to Theorem 2.9 and Theorem 2.10.

Before we end this section, we shall illustrate what happens to a NLP solver in practice when it is given a nonsmooth problem.

Problem 2.11. Consider the following problem,

$$\min_{\mathbf{p}\in[-10,10]^2} \max(0.01(p_1 - p_2), 3p_1 + 5p_2).$$

Note that this problem is convex because the maximum of two convex functions is also convex (in this case, we have the maximum of two hyperplanes). If we feed this problem with initial guess of $\mathbf{p} = (0, 0)$ to the solver SNOPT version 6.1 with default settings [66], the solver returns $\mathbf{p} = (0, 0)$ for an objective value of 0 with a message that the current point cannot be improved. If we feed a different initial guess of (5,-5), the solver again returns $\mathbf{p} = (0, 0)$ for an objective value of 0 with a message that the current point cannot be improved. As can be seen, the solver is having problems with a convex objective function when it is nonsmooth. The problem can be reformulated into the following smooth problem,

$$\begin{aligned} \min_{\mathbf{p}\in[-10,10]^2, z\in[-10^{20},10^{20}]} \quad & z \\ \text{s.t.} \quad & z \geq 0.01(p_1 - p_2), \\ & z \geq 3p_1 + 5p_2. \end{aligned}$$

When this smooth problem is solved with the same solver with an initial guess of $\mathbf{p} = (0, 0)$ and $z = 0$, the solver returns the correct optimal solution of $z = -0.1597$ at $\mathbf{p} = (-10, 5.97)$. This simple example illustrates that even with a convex problem, gradient based NLP solvers cannot perform robustly when the problem involves

nonsmooth functions.

2.6 Constructing Convex Relaxations

In this section, we will show how the aforementioned methods for constructing convex relaxations of functionals on Euclidean spaces can be harnessed to construct convex relaxations of the Bolza type functionals $F(\mathbf{p})$ introduced in the problem formulation of Section 2.4.

Theorem 2.12. *Consider the function F as defined by Problem 2.7. If $f_i(\cdot, \underline{t})$ is convex on $\dot{\mathcal{X}}^{(i)}(\underline{t}; P) \times \mathcal{X}^{(i)}(\underline{t}; P) \times P$ for all $\underline{t} \in [\sigma_i, \tau_i]$, $i = 1, \ldots, n_e$, and ϕ_{ij} is convex on $\dot{\mathcal{X}}^{(i)}(\hat{t}_{ij}; P) \times \mathcal{X}^{(i)}(\hat{t}_{ij}; P) \times P$ for all $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$, then F is convex on P.*

Proof. As explained in [120, Equation (3)], the structural form of the solution to the LTV ODE in the first mode m_1 is an affine function of $\mathbf{p}$:

$$\mathbf{x}(\mathbf{p}, t) = \mathbf{M}_1(t)\mathbf{p} + \mathbf{n}_1(t), \quad t \in [\sigma_1, \tau_1].$$

Applying the appropriate transition functions at τ_1, we obtain the *initial conditions* for mode m_2 as

$$\mathbf{x}(\mathbf{p}, \sigma_2) = \big(\mathbf{D}_1\mathbf{M}_1(\tau_1) + \mathbf{E}_1\big)\mathbf{p} + \mathbf{D}_1\mathbf{n}_1(\tau_1) + \mathbf{k}_1,$$

which is an affine function of $\mathbf{p}$ satisfying the condition in [120, Theorem 3.1]. By induction, we have

$$\mathbf{x}(\mathbf{p}, t) = \mathbf{M}_i(t)\mathbf{p} + \mathbf{n}_i(t), \quad t \in [\sigma_i, \tau_i], \ \forall i = 1, \ldots, n_e, \tag{2.9}$$

$$\mathbf{x}(\mathbf{p}, \sigma_{i+1}) = \big(\mathbf{D}_i\mathbf{M}_i(\tau_i) + \mathbf{E}_i\big)\mathbf{p} + \mathbf{D}_i\mathbf{n}_i(\tau_i) + \mathbf{k}_i, \ \forall i = 1, \ldots, n_e - 1. \tag{2.10}$$

From (2.9), [113, Theorems 3.1, 3.4, 3.5] and Definition 2.5, it follows that the set $\dot{\mathcal{X}}^{(i)}(\underline{t}; P) \times \mathcal{X}^{(i)}(\underline{t}; P) \times P$ is convex for all $\underline{t} \in [\sigma_i, \tau_i]$, $i = 1, \ldots, n_e$. Hence, we

can apply [120, Lemma 3.1] to obtain ϕ_{ij} is convex on P for all $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$. Next, consider any arbitrary epoch I_i, for any $i \in \{1, \ldots, n_e\}$, and its integral objective function with initial condition given by (2.10). We can apply [120, Theorem 3.1] to obtain F_i (defined in (2.8)) convex on P. Since i was arbitrary, we have F_i is convex on P for all $i = 1, \ldots, n_e$, from which it follows that F is convex on P. □

Corollary 2.13. *Consider the following function:*

$$U(\mathbf{p}) = \sum_{i=1}^{n_e} \left\{ \sum_{j=1}^{n_{\phi i}} \psi_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}, \hat{t}_{ij}), \mathbf{x}(\mathbf{p}, \hat{t}_{ij}), \mathbf{p}\Big) + \int_{\sigma_i}^{\tau_i} u_i\big(\dot{\mathbf{x}}(\mathbf{p}, t), \mathbf{x}(\mathbf{p}, t), \mathbf{p}, t\big)\, \mathrm{d}t \right\}$$

subject to the conditions of Problem 2.7, where u_i is a piecewise continuous mapping $u_i : \dot{\mathcal{X}}^{(i)}(P) \times \mathcal{X}^{(i)}(P) \times P \times [\sigma_i, \tau_i] \to \mathbb{R}$ for all $i = 1, \ldots, n_e$ where only a finite number of stationary simple discontinuities are allowed, and ψ_{ij} is a continuous mapping $\psi_{ij} : \dot{\mathcal{X}}^{(i)}(\hat{t}_{ij}; P) \times \mathcal{X}^{(i)}(\hat{t}_{ij}; P) \times P \to \mathbb{R}$ for all $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$. If, for all $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$, we have

$$\psi_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}, \hat{t}_{ij}), \mathbf{x}(\mathbf{p}, \hat{t}_{ij}), \mathbf{p}\Big) \leq \phi_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}, \hat{t}_{ij}), \mathbf{x}(\mathbf{p}, \hat{t}_{ij}), \mathbf{p}\Big), \ \forall \mathbf{p} \in P, \tag{2.11}$$

$$u_i\big(\dot{\mathbf{x}}(\mathbf{p}, t), \mathbf{x}(\mathbf{p}, t), \mathbf{p}, t\big) \leq f_i\big(\dot{\mathbf{x}}(\mathbf{p}, t), \mathbf{x}(\mathbf{p}, t), \mathbf{p}, t\big), \ \forall (\mathbf{p}, t) \in P \times [\sigma_i, \tau_i], \tag{2.12}$$

$u_i\big(\cdot, \underline{t}\big)$ is convex on $\dot{\mathcal{X}}^{(i)}(\underline{t}; P) \times \mathcal{X}^{(i)}(\underline{t}; P) \times P \quad \forall \underline{t} \in [\sigma_i, \tau_i]$, and ψ_{ij} is convex on $\dot{\mathcal{X}}^{(i)}(\hat{t}_{ij}; P) \times \mathcal{X}^{(i)}(\hat{t}_{ij}; P) \times P$, then U is convex on P such that $U(\mathbf{p}) \leq F(\mathbf{p}), \forall \mathbf{p} \in P$.

Proof. The proof is clear from Theorem 2.12 and [120, Lemma 3.2]. □

Differentiation of (2.9) with respect to $\mathbf{p}$ reveals that $\mathbf{M}_i$ are the parametric sensitivities of the hybrid system, which are important in obtaining the implied state bounds of the embedded linear hybrid system (see below). Corollary 2.13 is especially useful because the aforementioned methods for constructing convex relaxations on Euclidean spaces [96, 1] can be harnessed to construct the relevant convex relaxations, ψ_{ij} from ϕ_{ij} in (2.11) and $u_i(\cdot, \underline{t})$ from $f_i(\cdot, \underline{t})$ for all $\underline{t} \in T$ in (2.12). This then allows Algorithm 2.3 to be applied to solve Problem 2.7. The next step is to

show the convergence properties of the algorithm when employing convex relaxations constructed from Corollary 2.13.

As mentioned in Section 2.1.1, the selection operation in Algorithm 2.9 is bound improving by definition. Hence, in order to show that it will be infinitely convergent (from [77, Theorem IV.3]), we will have to show that the bounding operation is consistent when Corollary 2.13 is used to construct the convex relaxation of Problem 2.7. This is what we will seek to establish in the following section.

2.7 Implied State Bounds for LTV Hybrid Systems

In order to establish the consistency of the bounding operation, we first have to bound the solution of the embedded linear hybrid system. The results in this section are important, and will be revisited in subsequent chapters.

Theorem 2.14. *Consider Problem 2.7. If* $\mathbf{p} \in P = [\mathbf{p}^L, \mathbf{p}^U]$*, then the implied state bounds for the real valued state variables are given pointwise in time by the following natural interval extension [99] of* (2.9)*:*

$$[\mathbf{x}]([\mathbf{p}], t) = \mathbf{M}_i(t)[\mathbf{p}] + \mathbf{n}_i(t), \ \forall t \in [\sigma_i, \tau_i], \quad \forall i = 1, \ldots, n_e. \tag{2.13}$$

Further, the implied bounds for $\dot{\mathbf{x}}(\mathbf{p}, t)$ *are given pointwise in time by the following interval equation:*

$$[\dot{\mathbf{x}}]([\mathbf{p}], t) = \left(\mathbf{A}^{(m_i)}(t)\mathbf{M}_i(t) + \mathbf{B}^{(m_i)}(t)\right)[\mathbf{p}] + \mathbf{A}^{(m_i)}(t)\mathbf{n}_i(t) + \mathbf{q}^{(m_i)}(t),$$
$$\forall t \in [\sigma_i, \tau_i], \quad \forall i = 1, \ldots, n_e. \tag{2.14}$$

Proof. Apply [120, Theorem 4.1 and Corollary 4.1] to each mode m_i for $i = 1, \ldots, n_e$. □

Applying interval arithmetic [99] to (2.13) and (2.14), the following equations are

obtained for all $t \in [\sigma_i, \tau_i]$, $j = 1, \ldots, n_x$, $i = 1, \ldots, n_e$,

$$x_j^L(t) = n_j^i(t) + \sum_{k=1}^{n_p} \min\left\{m_{jk}^i(t)p_k^L, m_{jk}^i(t)p_k^U\right\},$$
$$x_j^U(t) = n_j^i(t) + \sum_{k=1}^{n_p} \max\left\{m_{jk}^i(t)p_k^L, m_{jk}^i(t)p_k^U\right\},$$
$$\dot{x}_j^L(t) = q_j^{(m_i)}(t) + \sum_{k=1}^{n_x} a_{jk}^{(m_i)}(t)n_j^i(t) + \sum_{k=1}^{n_p} \min\left\{z_{jk}^{(m_i)}(t)p_k^L, z_{jk}^{(m_i)}(t)p_k^U\right\},$$
$$\dot{x}_j^U(t) = q_j^{(m_i)}(t) + \sum_{k=1}^{n_x} a_{jk}^{(m_i)}(t)n_j^i(t) + \sum_{k=1}^{n_p} \max\left\{z_{jk}^{(m_i)}(t)p_k^L, z_{jk}^{(m_i)}(t)p_k^U\right\},$$

where $m_{jk}^i(t)$ is the (j,k)th element of $\mathbf{M}_i(t)$, $n_j^i(t)$ is the jth element of $\mathbf{n}_i(t)$, and $\mathbf{Z}^{(m_i)}(t) = \mathbf{A}^{(m_i)}(t)\mathbf{M}_i(t) + \mathbf{B}^{(m_i)}(t)$. From (2.2), (2.5), (2.4), (2.9) and (2.10), the vector $\mathbf{n}_i(t)$ is given by the following equations,

$$\dot{\mathbf{n}}_i(t) = \mathbf{A}^{(m_i)}(t)\mathbf{n}_i(t) + \mathbf{q}^{(m_i)}(t),\ \forall t \in (\sigma_i, \tau_i],\ i = 1, \ldots, n_e, \tag{2.15}$$
$$\mathbf{n}_1(\sigma_1) = \mathbf{k}_0,$$
$$\mathbf{n}_{i+1}(\sigma_{i+1}) = \mathbf{D}_i\mathbf{n}_i(\tau_i) + \mathbf{k}_i,\ \forall i = 1, \ldots, n_e - 1.$$

Theorem 2.15. *The implied state bounds $\mathbf{x}^L(t)$, $\mathbf{x}^U(t)$, $\dot{\mathbf{x}}^L(t)$ and $\dot{\mathbf{x}}^U(t)$ as determined from Theorem 2.14 are piecewise continuous on $[\sigma_1, \tau_{n_e}]$, and defined at any point of discontinuity.*

Proof. The proof follows from [120, Proposition 4.1 and Remark 4.1] and the fact that we have a finite number of epochs. □

Theorem 2.16. *The implied state bounds $\mathbf{x}^L(t)$ and $\mathbf{x}^U(t)$ as determined from Theorem 2.14 are* exact *in the following sense: For any $i \in \{1, \ldots, n_e\}$, $j \in \{1, \ldots, n_x\}$, $t \in [\sigma_i, \tau_i]$, the following relationship holds,*

$$x_j(\mathbf{p}^*, t) = x_j^L(t) \leq x_j(\mathbf{p}, t) \leq x_j^U(t) = x_j(\mathbf{p}^\dagger, t),\ \forall \mathbf{p} \in P,$$

for some $\mathbf{p}^*, \mathbf{p}^\dagger \in P$.

Proof. The proof is elementary from the application of interval arithmetic on the interval equation (2.13). □

Theorem 2.17. *Consider the function F and convex relaxation U as defined by Corollary 2.13 with implied state bounds defined by Theorem 2.14 subject to the conditions of Problem 2.7. If the constructed convex underestimators u_i and ψ_{ij} possess consistent bounding operations with monotonic convergence to f_i and ϕ_{ij} respectively, and the interval in any partition of P approaches degeneracy, then the lower bound in this partition converges pointwise to the upper bound of this same partition.*

Proof. Choose any partition and any fixed $\underline{t} \in [\sigma_i, \tau_i]$ for any $i \in \{1, \ldots, n_e\}$. As the interval $[\mathbf{p}^L, \mathbf{p}^U]$ approaches the degenerate value of $\mathbf{p}^*$, it follows from (2.13) and (2.14) that the intervals $[\mathbf{x}^L(\underline{t}), \mathbf{x}^U(\underline{t})]$ and $[\dot{\mathbf{x}}^L(\underline{t}), \dot{\mathbf{x}}^U(\underline{t})]$ respectively approach implied degenerate values $\mathbf{x}^*(\underline{t})$ and $\dot{\mathbf{x}}^*(\underline{t})$. Since ψ_{ij} is convex on $\dot{\mathcal{X}}^{(i)}(\hat{t}_{ij}; P) \times \mathcal{X}^{(i)}(\hat{t}_{ij}; P) \times P$, it is generated in the partition by the intervals $[\mathbf{p}^L, \mathbf{p}^U]$, $[\mathbf{x}^L(\hat{t}_{ij}), \mathbf{x}^U(\hat{t}_{ij})]$ and $[\dot{\mathbf{x}}^L(\hat{t}_{ij}), \dot{\mathbf{x}}^U(\hat{t}_{ij})]$. Suppose that at each step k, the interval $[\mathbf{p}^L, \mathbf{p}^U]_k$ is bisected (or partitioned in some other manner) such that as $k \to \infty$, $[\mathbf{p}^L, \mathbf{p}^U]_k \to \mathbf{p}^*$, which in turn implies $[\mathbf{x}^L(\hat{t}_{ij}), \mathbf{x}^U(\hat{t}_{ij})]_k \to \mathbf{x}^*(\hat{t}_{ij})$ and $[\dot{\mathbf{x}}^L(\hat{t}_{ij}), \dot{\mathbf{x}}^U(\hat{t}_{ij})]_k \to \dot{\mathbf{x}}^*(\hat{t}_{ij})$. Since this holds for arbitrary $\hat{t}_{ij} \in [\sigma_i, \tau_i]$ for any $i \in \{1, \ldots, n_e\}$, we have the following sequence:

$$(\psi_{ij})_k \ \uparrow \ \phi_{ij} \ \text{ as } \ k \to \infty, \ \forall j = 1, \ldots, n_{\phi i}, \ i = 1, \ldots, n_e$$

where the convergence arises because the bounds on ψ_{ij} are all approaching degeneracy and the underestimator $(\psi_{ij})_k$ for each step k is assumed to possess a consistent bounding operation with monotonic convergence to ϕ_{ij}. Since the partition was arbitrary, the convergence result is applicable to any partition. Applying [116, Theorem 3.3(a)], it is evident that

$$\sum_{i=1}^{n_e} \sum_{j=1}^{n_{\phi i}} \phi_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}^*, \hat{t}_{ij}), \mathbf{x}(\mathbf{p}^*, \hat{t}_{ij}), \mathbf{p}^*\Big) = \lim_{k \to \infty} \sum_{i=1}^{n_e} \sum_{j=1}^{n_{\phi i}} \psi_{ij}\Big(\dot{\mathbf{x}}(\mathbf{p}^*, \hat{t}_{ij}), \mathbf{x}(\mathbf{p}^*, \hat{t}_{ij}), \mathbf{p}^*\Big)_k.$$

The convergence of the integral term can be shown by considering the integral objec-

tive function in each epoch I_i with the following convex underestimator:

$$U_i(\mathbf{p})_k = \int_{\sigma_i}^{\tau_i} u_i\big(\dot{\mathbf{x}}(\mathbf{p},t), \mathbf{x}(\mathbf{p},t), \mathbf{p}, t\big)_k \,\mathrm{d}t, \tag{2.16}$$

and applying [120, Theorem 5.6], from which it follows that $F(\mathbf{p}^*) = \lim_{k\to\infty} U(\mathbf{p})_k$.

□

Theorem 2.18. *A BB algorithm such as Algorithm 2.3 utilizing the convex underestimators defined by Corollary 2.13 with implied state bounds defined by Theorem 2.14 is infinitely convergent.*

Proof. From Theorem 2.17, we have shown that the bounding operation is consistent. Since the selection operation is bound improving by definition, the algorithm is infinitely convergent by applying [77, Theorem IV.3]. □

Thus far, we have focused on constructing convex underestimators for the objective function. The exact same technique can be applied for the point and isoperimetric inequality constraints to construct convex relaxations of the feasible region. This will enable rigorous lower bounds to be obtained in the bounding step of the BB algorithm. The added advantage of expressing the constraints in their *canonical form* [126] is that the objective and all the constraint functions are treated the same way in as far as the computations of their values, relaxations and respective gradients are concerned in the numerical solution of the mathematical programming problem.

2.8 Illustrative Examples

Example 2.19. In this example, the implied state bounds $[\mathbf{x}^L(t), \mathbf{x}^U(t)]$ for the following hybrid system are obtained for $\mathbf{p} \in [-2,2]^2$ and $t \in [0,20]$,

$$\text{Mode 1} \;:\; \begin{cases} \dot{x}_1 = 0.1x_1 - x_2 + p_1 \\ \dot{x}_2 = x_1 + 0.1x_2 - p_2 \end{cases},$$

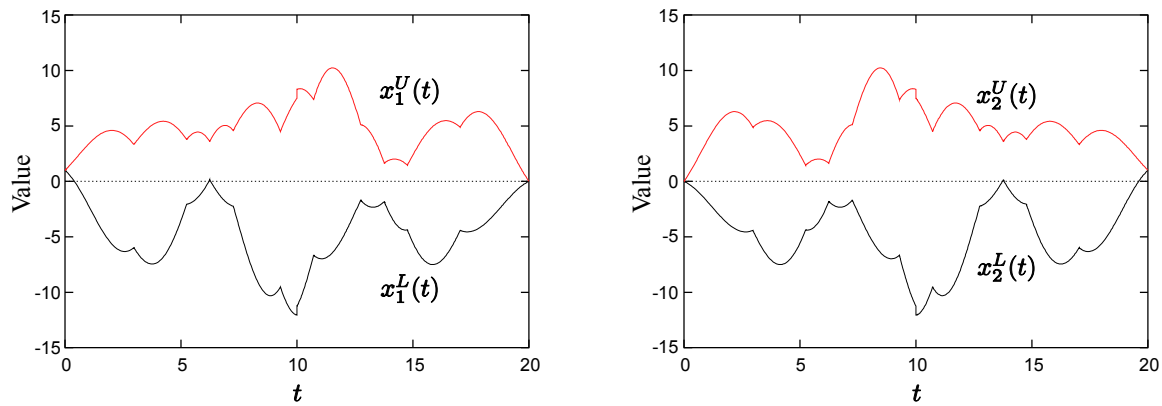


Figure 2-2: Implied state bounds for Example 2.19.

$$\text{Mode 2} \ : \begin{cases} \dot{x}_1 = -0.1x_1 - x_2 + p_2 \\ \dot{x}_2 = x_1 - 0.1x_2 - p_1 \end{cases},$$

where $\mathbf{x}(0) = (1, 0)$, $T_\mu = 1, 2$ and the transition functions at $\tau_1 = 10$ are given by

$$x_1(\mathbf{p}, \sigma_2) = x_2(\mathbf{p}, \tau_1),$$
$$x_2(\mathbf{p}, \sigma_2) = x_1(\mathbf{p}, \tau_1).$$

The implied state bounds, shown in Figure 2-2, are easily obtained from Theorem 2.14, where $\mathbf{M}_i(t)$ are obtained from the solution of (2.5), and $\mathbf{n}_i(t)$ from the solution of (2.15).

Example 2.20. Consider the following problem

$$\min_{u,x} \ F(u, x) \equiv \int_0^3 -x^2 \ \mathrm{d}t,$$

where $-4 \leq u(t) \leq 4$, subject to the following hybrid system

$$\text{Mode 1} \ : \dot{x} = x + u, \quad \text{Mode 2} \ : \dot{x} = -x - 2u,$$

with $x(0) = 1$, $T_\mu = 1, 2, 1$, and state continuity enforced at the explicit transitions at $\tau_1 = 1$ and $\tau_2 = 2$. The control parameterization employed is a piecewise constant

profile over two equal finite elements, where

$$u(\mathbf{p},t) = \begin{cases} p_1 & \text{for } 0 \le t \le 1.5 \\ p_2 & \text{for } 1.5 \le t \le 3 \end{cases} .$$

This problem can be solved using a BB algorithm such as Algorithm 2.3 utilizing the theory developed in this chapter. The resulting control parameterized problem is:

$$\min_{\mathbf{p}} \;\; F(\mathbf{p}) \equiv \int_0^3 -x(\mathbf{p},t)^2 \; \mathrm{d}t,$$

where $\mathbf{p} \in [-4,4]^2$, subject to the following hybrid system

$$\text{Mode 1} \;: \dot{x}(\mathbf{p},t) = x(\mathbf{p},t) + \mathbf{s}(t)^{\mathrm{T}}\mathbf{p}, \quad \text{Mode 2} \;: \dot{x}(\mathbf{p},t) = -x(\mathbf{p},t) - 2\mathbf{s}(t)^{\mathrm{T}}\mathbf{p},$$

$s_2(t) = 1 - s_1(t)$ and

$$s_1(t) = \begin{cases} 1 & \text{for } 0 \le t < 1.5 \\ 0 & \text{for } 1.5 \le t \le 3 \end{cases} ,$$

with $x(\mathbf{p},0) = 1$, $T_\mu = 1,2,1$, and state continuity enforced at the explicit transitions at $\tau_1 = 1$ and $\tau_2 = 2$. Since the integrand is a univariate concave function, the natural convex underestimator to use is its convex envelope:

$$U(\mathbf{p}) \equiv \int_0^3 \big(x^U(t) + x^L(t)\big)\big(x^L(t) - x(\mathbf{p},t)\big) - x^L(t)^2 \;\; \mathrm{d}t. \tag{2.17}$$

The implied state bounds constructed from Theorem 2.14, $x^L(t)$ and $x^U(t)$, are shown in Figure 2-3(a). The nonconvex objective function, $F(\mathbf{p})$, and its constructed convex underestimator, $U(\mathbf{p})$, over the entire feasible region P are shown in Figure 2-3(b). The lower bound at the first iteration will be obtained at $\mathbf{p} = (4,-4)$. The upper bound obtained at the first iteration will depend on the initial guess for $\mathbf{p}$. Suppose that the feasible region is partitioned along the lines $p_1 = 0$ and $p_2 = 0$ into 4 quadrants. Each of these partitions imply new state bounds, which update the convex

relaxation in (2.17). The constructed convex relaxation for the quadrant $[-4, 0] \times [0, 4]$ is shown in Figure 2-3(c). In this example, the algorithm terminates with at most 4 iterations utilizing the bisection heuristic described above, with a global solution of $F = -82.04$, $\mathbf{p}^* = (4, -4)$. However, in general, only ε convergence can be achieved in a finite number of iterations.

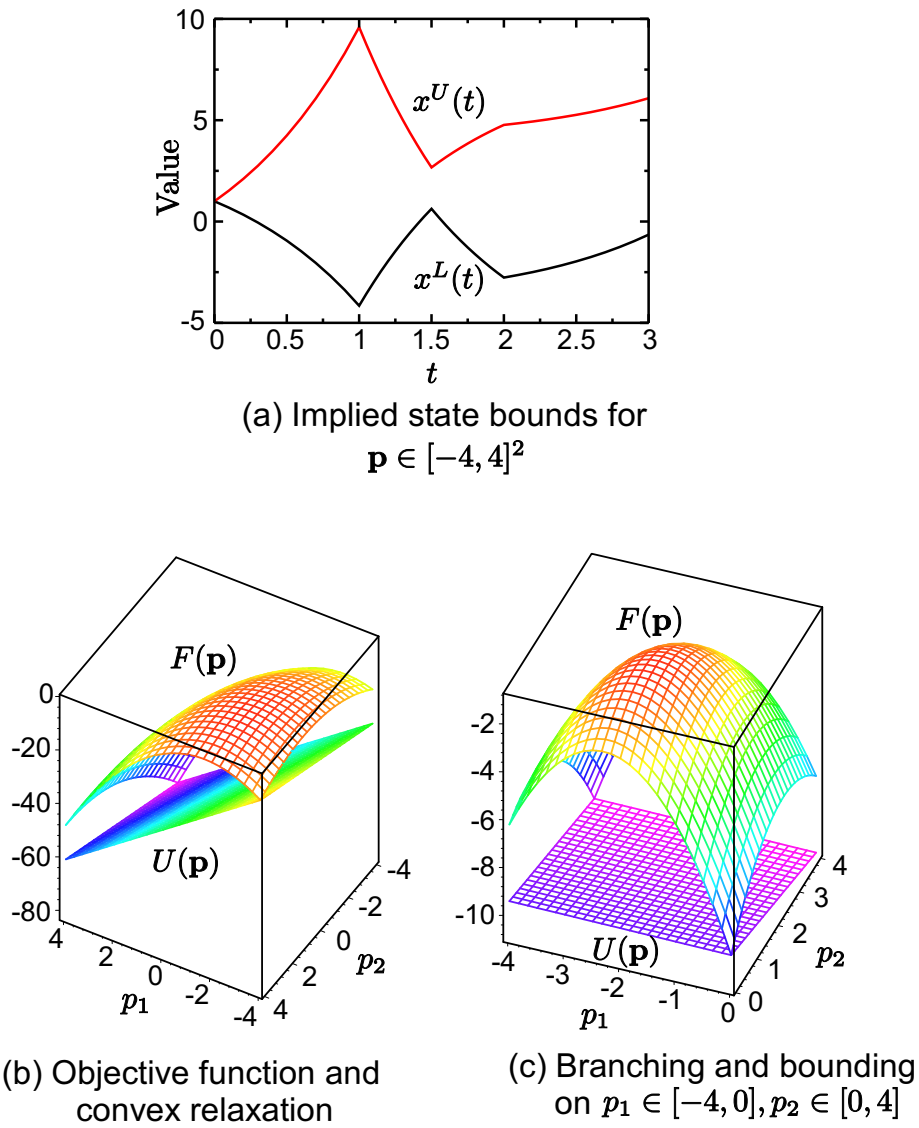


Figure 2-3: Implied state bounds, objective function and convex relaxations for Example 2.20.

Chapter 3

Determining the Optimal Mode Sequence

In this chapter, we shall examine the class of optimization problems with hybrid systems embedded where the mode sequence of the embedded hybrid system is to be determined by the optimization procedure, i.e., T_μ is also an optimization variable in the problem. To keep the analysis simple, we shall make the timings of the transitions fixed. It turns out that even with this restriction, the problem is very difficult to solve, as reflected by the length of this chapter.

This chapter is organized as follows. Section 3.1 introduces the general formulation of the optimal control problem, and defines the linear hybrid systems under consideration in this chapter. In Section 3.2, we explore the use of dynamic programming as a possible tool for solving this problem, and show that the extension of dynamic programming techniques to determining the optimal mode sequences for continuous time linear hybrid systems has considerable technical hurdles to overcome. In Section 3.3, a hybrid superstructure is postulated for the problem, and the problem is reformulated via the introduction of binary variables, while Section 3.4 describes various bounding strategies that we have developed for linear hybrid systems whose mode sequences are allowed to vary. Section 3.5 describes a branch-and-cut (BC) algorithm for solving the reformulated problem, where we have devised a dynamic bounds tightening heuristic, based on the bounding strategies presented, that can

greatly accelerate the convergence of the BC algorithm. Section 3.6 discusses the illustrative example problems that are solved with the BC algorithm, and highlights the effect of the dynamic bounds tightening heuristic.

3.1 Problem Formulation

First, we shall define the linear hybrid system of interest, based on the modeling framework presented in Section 1.1.

Definition 3.1. The LTV ODE hybrid system of interest is defined by the following.

1. An index set M of modes potentially visited along T_μ, $M = \{1, \ldots, n_m\}$, and a fixed T_τ with given time events (i.e., explicit transition times) $\sigma_1, \tau_1, \tau_2, \ldots, \tau_{n_e}$.

2. An invariant structure system where the number of continuous state variables is constant between modes, $V = \{\mathbf{x}, \mathbf{u}, \mathbf{p}, t\}$, where $\mathbf{p} \in P \subset \mathbb{R}^{n_p}$, $\mathbf{u}(\mathbf{p}, t) \in U(t) \subset \mathbb{R}^{n_u}$ for all $(\mathbf{p}, t) \in P \times I_i,\ i = 1, \ldots, n_e$, and $\mathbf{x}(\mathbf{p}, T_\mu, t) \in \mathbb{R}^{n_x}$ for all $(\mathbf{p}, T_\mu, t) \in P \times M^{n_e} \times I_i,\ i = 1, \ldots, n_e$.

3. The parameterization of the bounded real valued controls,

$$\begin{aligned} \mathbf{u}(\mathbf{p}, t) &= \mathbf{S}(t)\mathbf{p} + \mathbf{v}(t), \\ \mathbf{u}^L(t) \leq \mathbf{u}(\mathbf{p}, t) &\leq \mathbf{u}^U(t), \quad \forall t \in [\sigma_1, \tau_{n_e}], \end{aligned} \tag{3.1}$$

   where $\mathbf{u}^L(t)$ and $\mathbf{u}^U(t)$ are known lower and upper bounds on the controls $\mathbf{u}(\mathbf{p}, t)$ that define the set $U(t)$, $\mathbf{S}(t)$ and $\mathbf{v}(t)$ are piecewise continuous on $[\sigma_1, \tau_{n_e}]$ and defined at any point of discontinuity.

4. The LTV ODE system for each mode $m \in M$, which is given by

$$\dot{\mathbf{x}}(\mathbf{p}, T_\mu, t) = \mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{p}, T_\mu, t) + \tilde{\mathbf{B}}^{(m)}(t)\mathbf{u}(\mathbf{p}, t) + \mathbf{C}^{(m)}(t)\mathbf{p} + \tilde{\mathbf{q}}^{(m)}(t),$$

   where $\mathbf{A}^{(m)}(t)$ is continuous on $[\sigma_1, \tau_{n_e}]$, $\tilde{\mathbf{B}}^{(m)}(t)$, $\mathbf{C}^{(m)}(t)$ and $\tilde{\mathbf{q}}^{(m)}(t)$ are piecewise continuous on $[\sigma_1, \tau_{n_e}]$ and defined at any point of discontinuity, for all

$m \in M$. After control parameterization (substitute Equation (3.1)), we have

$$\dot{\mathbf{x}}(\mathbf{p}, T_\mu, t) = \mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{p}, T_\mu, t) + \mathbf{B}^{(m)}(t)\mathbf{p} + \mathbf{q}^{(m)}(t), \tag{3.2}$$

where $\mathbf{B}^{(m)}(t) \equiv \tilde{\mathbf{B}}^{(m)}(t)\mathbf{S}(t) + \mathbf{C}^{(m)}(t)$ and $\mathbf{q}^{(m)}(t) \equiv \tilde{\mathbf{B}}^{(m)}(t)\mathbf{v}(t) + \tilde{\mathbf{q}}^{(m)}(t)$ are piecewise continuous on $[\sigma_1, \tau_{n_e}]$ and defined at any point of discontinuity, for all $m \in M$.

5. The transition conditions for the transitions between epochs I_i and I_{i+1}, $i = 1, \ldots, n_e - 1$, which are explicit time events:

$$L^{(m_i)} := (t \geq \tau_i), \tag{3.3}$$

indicating the transition from mode m_i in epoch I_i to mode m_{i+1} in epoch I_{i+1} at time τ_i.

6. The collection of transition functions, which is given by the following equation,

$$\begin{aligned}\mathbf{x}(\mathbf{p}, T_\mu, \sigma_{i+1}) = \mathbf{D}_i(m_i, m_{i+1})\mathbf{x}(\mathbf{p}, T_\mu, \tau_i) \\ + \mathbf{E}_i(m_i, m_{i+1})\mathbf{p} + \mathbf{k}_i(m_i, m_{i+1}), \quad \forall i = 1, \ldots, n_e - 1,\end{aligned} \tag{3.4}$$

for the transition from mode m_i in epoch I_i to mode m_{i+1} in epoch I_{i+1}, where $\mathbf{D}_i(m_i, m_{i+1})$, $\mathbf{E}_i(m_i, m_{i+1})$ and $\mathbf{k}_i(m_i, m_{i+1})$ are known for all $(m_i, m_{i+1}) \in M^2$, $\forall i = 1, \ldots, n_e - 1$.

7. A given initial condition for mode m_1,

$$\mathbf{x}(\mathbf{p}, T_\mu, \sigma_1) = \mathbf{E}_0\mathbf{p} + \mathbf{k}_0. \tag{3.5}$$

A solution, $\mathbf{x}(\mathbf{p}, T_\mu, t)$, $t \in I_i$, $\forall i = 1, \ldots, n_e$, will exist and be unique for all $(\mathbf{p}, T_\mu) \in P \times M^{n_e}$, at least in the weak or extended sense (this follows from [42], see e.g., Theorem 4.4). Since the transitions occur at known time events, Zeno behavior [79, 139] will not occur for the hybrid systems considered. In addition, the same

observations for control parameterization in Section 2.3 apply here. We now introduce the general problem that we are interested in solving.

Definition 3.2. Let P be a nonempty compact convex subset of $\mathbb{R}^{n_p}$. Define the following sets for all $i = 1, \ldots, n_e$:

$$\begin{aligned}
\mathcal{X}^{(i)}(t; P) &\equiv \left\{\mathbf{x}(\mathbf{p}, T_\mu, t) \mid \mathbf{p} \in P, T_\mu \in M^{n_e}\right\}, \ \forall t \in I_i, \\
\dot{\mathcal{X}}^{(i)}(t; P) &\equiv \left\{\dot{\mathbf{x}}(\mathbf{p}, T_\mu, t) \mid \mathbf{p} \in P, T_\mu \in M^{n_e}\right\}, \ \forall t \in I_i, \\
\mathcal{X}^{(i)}(P) &\equiv \bigcup_{t \in I_i} \mathcal{X}^{(i)}(t; P), \quad \dot{\mathcal{X}}^{(i)}(P) \equiv \bigcup_{t \in I_i} \dot{\mathcal{X}}^{(i)}(t; P).
\end{aligned}$$

These sets represent the various images of the parameter space under the solution of the linear hybrid system. When T_μ is fixed, we can define the following sets.

Definition 3.3. Let P be a nonempty compact convex subset of $\mathbb{R}^{n_p}$. Define the following sets for all $i = 1, \ldots, n_e$ where $T_\mu = T_\mu^* = \{m_j^*\}_{j=1}^{n_e}$ is a fixed mode sequence:

$$\begin{aligned}
\mathcal{X}^{*(i)}(T_\mu^*, t; P) &\equiv \left\{\mathbf{x}(\mathbf{p}, T_\mu^*, t) \mid \mathbf{p} \in P\right\}, \ \forall t \in I_i, \\
\dot{\mathcal{X}}^{*(i)}(T_\mu^*, t; P) &\equiv \left\{\dot{\mathbf{x}}(\mathbf{p}, T_\mu^*, t) \mid \mathbf{p} \in P\right\}, \ \forall t \in I_i, \\
\mathcal{X}^{*(i)}(T_\mu^*; P) &\equiv \bigcup_{t \in I_i} \mathcal{X}^{*(i)}(T_\mu^*, t; P), \quad \dot{\mathcal{X}}^{*(i)}(T_\mu^*; P) \equiv \bigcup_{t \in I_i} \dot{\mathcal{X}}^{*(i)}(T_\mu^*, t; P).
\end{aligned}$$

Problem 3.4. Consider the following problem,

$$\begin{aligned}
\min_{\mathbf{p} \in P, T_\mu \in M^{n_e}} F(\mathbf{p}, T_\mu) \equiv \sum_{i=1}^{n_e} \Big(& \phi_i\big(\dot{\mathbf{x}}(\mathbf{p}, T_\mu, \tau_i), \mathbf{x}(\mathbf{p}, T_\mu, \tau_i), \mathbf{p}\big) \\
& + \int_{\sigma_i}^{\tau_i} f_i\big(\dot{\mathbf{x}}(\mathbf{p}, T_\mu, t), \mathbf{x}(\mathbf{p}, T_\mu, t), \mathbf{p}, t\big) \, \mathrm{d}t \Big), \quad (3.6)
\end{aligned}$$

$$\begin{aligned}
\text{s.t.} \quad \mathbf{G}(\mathbf{p}, T_\mu) \equiv \sum_{i=1}^{n_e} \Big(& \boldsymbol{\eta}_i\big(\dot{\mathbf{x}}(\mathbf{p}, T_\mu, \tau_i), \mathbf{x}(\mathbf{p}, T_\mu, \tau_i), \mathbf{p}\big) \\
& + \int_{\sigma_i}^{\tau_i} \mathbf{g}_i\big(\dot{\mathbf{x}}(\mathbf{p}, T_\mu, t), \mathbf{x}(\mathbf{p}, T_\mu, t), \mathbf{p}, t\big) \, \mathrm{d}t \Big) \leq \mathbf{0}, \quad (3.7)
\end{aligned}$$

where $\mathbf{x}(\mathbf{p}, T_\mu, t)$ is given by the solution of the embedded LTV ODE hybrid system

(Definitions 3.1); f_i and $\mathbf{g}_i$ are piecewise continuous mappings with a finite number of stationary simple discontinuities in time [120, Def. 2.1], $f_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P \times I_i \to \mathbb{R}$ and $\mathbf{g}_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P \times I_i \to \mathbb{R}^{n_c}$, for all $i = 1, \ldots, n_e$; ϕ_i and $\boldsymbol{\eta}_i$ are continuous mappings $\phi_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P \to \mathbb{R}$, and $\boldsymbol{\eta}_i : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P \to \mathbb{R}^{n_c}$, for all $i = 1, \ldots, n_e$; and n_c is the number of constraints in (3.7). Additionally, for the set $G = \{\mathbf{p}, T_\mu \mid \mathbf{G}(\mathbf{p}, T_\mu) \leq \mathbf{0}\}$, we require that $(P \times M^{n_e}) \cap G \neq \emptyset$, i.e., the feasible region is non-empty.

The objective function in (3.6) and the inequality constraints in (3.7) are written in the form of Bolza type functionals. As explained in the previous chapter, the advantage of expressing them in this canonical form [126] is that they are all treated in the same way in as far as the computations of their values, convex relaxations, and respective gradients are concerned in the numerical solution of the optimization problem. In addition, there exist constraint transcriptions [126] that will transform general constraints, e.g., equality or inequality path constraints, into the canonical form in (3.7).

We will end this section by establishing the existence of a minimum to Problem 3.4, which will be assumed for the rest of this chapter.

Theorem 3.5. *Let $T_\mu^* \in M^{n_e}$ be fixed. Also, let $P^o \supset P$, $\mathcal{X}^{*(i)o}(\tau_i) \supset \mathcal{X}^{*(i)}(T_\mu^*, \tau_i; P^o)$, $\mathcal{X}^{*(i)o} \supset \mathcal{X}^{*(i)}(T_\mu^*; P^o)$, $\dot{\mathcal{X}}^{*(i)o}(\tau_i) \supset \dot{\mathcal{X}}^{*(i)}(T_\mu^*, \tau_i; P^o)$ and $\dot{\mathcal{X}}^{*(i)o} \supset \dot{\mathcal{X}}^{*(i)}(T_\mu^*; P^o)$ be open subsets of $\mathbb{R}^{n_p}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$ and $\mathbb{R}^{n_x}$ respectively, for all $i = 1, \ldots, n_e$. If the following conditions are satisfied, then the objective function $F(\cdot, T_\mu^*)$ is continuously differentiable on P^o.*

1. *$\frac{\partial \phi_i}{\partial \dot{\mathbf{x}}}$, $\frac{\partial \phi_i}{\partial \mathbf{x}}$ and $\frac{\partial \phi_i}{\partial \mathbf{p}}$ exist, and are continuous on $\dot{\mathcal{X}}^{*(i)o}(\tau_i) \times \mathcal{X}^{*(i)o}(\tau_i) \times P^o$ for all $i = 1, \ldots, n_e$.*

2. *$\frac{\partial f_i}{\partial \dot{\mathbf{x}}}$, $\frac{\partial f_i}{\partial \mathbf{x}}$ and $\frac{\partial f_i}{\partial \mathbf{p}}$ are piecewise continuous on $\dot{\mathcal{X}}^{*(i)o} \times \mathcal{X}^{*(i)o} \times P^o \times I_i$ for all $i = 1, \ldots, n_e$ where only a finite number of stationary simple discontinuities are allowed.*

Proof. Since T_μ^* is fixed, we can apply Theorem 2.9 to obtain the desired result. □

Theorem 3.6. *If the sufficient conditions in Theorem 3.5 are satisfied for all $T_\mu \in M^{n_e}$, then a minimum exists for Problem 3.4.*

Proof. Since the number of modes and epochs, n_m and n_e, are finite integers, M^{n_e} is a finite discrete set. Define the following (possibly empty) set for some fixed $T_\mu^* \in M^{n_e}$,

$$G(T_\mu^*) = \{\mathbf{p} \mid \mathbf{G}(\mathbf{p}, T_\mu^*) \leq \mathbf{0}\}.$$

Partition the set M^{n_e} into the following disjoint sets,

$$Q^0 = \{T_\mu^* \mid G(T_\mu^*) \cap P = \emptyset\}, \qquad Q^1 = \{T_\mu^* \mid G(T_\mu^*) \cap P \neq \emptyset\}$$

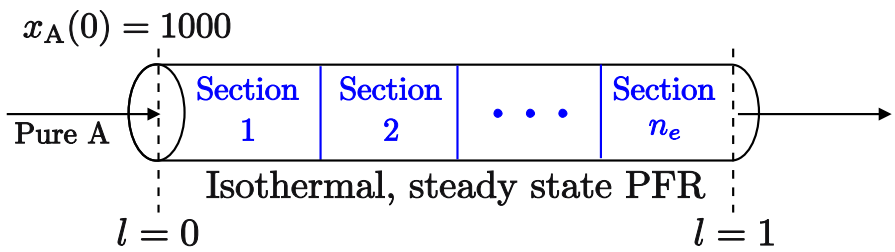
where $M^{n_e} = Q^0 + Q^1$. By assumption in Problem 3.4, $Q^1 \neq \emptyset$. For every fixed $T_\mu^* \in Q^1$, Theorem 2.10 shows that a minimum exists for Problem 3.4 with fixed $T_\mu = T_\mu^*$ if the sufficient conditions in Theorem 3.5 are satisfied for T_μ^*. Since $Q^1 \subset M^{n_e}$ is a finite discrete set, a minimum exists for Problem 3.4. □

3.1.1 An Illustrative Example: Catalyst Loading in a PFR

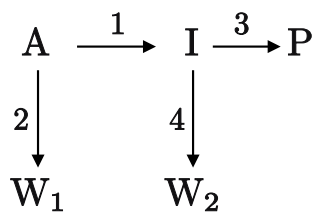
Here, we present a simple example problem inspired from chemical reaction engineering, which will be a common case study for the rest of this thesis.

Example 3.7. Consider an isothermal plug flow reactor (PFR) operating at steady state, and 3 possible choices of catalyst. The reaction scheme, initial conditions and associated rate constants are shown in Figure 3-1, where x_i represents the molar concentration of species i (mol m^{-3}) and k_j represents the rate constant of reaction j (min^{-1}). The PFR has a uniform cross-sectional area of 0.05 m^2, and a constant volumetric flow rate of 0.05 m^3 min^{-1}. In this example, the independent variable t is the length, l (m), of the reactor.

Note that the choice of catalyst corresponds to the choice of the sequence of modes in a linear hybrid system with 3 modes (each mode corresponds to the choice of a different catalyst) and n_e epochs (each epoch corresponds to a section of the reactor),



Reaction scheme



Catalyst

	k_1	k_2	k_3	k_4
1	2.098	1.317	0.021	0.033
2	29.53	110.2	0.295	0.079
3	182.6	2325	1.826	0.143

Kinetics

$$\dot{x}_A = -(k_1 + k_2)x_A \qquad \dot{x}_I = k_1 x_A - (k_3 + k_4)x_I$$
$$\dot{x}_{W_1} = k_2 x_A \qquad \dot{x}_{W_2} = k_4 x_I$$
$$\dot{x}_P = k_3 x_I$$

Figure 3-1: Chemical reaction scheme and kinetics for PFR example

with state continuity at the transitions. The initial feed to the reactor comprises a stream of pure component A, with an inlet molar concentration of 1000 mol m^{-3}. The objective function is to maximize the profit from the process, which is expressed as a scaled function of the anticipated sales of the product P, as well as the treatment costs of the by-products, W_1 and W_2,

$$\max_{T_\mu} \; x_P(1) - 0.01 x_{W_1}(1) - 0.1 x_{W_2}(1).$$

In terms of the notation of Section 3.1, the problem can be stated as the following:

$$\min_{T_\mu \in M^{n_e}} x_5(T_\mu, 1) - 0.01 x_2(T_\mu, 1) - 0.1 x_4(T_\mu, 1)$$

where $\mathbf{x}(T_\mu, t) \equiv (x_A, x_{W_1}, x_I, x_{W_2}, x_P) \equiv (x_1, x_2, x_3, x_4, x_5)$ is given by the solution of

the following hybrid system with $M = \{1, 2, 3\}$:

$$\text{Mode 1:} \begin{cases} \dot{x}_1 = -3.415x_1 \\ \dot{x}_2 = 1.317x_1 \\ \dot{x}_3 = 2.098x_1 - 0.054x_3 \\ \dot{x}_4 = 0.033x_3 \\ \dot{x}_5 = 0.021x_3 \end{cases},$$

$$\text{Mode 2:} \begin{cases} \dot{x}_1 = -139.73x_1 \\ \dot{x}_2 = 110.2x_1 \\ \dot{x}_3 = 29.53x_1 - 0.374x_3 \\ \dot{x}_4 = 0.079x_3 \\ \dot{x}_5 = 0.295x_3 \end{cases},$$

$$\text{Mode 3:} \begin{cases} \dot{x}_1 = -2507.6x_1 \\ \dot{x}_2 = 2325x_1 \\ \dot{x}_3 = 182.6x_1 - 1.969x_3 \\ \dot{x}_4 = 0.143x_3 \\ \dot{x}_5 = 1.826x_3 \end{cases},$$

$\mathbf{x}(T_\mu, 0) = (1000, 0, 0, 0, 0)$, $T_\tau = \{I_i\}, i = 1, \ldots, n_e$, $\sigma_1 = 0$ and $\tau_i = i/n_e$ for all $i = 1, \ldots, n_e$. The transition functions for all transitions are given by state continuity.

3.2 Dynamic Programming Approaches

In this section, we shall examine the use of dynamic programming [23] techniques for the solution of Problem 3.4. First, we shall introduce dynamic programming, which concept is best illustrated through examples. For example, consider the simple shortest path problem shown in Figure 3-2. The figure represents one-way streets in

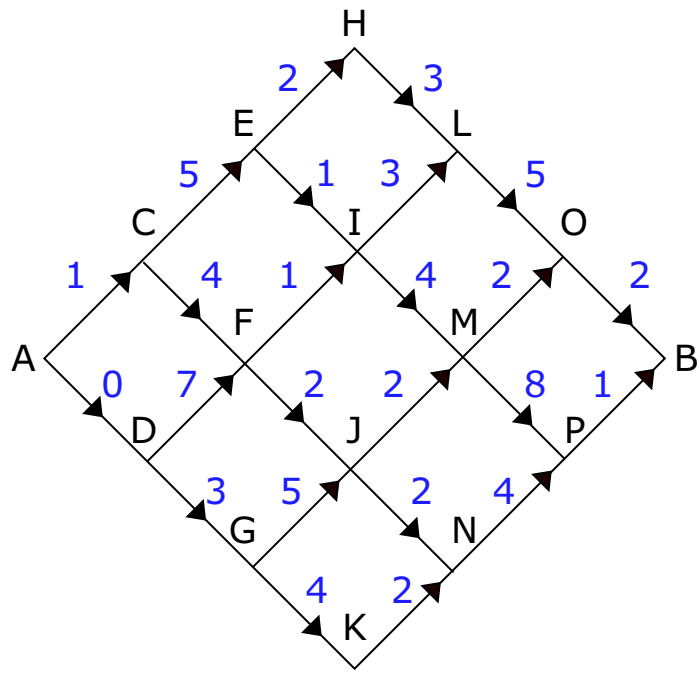


Figure 3-2: Shortest path problem from A to B

a city, and the numbers shown on the figure represent the effort (this could be time, cost or distance) required to traverse from point to point. Starting from A, we wish to reach B with minimum total effort.

There are a total of 20 paths from A to B, and so a total enumeration of all possible paths from A to B would involve 100 additions and 19 comparisons to solve the problem. A more efficient way to solve the problem would be to apply the dynamic programming method, which is based almost entirely on two key ideas, described below in the context of the problem stated above:

1. The best path from A to B has the property that, whatever the initial decision at A, the remaining path to B, starting from the next point after A, say Z, must be the best path from that point to B. This is also known as the principle of optimality, attributed to Bellman. The proof is simple. Let the best path from A to B be PQ, where P is the path from A to Z, and Q is the path from Z to B. Assume that Q is not the optimal path from Z to B, and that Q' is a different, optimal path from Z to B. Clearly, this means that the optimal path from A to B is PQ', which is a contradiction. Note that this arises from the addictive

contributions of each arc to the overall cost.

2. Starting from A, we do not know whether to go from A to C, or A to D, but if we knew two additional numbers, namely, the total effort required to get from C to B by the best path, and the total effort required to get from D to B by the best path, we could make the choice easily at A. Denoting the minimum effort from i to B by S_i, the minimum effort to get from A to B can be calculated as

$$S_{\mathrm{A}} = \min(1 + S_{\mathrm{C}}, 0 + S_{\mathrm{D}})$$

While the numbers S_{C} and S_{D} are not known, we could compute them recursively if we knew S_{E}, S_{F} and S_{G}:

$$S_{\mathrm{C}} = \min(5 + S_{\mathrm{E}}, 4 + S_{\mathrm{F}})$$
$$S_{\mathrm{D}} = \min(7 + S_{\mathrm{F}}, 3 + S_{\mathrm{G}})$$

These numbers in turn depend on S_{H}, S_{I}, S_{J} and S_{K}, which in turn depend on S_{L}, S_{M}. S_{N}, which in turn depend on S_{O} and S_{P}. However, note that $S_{\mathrm{O}} = 2$ and $S_{\mathrm{P}} = 1$ are trivial to obtain, and so, we can compute the lengths of the minimum-effort paths by considering starting points further and further away from B, finally working our way back to A.

The cost of performing dynamic programming on the above problem involves 24 additions and 9 comparisons (which is clearly superior to explicit enumeration), and gives the optimal solution as $S_{\mathrm{A}} = 13$ with the optimal path of ACFJMOB.

Forward Dynamic Programming

Now, we shall describe a variation on the dynamic programming procedure described above, which was to calculate the optimal paths *backward* from B. The subproblems solved in the backward dynamic programming procedure produced optimal paths from any stage to B. The forward dynamic programming approach is essentially a reverse procedure, based on the same optimality principle: the best path from A to

B has the property that, whatever the vertex before B, say Z, the same path starting from A to Z must be the best path from A to Z. In this case, we can set up the appropriate recurrence relations for the new optimal value functions, S_i, to denote the minimum effort from A to i. Hence, the minimum effort to get from A to B can be calculated as

$$S_{\mathrm{B}} = \min(S_{\mathrm{O}} + 2, S_{\mathrm{P}} + 1)$$

While the numbers S_{O} and S_{P} are not known, we could compute them recursively if we knew S_{L}, S_{M} and S_{N}:

$$S_{\mathrm{O}} = \min(S_{\mathrm{L}} + 5, S_{\mathrm{M}} + 2)$$
$$S_{\mathrm{P}} = \min(S_{\mathrm{M}} + 8, S_{\mathrm{N}} + 4)$$

These numbers in turn depend on S_{H}, S_{I}, S_{J} and S_{K}, which in turn depend on S_{E}, S_{F}. S_{G}, which in turn depend on S_{C} and S_{D}. Again, $S_{\mathrm{C}} = 1$ and $S_{\mathrm{D}} = 0$ are trivial to obtain, and so, we can compute the lengths of the minimum-effort paths by considering starting points further and further away from A, finally working our way forward to B. The subproblems solved in the forward dynamic programming procedure produce optimal paths from A to any stage. For this shortest path problem, the computational effort for the forward dynamic programming approach is the same as the backward dynamic programming approach.

It is interesting to note that according to Dreyfus and Law [47], there are no further key ideas in dynamic programming. We will now summarize some known results concerning dynamic programming in the literature.

3.2.1 Discrete Time Linear Dynamical Systems

Let $x(i)$ denote the state at stage i and let $u(i)$ denote the decision or control variable. Then we assume that the state at stage $i + 1$ is given by

$$x(i + 1) = g(i)x(i) + h(i)u(i), \tag{3.8}$$

where $g(i)$ and $h(i)$, $i = 0, \ldots, N-1$ are known constants. This is called a linear dynamical system because the rule giving the new state is linear in the old state and the decision. Here, we assume that $x(i)$ and $u(i)$ are continuous variables that can assume any real values. For simplicity, we are going to abuse notation by following that used in Dreyfus and Law [47], where $x(i)$ and x are used interchangeably, and similarly for $u(i)$ and u, when the stage i can be inferred from the context.

The objective function is a summation of costs over N stages plus a terminal cost depending on $x(N)$, where the cost of each stage is a quadratic function of $x(i)$ and $u(i)$. We assume that the objective function is given by

$$J = lx^2(N) + \sum_{i=0}^{N-1} \left(a(i)x^2(i) + c(i)u^2(i) \right). \tag{3.9}$$

Given the state at stage zero, $x(0)$, the problem is to choose $u(0), u(1), \ldots, u(N-1)$ so as to minimize J given by (3.9) where the states $x(i)$ evolve by the rule (3.8).

Notice that this is formulated as a typical Quadratic Programming (QP) problem with linear constraints and a quadratic objective function [21]. This problem can also be solved using modified simplex or interior point methods, e.g., with the commercial solver CPLEX [78]. Another way to solve the problem is to first solve for x in terms of u, and substitute the constraints into the objective function to obtain a quadratic function of u. Finally, we could solve the problem using the Karush-Kuhn-Tucker (KKT) optimality conditions (solving a linear system of equations) [21]. According to Dreyfus and Law [47], the dynamic programming approach is easier and more systematic. We will describe the dynamic programming approach next as it will be useful in understanding how these techniques can be applied to solving Problem 3.4.

First, we begin by defining the optimal value function $V_i(x)$ for the problem given by (3.8) and (3.9) as

$V_i(x) =$ the minimum cost of the remaining process if it starts stage i in state x.

Then, by the principle of optimality, for $i = 0, 1, \ldots, N-1$,

$$V_i(x) = \min_u \left[a(i)x^2 + c(i)u^2 + V_{i+1}(g(i)x + h(i)u)\right] \tag{3.10}$$

with the boundary condition

$$V_N(x) = lx^2. \tag{3.11}$$

We can then solve the problem in two ways: (a) a discretization of the state space; and (b) solving for $V_i(x)$ using optimality conditions. The former approach proceeds as follows. Evaluate $V_N(x)$ at a discrete grid of points taken between some arbitrary upper and lower bounds that we are sure will include the optimal solution for the given initial condition. For example, if $x(0) = 2$, we might use a grid consisting of $-5, -4.9, -4.8, \ldots, 0, 0.1, 0.2, \ldots, 4.9, 5$. Then, we could determine $V_{N-1}(x)$ at these same points by either considering only those values of $u(N-1)$, given $x(N-1)$, that lead to the grid points at which $V_N(x)$ has been computed or else by using a fixed grid of reasonable decisions for u (say $u = -5, -4.9, \ldots, 0, \ldots, 4.9, 5$) and when we need to know $V_N(x)$ at a point not actually computed, use an interpolation formula. Having determined $V_{N-1}(x)$, we apply the same procedure to determine $V_{N-2}(x)$, and so on.

Next, we consider the approach utilizing optimality conditions. From (3.10) and (3.11), we have

$$\begin{aligned} V_{N-1}(x) &= \min_u \big(a(N-1)x^2 + c(N-1)u^2 + V_N(g(N-1)x + h(N-1)u)\big) \\ &= \min_u \Big(a(N-1)x^2 + c(N-1)u^2 + lg^2(N-1)x^2 \\ &\qquad + 2lg(N-1)h(N-1)xu + lh^2(N-1)u^2\Big). \end{aligned} \tag{3.12}$$

From the first order optimality conditions, we have

$$\begin{aligned} \frac{\partial V_{N-1}(x)}{\partial u} &= 0 = 2c(N-1)u + 2lg(N-1)h(N-1)x + 2lh^2(N-1)u \\ &\Rightarrow u = -\frac{lg(N-1)h(N-1)x}{c(N-1) + lh^2(N-1)}. \end{aligned}$$

From the second order conditions, we know that this value of u yields the minimum if

$$c(N-1) + lh^2(N-1) > 0, \tag{3.13}$$

otherwise the problem has no solution. Henceforth, we assume that (3.13) holds. Substituting for u back into (3.12), we obtain

$$\begin{aligned} V_{N-1}(x) = a(N-1)x^2 + c(N-1)\left(-\frac{lg(N-1)h(N-1)x}{c(N-1)+lh^2(N-1)}\right)^2 + lg^2(N-1)x^2 \\ + 2lg(N-1)h(N-1)x\left(-\frac{lg(N-1)h(N-1)x}{c(N-1)+lh^2(N-1)}\right) \\ + lh^2(N-1)\left(-\frac{lg(N-1)h(N-1)x}{c(N-1)+lh^2(N-1)}\right)^2 \\ = \left[a(N-1) + lg^2(N-1) - \frac{lg^2(N-1)h^2(N-1)}{c(N-1)+lh^2(N-1)}\right]x^2 = p(N-1)x^2. \end{aligned} \tag{3.14}$$

Note that the coefficient of x^2 can easily be computed from the given data, by $p(N-1)$, and that the optimal value function for the process starting at stage $N-1$ is a quadratic function of the state x.

Now, we can repeat the same procedure to compute $V_{N-2}(x)$ from (3.10). We will obtain the same equation as (3.12) except that l is replaced by $p(N-1)$. Hence, we obtain u at stage $N-2$ as follows,

$$u = -\frac{p(N-1)g(N-2)h(N-2)x}{c(N-2)+p(N-1)h^2(N-2)}$$

and from (3.14) we deduce that $V_{N-2}(x)$ is given by $V_{N-2}(x) = p(N-2)x^2$ where

$$p(N-2) = a(N-2) + p(N-1)g^2(N-2) - \frac{p^2(N-1)g^2(N-2)h^2(N-2)}{c(N-2)+p(N-1)h^2(N-2)}.$$

By induction, $V_i(x)$ is given by $V_i(x) = p(i)x^2$, where $p(i)$ is determined recursively from $p(i+1)$ by

$$p(i) = a(i) + p(i+1)g^2(i) - \frac{p^2(i+1)g^2(i)h^2(i)}{c(i)+p(i+1)h^2(i)}$$

and the boundary condition $p(N) = l$. The optimal u at stage i, for a given $x(i)$, is determined by

$$u(i) = -\frac{p(i+1)g(i)h(i)x(i)}{c(i) + p(i+1)h^2(i)}.$$

Note that we have not mentioned how constraints can be handled in the dynamic programming approach. Although it is possible to incorporate constraints, e.g., specifying a terminal condition of the form $x(N) = t$, it appears that the treatment of general constraints such as bounded controls is complicated using dynamic programming, and is a topic that the traditional texts on dynamic programming do not provide guidance on. The constraints will have to be incorporated in some form in either the expression of the optimal value function (from the principle of optimality) or in the boundary conditions of the problem. Indeed, it appears that the difficulties with incorporating general constraints into dynamic programming approaches is a major drawback with using these approaches in the solution of Problem 3.4, as we shall see later in this section.

3.2.2 Optimal State-feedback Quadratic Regulation of Linear Hybrid Automata

In the hybrid systems literature, dynamic programming has been used in [26] to obtain a state-feedback control law for solving certain classes of optimal control problems, where the hybrid systems considered are discrete time linear hybrid systems with a quadratic objective function. An optimal control law is obtained for the optimization problem that takes the form of a state-feedback, i.e., it is only necessary to look at the current system state $\mathbf{x}$ in order to determine if a switch should occur. This is done by computing tables which determine whether or not a switch should occur, and the tables are formed by total discretization of the state space. There are elaborate rules in [26] that govern how each table should be calculated, which will not be presented here. However, it is worth noting the computational cost of solving the problem using the proposed approach:

1. Cost of gridding: If the state space is $\mathbb{R}^n$ and r samples are taken along each

direction, then the computational complexity for constructing each table is $\mathcal{O}(r^{n-1})$ if all switching costs are null (because the table contains two regions that can be determined by solving a one-parameter optimization problem for each vector y on the unitary semi-sphere). Otherwise, if not all switching costs are null, the complexity is $\mathcal{O}(r^n)$ because it is necessary to grid all the state space.

2. Cost of switching: There are N switches, and so the corresponding complexities become $\mathcal{O}(Nr^{n-1})$ and $\mathcal{O}(Nr^n)$ because for each switch a new table must be determined.

3. Cost for each mode: For each switch, it is necessary to compute s tables, one for each discrete mode. Furthermore, the complexity of computing the tables is equal to $\mathcal{O}((s_i-1)r^{n-1})$ and $\mathcal{O}((s_i-1)r^n)$ respectively, since each table contains s_i regions, where s_i is the number of possible successor modes. Because $s_i \leq s$, we have, for the case where there are no switching costs,

$$\mathcal{O}(Nr^{n-1}\sum_{i=1}^{s}(s_i-1)) \leq \mathcal{O}(Nr^{n-1}s^2).$$

Hence, the complexity of solving the problem is $\mathcal{O}(Nr^{n-1}s^2)$ and $\mathcal{O}(Nr^n s^2)$ respectively. Note that because this is a closed-loop problem, the inevitable *curse of dimensionality* kicks in for the state discretization (gridding). Thus, this approach would not be attractive in solving Problem 3.4, not to mention the deficiencies of the approach towards handling constraints (3.7). In the next section, we shall discuss this in more detail, including whether it is possible to use dynamic programming approaches within a continuous time formulation, without having to discretize the continuous state space at the transitions (epoch boundaries).

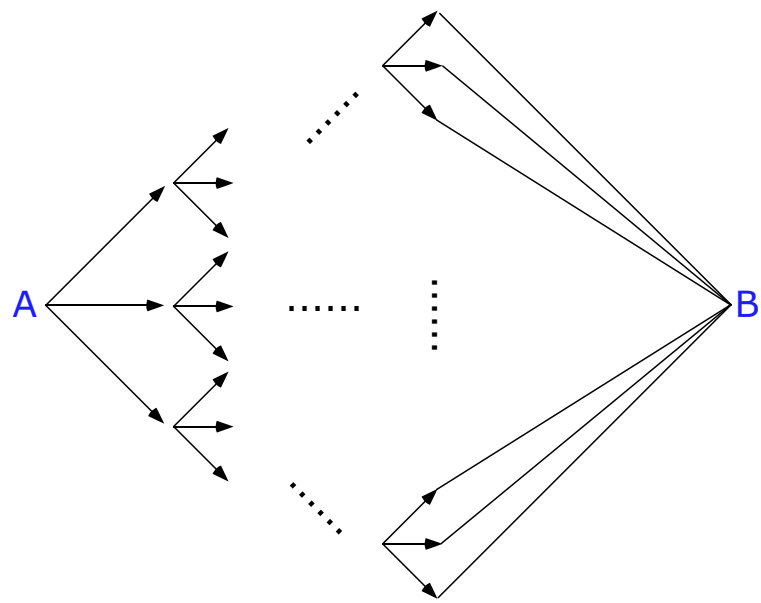


Figure 3-3: Tree for shortest path form of catalyst loading problem

3.2.3 Application to Global Optimization of Continuous State and Time Linear Hybrid Systems

In this section, we discuss the feasibility of employing dynamic programming techniques for the global optimization of continuous state hybrid systems in the continuous time domain. First, consider the catalyst loading problem presented Example 3.7. A naive way to transcribe that problem into a shortest path problem (see Figure 3-3) would require exponential complexity ($\mathcal{O}(n_m^{n_e})$) starting forward from $l = 0$, i.e., the path would look like the following tree-like structure shown in Figure 3-3, tracing each possible mode selection for each epoch, before collapsing to the end point B at the final mode.

This is essentially the explicit enumeration approach. Note that in this approach, there are no costs associated with the intermediate paths, and the costs only appear at the final $n_m^{n_e}$ nodes before the final destination node. This will be shown with a simple example (see Example 3.8 below). On the other hand, it is possible to introduce intermediate costs in the tree by transforming the objective function into an integral with the respective derivatives of the states as integrands. In this way, the integral can be split according to the epochs into intermediate costs at each node

in the tree. However, this does not change the fact that an exponential number of nodes exist in the tree.

Although the primary concepts of dynamic programming are easy to understand, not all problems exhibit a structure that can be solved effectively using the principle of optimality. Much ingenuity (and effort) has to go into formulating a suitable optimal value function that will enable the structure of the problem to be solved efficiently. It is noted here that so far, the dynamic programming approaches have relied on the discrete time formulation to reduce the costs associated with calculating the tables, assumed no (or simple terminal) constraints in the problem, and assumed quadratic (convex) objective functions. We expect that it will be very difficult to treat general problems in the form of Problem 3.4, which is nonconvex, and contains arbitrary Bolza type objective function and constraints, using dynamic programming approaches. Nevertheless, we shall try to see what types of problems can be solved in this section.

The main attraction in using dynamic programming approaches for solving the optimal control problem in Bemporad et al. [26] is that its complexity is $\mathcal{O}(n_e r^{n_x} n_m^2)$ compared to $\mathcal{O}(n_e r^{n_x} n_m^{n_e})$. In my opinion, these are the disadvantages of the method:

1. The curse of dimensionality in constructing the tables: r^{n_x}. Note that open loop approaches do not involve the construction of tables, but instead, relies on deterministic global optimization algorithms (see Section 2.1.1 and 2.1.2) which do not suffer from the curse of dimensionality which arises from discretizing the state space. Consider the catalyst loading problem. We have $n_x = 5$, and if we assume that we take 1000 points for sampling (this does not guarantee ε convergence; perhaps some form of adaptive meshing (no clear form of global guarantees) could help here, but one still needs to take a coarse enough grid at the first iteration) since $x_1(0) = 1000$. The complexity of constructing the tables would then involve the evaluation of at least $1000^5 = 10^{15}$ integrations or function evaluations, depending on whether the continuous time or discrete time formulation is used. It is thus not feasible to employ integration in the calculation of the tables, as we will demonstrate. Suppose that the integration

of a mode in an epoch takes 10^{-3} seconds. This would still require, for our example, 10^{12} seconds $>$ 30 thousand years. This was not possible within the time scale of this thesis. Clearly, this cost becomes prohibitively large as r and n_x increases. Consider a system with $n_x = 10$ and $r = 1000$. This would require at least 10^{30} function evaluations, and even if we employ a supercomputer of 10^{14} flops per second and only require a single function evaluation for each discretized point, we would need at least 10^{16} seconds $>$ 300 million years. We have also not considered the costs of storing the tables here.

2. The limitations of the dynamic programming approach, as discussed above, i.e., the assumptions of a quadratic objective function and simple (if any) constraints only.

3. The discrete time formulation is required to compute the required tables, since the integration of dynamic systems could be computationally prohibitive.

4. It is difficult to place a bound on how many sampling points, r, are needed to achieve ε optimality, especially for the case of stiff LTV systems.

5. It is not clear how to incorporate controls, $\mathbf{u}(t)$, into the optimal control problem. We suspect that the gridding of the parameter space would be needed (i.e., a total discretization approach), thus adding to the complexity of the approach.

Discretization of the state (and parameter) space is necessary as discussed above, because the principle of optimality cannot be otherwise applied at each epoch, as the state variables $\mathbf{x}$ are given by the solution of the embedded dynamic systems, and could take values anywhere in the state space. An alternative method to avoid the construction of the tables via discretization of the state (and parameter) space could be to solve the optimality conditions for each epoch directly, as demonstrated in Section 3.2.1. However, this approach is complicated by the fact that the optimization variables in this case are integer variables. As such, traditional approaches for deriving first or second order optimality conditions based on calculus and convex analysis

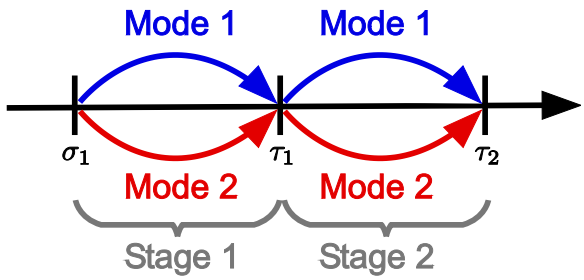


Figure 3-4: Time horizon for Example 3.8 ($n_e = 2$)

cannot be used. It is nontrivial to derive necessary and sufficient conditions for the optimality of such problems.

Consider the following open loop optimal control problem,

Example 3.8.

$$\min_{T_\mu} x(\tau_{n_e})$$

where $x(t)$ is given by the solution of the following hybrid system,

$$\text{Mode 1} : \dot{x} = x(t), \quad \text{Mode 2} : \dot{x} = -x(t),$$

$x(0) = 100$, $\tau_{n_e} = 1$, and $T_\mu \in M^{n_e}$, where n_e is the fixed number of epochs.

For epoch $I_i = [\sigma_i, \tau_i]$, $x(\sigma_i) = x_0$, the analytical solution of the differential equations is given by

$$\text{Mode 1} : x(\tau_i) = x_0 \mathrm{e}^{\Delta t}, \tag{3.15}$$

$$\text{Mode 2} : x(\tau_i) = x_0 \mathrm{e}^{-\Delta t}, \tag{3.16}$$

where $\Delta t = \tau_i - \sigma_i$.

Consider the case where $n_e = 2$, shown in Figure 3-4. Assume that $\sigma_1 = 0$, $\tau_1 = 0.5$ and $\tau_2 = 1$. In order to apply the dynamic programming approach, we have to break the problem into stages, formulate a suitable optimal value function, and apply the principle of optimality.

For this problem, it is natural to treat the epochs as stages. Note that this is different from the standard shortest path problem because the value of the state

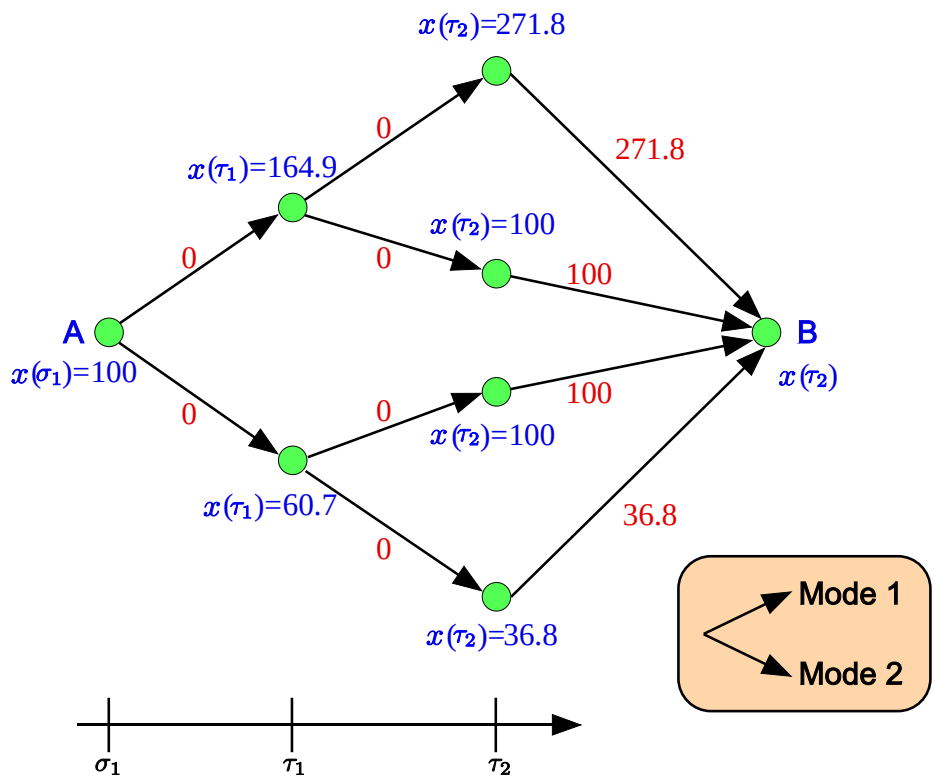


Figure 3-5: Enumerated tree for Example 3.8

variable x is not defined a priori at time τ_1. In general, x can take any real value at τ_1, depending on the form of the embedded hybrid system, and more importantly, depending on the choice of modes in epoch/stage 1. One way to fix the values of x is to derive an equivalent tree structure for a shortest path problem, as shown in Figure 3-5.

Because $\Delta t = 0.5$ is fixed, we can enumerate the values of $x(t_i)$ for all the nodes (epochs/stages) $i = 1, 2$ forward from the initial condition using (3.15) and (3.16), as shown in Figure 3-5. At τ_2, we have four nodes, and these are joined to the end node B. The value of $x(\tau_2)$ is assigned to the cost of these end paths, while all other paths have null costs. Once this problem has been formulated as a shortest path problem, the application of the principle of optimality to the problem is straightforward and standard dynamic programming approaches can be applied to this problem. However, this is clearly not an attractive way to solve the problem, because in deriving the tree, we have to enumerate all the nodes of the tree, starting from the given initial

condition, in order to obtain the cost of terminal paths. This is essentially a brute force search which suffers from exponential complexity ($\mathcal{O}(n_m^{n_e})$), and does not exploit the technique of dynamic programming to solve the problem.

An alternative dynamic programming formulation

To apply dynamic programming ideas without explicit enumeration of the tree, we have to come up with an alternative formulation of the problem. Consider the following general problem:

Problem 3.9 ($DP(1, \mathbf{x}(0))$)**.**

$$\min_{T_\mu \in M^{n_e}} \sum_{i=1}^{n_e} F_i,$$
$$\text{s.t. } F_i = \phi_i\Big(\dot{\mathbf{x}}(T_\mu, \tau_i), \mathbf{x}(T_\mu, \tau_i), \tau_i\Big) + \int_{\sigma_i}^{\tau_i} f_i\Big(\dot{\mathbf{x}}(T_\mu, t), \mathbf{x}(T_\mu, t), t\Big)\ \mathrm{d}t,$$

where $\mathbf{x}(T_\mu, t)$ is given by the solution of the embedded hybrid system, and $\mathbf{x}(\sigma_1 = 0)$ is given.

For simplicity and ease of presentation, we will assume state continuity for all transitions. It is straightforward to incorporate jumps in the values of the continuous state variables at the transitions provided that the systems of transition functions for such jumps are known a priori. Let us define the following optimal value function: $V_i(\mathbf{u})$ = the minimum cost of the remaining process if it starts in epoch i with initial condition $\mathbf{x}(\sigma_i) = \mathbf{u}$. In other words, $V_i(\mathbf{u})$ is the optimal solution value of the following subproblem $DP(i, \mathbf{u})$.

Problem 3.10 ($DP(i, \mathbf{u})$)**.**

$$\min_{T_\mu \in M^{n_e - i + 1}} \sum_{j=i}^{n_e} F_j,$$
$$\text{s.t. } F_j = \phi_j\Big(\dot{\mathbf{x}}(T_\mu, \tau_j), \mathbf{x}(T_\mu, \tau_j), \tau_j\Big) + \int_{\sigma_j}^{\tau_j} f_j\Big(\dot{\mathbf{x}}(T_\mu, t), \mathbf{x}(T_\mu, t), t\Big)\ \mathrm{d}t,$$
$$\mathbf{x}(\sigma_i) = \mathbf{u},$$

where $\mathbf{x}(T_\mu, t)$ is given by the solution of the embedded hybrid system from $\mathbf{x}(\sigma_i) = \mathbf{u}$.

We can then state the principle of optimality as follows.

Theorem 3.11. *Let the optimal solution value of $DP(1, \mathbf{x}(0))$ be attained at $T_\mu^* = m_1^*, m_2^*, \ldots, m_{n_e}^*$. Consider any arbitrary epoch $i > 1$. Let $\mathbf{x}^*(\tau_{i-1})$ be the state variables after $i-1$ epochs have evolved along the mode trajectory $m_1^*, \ldots, m_{i-1}^*$. Then, the solution of the subproblem $DP(i, \mathbf{x}^*(\tau_{i-1}))$ is $T_\mu = m_i^*, \ldots, m_{n_e}^*$.*

Proof. Let the optimal solution value for $DP(1, \mathbf{x}(0))$ be G^*. We can write

$$G^* = G_1 + G_2,$$

where

$$G_1(m_1^*, \ldots, m_{i-1}^*) = \sum_{j=1}^{i-1} F_j, \qquad G_2(\mathbf{x}^*(t_{i-1}), m_i^*, \ldots, m_{n_e}^*) = \sum_{j=i}^{n_e} F_j.$$

It is then clear that G_2 is also the solution to subproblem $DP(i, \mathbf{x}^*(t_{i-1}))$ at $T_\mu = m_i^*, \ldots, m_{n_e}^*$. Assume, for contradiction, that there exists a mode trajectory, $T_\mu^\dagger \neq m_i^*, \ldots, m_{n_e}^*$ that provides a better solution, say $G_2^\dagger$, than G_2, i.e.,

$$G_2^\dagger(\mathbf{x}^*(t_{i-1}), T_\mu^\dagger) < G_2(\mathbf{x}^*(t_{i-1}), m_i^*, \ldots, m_{n_e}^*).$$

It follows that the value of $DP(1, \mathbf{x}(0))$ attained at $m_1^*, \ldots, m_{i-1}^*, T_\mu^\dagger$ is given by

$$G^\dagger = G_1 + G_2^\dagger < G_1 + G_2 = G^*,$$

which is a contradiction that G^* is the optimal solution to $DP(1, \mathbf{x}(0))$. □

Corollary 3.12. *Consider any arbitrary epoch $i > 1$. If $\mathbf{x}^*(\tau_{i-1})$ is the value of the state variables after $i-1$ epochs have evolved along the optimal mode trajectory T_μ^* for $DP(1, \mathbf{x}(0))$, and the solution value of the subproblem $DP(i, \mathbf{x}^*(\tau_{i-1}))$ is attained at $T_\mu = m_i^*, \ldots, m_{n_e}^*$, then the last $n_e - i + 1$ modes of T_μ^* are given by $m_i^*, \ldots, m_{n_e}^*$.*

Proof. The proof is elementary from Theorem 3.11. □

From Corollary 3.12, we have a suitable principle of optimality that can be applied to the problem. Working backwards from $\mathbf{x}(\tau_{n_e})$, we can apply the principles of dynamic programming. Consider the last epoch. Let us assume that $\mathbf{x}(\tau_{n_e-1})$ is known. Then, the solution to subproblem $DP(n_e, \mathbf{x}(\tau_{n_e-1}))$ is given by

$$V_{n_e}(\mathbf{x}(\tau_{n_e-1})) = \min_{m_{n_e} \in M} F_{n_e}. \tag{3.17}$$

Given $\mathbf{x}(\tau_{n_e-1})$, (3.17) can be solved easily to yield $m^*_{n_e}$. Let us now consider the penultimate epoch. Again, let us assume that $\mathbf{x}(\tau_{n_e-2})$ is known. Then, the solution to subproblem $DP(n_e - 1, \mathbf{x}(\tau_{n_e-2}))$ is given by

$$V_{n_e-1}(\mathbf{x}(\tau_{n_e-2})) = \min_{m_{n_e-1} \in M} \left\{ F_{n_e-1} + V_{n_e}(\mathbf{x}(\tau_{n_e-1})) \right\}. \tag{3.18}$$

We can repeat this procedure to obtain the recursive formula (for $i = 1, \ldots, n_e - 1$),

$$V_i(\mathbf{x}(\tau_{i-1})) = \min_{m_i \in M} \left\{ F_i + V_{i+1}(\mathbf{x}(\tau_i)) \right\}. \tag{3.19}$$

$V_1(\mathbf{x}(0))$ is clearly the optimal solution value of $DP(1, \mathbf{x}(0))$. While the solution can be obtained from (3.19) via solving a sequence of n_e single-epoch optimization problems, this approach is different from simply making local decisions at each epoch due to the inclusion of the optimal value function V_i in (3.19). Indeed, obtaining V_i is the key to applying this approach effectively.

To obtain V_i, we can apply the backward dynamic programming approach, i.e., calculate V_i for $i = n_e, \ldots, 1$. Consider (3.17). If m_{n_e} was a continuous variable, we could apply the optimality conditions from calculus (subject to suitable conditions, e.g., when F_{n_e} is convex) to obtain an explicit expression for $V_{n_e}(\mathbf{x}(\tau_{n_e-1}))$. Following that, we could do the same to (3.18), where $V_{n_e}(\mathbf{x}(\tau_{n_e-1}))$ is treated as an end point objective added to F_{n_e-1}. This procedure can then be repeated (subject to suitable conditions, e.g., if m_{n_e-1} was continuous, F_{n_e-1} and V_{n_e} were convex, and so on) backward in time to obtain all V_i. Then, we can integrate the system forward in time

from $\mathbf{x}(\sigma_1)$ to obtain T_μ^*, i.e., starting from $\mathbf{x}(\sigma_1)$, we determine V_1, m_1^*, and $\mathbf{x}(\tau_1)$, from which we can determine V_2, m_2^*, and $\mathbf{x}(\tau_2)$, and so on.

Note that this is exactly the approach that is presented in Section 3.2. The key point is to obtain $V_i(\mathbf{x}(\tau_{i-1}))$ as an explicit function of $\mathbf{x}(\tau_{i-1})$, which is not known a priori. This requires constructing explicitly the parametric solution of an optimization problem, which in general can be constrained and nonconvex. However, for the problems considered, m_i are integer variables, and so the method of applying optimality conditions and solving for the values of $\mathbf{x}(\tau_{i-1})$ cannot be used. A possible way around the problem is to simply discretize the state space at the beginning of each epoch, and calculate the values of $V_i(\mathbf{y}_k)$ at the discretized points $\mathbf{y}_k$. This is the approach that is presented in Bemporad et al. [26]. However, it appears that without appealing to discretization, the problem has to exhibit some form of special structure for the dynamic programming approach to be useful. It would perhaps be useful at this point to apply the method to some examples.

Consider again Example 3.8 with $n_e = 2$, $\sigma_1 = 0$, $\tau_1 = 0.5$ and $\tau_2 = 1$. We can calculate

$$\begin{aligned} V_2(x(0.5)) &= \min_{m_2 \in \{1,2\}} x(1) \\ &= \min_{m_2 \in \{1,2\}} \left\{ x(0.5)\mathrm{e}^{0.5}, x(0.5)\mathrm{e}^{-0.5} \right\} \\ &= \min_{m_2 \in \{1,2\}} \left\{ 1.649x(0.5), 0.607x(0.5) \right\}. \end{aligned}$$

It is clear that

$$V_2(x(0.5)) = \begin{cases} 0.607x(0.5), m_2 = 2 \text{ if } x(0.5) \geq 0, \\ 1.649x(0.5), m_2 = 1 \text{ if } x(0.5) < 0. \end{cases} \tag{3.20}$$

Working backwards, we can calculate $V_1(x(0))$ as follows.

$$V_1(x(0)) = \min_{m_1 \in \{1,2\}} V_2(x(0.5))$$
$$V_1(x(0)) = \min_{m_1 \in \{1,2\}} \left\{ \min_{m_2 \in \{1,2\}} \{1.649x(0.5), 0.607x(0.5)\} \right\}.$$

The brute force way to solve the nested minimization problem is to enumerate the tree as in Figure 3-5 to determine the values of $x(0.5)$. However, this approach is equivalent to enumerating the full tree and thus is unattractive. For this problem, we can perform the following analysis. Suppose that $x(0.5) \geq 0$. Then, we have

$$\begin{aligned} V_1(x(0)) &= \min_{m_1 \in \{1,2\}} 0.607x(0.5) \\ &= \min_{m_1 \in \{1,2\}} \left\{ x(0)\mathrm{e}^0, x(0)\mathrm{e}^{-1} \right\} \\ &= \min_{m_1 \in \{1,2\}} \{x(0), 0.368x(0)\}. \end{aligned}$$

It is clear that

$$V_1(x(0)) = \begin{cases} 0.368x(0), m_1 = 2 \text{ if } x(0) \geq 0 \text{ and } x(0.5) \geq 0, \\ x(0), m_1 = 1 \text{ if } x(0) < 0 \text{ and } x(0.5) \geq 0. \end{cases} \tag{3.21}$$

The appearance of $x(0.5)$ in the conditional statement is undesirable because we do not know what $x(0.5)$ is at the point where we are making the decision based on $V_1(x(0))$. Hence, we have to examine which cases are valid for our assumption of $x(0.5) \geq 0$. For this example, if $x(0) \geq 0$ and $m_2 = 2$, we will satisfy the assumption. However, if $x(0) < 0$ and $m_1 = 1$, we have $x(0.5) < 0$ which violates the assumption. Hence, only the first case is valid in (3.21). We can repeat for the assumption that $x(0.5) < 0$ and obtain

$$V_1(x(0)) = \begin{cases} x(0), m_1 = 2 \text{ if } x(0) \geq 0 \text{ and } x(0.5) < 0, \\ 2.718x(0), m_1 = 1 \text{ if } x(0) < 0 \text{ and } x(0.5) < 0. \end{cases}$$

Again, testing our assumptions, we find that only the second case is valid, resulting in the following optimal solution to the problem,

$$V_1(x(0)) = \begin{cases} 0.368x(0), m_1 = 2 \text{ if } x(0) \geq 0, \\ 2.718x(0), m_1 = 1 \text{ if } x(0) < 0. \end{cases} \tag{3.22}$$

Equations (3.22) and (3.20) provide the optimal mode trajectory as we move forward in time.

In general, calculating V_i for $n_e, \ldots, 1$ would require solving n_e nested parametric minimization problems (in discrete variables), which is difficult. For the problem considered above, we have avoided enumerating all possible cases of the tree (Figure 3-5) by effectively eliminating possible regions from the state space using the arguments made to obtain (3.22). The situation becomes harder to analyze as n_x increases, because it is not clear that the number of regions in the state space $\mathbb{R}^{n_x}$ corresponding to the optimal choice of mode at each epoch can be bounded by n_m. What is perhaps more devastating is that, in the general case, there does not seem to be a way to systematically and automatically remove "dead" regions in the state space from the list of all possible regions which is exponential in the number of epochs, hence limiting the application of dynamic programming principles in a continuous setting.

As will be described in Section 3.3, it is possible to formulate the problem of obtaining the optimal mode sequence into a mixed-integer framework by introducing binary decision variables to represent T_μ. Viewed in this framework, these regions of the state space, V_i, are in essence a description of the parametric solution of the integer programming (IP) problem. It is beyond the scope of this thesis to elaborate on the properties and methods for solving multiparametric optimization problems. For a review on the multiparametric 0-1 integer linear programming problem (ILP), see [44]. Note that the parameters considered in [44] are assumed to be integers. For a review and discussion on algorithms and solution of multiparametric MILPs, see [97].

If we assume a linear cost function for the problem, state continuity at the tran-

sitions and no constraints, the problem of obtaining the regions V_{n_e} is given by the following problem,

$$\begin{aligned} \min_{\mathbf{y}\in\{0,1\}^{n_m}} \quad & \mathbf{c}^{\mathrm{T}}\mathbf{x}(\tau_{n_e}) \\ \text{s.t.} \quad & \sum_{i=1}^{n_m} y_i = 1, \\ \mathbf{x}(\tau_{n_e}) = & \sum_{i=1}^{n_m} y_i \left(\mathbf{M}^{(i)}\mathbf{x}(\tau_{n_e-1}) + \mathbf{n}^{(i)}\right). \end{aligned}$$

where $\mathbf{c}$, and $\mathbf{M}^{(i)}$ and $\mathbf{n}^{(i)}$ for all $i = 1, \ldots, n_m$ are known. If we eliminate the last constraint, we have the following, equivalent problem,

$$\begin{aligned} \min_{\mathbf{y}\in\{0,1\}^{n_m}} \quad & \mathbf{c}^{\mathrm{T}} \left(\sum_{i=1}^{n_m} y_i \left(\mathbf{M}^{(i)}\mathbf{x}(\tau_{n_e-1}) + \mathbf{n}^{(i)}\right) \right) \\ \text{s.t.} \quad & \sum_{i=1}^{n_m} y_i = 1. \end{aligned}$$

Treating $\mathbf{x}(\tau_{n_e-1})$ as a vector of continuous parameters, the above problem is a ILP in the integer (binary) variables $\mathbf{y} \in \{0,1\}^{n_m}$. Hence, the problem of determining V_{n_e} is equivalent to solving a multiparametric ILP with an affine dependence of parameters on the objective function. In [97], a tailored algorithm was developed for solving exactly such a multiparametric problem. The algorithm involves solving at least as many MILPs as there are possible optimality regions. Note that this is potentially a very large number of optimality regions; see [97] for a discussion on the number and properties of the optimality regions and complexity of parametric optimization.

Clearly, in the general case, the number of regions for this multiparametric ILP is given by n_m. Propagating backward to epoch $n_e - 1$, we will have to formulate an ILP for every region that was obtained in epoch n_e (the multiparametric ILP solved above). Thus, in the worst case, the number of regions in the state space could increase exponentially as we move backward in time. This clearly makes such an approach unattractive. This will be illustrated and discussed in the following example.

Dynamic Programming on Example 3.7

Consider Example 3.7. For epoch $I_i = [\sigma_i, \tau_i]$, $\mathbf{x}(\sigma_i) = \mathbf{x}_0$, and $\Delta t = \sigma_i - \tau_i$, it is possible to obtain the analytical solution of the hybrid system for each mode, because the dynamics of the system in each mode is LTI. Let us consider the case where $n_e = 2$ and $t_1 = 0.5$. The analytical solution is given by the following, where the index set for $\mathbf{x}$ is given by { A, W_1, I, W_2, P },

$$\text{Mode 1}: \begin{cases} x_1(t_i) = 0.181x_{01} \\ x_2(t_i) = 0.316x_{01} + x_{02} \\ x_3(t_i) = 0.494x_{01} + 0.973x_{03} \\ x_4(t_i) = 0.00530x_{01} + 0.0165x_{03} + x_{04} \\ x_5(t_i) = 0.00332x_{01} + 0.0103x_{03} + x_{05} \end{cases} \tag{3.23}$$

$$\text{Mode 2}: \begin{cases} x_1(t_i) \approx 0 \\ x_2(t_i) = 0.788x_{01} + x_{02} \\ x_3(t_i) = 0.176x_{01} + 0.829x_{03} \\ x_4(t_i) = 0.00753x_{01} + 0.0361x_{03} + x_{04} \\ x_5(t_i) = 0.0281x_{01} + 0.135x_{03} + x_{05} \end{cases} \tag{3.24}$$

$$\text{Mode 3}: \begin{cases} x_1(t_i) \approx 0 \\ x_2(t_i) = 0.927x_{01} + x_{02} \\ x_3(t_i) = 0.0272x_{01} + 0.374x_{03} \\ x_4(t_i) = 0.00332x_{01} + 0.0456x_{03} + x_{04} \\ x_5(t_i) = 0.0423x_{01} + 0.581x_{03} + x_{05}. \end{cases} \tag{3.25}$$

At the last epoch, we can calculate

$$V_2(\mathbf{x}(0.5)) = \min_{m_2 \in \{1,2,3\}} 0.01x_2(1) + 0.1x_4(1) - x_5(1) = \min_{m_2 \in \{1,2,3\}} F_{n_e}(m_2)$$

where

$$F_{n_e}(1) = 0.000365x_1(0.5) + 0.01x_2(0.5) - 0.0087x_3(0.5) + 0.1x_4(0.5) - x_5(0.5),$$
$$F_{n_e}(2) = -0.0195x_1(0.5) + 0.01x_2(0.5) - 0.131x_3(0.5) + 0.1x_4(0.5) - x_5(0.5),$$
$$F_{n_e}(3) = -0.0327x_1(0.5) + 0.01x_2(0.5) - 0.576x_3(0.5) + 0.1x_4(0.5) - x_5(0.5).$$

To obtain $V_2(\mathbf{x}(0.5))$ as an explicit function of $\mathbf{x}(0.5)$, we have to divide the state space of $\mathbf{x}(0.5) \in \mathbb{R}^5$ into regions in which we know what the optimal mode m_2^* should be. Consider the following functions,

$$g_{n_e,1} = F_{n_e}(1) - F_{n_e}(2) = 0.019865x_1(0.5) + 0.1223x_3(0.5),$$
$$g_{n_e,2} = F_{n_e}(1) - F_{n_e}(3) = 0.033065x_1(0.5) + 0.5673x_3(0.5),$$
$$g_{n_e,3} = F_{n_e}(2) - F_{n_e}(3) = 0.0132x_1(0.5) + 0.445x_3(0.5).$$

It is clear that the following conditional statements are equivalent:

$$\text{If } F_{n_e}(1) \leq F_{n_e}(2) \text{ and } F_{n_e}(1) \leq F_{n_e}(3), \text{ then } m_2^* = 1,$$
$$\text{If } g_{n_e,1} \leq 0 \quad \text{and } g_{n_e,2} \leq 0, \quad \text{then } m_2^* = 1.$$

Hence, the feasible region for $m_2^* = 1$ is defined by the conditional statement $g_{n_e,1} \leq 0$ and $g_{n_e,2} \leq 0$. Similar conditional statements can be obtained for $m_2^* = 2$ ($g_{n_e,1} \geq 0$ and $g_{n_e,3} \leq 0$) and $m_2^* = 3$ ($g_{n_e,2} \geq 0$ and $g_{n_e,3} \geq 0$). Figure 3-6 shows how the regions of the state space of $x_1(0.5)$ and $x_3(0.5)$ are partitioned according to the modes and their corresponding conditional statements. Note that we have excluded $x_2(0.5)$, $x_4(0.5)$ and $x_5(0.5)$ because they do not participate in $g_{n_e,1}$, $g_{n_e,2}$ and $g_{n_e,3}$ (note that this is a special feature of this particular problem). We can see that if $x_1(0.5) \geq 0$ and $x_3(0.5) \geq 0$, $m_2^* = 3$, and this is easily confirmed by inspection of $g_{n_e,1}$, $g_{n_e,2}$ and $g_{n_e,2}$. Note that at the boundaries of each region, there may be more than a single choice of an optimal mode, i.e., the set $\arg\min_{m_2 \in \{1,2,3\}}$ is no longer a singleton.

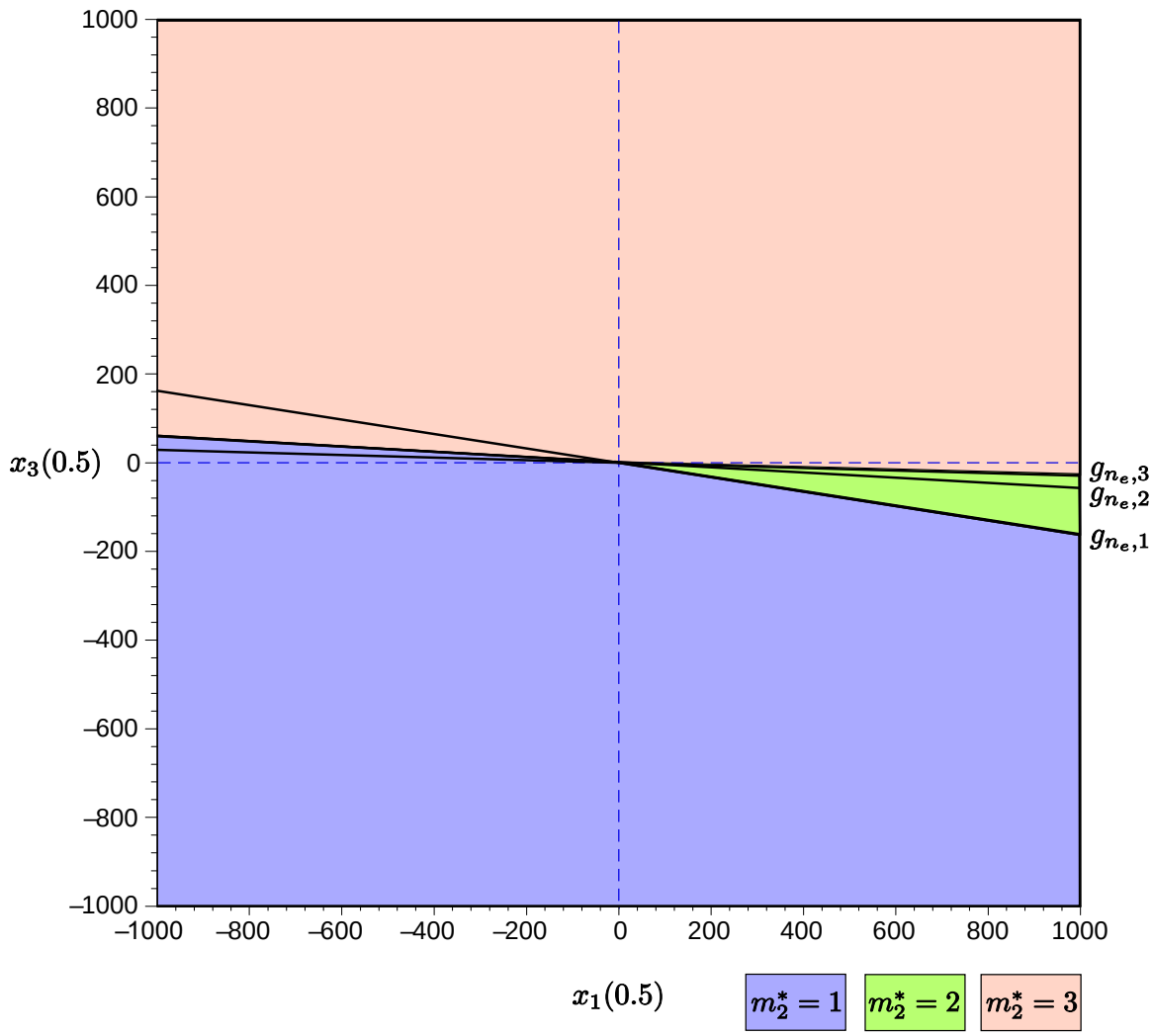


Figure 3-6: State space of $x_1(0.5)$ and $x_3(0.5)$ for $V_2(\mathbf{x}(0.5))$

From an overall mass balance, we know that $\sum_{i=1}^{5} x_i(t) = 1000$, and that $x_i(t) \geq 0$, $\forall i = 1, \ldots, n_x$, and so we can focus on the positive quadrant in Figure 3-6. Hence, $m_2^* = 3$, and we can calculate

$$V_1(\mathbf{x}(0)) = \min_{m_1 \in \{1,2,3\}} V_2(\mathbf{x}(0.5)) = \min_{m_1 \in \{1,2,3\}} F_{n_e}(3) = \min_{m_1 \in \{1,2,3\}} F_1(m_1)$$

where

$$\begin{aligned} F_1(1) &= -0.290x_1(0) + 0.01x_2(0) - 0.569x_3(0) + 0.1x_4(0) - x_5(0), \\ F_1(2) &= -0.121x_1(0) + 0.01x_2(0) - 0.609x_3(0) + 0.1x_4(0) - x_5(0), \\ F_1(3) &= -0.0483x_1(0) + 0.01x_2(0) - 0.791x_3(0) + 0.1x_4(0) - x_5(0). \end{aligned} \tag{3.26}$$

Again, we can construct the following functions,

$$\begin{aligned} g_{1,1} &= F_1(1) - F_1(2) = -0.170x_1(0) + 0.0395x_3(0), \\ g_{1,2} &= F_1(1) - F_1(3) = -0.242x_1(0) + 0.222x_3(0), \\ g_{1,3} &= F_1(2) - F_1(3) = -0.0723x_1(0) + 0.183x_3(0). \end{aligned}$$

Figure 3-7 shows how the regions of the state space of $x_1(0)$ and $x_3(0)$ are partitioned according to the optimal modes. From an initial condition of $\mathbf{x}(0) = (1000, 0, 0, 0, 0)$, it is clear that $m_1^* = 1$, and from $F_1(1)$, the optimal solution to the problem is -290. To obtain m_2^*, we integrate forward in time from $\mathbf{x}(0)$ with $m_1^* = 1$, and obtain $m_2^* = 3$ from $V_2(\mathbf{x}(0.5))$ (see Figure 3-6).

Note that Figure 3-7 is only valid assuming that $\sum_{i=1}^{n_x} x_i(t) = 1000$, and so obtaining the regions was straightforward since we knew that $m_2^* = 3$. It would be instructive to obtain the total state space without making that assumption. Consider $V_2(\mathbf{x}(0.5))$. It is clear that it has $n_m = 3$ regions corresponding to the different choices of m_2 as seen in Figure 3-6. The functional form of $V_2(\mathbf{x}(0.5))$ itself depends on m_2, and can be either $F_{n_e}(1)$, $F_{n_e}(2)$ or $F_{n_e}(3)$. Let us first consider the case where $m_2^* = 3$. We have already worked out the functions above for $g_{n_e,1}$, $g_{n_e,2}$ and $g_{n_e,3}$. However,

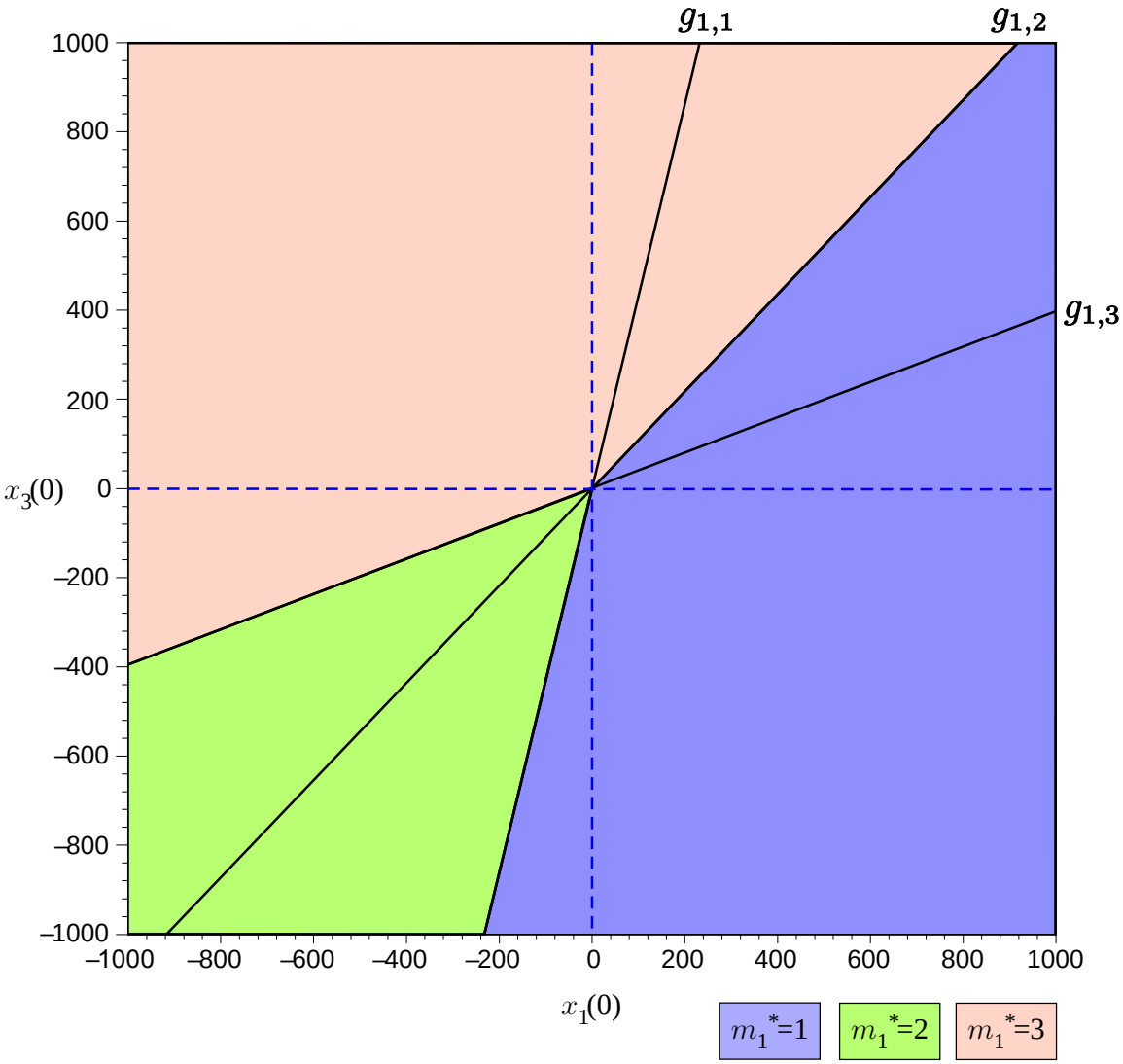


Figure 3-7: State space of $x_1(0)$ and $x_3(0)$ for $V_1(\mathbf{x}(0))$, assuming $\sum_{i=1}^{n_x} x_i(t) = 1000$.

the conditional statements will have to be changed to incorporate the condition of $m_2^* = 3$, i.e.,

$$\text{If } g_{1,1} \leq 0 \text{ and } g_{1,2} \leq 0, \text{ and } m_2^* = 3, \text{ then } m_1^* = 1$$
$$\text{If } g_{1,1} \geq 0 \text{ and } g_{1,3} \leq 0, \text{ and } m_2^* = 3, \text{ then } m_1^* = 2$$
$$\text{If } g_{1,2} \geq 0 \text{ and } g_{1,3} \geq 0, \text{ and } m_2^* = 3, \text{ then } m_1^* = 3$$

The condition of $m_2^* = 3$ is given by $g_{n_e,2} \geq 0$ and $g_{n_e,3} \geq 0$, and given m_1^*, we can express it as a function of $\mathbf{x}(0)$. For example, consider when $m_1^* = 1$, then

$$g_{n_e,2} = 0.033065x_1(0.5) + 0.5673x_3(0.5) = 0.286x_1(0) + 0.552x_3(0),$$
$$g_{n_e,3} = 0.0132x_1(0.5) + 0.445x_3(0.5) = 0.222x_1(0) + 0.433x_3(0),$$

and $V_1(\mathbf{x}(0))$ is given by (3.26). The feasible region for $T_\mu = 1, 3$ in the state space of $x_1(0)$ and $x_3(0)$ is then given by Figure 3-8. The superscript (i) denotes that the function is calculated based on the assumption of mode i being active in the current epoch. It can be seen that the conditions $g_{n_e,2} \geq 0$ and $g_{n_e,3} \geq 0$ reduce the size of the feasible region of $m_1^* = 1$ compared to that in Figure 3-7.

We can repeat the procedure for $m_1^* = 1$ and $m_1^* = 2$ to obtain the state space given by Figure 3-9. It can be seen that there is no feasible region for $T_\mu = 2, 3$. Although it is easy to see from Figure 3-9 that this region is excluded for future consideration (e.g., if $n_e > 2$), it is not trivial to obtain rules or criteria for the exclusion of such regions in general.

We can perform the same analysis for $m_2^* = 2$ and $m_2^* = 1$ to obtain the overall state space shown in Figure 3-10, where the values of $V_1(\mathbf{x}(0))$ have been included as well. Note that there are now 5 regions of the state space. Although it would seem that the regions could be combined into just two where $m_1^* = 1$ and $m_1^* = 3$ (thus reducing the number of regions in the state space), we cannot do so because the optimal value function $V_1(\mathbf{x}(0))$ is different between the regions, e.g., between the region $T_\mu = 1, 3$ and $T_\mu = 1, 2$. In the worst case, the number of regions will grow

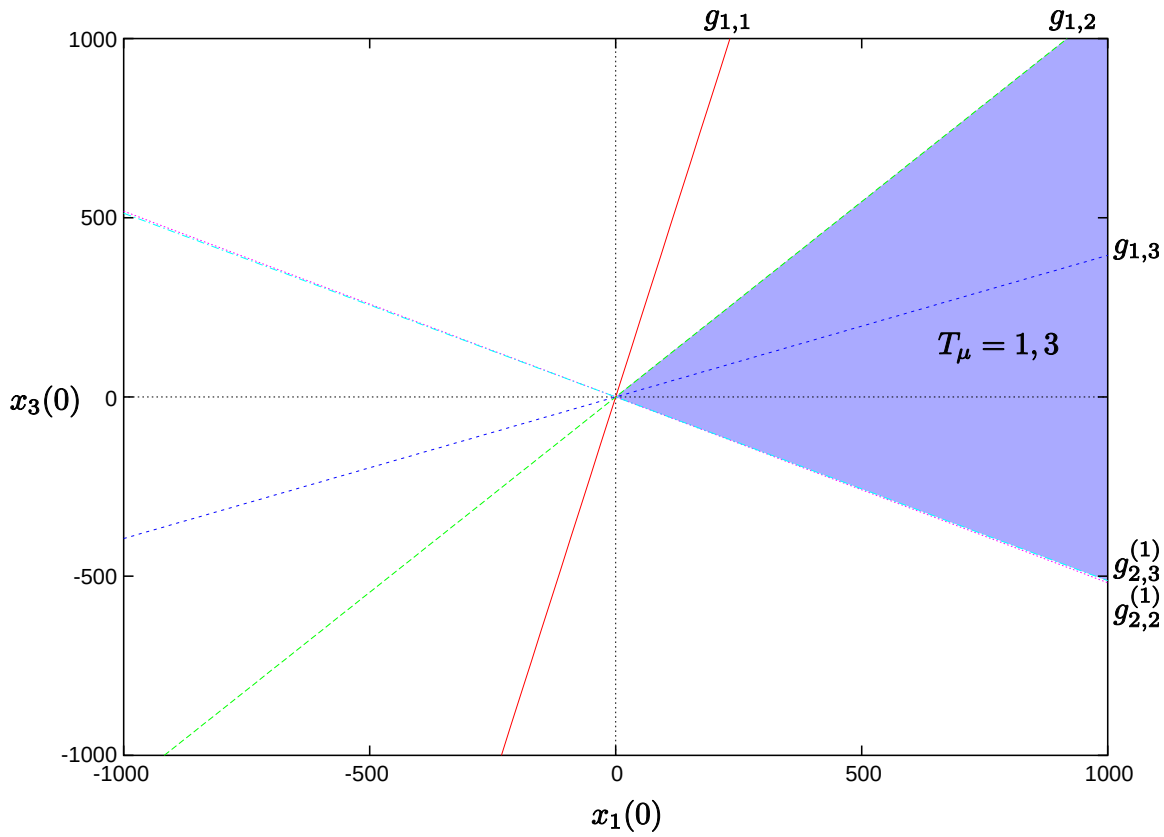


Figure 3-8: State space of $x_1(0)$ and $x_3(0)$ for $T_\mu = 1, 3$

exponentially as we propagate backward in time. In other words, we are growing the exponential tree backward in time, where we will have $n_m^{n_e}$ possible regions at time t_0. In this example, we have 9 possible regions, out of which 4 regions have been ruled out in Figure 3-10 (although we would not have known this without plotting the graph).

This exponential complexity in the number of epochs does not arise in the discretized case [26] because there is no concept of "regions" in the discretized case as there are only discretized grid points. Alternatively, one can think of the $n_d^{n_x}$ (where n_d is the number of discretization points per state variable) discrete elements (or differences) to be $n_d^{n_x}$ separate "regions", and this number of regions stays constant for each table that is constructed at each epoch. For *each* discrete point, or "region", one has to record the optimal mode for the epoch, as well as the optimal value function. This is illustrated in Figure 3-11, and summarized in the following:

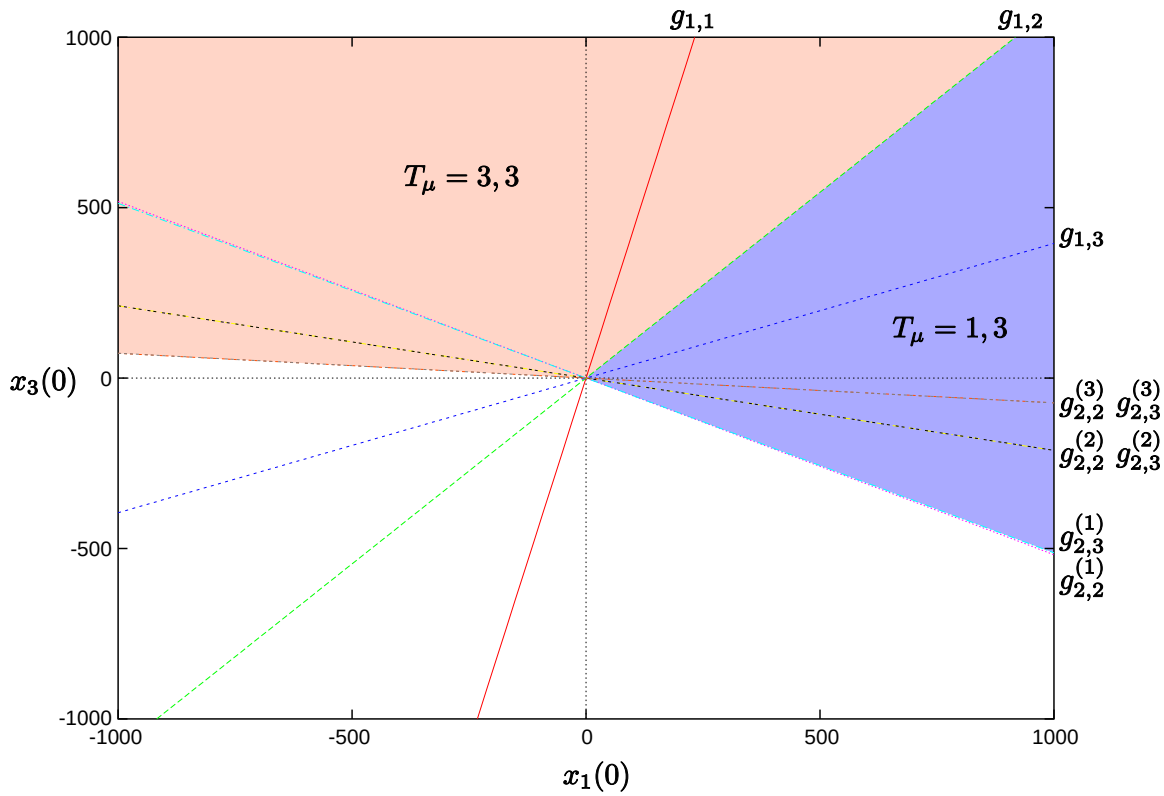


Figure 3-9: State space of $x_1(0)$ and $x_3(0)$ for $m_2^* = 3$

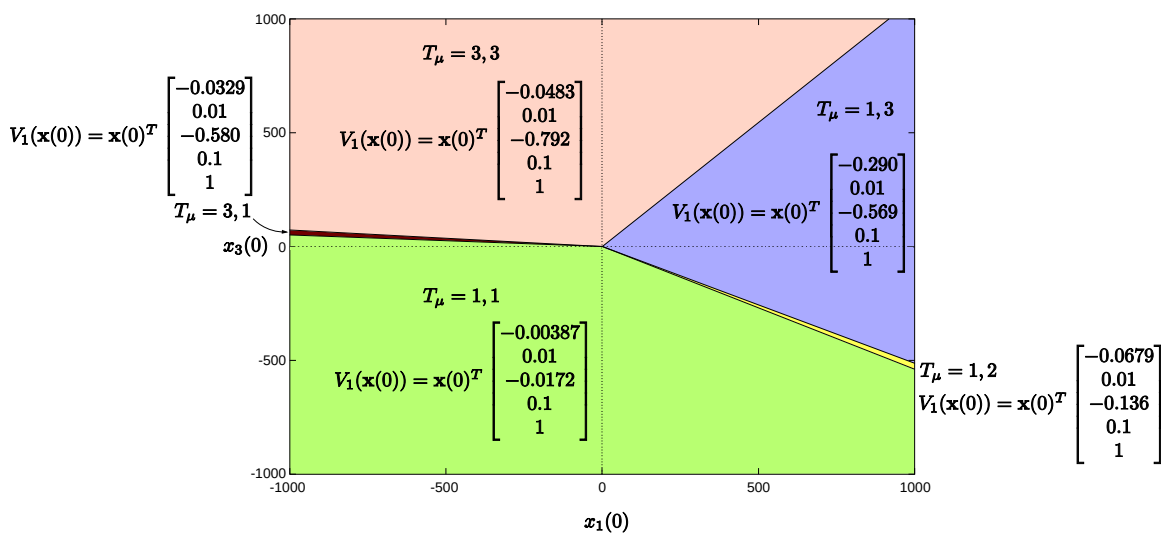


Figure 3-10: State space of $x_1(0)$ and $x_3(0)$

1. The continuous approach:

   (a) The number of regions per table increases exponentially as the number of epochs increases.

(b) Associated with each region is a (distinct) optimal value function (e.g., $V_1(\mathbf{x}(0))$ for $T_\mu = 1, 3$ is different from $V_1(\mathbf{x}(0))$ for $T_\mu = 3, 1$).

2. The state discretization approach:

   (a) The number of regions per table stays the same, and increases exponentially as the number of state variables increases.

   (b) Associated with each region is an optimal value function which is obtained by taking the minimum over all successor modes (a finite number of modes, bounded by n_m).

   (c) Could potentially lead to qualitatively wrong sequences when a discretized region includes a boundary between two regions in the continuous case.

An algorithm for the continuous approach based on the backward propagation of the tree is shown in Figure 3-12. Note that steps 1 and 2 in the main program are redundant, but are included to illustrate the idea of working backwards in time using the dynamic programming approach. The exponential complexity cannot be avoided because the optimal value function $V_i(\mathbf{x})$ changes according to *all* possible mode trajectories that come after the current epoch i (that make up all possible discrete combinations).

In the case of the reactor example, applying the algorithm shown in Figure 3-14 is equivalent to a brute force enumeration of the tree in the forward direction. The fact that we know all states are bounded below by zero will not help in reducing the complexity of the execution of the algorithm (neither does the fact that mass must be conserved at all times).

3.2.4 Elimination of Regions that are Linearly Bounded

In this section, we discuss how "dead" regions can potentially be identified and eliminated from the dynamic programming tree as we propagate backward in time, at the expense of solving infeasibility problems as LPs. First, we define a "dead" region to be one that can be removed from consideration when calculating the optimal value

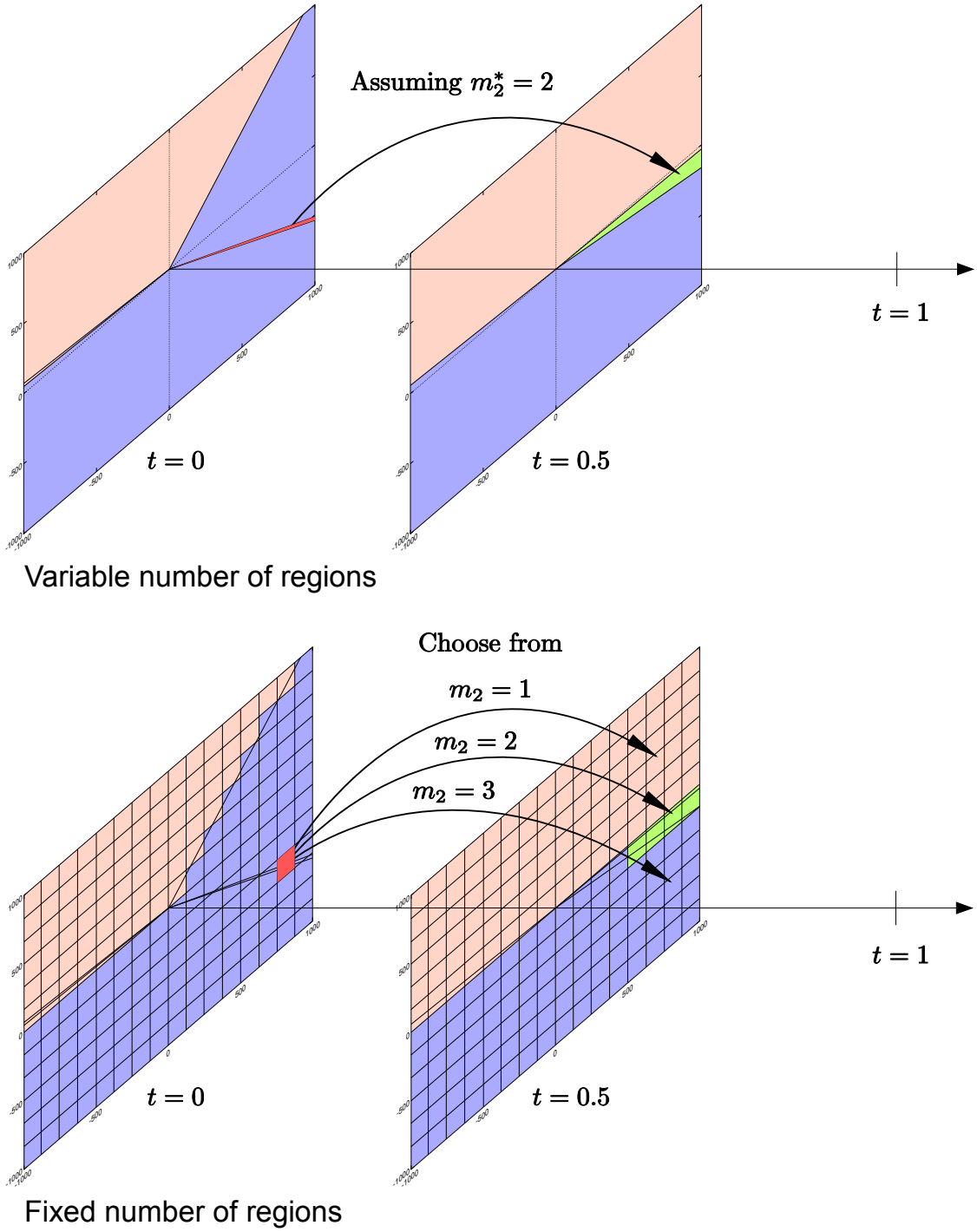


Figure 3-11: Illustration of state discretization and continuous dynamic programming approaches

Main

---

1. Set $i = n_e$.
2. Do while $i \geq 1$.
   (a) Construct subroutine $S_i(\mathbf{x}_0)$ that returns m_i^* and $V_i(\mathbf{x}_0)$.
   (b) Set $i = i - 1$.
3. Run subroutine $S_1(\mathbf{x}(0))$. The optimal mode trajectory is given by $T_\mu^* = \{m_i^*\}$, and the optimal solution is given by $V_1(\mathbf{x}(0))$.

Subroutine $S_i(\mathbf{x}_0)$

---

1. Set $m_i^* = j = 1$, $V_i(\mathbf{x}_0) = +\infty$.
2. Do while $j \leq n_m$
   (a) Calculate

$$F_i^{(j)} = \phi_i\Big(\dot{\mathbf{x}}(\tau_i; j), \mathbf{x}(\tau_i; j), \tau_i\Big) + \int_{\sigma_i}^{\tau_i} f_i\Big(\dot{\mathbf{x}}(t; j), \mathbf{x}(t; j), t\Big)\ \mathrm{d}t$$

   by integrating forward in time in epoch i according to the dynamics of mode j with initial conditions $\mathbf{x}_0$.
   (b) If $i < n_e$, call subroutine $S_{i+1}(\mathbf{x}(\tau_i; j))$.
   (c) If $i < n_e$, set $Z = F_i^{(j)} + V_{i+1}(\mathbf{x}(\tau_i; j))$, else $Z = F_i^{(j)}$.
   (d) If $Z < V_i(\mathbf{x}_0)$, set $m_i^* := j$ and $V_i(\mathbf{x}_0) := Z$.
   (e) Set $j = j + 1$.
3. Return m_i^* and $V_i(\mathbf{x}_0)$.

Figure 3-12: Algorithm for continuous dynamic programming approach

functions V_i in the continuous setting. For example, in Figure 3-10, the regions corresponding to the trajectories $T_\mu = 2,1;\ 2,2;\ 2,3;3,2$ are "dead" and can removed from further consideration when calculating the optimal value function for the preceding epochs. Note that this does not mean that the said trajectories are infeasible (although it might be tempting to think of it that way for verification purposes), as it only means that these trajectories will not produce optimal solutions anywhere in the state space of interest. As another example, consider Figure 3-6. The regions $m_2^* = 1$ and $m_2^* = 2$ would be considered "dead" regions if the constraints $x_i \geq 0$ are considered as the state space of interest.

The proposed method comes naturally from posing the following feasibility question: Is the region formed by a given set of inequality constraints an empty set? Let us illustrate this by revisiting the examples shown in the previous section. Consider again Example 3.7 with $n_e = 2$ and $\tau_1 = 0.5$. We have already obtained the feasible region for $m_2^* = 1$ in the previous section. We can pose the feasibility question as the following LP.

Problem 3.13.

$$\min_{\mathbf{x}\in\mathbb{R}^5} \ x_1$$

$$\text{s.t.} \quad 0.019865x_1 + 0.1223x_3 \leq 0 \tag{3.27}$$

$$0.033065x_1 + 0.5673x_3 \leq 0 \tag{3.28}$$

$$\mathbf{x} \geq \mathbf{0} \tag{3.29}$$

Note that the objective function is arbitrary since we are only interested in whether the LP has a feasible solution or not. Alternatively, one can simply carry out the Phase I algorithm of the simplex method [27, page 116] to determine the feasibility of the region. However, when we solve this problem using CPLEX [78], we get a solution at $\mathbf{x} = (0,0,0,0,1000)$, and so we cannot eliminate the region. It appears that the problem here is that the origin $x_1 = 0$, $x_3 = 0$ is the only feasible solution to the problem. One way to mitigate this problem is to replace (3.29) with the following

constraints for some small $\varepsilon > 0$,

$$x_i \geq \varepsilon, \ \forall i = 1, \ldots, 5.$$

Note that here, we actually have to make use of the fact that we know $x_i \geq 0$. To attempt to exclude only a small region around the origin with the constraints $|x_i| \geq \varepsilon$ will not eliminate the region as x_i will then be allowed to take on negative values. When we solve this problem again in CPLEX with a value of $\varepsilon = 10^{-4}$, it tells us that the presolve stage determines that the problem is infeasible or unbounded, which does not really help us in eliminating the region.

We then have to directly use the Phase I algorithm mentioned above. To do this, we first convert the inequality constraints to equality constraints through adding auxiliary variables $\mathbf{y} \geq \mathbf{0}$ into (3.27) and (3.28),

$$\begin{aligned} 0.019865x_1 + 0.1223x_3 + y_1 &= 0, \\ 0.033065x_1 + 0.5673x_3 + y_2 &= 0. \end{aligned}$$

Next, we introduce auxiliary variables $\mathbf{z} \geq \mathbf{0}$ into (3.27) and (3.28),

$$\begin{aligned} 0.019865x_1 + 0.1223x_3 + y_1 + z_1 &= 0, \\ 0.033065x_1 + 0.5673x_3 + y_2 + z_2 &= 0, \end{aligned}$$

and replace the objective function with

$$z_1 + z_2.$$

Finally, when we solve this problem in CPLEX, it again returns the information that presolve stage determines that the problem is infeasible or unbounded. However, now that we know the optimal solution is non-zero, we can safely eliminate the region from consideration.

It is clear that this method can be applied to any of the other regions to be

considered, as long as the regions are linearly bounded. As we propagate backwards in time, the number of inequality constraints in the feasibility problem increases with $\mathcal{O}(n_m n_e)$. We will end this section with the following remarks:

1. This method only works for regions bounded by linear inequality constraints. Although outer approximation type techniques could theoretically be applied for regions defined by nonlinear or nonconvex constraints, it is hypothesized that very few dead regions could potentially be identified to justify the cost of solving those outer approximation problems, as the relaxation of these nonlinear regions would most likely overlap in some region of state space.

2. Thus far, we have only considered the case where the regions are excluded based on optimality of the objective function (optimal value function). It is not clear at present how this can be extended to handle constraints, although it seems that linear constraints can be incorporated into the feasibility subproblems easily.

3.3 A Hybrid Superstructure - Mixed-Integer Reformulation

Problem 3.4 is combinatorial in T_μ. A possible approach to solve the problem is to fix $T_\mu = T_\mu^*$ and apply Algorithm 2.3 to the smooth subproblem in order to obtain the global solution for T_μ^*; this is then repeated for all possible $n_m^{n_e}$ mode sequences to determine the global minimum to Problem 3.4. Conventional wisdom dictates that this explicit enumeration approach greatly limits the size of problems that can be handled, and should be avoided. We will discuss the performance of the explicit enumeration approaches in greater detail in Section 3.6.

The application of mathematical programming techniques for mixed-integer nonlinear programming problems (MINLPs) [50, 59, 60, 117, 83] is proposed as a much more attractive approach for solving Problem 3.4. Although the worst case performance of any such algorithm will approach explicit enumeration of all possible mode sequences, it should outperform explicit enumeration in many instances due to the

ability to exclude entire classes of mode sequences rigorously, thus reducing the number of subproblems that needs to be solved.

We start with the introduction of binary decision variables to represent T_μ. In particular, we introduce $n_m n_e$ binary variables $y_{mi}, m = 1, \ldots, n_m, i = 1, \ldots, n_e$. By adding the following logical constraints, we ensure that exactly one mode is active in each epoch,

$$\sum_{m=1}^{n_m} y_{mi} = 1, \quad \forall i = 1, \ldots, n_e.$$

Hence, if m_i^* is the active mode in epoch I_i, then $y_{m_i^*,i} = 1$ and $y_{m \neq m_i^*,i} = 0$, for all $i = 1, \ldots, n_e$. This transforms Problem 3.4 into the following mixed-integer dynamic optimization (MIDO) problem.

Problem 3.14.

$$\min_{\mathbf{p} \in P, \mathbf{Y} \in Y^b} F(\mathbf{p}, \mathbf{Y}) \equiv \sum_{i=1}^{n_e} \Big(\phi_i\big(\dot{\mathbf{x}}(\mathbf{p}, \mathbf{Y}, \tau_i), \mathbf{x}(\mathbf{p}, \mathbf{Y}, \tau_i), \mathbf{p}\big) + \int_{\sigma_i}^{\tau_i} f_i\big(\dot{\mathbf{x}}(\mathbf{p}, \mathbf{Y}, t), \mathbf{x}(\mathbf{p}, \mathbf{Y}, t), \mathbf{p}, t\big)\, \mathrm{d}t \Big), \quad (3.30)$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{p}, \mathbf{Y}) \equiv \sum_{i=1}^{n_e} \Big(\boldsymbol{\eta}_i\big(\dot{\mathbf{x}}(\mathbf{p}, \mathbf{Y}, \tau_i), \mathbf{x}(\mathbf{p}, \mathbf{Y}, \tau_i), \mathbf{p}\big) + \int_{\sigma_i}^{\tau_i} \mathbf{g}_i\big(\dot{\mathbf{x}}(\mathbf{p}, \mathbf{Y}, t), \mathbf{x}(\mathbf{p}, \mathbf{Y}, t), \mathbf{p}, t\big)\, \mathrm{d}t \Big) \leq \mathbf{0}, \quad (3.31)$$

$$\sum_{m=1}^{n_m} y_{mi} = 1, \quad \forall i = 1, \ldots, n_e, \quad (3.32)$$

where $\mathbf{x}(\mathbf{p}, \mathbf{Y}, t)$ is given by the solution of the embedded LTV ODE hybrid system (Definition 3.1) with (3.2), (3.4) and (3.5) replaced by the following equations,

$$\dot{\mathbf{x}}(\mathbf{p}, \mathbf{Y}, t) = \sum_{m=1}^{n_m} y_{mi} \Big(\mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{p}, \mathbf{Y}, t) + \mathbf{B}^{(m)}(t)\mathbf{p} + \mathbf{q}^{(m)}(t) \Big), \ \forall t \in (\sigma_i, \tau_i], \ i = 1, \ldots, n_e, \quad (3.33)$$

$$\mathbf{x}(\mathbf{p},\mathbf{Y},\sigma_{i+1}) = \sum_{j=1}^{n_m}\sum_{k=1}^{n_m} y_{ji}y_{ki+1}\Big(\mathbf{D}_i(j,k)\mathbf{x}(\mathbf{p},\mathbf{Y},\tau_i) + \mathbf{E}_i(j,k)\mathbf{p} + \mathbf{k}_i(j,k)\Big),\ \forall i = 1,\ldots,n_e-1, \quad (3.34)$$

$$\mathbf{x}(\mathbf{p},\mathbf{Y},\sigma_1) = \mathbf{E}_0\mathbf{p} + \mathbf{k}_0, \quad (3.35)$$

$Y^b \equiv \{0,1\}^{n_m\times n_e} \subset Y \equiv [0,1]^{n_m\times n_e}$; f_i, $\mathbf{g}_i$, ϕ_i and $\boldsymbol{\eta}_i$ are mappings as defined in Problem 3.4. Additionally, we require that the set $G = \{(\mathbf{p},\mathbf{Y}) \in P\times Y^b \mid \mathbf{G}(\mathbf{p},\mathbf{Y}) \leq \mathbf{0}\}$ is nonempty.

Equations (3.32), (3.33), (3.34) and (3.35) clearly establish the following proposition.

Proposition 3.15. *Problem 3.4 has the solution* $(\mathbf{p}^*, T_\mu^*)$ *if and only if* $(\mathbf{p}^\dagger, \mathbf{Y}^\dagger)$ *is a solution to Problem 3.14, where* $\mathbf{p}^* = \mathbf{p}^\dagger$ *and* $T_\mu^* = \{m_i \mid y_{m_i i}^\dagger = 1\}$.

Having established a MINLP formulation in Problem 3.14, the aforementioned algorithms for solving MINLPs can be applied. However, the following issues need to be addressed. When the objective (3.30) and constraint functionals (3.31) are nonconvex on $P \times Y$, rigorous global optimization algorithms have to be employed, e.g., [117, 83], because conventional algorithms which rely on convexity of the participating functions on $P \times Y$ can potentially generate arbitrary suboptimal solutions. These global optimization algorithms rely on the ability to construct convex relaxations of the objective and constraint functionals. Note that because bilinear terms appear in (3.33) this would require a theory for the construction of such convex relaxations with nonlinear hybrid systems embedded. The LTV structure of the embedded hybrid system, (3.2), in Problem 3.4 has been lost in the reformulation into the bilinear form, (3.33), of Problem 3.14, whereas it would be highly desirable to exploit the LTV structure of Problem 3.4, as in Chapter 2. These considerations motivate the following reformulation of Problem 3.14 that retains the LTV structure of Problem 3.4, and transfers the nonlinearity of Problem 3.14 from the embedded dynamic system into the objective and constraint functionals.

Before we proceed, we will have to define the following sets, based on the sets defined in Definition 3.2.

Definition 3.16. Define the following sets for all $i = 1, \ldots, n_e$:

$$\mathcal{X}^{a(i)}(t; P) = [\mathbf{v}, \mathbf{w}] \mid \mathbf{v} \leq \mathbf{z} \leq \mathbf{w}, \quad \forall \mathbf{z} \in \mathcal{X}^{(i)}(t; P), \ \forall t \in I_i,$$
$$\dot{\mathcal{X}}^{a(i)}(t; P) = [\mathbf{v}, \mathbf{w}] \mid \mathbf{v} \leq \mathbf{z} \leq \mathbf{w}, \quad \forall \mathbf{z} \in \dot{\mathcal{X}}^{(i)}(t; P), \ \forall t \in I_i.$$

Note that these sets are nonempty, compact and convex because they are defined to be interval vectors, while the sets defined in Definition 3.2 are not necessarily convex because they may be disjoint. While the sets defined in Definition 3.2 characterize the exact image of $(\mathbf{p}, T_\mu, t) \in P \times M^{n_e} \times I_i$ under the solution of the hybrid system for all $i = 1, \ldots, n_e$, the sets defined in Definition 3.16 represent a convex relaxation of the sets in Definition 3.2, i.e., they represent a conservative estimate of the exact image under the solution of the hybrid system. These convex sets are important, because they enable a convex, compact set, Z, to be constructed for the auxiliary variables that will be introduced in the reformulation to be presented. This in turn enables a convex relaxation to be constructed for each linear dynamic system that will be introduced, because the set Z contains bounds for the initial conditions for each dynamic system, and will thus allow a convex relaxation to be constructed according to the theory presented in Section 2.6.

Problem 3.17.

$$\min_{\mathbf{p} \in P, \mathbf{Y} \in Y^b, \ \mathbf{Z} \in Z} \quad F(\mathbf{p}, \mathbf{Y}, \mathbf{Z}) \equiv \sum_{m=1}^{n_m} \sum_{i=1}^{n_e} y_{mi} \Bigg(\phi_i \Big(\dot{\mathbf{x}}_{mi}(\mathbf{p}, \mathbf{Z}, \tau_i), \mathbf{x}_{mi}(\mathbf{p}, \mathbf{Z}, \tau_i), \mathbf{p} \Big) + \int_{\sigma_i}^{\tau_i} f_i \Big(\dot{\mathbf{x}}_{mi}(\mathbf{p}, \mathbf{Z}, t), \mathbf{x}_{mi}(\mathbf{p}, \mathbf{Z}, t), \mathbf{p}, t \Big) \, \mathrm{d}t \Bigg), \quad (3.36)$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{p},\mathbf{Y},\mathbf{Z}) \equiv \sum_{m=1}^{n_m}\sum_{i=1}^{n_e} y_{mi}\Bigg(\boldsymbol{\eta}_i\Big(\dot{\mathbf{x}}_{mi}(\mathbf{p},\mathbf{Z},\tau_i),\mathbf{x}_{mi}(\mathbf{p},\mathbf{Z},\tau_i),\mathbf{p}\Big) + \int_{\sigma_i}^{\tau_i} \mathbf{g}_i\Big(\dot{\mathbf{x}}_{mi}(\mathbf{p},\mathbf{Z},t),\mathbf{x}_{mi}(\mathbf{p},\mathbf{Z},t),\mathbf{p},t\Big)\,\mathrm{d}t\Bigg) \leq \mathbf{0}, \quad (3.37)$$

$$\sum_{m=1}^{n_m} y_{mi} = 1, \quad \forall i = 1,\ldots,n_e, \quad (3.32)$$

$$\mathbf{z}_{i+1} = \sum_{j=1}^{n_m}\sum_{k=1}^{n_m} y_{ji}y_{ki+1}\Big(\mathbf{D}_i(j,k)\mathbf{x}_{ji}(\mathbf{p},\mathbf{Z},\tau_i) + \mathbf{E}_i(j,k)\mathbf{p} + \mathbf{k}_i(j,k)\Big), \ \forall i = 1,\ldots,n_e-1, \quad (3.38)$$

where $Z \equiv \mathcal{X}^{a(1)}(\sigma_1;P)\times\mathcal{X}^{a(2)}(\sigma_2;P)\times\cdots\times\mathcal{X}^{a(n_e)}(\sigma_{n_e};P) \subset \mathbb{R}^{n_x\times n_e}$ and $\mathbf{x}_{mi}(\mathbf{p},\mathbf{Z},t)$ are given by the solutions of the following embedded LTV ODE systems,

$$\dot{\mathbf{x}}_{mi}(\mathbf{p},\mathbf{Z},t) = \mathbf{A}^{(m)}(t)\mathbf{x}_{mi}(\mathbf{p},\mathbf{Z},t) + \mathbf{B}^{(m)}(t)\mathbf{p} + \mathbf{q}^{(m)}(t), \ \forall t \in (\sigma_i,\tau_i], \ m \in M, \ i = 1,\ldots,n_e, \quad (3.39)$$

$$\mathbf{x}_{mi}(\mathbf{p},\mathbf{Z},\sigma_i) = \mathbf{z}_i, \quad \forall m \in M, \ i = 1,\ldots,n_e, \quad (3.40)$$

$$\mathbf{z}_1 = \mathbf{E}_0\mathbf{p} + \mathbf{k}_0. \quad (3.41)$$

f_i, $\mathbf{g}_i$, ϕ_i and $\boldsymbol{\eta}_i$ are mappings as defined in Problem 3.4. Additionally, we require that the set $G = \{(\mathbf{p},\mathbf{Y},\mathbf{Z}) \in P\times Y^b\times Z \mid \mathbf{G}(\mathbf{p},\mathbf{Y},\mathbf{Z}) \leq \mathbf{0}\}$ is nonempty.

Remark. When state continuity is enforced for all transition functions in (3.4), (3.38) simplifies to

$$\mathbf{z}_{i+1} = \sum_{j=1}^{n_m} y_{mi}\mathbf{x}_{mi}(\mathbf{p},\mathbf{Z},\tau_i), \ \forall i = 1,\ldots,n_e-1. \quad (3.42)$$

The key step in the reformulation of Problem 3.14 into Problem 3.17 is the introduction of additional continuous parameters $\mathbf{Z} = (\mathbf{z}_1,\ldots,\mathbf{z}_{n_e})$ to serve as initial conditions for each epoch, enabling the transformation of the nonlinear hybrid sys-

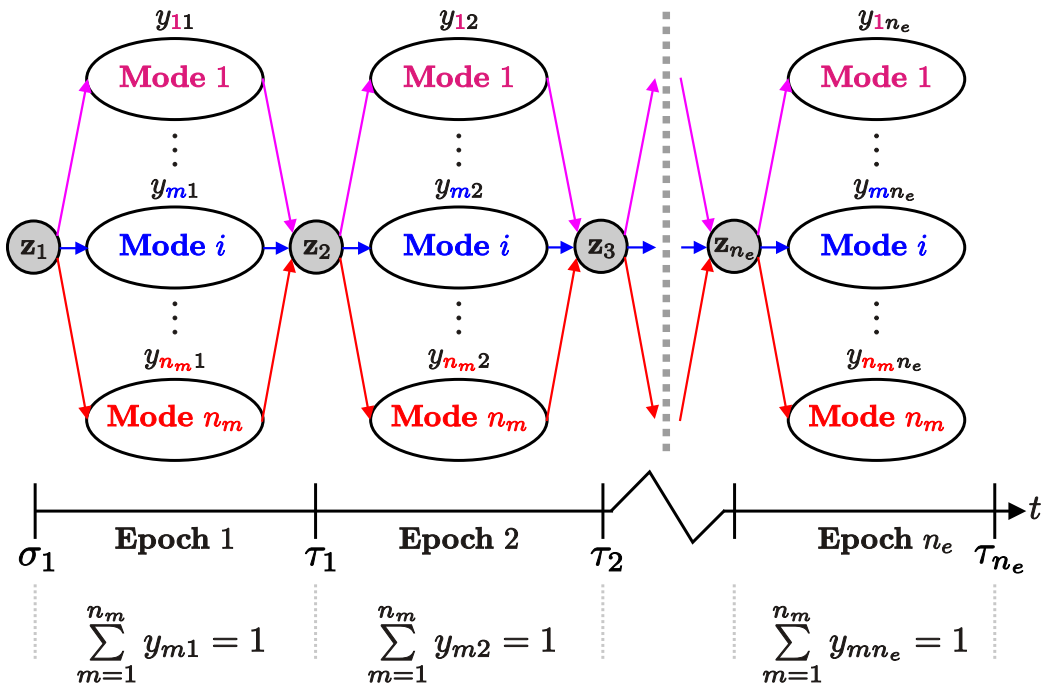


Figure 3-13: Superstructure for Problem 3.17.

tem into $n_m n_e$ equivalent LTV dynamic systems. Equation (3.38) then assigns the correct values for the initial conditions of epoch I_{i+1}, depending on the predecessor mode m_i that is active, which is enforced by (3.32). As discussed above, the set Z is constructed to be a convex relaxation of the image of $P \times M^{m_e}$ under the solution of the hybrid system, and thus (3.38) is always guaranteed to be feasible. This superstructure is illustrated in Figure 3-13. Equations (3.32), (3.38), (3.39), (3.40) and (3.41) clearly establish the equivalence of Problem 3.14 and Problem 3.17, expressed as the following proposition.

Proposition 3.18. $(\mathbf{p}^*, \mathbf{Y}^*, \mathbf{Z}^*)$ *is a solution to Problem 3.17 if and only if* $(\mathbf{p}^\dagger, \mathbf{Y}^\dagger)$ *is a solution to Problem 3.14, where* $\mathbf{p}^* = \mathbf{p}^\dagger$ *and* $\mathbf{Y}^* = \mathbf{Y}^\dagger$.

Henceforth, we will concentrate on solving Problem 3.17.

3.3.1 Constructing Convex Relaxations

In order to solve Problem 3.17 using the aforementioned deterministic global optimization algorithms for nonconvex MINLPs, the ability to construct convex relaxations

of (3.36) and (3.37) subject to the embedded system (3.38), (3.40) and (3.41) is key. Consider the following subproblem.

Problem 3.19.

$$\min_{\mathbf{p}\in P,\tilde{y}\in\tilde{Y},\tilde{\mathbf{z}}\in\tilde{Z}} \quad F(\mathbf{p},\tilde{y},\tilde{\mathbf{z}}) \equiv \phi\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{p},\tilde{y}\Big) + \int_{\sigma}^{\tau} f\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{p},\tilde{y},t\Big)\,\mathrm{d}t, \quad (3.43)$$

$$\text{s.t.} \quad \dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},t) = \mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},t) + \mathbf{B}^{(m)}(t)\mathbf{p} + \mathbf{q}^{(m)}(t), \ \forall t \in (\sigma,\tau] \quad (3.44)$$

$$\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},\sigma) = \tilde{\mathbf{z}}, \quad (3.45)$$

for some $m \in M$, where $\sigma < \tau, T \equiv [\sigma,\tau]$, $P \subset \mathbb{R}^{n_p}$, $\tilde{Y} \equiv [0,1]$, $\tilde{Z} \subset \mathbb{R}^{n_x}$; f is a piecewise continuous mapping with a finite number of stationary simple discontinuities in time, $f : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P \times \tilde{Y} \times T \to \mathbb{R}$; ϕ is a continuous mapping $\phi : \mathbb{R}^{n_x} \times \mathbb{R}^{n_x} \times P \times \tilde{Y} \to \mathbb{R}$.

Definition 3.20. Let P be a nonempty compact convex subset of $\mathbb{R}^{n_p}$, and $\tilde{Z}$ be a nonempty compact convex subset of $\mathbb{R}^{n_x}$. Define the following sets:

$$\mathcal{X}^b(t;P,\tilde{Z}) \equiv \big\{\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},t) \mid \mathbf{p}\in P, \tilde{\mathbf{z}}\in\tilde{Z}\big\}, \ \forall t\in T,$$
$$\dot{\mathcal{X}}^b(t;P,\tilde{Z}) \equiv \big\{\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},t) \mid \mathbf{p}\in P, \tilde{\mathbf{z}}\in\tilde{Z}\big\}, \ \forall t\in T,$$
$$\mathcal{X}^b(P,\tilde{Z}) \equiv \bigcup_{t\in T}\mathcal{X}^b(t;P,\tilde{Z}), \quad \dot{\mathcal{X}}^b(P,\tilde{Z}) \equiv \bigcup_{t\in T}\dot{\mathcal{X}}^b(t;P,\tilde{Z}).$$

The following results demonstrate how the convexity theory presented in Section 2.6 can be used to establish a convexity theory for Problem 3.19.

Theorem 3.21. *Consider the function F in Problem 3.19. If $f(\cdot,t)$ is convex on $\dot{\mathcal{X}}^b(t;P,\tilde{Z}) \times \mathcal{X}^b(t;P,\tilde{Z}) \times P \times \tilde{Y}$ for each $t \in T$, and ϕ is convex on $\dot{\mathcal{X}}^b(\tau;P,\tilde{Z}) \times \mathcal{X}^b(\tau;P,\tilde{Z}) \times P \times \tilde{Y}$, then F is convex on $P \times \tilde{Y} \times \tilde{Z}$.*

Proof. Let $\mathbf{w} = (\mathbf{p},\tilde{y},\tilde{\mathbf{z}})$. The embedded LTV system in (3.44) and (3.45) can then

be written as the following equivalent system,

$$\dot{\mathbf{x}}(\mathbf{w},t) = \mathbf{A}^{(m)}(t)\mathbf{x}(\mathbf{w},t) + \mathbf{H}^{(m)}(t)\mathbf{w} + \mathbf{q}^{(m)}(t),\ \forall t \in (\sigma,\tau], \tag{3.46}$$

$$\mathbf{x}(\mathbf{w},\sigma) = \mathbf{E}\mathbf{w}, \tag{3.47}$$

where $\mathbf{H}^{(m)}(t) = [\mathbf{B}^{(m)}(t)\ \mathbf{0}]$, $\mathbf{E} = [\mathbf{0}\ \mathbf{I}]$, and $\mathbf{I}$ is the identity matrix of rank n_x; $\mathbf{w} \in W \equiv P \times \tilde{Y} \times \tilde{Z}$. The system (3.46) and (3.47) is a trivial example of the LTV hybrid systems considered in Section 2.6, hence Theorem 2.12 can be applied to obtain the desired result. □

Remark. The set $\dot{\mathcal{X}}^b(t;P,\tilde{Z}) \times \mathcal{X}^b(t;P,\tilde{Z}) \times P \times \tilde{Y}$ is convex for each $t \in T$ (see proof of Theorem 2.12).

Corollary 3.22. *Consider the following function:*

$$U(\mathbf{p},\tilde{y},\tilde{\mathbf{z}};P,\tilde{Z}) = \psi\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{p},\tilde{y}\Big) + \int_\sigma^\tau u\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{p},\tilde{y},t\Big)\,\mathrm{d}t, \tag{3.48}$$

subject to the constraints of Problem 3.19, u is a piecewise continuous mapping with a finite number of stationary simple discontinuities in time, $u : \dot{\mathcal{X}}^b(P,\tilde{Z}) \times \mathcal{X}^b(P,\tilde{Z}) \times P \times \tilde{Y} \times T \to \mathbb{R}$; and ψ is a continuous mapping $\psi : \dot{\mathcal{X}}^b(\tau;P,\tilde{Z}) \times \mathcal{X}^b(\tau;P,\tilde{Z}) \times P \times \tilde{Y} \to \mathbb{R}$. If the following conditions are satisfied, then U is convex on $P \times \tilde{Y} \times \tilde{Z}$ such that $U(\mathbf{p},\tilde{y},\tilde{\mathbf{z}};P,\tilde{Z}) \le F(\mathbf{p},\tilde{y},\tilde{\mathbf{z}})$, $\forall(\mathbf{p},\tilde{y},\tilde{\mathbf{z}}) \in P \times \tilde{Y} \times \tilde{Z}$.

(D1) $\psi\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{p},\tilde{y}\Big) \le \phi\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},\tau),\mathbf{p},\tilde{y}\Big)$, $\forall(\mathbf{p},\tilde{y},\tilde{\mathbf{z}}) \in P \times \tilde{Y} \times \tilde{Z}$,

(D2) $u\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{p},\tilde{y},t\Big) \le f\Big(\dot{\mathbf{x}}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{x}(\mathbf{p},\tilde{\mathbf{z}},t),\mathbf{p},\tilde{y},t\Big)$, $\forall(\mathbf{p},\tilde{y},\tilde{\mathbf{z}}) \in P \times \tilde{Y} \times \tilde{Z}$, *for each* $t \in T$,

(D3) ψ *is convex on* $\dot{\mathcal{X}}^b(\tau;P,\tilde{Z}) \times \mathcal{X}^b(\tau;P,\tilde{Z}) \times P \times \tilde{Y}$,

(D4) $u(\cdot,t)$ *is convex on* $\dot{\mathcal{X}}^b(t;P,\tilde{Z}) \times \mathcal{X}^b(t;P,\tilde{Z}) \times P \times \tilde{Y}$ *for each* $t \in T$.

Proof. See Corollary 2.13. □

Remark. Corollary 3.22 allows convex relaxations of (3.43) to be constructed, by harnessing McCormick's composition theorem and αBB [96, 1] to build the required convex relaxations, ψ from ϕ, and $u(\cdot, \underline{t})$ from $f(\cdot, \underline{t})$ for all $\underline{t} \in T$ in (3.43). Note also that the functional form of U includes a dependence on the parameter set, $P \times \tilde{Z}$, which is used for its construction.

Remark. Problem 3.19 is a subproblem obtained by considering one of the $n_m n_e$ embedded systems in Problem 3.17, i.e., by fixing some $m \in M$ and $i \in \{1, \ldots, n_e\}$. Thus, the repeated application of Corollary 3.22 for all $n_m n_e$ embedded systems allows convex relaxations of (3.36) and (3.37) to be constructed, i.e., the construction of a lower bounding convex MINLP to Problem 3.17. This is important as it allows the deterministic global optimization algorithm described later in this chapter to be applied for the solution of Problem 3.17.

For Corollary 3.22, the sets P, $\tilde{Y}$ and T are known exactly. Next, we show how the sets $\dot{\mathcal{X}}^b(t; P, \tilde{Z})$ and $\mathcal{X}^b(t; P, \tilde{Z})$, $t \in T$, can be calculated from some given interval vector $\tilde{Z}$. Recall that the interval notation $\mathbf{a} \in [\mathbf{a}^L, \mathbf{a}^U]$ indicates that $\mathbf{a}^L \leq \mathbf{a} \leq \mathbf{a}^U$.

Theorem 3.23. *Consider Problem 3.19. Given intervals $P \equiv [\mathbf{p}^L, \mathbf{p}^U]$ and $\tilde{Z} \equiv [\tilde{\mathbf{z}}^L, \tilde{\mathbf{z}}^U]$, the convex set $\mathcal{X}^b(t; P, \tilde{Z}) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)]$ for $t \in T$ can be calculated pointwise in time from the following interval equation,*

$$[\mathbf{x}](t) = \mathbf{M}(t)[\mathbf{w}] + \mathbf{n}(t), \tag{3.49}$$

where $\mathbf{w} = (\mathbf{p}, \tilde{y}, \tilde{\mathbf{z}})$, $\mathbf{w} \in W \equiv [\mathbf{w}^L, \mathbf{w}^U]$, $\mathbf{w}^L = (\mathbf{p}^L, 0, \tilde{\mathbf{z}}^L)$, $\mathbf{w}^U = (\mathbf{p}^U, 1, \tilde{\mathbf{z}}^U)$, and $\mathbf{M}(t)$, $\mathbf{n}(t)$ are given by the solution of the following LTV systems,

$$\dot{\mathbf{M}}(t) = \mathbf{A}^{(m)}(t)\mathbf{M}(t) + \mathbf{H}^{(m)}(t), \ \forall t \in (\sigma, \tau], \tag{3.50}$$

$$\dot{\mathbf{n}}(t) = \mathbf{A}^{(m)}(t)\mathbf{n}(t) + \mathbf{q}^{(m)}(t), \ \forall t \in (\sigma, \tau], \tag{3.51}$$

$$\mathbf{M}(\sigma) = \mathbf{L}, \tag{3.52}$$

$$\mathbf{n}(\sigma) = \mathbf{0}, \tag{3.53}$$

where $\mathbf{H}^{(m)}(t) = [\mathbf{B}^{(m)}(t)\ \mathbf{0}]$, $\mathbf{L} = [\mathbf{0}\ \mathbf{I}]$, *and* $\mathbf{I}$ *is the identity matrix of rank* n_x.

Proof. The embedded LTV system in Problem 3.19 can be written as the equivalent system in (3.46) and (3.47). The desired result is then obtained by applying the theory in [120]. □

Corollary 3.24. *Consider Theorem 3.23. Then, the convex set* $\dot{\mathcal{X}}^b(t; P, \tilde{Z}) \equiv [\dot{\mathbf{x}}^L(t), \dot{\mathbf{x}}^U(t)]$ *for* $t \in T$ *can be calculated pointwise in time from the following interval equation,*

$$[\dot{\mathbf{x}}](t) = \Big(\mathbf{A}^{(m)}(t)\mathbf{M}(t) + \mathbf{H}^{(m)}(t)\Big)[\mathbf{w}] + \mathbf{A}^{(m)}(t)\mathbf{n}(t) + \mathbf{q}^{(m)}(t),\ \forall t \in T. \tag{3.54}$$

Proof. See Theorem 2.14. □

Remark. The functional form of the solution of the equivalent LTV system, (3.46) and (3.47), is affine in the parameters $\mathbf{w}$,

$$\mathbf{x}(\mathbf{w}, t) = \mathbf{M}(t)\mathbf{w} + \mathbf{n}(t). \tag{3.55}$$

The entries in $\mathbf{M}$ are clearly the parametric sensitivities of the dynamic system, $\frac{\partial \mathbf{x}}{\partial \mathbf{w}}$. Hence, (3.50) and (3.52) are simply the forward sensitivity equations of the embedded dynamic system in Problem 3.19. For problems where the number of parameters is much greater than the state variables, it might be more attractive to employ adjoint methods to calculate the required parametric sensitivities at the specified final time, i.e., calculating $\mathbf{M}(\tau)$ to construct the sets $\mathcal{X}^b(\tau; P, \tilde{Z})$ and $\dot{\mathcal{X}}^b(\tau; P, \tilde{Z})$. However, adjoint methods cannot be applied for constructing the bounding trajectories $\mathbf{x}^L(t)$, $\mathbf{x}^U(t)$, $\dot{\mathbf{x}}^L(t)$ and $\dot{\mathbf{x}}^U(t)$, i.e., the sets $\mathcal{X}^b(t; P, \tilde{Z})$ and $\dot{\mathcal{X}}^b(t; P, \tilde{Z})$ $\forall t \in [\sigma, \tau]$ (which are needed for constructing convex relaxations of the integral term, see e.g., (D4), since they cannot provide the sensitivity trajectories pointwise in time). Hence, the forward sensitivities must be used whenever there are integral terms in the objective or constraint functionals, and efficient methods exist for computing these sensitivities [57].

Remark. The bounds $\mathbf{x}^L(t)$ and $\mathbf{x}^U(t)$ from (3.49) are *exact* in the following sense. For any $i \in \{1, \ldots, n_x\}$, and any $t \in T$, the following relationship holds,

$$x_i(\mathbf{w}^*, t) = x_i^L(t) \leq x_i(\mathbf{w}, t) \leq x_i^U(t) = x_i(\mathbf{w}^\dagger, t), \ \forall \mathbf{w} \in W,$$

for some $\mathbf{w}^*, \mathbf{w}^\dagger \in W$.

The final prerequisite for utilizing Corollary 3.22 lies in the construction, or estimation, of the set Z in Problem 3.17. From Figure 3-13, we see that Z is simply an estimate of the set of bounds for the state variables $\mathbf{x}$ at the start of each epoch, over all possible T_μ. For each epoch I_i, where $\mathbf{z}_i$ are the initial conditions, it is desirable to have a good estimate for the bounds $[\mathbf{z}_i^L, \mathbf{z}_i^U]$, because the constructed convex relaxations become tighter (more accurate) as the bounds become tighter. Consequently, the solution of the lower bounding convex MINLP constructed becomes a better lower bound as Z becomes smaller, increasing the efficiency of the global optimization algorithm. We will discuss how the set Z can be estimated in the next Section. Note that the estimation of the set Z can have a dramatic effect on algorithms for the solution of Problem 3.17, as will be shown later. However, given a set Z, the nonconvex OA algorithm described in Section 2.1.2 will still terminate finitely when applied to Problem 3.17. Another thing to note is the solution of the primal problem. This occurs with $\mathbf{Y}^*$ fixed, which implies that T_μ^* is fixed. Clearly, the auxiliary variables $\mathbf{Z}$ can then be eliminated from the primal problem, which can be solved as an optimization problem involving only the set P.

Having shown how to construct convex relaxations for (3.36) and (3.37), we now construct relaxations for the remaining nonconvex constraints, (3.38). Consider the trilinear term, $\mathbf{s} = y_1 y_2 \mathbf{v}$ with the following bounds, $0 \leq y_1, y_2 \leq 1$, $\mathbf{v}^L \leq \mathbf{v} \leq \mathbf{v}^U$, where $\mathbf{s}, \mathbf{v} \in \mathbb{R}^{n_x}$. By applying the exact linearization methods in [67], we can obtain an exact linear representation of the trilinear term. For notational convenience, let $\langle \mathbf{s}, y_1, y_2, \mathbf{v} \rangle_{trilin}$ denote the following collection of linear constraints for all elements $i = 1, \ldots, n_x$,

$$
\text{if } v_i^L \geq 0: \quad \left.\begin{array}{r} v_i^U(y_1+y_2-2)+v_i \\ v_i^L(y_1+y_2-1) \\ 0 \end{array}\right\} \leq s_i \leq \left\{\begin{array}{l} v_i^L(y_1-1)+v_i \\ v_i^L(y_2-1)+v_i \\ v_i^U y_1 \\ v_i^U y_2 \end{array}\right. ,
$$

$$
\text{if } v_i^U \leq 0: \quad \left.\begin{array}{r} v_i^U(y_1-1)+v_i \\ v_i^U(y_2-1)+v_i \\ v_i^L y_1 \\ v_i^L y_2 \end{array}\right\} \leq s_i \leq \left\{\begin{array}{l} v_i^L(y_1+y_2-2)+v_i \\ v_i^U(y_1+y_2-1) \\ 0 \end{array}\right. ,
$$

$$
\text{if } v_i^L < 0 < v_i^U: \quad \left.\begin{array}{r} v_i^U(y_1+y_2-2)+v_i \\ v_i - v_i^U \\ v_i^L y_1 \\ v_i^L y_2 \end{array}\right\} \leq s_i \leq \left\{\begin{array}{l} v_i^L(y_1+y_2-2)+v_i \\ v_i - v_i^L \\ v_i^U y_1 \\ v_i^U y_2 \end{array}\right. .
$$

Then, the following constraints are the exact linearizations for (3.38):

$$
\mathbf{z}_{i+1} = \sum_{j=1}^{n_m}\sum_{k=1}^{n_m} \mathbf{s}_{jki}, \ \forall i = 1,\ldots,n_e-1, \tag{3.56}
$$

$$
\langle \mathbf{s}_{jki}, y_{ji}, y_{k,i+1}, \mathbf{v}_{jki}\rangle_{trilin}, \ \forall j,k = 1,\ldots,n_m, \ i = 1,\ldots,n_e-1, \tag{3.57}
$$

$$
\begin{aligned}
\mathbf{v}_{jki} = &\Big(\mathbf{D}_i(j,k)\mathbf{M}_i^{(j)}(\tau_i) + \mathbf{E}_i(j,k)\Big)\mathbf{w}_i \\
&+ \mathbf{D}_i(j,k)\mathbf{n}_i^{(j)}(\tau_i) + \mathbf{k}_i(j,k), \ \forall j,k = 1,\ldots,n_m, \ i = 1,\ldots,n_e-1,
\end{aligned} \tag{3.58}
$$

$$
\mathbf{w}_i = (\mathbf{p}, 0, \mathbf{z}_i), \ \forall i = 1,\ldots,n_e-1, \tag{3.59}
$$

where $\mathbf{V} \in V \subset \mathbb{R}^{n_x\times n_m\times n_m\times(n_e-1)}$, $\mathbf{W} \in \mathbb{R}^{(n_p+n_x+1)\times(n_e-1)}$, $\mathbf{S} \in \mathbb{R}^{n_x\times n_m\times n_m\times(n_e-1)}$; and $\mathbf{M}_i^{(j)}$ and $\mathbf{n}_i^{(j)}$ are the quantities in Theorem 3.23 for mode j and epoch i. The

bounds on the auxiliary variables, V, can be calculated from the natural interval extensions of (3.58) and (3.59) once the set Z has been estimated,

$$[\mathbf{v}_{jki}^L, \mathbf{v}_{jki}^U] = \Big(\mathbf{D}_i(j,k)\mathbf{M}_i^{(j)}(\tau_i) + \mathbf{E}_i(j,k)\Big)[\mathbf{w}_i^L, \mathbf{w}_i^U] \\ + \mathbf{D}_i(j,k)\mathbf{n}_i^{(j)}(\tau_i) + \mathbf{k}_i(j,k),\ \forall j,k = 1,\ldots,n_m,\ i = 1,\ldots,n_e-1,$$

$$[\mathbf{w}_i^L, \mathbf{w}_i^U] = [(\mathbf{p}^L, 0, \mathbf{z}_i^L), (\mathbf{p}^U, 1, \mathbf{z}_i^U)],\ \forall i = 1,\ldots,n_e-1.$$

Before we end this section, we establish conditions for which the constructed convex relaxations are continuously differentiable on an open set containing P as a corollary of Theorem 3.5, which is important since a gradient based NLP solver will be used to solve the lower bounding problems.

Corollary 3.25. *Consider Corollary 3.22. Let $P^o \supset P$, $\tilde{Y}^o \supset \tilde{Y}$, $\tilde{Z}^o \supset \tilde{Z}$, $\mathcal{X}^{bo}(\tau) \supset \mathcal{X}^b(\tau; P^o, \tilde{Z}^o)$, $\mathcal{X}^{bo} \supset \mathcal{X}^b(P^o, \tilde{Z}^o)$, $\dot{\mathcal{X}}^{bo}(\tau) \supset \dot{\mathcal{X}}^b(\tau; P^o, \tilde{Z}^o)$ and $\dot{\mathcal{X}}^{bo} \supset \dot{\mathcal{X}}^b(P^o, \tilde{Z}^o)$ be open subsets of $\mathbb{R}^{n_p}$, $\mathbb{R}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$, $\mathbb{R}^{n_x}$ and $\mathbb{R}^{n_x}$ respectively. If the following conditions are satisfied, then the function U is continuously differentiable on $P^o \times \tilde{Y}^o \times \tilde{Z}^o$.*

1. *$\frac{\partial\psi}{\partial\dot{\mathbf{x}}}$, $\frac{\partial\psi}{\partial\mathbf{x}}$, $\frac{\partial\psi}{\partial\mathbf{p}}$ and $\frac{\partial\psi}{\partial\tilde{y}}$ exist, and are continuous on $\dot{\mathcal{X}}^{bo}(\tau) \times \mathcal{X}^{bo}(\tau) \times P^o \times \tilde{Y}^o$.*
2. *$\frac{\partial u}{\partial\dot{\mathbf{x}}}$, $\frac{\partial u}{\partial\mathbf{x}}$, $\frac{\partial u}{\partial\mathbf{p}}$ and $\frac{\partial u}{\partial\tilde{y}}$ are piecewise continuous on $\dot{\mathcal{X}}^{bo} \times \mathcal{X}^{bo} \times P^o \times \tilde{Y}^o \times T$ where only a finite number of stationary simple discontinuities are allowed.*

Proof. The embedded LTV system in Problem 3.19 can be written as the equivalent system in (3.46) and (3.47). The proof is then elementary from the proof of Theorem 2.9 by treating $\tilde{y}$ and $\tilde{\mathbf{z}}$ as additional optimization decision variables. □

McCormick's composition theorem [96] will most often produce smooth convex relaxations; however, if the factorable representation is employed, the convex relaxations constructed have no guarantee of smoothness, as discussed in Section 2.1. For point objectives and constraints, this nonsmoothness can be easily eliminated by introducing additional variables and constraints (see e.g., [125, Chapter 4]. For general terms which include the isoperimetric (integral) term, the αBB method [1] for constructing convex relaxations can be used to guarantee smoothness. This method

is applicable to a broad class of twice-differentiable functions, and the constructed convex relaxations are also guaranteed to be twice-differentiable. Thus, the use of Corollary 3.22 in conjunction with the aforementioned methods guarantees that if Theorem 3.5 holds, Corollary 3.25 also holds, and both the original functions and their constructed convex relaxations are at least continuously differentiable on some open set containing $P \times \tilde{Y} \times \tilde{Z}$. We will assume that this is true for the rest of this Chapter.

3.4 Bounding Strategies for Hybrid Systems with Varying Mode Sequences

In this section, we shall present different bounding strategies for estimating the set Z in Problem 3.17. In general, more computational effort has to be expended in order to obtain tighter estimates for the set Z. However, it is interesting to note that with the development of the dynamic bounds tightening heuristic (see Section 3.5.1), that cheap (in terms of computational time) bounding strategies can also make a big difference in the convergence rate of global deterministic algorithms.

3.4.1 Extended Affine Bounding

From the previous section, we know that exact bounds for $\mathbf{x}(\tau)$ can be constructed for Problem 3.19 once the bounds for $\tilde{\mathbf{z}}$ are known. This can be further extended to problems with multiple epochs by simply stepping through each epoch sequentially. This motivates the following algorithm for estimating the set Z, where $\mathcal{A}$ is a (possibly empty) set of tuples (m, i) indicating mode m is fixed in epoch i,

$$\mathcal{A} \equiv \{\ (m, i) \mid \text{ mode } m \text{ is active in epoch } i\ \}.$$

For example, $\mathcal{A} = \{(1, 1), (3, 5)\}$ denotes mode 1 is fixed in epoch 1, mode 3 is fixed in epoch 5, and the mode is free in all other epochs. Each epoch can appear in at most one element of $\mathcal{A}$. The bounds on $\mathbf{z}_1$ are given by the natural interval extension

[99] of (3.41),

$$[\mathbf{z}_1^L, \mathbf{z}_1^U] = \mathbf{E}_0[\mathbf{p}^L, \mathbf{p}^U] + \mathbf{k}_0.$$

Algorithm 3.26 (A1($\mathcal{A}$)).

1. (***Preprocessing***) For $m = 1$ to n_m do:

   (a) For $i = 1$ to $(n_e - 1)$ do:

      i. Integrate the system (3.50), (3.51), (3.52) and (3.53) from $\sigma = \sigma_i$ to $\tau = \tau_i$, and store $\mathbf{M}_i^{(m)}(\tau_i) := \mathbf{M}(\tau)$ and $\mathbf{n}_i^{(m)}(\tau_i) := \mathbf{n}(\tau)$.

2. (***Initialization***) Set $a_1 = a_2 = 0$, $b_i = \texttt{FALSE}, c_i = 0\ \forall i = 1, \ldots, n_e$. Set bounds $\mathbf{x}^{(m,k)^L}(\sigma_{i+1}) = +\infty, \mathbf{x}^{(m,k)^U}(\sigma_{i+1}) = -\infty$ for all $m, k = 1, \ldots, n_m$, and $i = 1, \ldots, n_e - 1$.

3. (***Active Mode Inclusion***) For each $(m, i) \in \mathcal{A}$, set $b_i := \texttt{TRUE}$, $c_i := m$.

4. (***Calculate Bounds***) For $i = 1$ to $(n_e - 1)$ do:

   (a) If (b_i) set $a_1 = 1$, else set $a_1 = n_m$. If (b_{i+1}) set $a_2 = 1$, else set $a_2 = n_m$.

   (b) For $j = 1$ to a_1 do:

      i. If (b_i) set $m = c_i$, else set $m = j$.

      ii. For $l = 1$ to a_2 do:

         A. If (b_{i+1}) set $k = c_{i+1}$, else set $k = l$.

         B. Calculate and store $[\mathbf{x}^{(m,k)^L}(\sigma_{i+1}), \mathbf{x}^{(m,k)^U}(\sigma_{i+1})]$ from

$$[\mathbf{x}^{(m,k)}](\sigma_{i+1}) = \Big(\mathbf{D}_i(m,k)\mathbf{M}_i^{(m)}(\tau_i) + \mathbf{L}_i(m,k)\Big)[\mathbf{w}] + \mathbf{D}_i(m,k)\mathbf{n}_i^{(m)}(\tau_i) + \mathbf{k}_i(m,k).$$

         where $\mathbf{w}^L = (\mathbf{p}^L, 0, \mathbf{z}_i^L)$, $\mathbf{w}^U = (\mathbf{p}^U, 1, \mathbf{z}_i^U)$, and $\mathbf{L}_i(m,k) = [\mathbf{E}_i(m,k) \ \ \mathbf{0}]$.

   (c) For $j = 1$ to n_x do:

i. Calculate and store the jth element of $[\mathbf{z}_{i+1}^L, \mathbf{z}_{i+1}^U]$ from

$$z_{j,i+1}^L = \min_{m,k} \; x_j^{(m,k)^L}(\sigma_{i+1}), \quad z_{j,i+1}^U = \max_{m,k} \; x_j^{(m,k)^U}(\sigma_{i+1}).$$

Remark. The system (3.50), (3.51), (3.52) and (3.53) is independent of the parameters $\mathbf{w}$, hence the values of $\mathbf{M}(\tau)$ and $\mathbf{n}(\tau)$ are also independent of $\mathbf{w}$. Hence, if the epochs are of equal duration, i.e., $\tau_i - \sigma_i$ is constant for all i, and we have a LTI hybrid system, step (1a) only needs to be executed once for $i = 1$.

Although Theorem 3.23 guarantees exact bounds for Problem 3.19, the bounds obtained from implementing Algorithm 3.26 have no guarantee of exactness for Problem 3.17 past the first epoch. This is not surprising, as bounds for different elements of $[\mathbf{z}_i^L, \mathbf{z}_i^U]$, $i > 2$, could come from different predecessor modes m_{i-1}, i.e., (3.32) is not enforced in any of the preceding modes for Step 4) in (A1). One way to enforce (3.32) is to obtain the bounds for all possible combinations of T_μ, i.e., solving the bounding equations in Theorem 2.14 through explicit enumeration. This would provide exact bounds for Problem 3.17; however, this method clearly suffers from exponential complexity.

3.4.2 Relaxed LP Bounding

Consider the following problem.

Problem 3.27 (LPB1(α,β))**.**

$$\min_{\mathbf{p}\in P, \mathbf{Y}\in Y^b, \mathbf{Z}} \mathbf{e}_\beta^{\mathrm{T}} \mathbf{z}_{\alpha+1}$$
$$\text{s.t.} \; \sum_{m=1}^{n_m} y_{mi} = 1, \; \forall i = 1, \ldots, \alpha,$$

$$\mathbf{z}_{i+1} = \sum_{j=1}^{n_m} \sum_{k=1}^{n_m} y_{ji} y_{ki+1} \Big(\mathbf{D}_i(j,k) \mathbf{x}_{ji}(\mathbf{p}, \mathbf{Z}, \tau_i) + \mathbf{E}_i(j,k)\mathbf{p} + \mathbf{k}_i(j,k) \Big), \; \forall i = 1, \ldots, \alpha, \quad (3.60)$$

$$\mathbf{z}_1 = \mathbf{E}_0\mathbf{p} + \mathbf{k}_0,$$

where $Y^b \equiv \{0,1\}^{n_m \times \alpha} \subset Y \equiv [0,1]^{n_m \times \alpha}$, $\mathbf{Z} \in \mathbb{R}^{n_x \times (\alpha+1)}$, and the unit vector $\mathbf{e}_\beta$ is the βth column of the rank n_x identity matrix; $\mathbf{x}_{mi}(\mathbf{p}, \mathbf{Z}, t)$ are given by the solution of the following embedded LTV ODE systems for all $m \in M$, $i = 1, \ldots, \alpha$,

$$\dot{\mathbf{x}}_{mi}(\mathbf{p}, \mathbf{Z}, t) = \mathbf{A}^{(m)}(t)\mathbf{x}_{mi}(\mathbf{p}, \mathbf{Z}, t) + \mathbf{B}^{(m)}(t)\mathbf{p} + \mathbf{q}^{(m)}(t), \ \forall t \in (\sigma_i, \tau_i],$$
$$\mathbf{x}_{mi}(\mathbf{p}, \mathbf{Z}, \sigma_i) = \mathbf{z}_i.$$

Problem LPB1(α, β) determines the exact lower bound for the βth component of $\mathbf{x}(\mathbf{p}, T_\mu, \sigma_{\alpha+1}) = \mathbf{z}_{\alpha+1}$. We can construct an exact linear reformulation for LPB1(α, β) by treating the trilinear terms in (3.60) using the exact linearizations described above for (3.38). We can then formulate the following, equivalent, MILP.

Problem 3.28 (LPB2(α,β))**.**

$$\min_{\mathbf{p},\mathbf{Y},\mathbf{Z},\mathbf{V},\mathbf{W},\mathbf{S}} \mathbf{e}_\beta^{\mathrm{T}}\mathbf{z}_{\alpha+1}$$
$$\text{s.t.} \ \sum_{m=1}^{n_m} y_{mi} = 1, \ \forall i = 1, \ldots, \alpha,$$
$$\mathbf{z}_{i+1} = \sum_{j=1}^{n_m}\sum_{k=1}^{n_m} \mathbf{s}_{jki}, \ \forall i = 1, \ldots, \alpha,$$
$$\langle \mathbf{s}_{jki}, y_{ji}, y_{k,i+1}, \mathbf{v}_{jki} \rangle_{trilin}, \ \forall j, k = 1, \ldots, n_m, \ i = 1, \ldots, \alpha,$$

$$\mathbf{v}_{jki} = \Big(\mathbf{D}_i(j,k)\mathbf{M}_i^{(j)}(\tau_i) + \mathbf{E}_i(j,k)\Big)\mathbf{w}_i$$
$$+ \mathbf{D}_i(j,k)\mathbf{n}_i^{(j)}(\tau_i) + \mathbf{k}_i(j,k), \ \forall j, k = 1, \ldots, n_m, \ i = 1, \ldots, \alpha,$$

$$\mathbf{w}_i = (\mathbf{p}, 0, \mathbf{z}_i), \ \forall i = 1, \ldots, \alpha,$$

where $\mathbf{Y} \in Y^b \equiv \{0,1\}^{n_m \times \alpha} \subset Y \equiv [0,1]^{n_m \times \alpha}$, $\mathbf{Z} \in \mathbb{R}^{n_x \times (\alpha+1)}$, $\mathbf{V} \in V \subset \mathbb{R}^{n_x \times n_m \times n_m \times \alpha}$, $\mathbf{W} \in \mathbb{R}^{(n_p+n_x+1) \times \alpha}$, $\mathbf{S} \in \mathbb{R}^{n_x \times n_m \times n_m \times \alpha}$, and the unit vector $\mathbf{e}_\beta$ is the βth column of

a rank n_x identity matrix; and $\mathbf{M}_i^{(j)}(\tau_i)$ and $\mathbf{n}_i^{(j)}(\tau_i)$ are given by the solution of the system (3.50), (3.51), (3.52) and (3.53) from $\sigma = \sigma_i$ to $\tau = \tau_i$, for $j \in M$, $i = 1, \ldots, \alpha$.

The required bounds on the auxiliary variables $\mathbf{V}$ constitute the set V, and can be determined sequentially for each epoch (see Algorithm 3.29 below). The variables $\mathbf{Z}$, $\mathbf{W}$, $\mathbf{S}$ are left as free or unrestricted variables. While it is impractical to solve a family of MILPs (LPB2(α, β)) to obtain the tightest bounds for $\mathbf{Z}$, it is much cheaper to solve (LPB2(α, β)) on the relaxed set $\mathbf{Y} \in Y$, resulting in solving a family of relaxed LPs to provide valid (but not exact) bounds for $\mathbf{Z}$. This constitutes the following algorithm.

Algorithm 3.29 (A2)**.**

1. (***Preprocessing***) For $m = 1$ to n_m do:

   (a) For $i = 1$ to $(n_e - 1)$ do:

      i. Integrate the system (3.50), (3.51), (3.52) and (3.53) from $\sigma = \sigma_i$ to $\tau = \tau_i$, and store $\mathbf{M}_i^{(m)}(\tau_i) := \mathbf{M}(\tau)$ and $\mathbf{n}_i^{(m)}(\tau_i) := \mathbf{n}(\tau)$.

2. (***Initialization***) Set bounds $\hat{\mathbf{x}}^{(m,k)L}(\sigma_{i+1}) = \mathbf{x}^{(m,k)^L}(\sigma_{i+1}) = +\infty, \hat{\mathbf{x}}^{(m,k)U}(\sigma_{i+1}) = \mathbf{x}^{(m,k)^U}(\sigma_{i+1}) = -\infty$ for all $m, k = 1, \ldots, n_m$, and $i = 1, \ldots, n_e - 1$.

3. (***Calculate Bounds***) For $i = 1$ to $(n_e - 1)$ do:

   (a) For $j = 1$ to n_m do:

      i. For $k = 1$ to n_m do:

         A. Calculate and store $[\hat{\mathbf{x}}^{(j,k)L}(\sigma_{i+1}), \hat{\mathbf{x}}^{(j,k)U}(\sigma_{i+1})]$ from

$$[\hat{\mathbf{x}}^{(j,k)}](\sigma_{i+1}) = \Big(\mathbf{D}_i(j,k)\mathbf{M}_i^{(j)}(\tau_i) + \mathbf{L}_i(j,k)\Big)[\mathbf{w}] + \mathbf{D}_i(j,k)\mathbf{n}_i^{(j)}(\tau_i) + \mathbf{k}_i(j,k).$$

where $\mathbf{w}^L = (\mathbf{p}^L, 0, \mathbf{z}_i^L)$, $\mathbf{w}^U = (\mathbf{p}^U, 1, \mathbf{z}_i^U)$, and $\mathbf{L}_i(j,k) = [\mathbf{E}_i(j,k) \;\; \mathbf{0}]$.

(b) For $j = 1$ to n_m do:

i. For $k = 1$ to n_m do:

A. For $l = 1$ to n_x do:

- Solve LPB2(i, l), with $[\mathbf{v}_{jk\theta}] = [\mathbf{x}^{(j,k)}(\sigma_{\theta+1})]$, $\theta = 1, \ldots, i-1$, and $[\mathbf{v}_{jki}] = [\hat{\mathbf{x}}^{(j,k)}(\sigma_{i+1})]$ for all $(j, k) \in M^2$, on the relaxed set Y, with the following constraint, $y_{ji} = 1$, and store ${x_l^{(j,k)}}^L(\sigma_{i+1}) :=$ optimal solution value.
- Repeat the step above as a maximization problem, and store ${x_l^{(j,k)}}^U(\sigma_{i+1}) :=$ optimal solution value.

(c) For $j = 1$ to n_x do:

i. Calculate and store the jth element of $[\mathbf{z}_{i+1}^L, \mathbf{z}_{i+1}^U]$ from

$$z_{j,i+1}^L = \min_{m,k} \; {x_j^{(m,k)}}^L(\sigma_{i+1}), \quad z_{j,i+1}^U = \max_{m,k} \; {x_j^{(m,k)}}^U(\sigma_{i+1}).$$

Note that an attractive feature of the algorithm is the ability to easily incorporate information gained from physical insight into the problem as constraints in the relaxed LPs. Although the LP relaxation can be solved in polynomial time, it is still significantly slower than Algorithm 3.26, which will be the algorithm of choice for the dynamic bounds tightening heuristic introduced in Section 3.5.1. Also, it is possible to incorporate the active mode inclusion step into Algorithm 3.29 similar to Step 3 of Algorithm 3.26, where the appropriate modifications to LPB2(α, β) are made through the removal of the auxiliary variables which are not needed, and the addition of constraints to enforce the active modes.

3.4.3 Harrison's Method and its Extension

In this section, we will consider the application of Harrison's method [71] to obtain estimates for the set Z as applied to Example 3.7. The reason that we have applied it specifically to Example 3.7 is that Harrison's method requires some assumptions and conditions to be met before it can be applied. We shall also discuss some simple

extensions to the method that could make it more widely applicable. For the example considered, Harrison's method scales quite nicely with the number of epochs considered, although the bounds obtained are not very tight, as will be discussed in Section 3.4.4.

In [71] Harrison introduced a method to compute upper and lower bounds for the flow rates in linear compartmental models. One of the main assumptions made is that the values of the state variables are nonnegative. In this section, we will describe a trivial extension to Harrison's theorem to deal with state variables that can take negative values. The extension is based on the assumption that valid lower bounds can be obtained for all state variables, however it does not guarantee exact bounds (and hence its usefulness is questionable). In addition, we will also trivially generalize Harrison's results to include time varying bounds on the rate coefficients a_{ij} and r_i. First, we shall describe Harrison's method.

Consider the following linear ODE system (linear compartmental model),

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{r}, \tag{3.61}$$

where x_i is the concentration of material in compartment i, r_i is the flow rate into compartment i from outside the system, and, for $i \neq j$, $a_{ij}x_j$ is the flow rate from compartment j into compartment i. The term $a_{ii}x_i$ is the total flow rate out of compartment i, hence we have

$$a_{ii} = -a_{0i} - \sum_{k \neq i} a_{ki}, \tag{3.62}$$

where $a_{0i}x_i$ is the flow rate from compartment i to outside the system.

Harrison was interested in applications where it was extremely difficult to determine the rate coefficients a_{ij} and r_i, due to uncertainty, incomplete knowledge of the process controlling the flow rates, and/or the inability to measure the flow rates precisely. He dealt with the uncertainty by assuming that a priori bounds were known

for $\mathbf{A}$ and $\mathbf{r}$ of the following form,

$$a_{ij}^L \leq a_{ij}(t) \leq a_{ij}^U, \quad \text{for } i \neq j,\ i = 0, \ldots, n_x,\ j = 1, \ldots, n_x, \tag{3.63}$$

$$r_i^L \leq r_i(t) \leq r_i^U, \quad \text{for } i = 1, \ldots, n_x, \tag{3.64}$$

and developed the following results for computing the (exact) intervals $[x_i^L(t), x_i^U(t)]$ which contain the elements $x_i(t)$ of the solution to (3.61), subject to (3.62), (3.63) and (3.64). He noted that while traditional methods for differential inequalities [22, 135] could be used to obtain the desired bounds, these methods tended to produce weak bounds when (a) the entries of the matrix $\mathbf{A}$ are not independent, as in (3.62); or (b) when the system is not quasimonotone as defined in [135, 84]. In the latter case, some improvement is obtained by using Moore's [98] correction for the wrapping effect [70], but there is still no guarantee that the bounds are exact. It is assumed throughout that $\mathbf{x}(t) \geq \mathbf{0}$ is met by any realistic compartmental model. Note that the coefficients of $\mathbf{A}$ and $\mathbf{r}$ are actually allowed to take on any functional form over the time horizon, and are not restricted to assume constant real values over time. Thus, intuitively, the state bounds obtained from applying Harrison's method will naturally be weaker than the exact bounds obtained for a system in which the coefficients are restricted to constant real values. This is also the reason why the extension to Harrison's method proposed below does not give the exact state bounds when applied to LTI or hybrid systems.

Harrison's theorem

Consider the following system,

$$\dot{\mathbf{x}}(t) = \mathbf{A}(\mathbf{u}(t))\mathbf{x}(t) + \mathbf{r}(t), \tag{3.65}$$

where $\mathbf{A}$ is a function of the parameter vector $\mathbf{u}(t) \in \mathbb{R}^{n_u}$ satisfying

$$\mathbf{u}^L \leq \mathbf{u}(t) \leq \mathbf{u}^U, \tag{3.66}$$

and $\mathbf{r}(t)$ is a vector of inputs satisfying

$$\mathbf{r}^L \leq \mathbf{r}(t) \leq \mathbf{r}^U. \tag{3.67}$$

Theorem 3.30 (Harrison [71])**.** *Let $\boldsymbol{\lambda}(t)$ be the solution of the adjoint equations*

$$\dot{\boldsymbol{\lambda}}(t) = -\mathbf{A}(\mathbf{u}^*(t))^{\mathrm{T}}\boldsymbol{\lambda}(t), \tag{3.68}$$

where $\mathbf{u}^*(t)$ satisfies

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}[\mathbf{A}(\mathbf{u}(t)) - \mathbf{A}(\mathbf{u}^*(t))]\mathbf{x}(t) \leq 0, \tag{3.69}$$

for all $\mathbf{x}(t) \geq \mathbf{0}$ and all $\mathbf{u}(t)$ satisfying (3.66)*, and let $\mathbf{r}^*(t)$ satisfy*

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}(t) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}^*(t) \tag{3.70}$$

for all $\mathbf{r}(t)$ satisfying (3.67)*. If $\mathbf{x}^*(t)$ is a solution of* (3.65) *with $\mathbf{u}(t) = \mathbf{u}^*(t)$ and $\mathbf{r}(t) = \mathbf{r}^*(t)$, and $\mathbf{x}(t)$ is a nonnegative solution of* (3.65) *with*

$$\boldsymbol{\lambda}(0)^{\mathrm{T}}\mathbf{x}(0) \leq \boldsymbol{\lambda}(0)^{\mathrm{T}}\mathbf{x}^*(0), \tag{3.71}$$

then for all $t \geq 0$

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{x}(t) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{x}^*(t). \tag{3.72}$$

Proof. First we adjoin the state variables, $\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{x}$, and take the time derivative,

$$\begin{aligned} v(t) &= \dot{\boldsymbol{\lambda}}(t)^{\mathrm{T}}\mathbf{x}(t) + \boldsymbol{\lambda}(t)^{\mathrm{T}}\dot{\mathbf{x}}(t) \\ &= -\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{A}(\mathbf{u}^*(t))\mathbf{x}(t) + \boldsymbol{\lambda}(t)^{\mathrm{T}}(\mathbf{A}(\mathbf{u}(t))\mathbf{x}(t) + \mathbf{r}(t)) \\ &= \boldsymbol{\lambda}(t)^{\mathrm{T}}[\mathbf{A}(\mathbf{u}(t)) - \mathbf{A}(\mathbf{u}^*(t))]\mathbf{x}(t) + \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}(t), \end{aligned} \tag{3.73}$$

where $v \equiv \frac{\mathrm{d}(\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{x})}{\mathrm{d}t}$. Combining (3.69) and (3.73), we have

$$v(t) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}(t). \tag{3.74}$$

From (3.73), it is clear that the following equation holds

$$v(t)|_{\mathbf{x}(t)=\mathbf{x}^*(t)} = \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}^*(t),$$

which together with (3.70) and (3.74) implies

$$v(t) \leq v(t)|_{\mathbf{x}(t)=\mathbf{x}^*(t)}\,.$$

Applying [116, Thm. 6.12(b)], we have

$$\int_0^t \frac{\mathrm{d}(\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{x})}{\mathrm{d}t}\,\mathrm{d}t \leq \int_0^t \left.\frac{\mathrm{d}(\boldsymbol{\lambda}^{\mathrm{T}}\mathbf{x})}{\mathrm{d}t}\right|_{\mathbf{x}(t)=\mathbf{x}^*(t)}\,\mathrm{d}t.$$

From the fundamental theorem of calculus, we have

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{x}(t) - \boldsymbol{\lambda}(0)^{\mathrm{T}}\mathbf{x}(0) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{x}^*(t) - \boldsymbol{\lambda}(0)^{\mathrm{T}}\mathbf{x}^*(0)$$

which, together with (3.71), clearly implies (3.72). □

Remark. Harrison does not impose any further restrictions in stating his theorem. Equation (3.66) clearly accommodates piecewise continuous functions of $\mathbf{u}(t)$, provided that $\mathbf{u}(t)$ is defined and bounded by (3.66) at points of discontinuity (stationary simple discontinuities in time). Harrison does not discuss the existence or uniqueness of solutions, although the usual restrictions on $\mathbf{A}(t)$ and $\mathbf{r}(t)$ would apply.

Harrison's theorem shows that (3.68), (3.69) and (3.70) can be viewed as the sufficient[1] conditions for bounding the adjoint of the states, (3.72). The connection to the linear compartmental models described above is made through the following corollary.

Corollary 3.31. *Let*

$$\dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{r}(t), \tag{3.75}$$

where the diagonal entries $a_{ii}(t)$ of $\mathbf{A}(t)$ are given by (3.62)*, and where $r_i(t)$ and*

[1] and necessary as remarked by Harrison, although not formally proven or stated as such

$a_{ij}(t)$ for $i \neq j$, $i = 0, \ldots, n_x$, $j = 1, \ldots, n_x$, satisfy the bounds given in (3.63) *and* (3.64)*. Let $\boldsymbol{\lambda}(t)$ be the solution of*

$$\dot{\boldsymbol{\lambda}}(t) = -\mathbf{A}^*(t)^{\mathrm{T}}\boldsymbol{\lambda}(t), \tag{3.76}$$

where

$$a_{0i}^*(t) = \begin{cases} a_{0i}^L, & \text{if} \quad \lambda_i(t) \geq 0, \\ a_{0i}^U, & \text{if} \quad \lambda_i(t) < 0, \end{cases} \tag{3.77}$$

$$a_{ij}^*(t) = \begin{cases} a_{ij}^U, & \text{if} \quad \lambda_i(t) - \lambda_j(t) \geq 0, \\ a_{ij}^L, & \text{if} \quad \lambda_i(t) - \lambda_j(t) < 0, \end{cases} \qquad \text{for } i \neq 0, i \neq j, \tag{3.78}$$

and a_{ii}^* is computed from (3.62)*,*

$$a_{ii}^*(t) = -a_{0i}^*(t) - \sum_{k \neq i} a_{ki}^*(t).$$

If $\mathbf{x}^*(t)$ is a solution of

$$\dot{\mathbf{x}}^*(t) = \mathbf{A}^*(t)\mathbf{x}^*(t) + \mathbf{r}^*(t), \tag{3.79}$$

where

$$r_i^*(t) = \begin{cases} r_i^U, & \text{if} \quad \lambda_i(t) \geq 0, \\ r_i^L, & \text{if} \quad \lambda_i(t) < 0, \end{cases} \tag{3.80}$$

then any nonnegative solution of (3.75) *subject to* (3.62)*,* (3.63) *and* (3.64) *with*

$$\boldsymbol{\lambda}(0)^{\mathrm{T}}\mathbf{x}(0) \leq \boldsymbol{\lambda}(0)^{\mathrm{T}}\mathbf{x}^*(0) \tag{3.71}$$

satisfies, for all $t \geq 0$,

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{x}(t) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{x}^*(t). \tag{3.81}$$

Proof. It suffices to show that (3.77), (3.78) and (3.80) imply the conditions (3.69) and (3.70) in Harrison's theorem with the parameters $\mathbf{u}(t)$ being the rate coefficients $a_{ij}(t)$ for $i \neq j$, $i = 0, \ldots, n_x$, $j = 1, \ldots, n_x$. First, (3.80) clearly implies (3.70). Next,

note that

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}[\mathbf{A}(t) - \mathbf{A}^*(t)]\mathbf{x}(t) = \sum_i (a_{0i}^*(t) - a_{0i}(t))\lambda_i(t)x_i(t)$$
$$+ \sum_i \sum_{k \neq i} (a_{ki}(t) - a_{ki}^*(t))(\lambda_k(t) - \lambda_i(t))x_i(t),$$

where the choice of (3.77) and (3.78) ensures that (3.69) holds for all $\mathbf{x}(t) \geq \mathbf{0}$. $\square$

Remark. The adjoint system is clearly a hybrid system due to the state conditions in (3.77), (3.78) and (3.80), which are reversible discontinuities. Note that the direction of the strict and regular (or weak) inequalities is arbitrary as long as the conditions are suitably defined, e.g., the following equation could replace (3.80),

$$r_i^*(t) = \begin{cases} r_i^U, & \text{if} \quad \lambda_i(t) > 0, \\ r_i^L, & \text{if} \quad \lambda_i(t) \leq 0. \end{cases} \tag{3.82}$$

In either case, the treatment of reversible discontinuities discussed in Section 1.1.5 applies. Note that all but one of the adjoint variables will have zero values at the start of the backward integration phase of the algorithm (described below), and so the simulator must be able to recognize correctly which branch of the state condition is active at the start of the simulation.

Note that thus far, nothing has been mentioned about the initial conditions of the original ODE system. In fact, the theorem and corollary only requires that (3.71) be satisfied. For problems with a set of fixed initial conditions, (3.71) is automatically satisfied, and so $\mathbf{x}^*(0)$ can be set to the given fixed initial conditions. For problems where the initial conditions belong to a range of values, say an interval vector,

$$\mathbf{x}(0) \in [\mathbf{x}^L(0), \mathbf{x}^U(0)],$$

1. Initialize $\boldsymbol{\lambda}(\tau) = e_i$ where the unit vector e_i is the ith column of the identity matrix.
2. Initialize the discontinuity functions by taking one integration step backwards from τ.
3. Integrate (3.76), (3.77), (3.78) and (3.80) backward in time from τ to 0, and store the timings of the events, as well as the state condition that triggers the event. This can be done, e.g., by storing the triple (t, i, j) for each event, where t stores the event time triggered by a zero crossing of $\lambda_i(t)$ when $j = 0$, and a zero crossing of $\lambda_i(t) - \lambda_j(t)$ otherwise.
4. Integrate (3.79), (3.77), (3.78) and (3.80) forward in time from 0 to τ with initial conditions $\mathbf{x}^*(0)$ given either by fixed initial conditions or (3.83).
5. The upper bound for $x_i(\tau)$ is then given by $x_i^*(\tau)$.
6. To obtain the lower bound, re-initialize with $\boldsymbol{\lambda}(\tau) = -e_i$ and repeat Steps 2, 3 and 4. The lower bound for $x_i(\tau)$ is then given by $x_i^*(\tau)$.

Figure 3-14: Algorithm for calculating the exact bounds for $x_i(\tau)$

these conditions can be satisfied if $\mathbf{x}^*(0)$ is computed by the following equations,

$$x_i^*(0) = \begin{cases} x_i^L(0), & \text{if } \lambda_i(0) < 0 \\ x_i^U(0), & \text{if } \lambda_i(0) \geq 0 \end{cases}, \ \forall i = 1, \ldots, n_x \tag{3.83}$$

for some given value of $\lambda(0)$, which is obtained from performing interval analysis. Also, note that the initial time is given by 0, without loss of generality. Clearly, the same results would hold if 0 was replaced by some fixed initial time σ.

The bounds for element $x_i(\tau)$ at a specified time τ, where $0 < \tau < \infty$, can be computed by the algorithm shown in Figure 3-14. We now show the correctness of the algorithm. Since we have chosen $\boldsymbol{\lambda}(\tau) = e_i$, (3.81) reduces to $x_i(\tau) \leq x_i^*(\tau)$ where $x_i^*(\tau)$ is clearly a valid upper bound for $x_i(\tau)$. The exactness of the upper bound follows from the fact that $x_i^*(\tau)$ itself is a solution to (3.79), and hence is a valid solution for (3.75) for some $\mathbf{A}(t)$ and $\mathbf{r}(t)$ satisfying (3.62), (3.63) and (3.64). The same argument holds true for the lower bound.

Remark. This algorithm is useful when the bounds are needed at specified times. Similar to adjoint methods for calculating sensitivities, which do not deliver the sensitivity trajectories but rather the end time sensitivities, this method does not produce the bounding trajectories. To calculate the bounds at intermediate times to some specified final time t_f, the algorithm will have to be repeated for each intermediate time point.

We will now describe some extensions to Harrison's theorem.

An Extension to Harrison's theorem - Time Varying Bounds

Since Harrison was interested in models with fixed rate coefficients, a_{ij} and r_i, that possibly varied between some known a priori bounds, the bounds in (3.63), (3.64), (3.66), (3.67), (3.77), (3.78) and (3.80) are all time invariant.

Theorem 3.32. *Theorem 3.30 and Corollary 3.31 presented above are valid when* (3.63)*,* (3.64)*,* (3.66)*,* (3.67)*,* (3.77)*,* (3.78) *and* (3.80) *are replaced by the following equations respectively,*

$$\begin{aligned} a_{ij}^L(t) \leq a_{ij}(t) \leq & a_{ij}^U(t), \quad \textit{for } i \neq j,\ i = 0, \ldots, n_x,\ j = 1, \ldots, n_x, \\ r_i^L(t) \leq r_i(t) \leq & \ r_i^U(t), \quad \textit{for } i = 1, \ldots, n_x, \end{aligned} \tag{3.84}$$

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

$$\mathbf{r}^L(t) \leq \mathbf{r}(t) \leq \mathbf{r}^U(t).$$

$$a_{0i}^*(t) = \begin{cases} a_{0i}^L(t), & \textit{if} \quad \lambda_i(t) \geq 0, \\ a_{0i}^U(t), & \textit{if} \quad \lambda_i(t) < 0, \end{cases}$$

$$a_{ij}^*(t) = \begin{cases} a_{ij}^U(t), & \textit{if} \quad \lambda_i(t) - \lambda_j(t) \geq 0, \\ a_{ij}^L(t), & \textit{if} \quad \lambda_i(t) - \lambda_j(t) < 0, \end{cases} \qquad \textit{for } i \neq 0, i \neq j,$$

$$r_i^*(t) = \begin{cases} r_i^U(t), & \textit{if} \quad \lambda_i(t) \geq 0, \\ r_i^L(t), & \textit{if} \quad \lambda_i(t) < 0, \end{cases}$$

Proof. The proof is trivial through substitution. □

This trivial extension is useful when we want to acquire tighter bounds for the case where we know the time varying bounds for $\mathbf{A}(t)$ and $\mathbf{r}(t)$. As discussed earlier, this is also why Harrison's bounds are weak, because they are valid for all functions between the bounds, not just scalar values between the bounds.

An Extension to Harrison's Theorem - Allowing Negative Trajectories

We are interested in removing the restriction $\mathbf{x} \geq \mathbf{0}$ for Harrison's theorem. The extension described here provides valid bounds for the system, however, there cannot be any guarantee of exactness, as will be shown below. Let us also restrict our interest to a finite time domain, $t \in [0, t_f]$. Assume that a valid lower bound is known a priori for the states, $\mathbf{x}(t)$ for $t \in [0, t_f]$,

$$\mathbf{v} \leq \mathbf{x}(t).$$

where $v_i > -\infty$ for all i. Let element z_i of the vector $\mathbf{z}$ be such that $z_i = \min\{0, v_i\}$ for all $i = 1, \ldots, n_x$. It follows that

$$\mathbf{y}(t) \equiv \mathbf{x}(t) - \mathbf{z} \geq \mathbf{0}.$$

First, we show how application of Harrison's theorem to the variable $\mathbf{y}(t)$ will not work to produce the exact bounds for $\mathbf{x}(t)$, even though $\mathbf{y}(t)$ is now a nonnegative state variable. We have

$$\dot{\mathbf{y}}(t) = \dot{\mathbf{x}}(t) = \mathbf{A}(t)\mathbf{x}(t) + \mathbf{r}(t) = \mathbf{A}(t)\mathbf{y}(t) + \mathbf{A}(t)\mathbf{z} + \mathbf{r}(t) \tag{3.85}$$

If we let $\boldsymbol{\lambda}(t)$ be the solution of the adjoint equations as in (3.68), the sufficient conditions (3.69) and (3.70) become, respectively,

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}[\mathbf{A}(\mathbf{u}(t)) - \mathbf{A}(\mathbf{u}^*(t))]\mathbf{y}(t) \leq 0, \tag{3.86}$$

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{A}(\mathbf{u}(t))\mathbf{z} + \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}(t) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{A}(\mathbf{u}^*(t))\mathbf{z} + \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}^*(t). \tag{3.87}$$

If we now consider the choice of $\mathbf{u}^*(t)$ in (3.77), (3.78) and (3.80), it is clear that

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}(t) \leq \boldsymbol{\lambda}(t)^{\mathrm{T}}\mathbf{r}^*(t). \tag{3.88}$$

However, together with (3.88), we must also have

$$\boldsymbol{\lambda}(t)^{\mathrm{T}}[\mathbf{A}(\mathbf{u}(t)) - \mathbf{A}(\mathbf{u}^*(t))]\mathbf{z} \leq 0 \tag{3.89}$$

to satisfy (3.87), which is impossible since $\mathbf{z} \leq \mathbf{0}$ and $\mathbf{u}^*(t)$ was chosen to satisfy (3.86) for $\mathbf{y}(t) \geq \mathbf{0}$. In other words, these sufficient conditions have very limited applicability because there is no choice of $\mathbf{u}^*(t)$ that can satisfy (3.86), (3.88) and (3.89) simultaneously.

One way to obtain the bounds, however, is to work with (3.85). We introduce

$$\mathbf{q}(t) = \mathbf{A}(t)\mathbf{z} + \mathbf{r}(t),$$

where we have the following bounds on $\mathbf{q}(t)$,

$$q_i^L(t) = r_i^L(t) + \left(a_{0i}^L(t) + \sum_{k\neq i} a_{ki}^L(t)\right) z_i + \sum_{k\neq i} a_{ik}^U(t) z_k, \tag{3.90}$$

$$q_i^U(t) = r_i^U(t) + \left(a_{0i}^U(t) + \sum_{k\neq i} a_{ki}^U(t)\right) z_i + \sum_{k\neq i} a_{ik}^L(t) z_k. \tag{3.91}$$

Consider now the system

$$\dot{\mathbf{y}} = \mathbf{A}(t)\mathbf{y}(t) + \mathbf{q}(t)$$

where the bounds on $\mathbf{A}(t)$ are given by (3.84), and the bounds on $\mathbf{q}(t)$ are given by (3.90) and (3.91). We can then apply Harrison's theorem to this relaxed system to compute the (exact) bounds for this system ($\mathbf{y}(\tau)$) at any specified time, $\tau \in [0, t_f]$,

$$\mathbf{y}^L(\tau) \leq \mathbf{y}(\tau) \leq \mathbf{y}^U(\tau).$$

We can then extract valid bounds on $\mathbf{x}(\tau)$ by adding $\mathbf{z}$,

$$\mathbf{y}^L(\tau) + \mathbf{z} \leq \mathbf{x}(\tau) \leq \mathbf{y}^U(\tau) + \mathbf{z}.$$

The fact that the bounds on $\mathbf{q}(t)$ are valid for any $\mathbf{A}(\mathbf{u}^*(t))$ and $\mathbf{r}^*(t)$ shows that the bounds on $\mathbf{x}(\tau)$ are valid. However, it is not possible to show exactness of the bounds $(\mathbf{x}(\tau))$ for the original system.

Applying Harrison's Method

It is easy to verify that for the general case where (3.62) does not hold, and (3.63) is replaced by the following equation,

$$a_{ij}^L(t) \leq a_{ij}(t) \leq a_{ij}^U(t), \quad \text{for} \;\; i = 1, \ldots, n_x, \;\; j = 1, \ldots, n_x, \tag{3.92}$$

that (3.77) and (3.78) reduce to the following state conditions,

$$a_{ij}^*(t) = \begin{cases} a_{ij}^L(t) & \text{if} \quad \lambda_j(t) \geq 0, \\ a_{ij}^U(t) & \text{if} \quad \lambda_j(t) < 0, \end{cases} \qquad \text{for } i = 1, \ldots, n_x, \;\; j = 1, \ldots, n_x. \tag{3.93}$$

As noted by Harrison [71], an application of Corollary 3.31 to calculate the bounds for the system at time τ is to start a backward integration of the adjoint system from τ to σ with an appropriate final condition to obtain the values of $\mathbf{A}^*(t)$ and $\mathbf{r}^*(t)$ on $[\sigma, \tau]$, and then to integrate the original system forward in time from σ to τ using $\mathbf{A}^*(t)$ and $\mathbf{r}^*(t)$. Clearly, the backward sweep of the algorithm requires rigorous state event detection. When there are multiple transitions becoming true at the same time, we assume that the integrator is able to: (a) detect the timing of such transitions; and (b) record all multiple transitions that have been taken.

Before we present the algorithm based on Harrison's method, we need to transcribe the hybrid system into the form of (3.61), or vice versa. For the sequel, we will only consider transitions with state continuity for simplicity.

Remark. It is possible to handle transition conditions of the following form:

$$\mathbf{x}(\mathbf{p}, T_\mu, \sigma_{i+1}) = \mathbf{D}_i \mathbf{x}(\mathbf{p}, T_\mu, \tau_i) + \mathbf{E}_i \mathbf{p} + \mathbf{k}_i, \quad \forall i = 1, \ldots, n_e - 1, \tag{3.94}$$

where $\mathbf{D}_i$, $\mathbf{E}_i$ and $\mathbf{k}_i$ are known, provided that nonnegativity of the state variables is preserved. These transition functions will not affect the solution of the adjoint system in the backward sweep, and Corollary 3.31 will still apply. If the transition functions cannot guarantee nonnegativity of the states, the extension of Harrison's theorem to nonnegative variables presented below can be applied. However, if the transition functions are expressed as a function of the predecessor or successor modes, as is the case for Problem 3.17, then the proof of Corollary 3.31 need not hold and Harrison's method cannot be applied directly.

Theorem 3.33. *Consider the hybrid system in Definition 3.1 with state continuity as the transition functions, where it is known that the continuous states of the hybrid system are nonnegative. Consider next the following system*

$$\dot{\tilde{\mathbf{x}}}(t) = \tilde{\mathbf{A}}(t)\tilde{\mathbf{x}}(t) + \tilde{\mathbf{r}}(t), \tag{3.95}$$

where

$$\tilde{a}_{ij}^L(t) \leq \tilde{a}_{ij}(t) \leq \tilde{a}_{ij}^U(t), \quad \textit{for} \;\; i = 1, \ldots, n_x, \;\; j = 1, \ldots, n_x, \tag{3.96}$$

$$\tilde{r}_i^L(t) \leq \tilde{r}_i(t) \leq \tilde{r}_i^U(t), \quad \textit{for } i = 1, \ldots, n_x, \tag{3.97}$$

$$\left.\begin{aligned} \tilde{a}_{ij}^L(t) &= \min_{m \in M} a_{ij}^{(m)}(t) \\ \tilde{a}_{ij}^U(t) &= \max_{m \in M} a_{ij}^{(m)}(t) \end{aligned}\right\} \quad \textit{for} \;\; i = 1, \ldots, n_x, \;\; j = 1, \ldots, n_x, \tag{3.98}$$

$$\left.\begin{aligned} \tilde{r}_i^L(t) &= \min_{m \in M} {r_i^{(m)}}^L(t) \\ \tilde{r}_i^U(t) &= \max_{m \in M} {r_i^{(m)}}^U(t) \end{aligned}\right\} \quad \textit{for} \;\; i = 1, \ldots, n_x, \tag{3.99}$$

$$[\mathbf{r}^{(m)}](t) = \mathbf{B}^{(m)}(t)[\mathbf{p}] + \mathbf{q}^{(m)}(t), \tag{3.100}$$

$$[\tilde{\mathbf{x}}](\sigma_1) = \mathbf{E}_0[\mathbf{p}] + \mathbf{k}_0. \tag{3.101}$$

Let the bounds obtained from the application of Corollary 3.31 (with (3.63) *replaced by* (3.92)*, and* (3.77) *and* (3.78) *replaced by* (3.93)*) on* (3.95) *at the fixed time* $\tau \in [t_0, t_f]$ *be given by* $[\tilde{\mathbf{x}}^L(\tau), \tilde{\mathbf{x}}^U(\tau)]$*. Then,*

$$\tilde{\mathbf{x}}^L(\tau) \leq \mathbf{x}(\mathbf{p}, T_\mu, \tau) \leq \tilde{\mathbf{x}}^U(\tau), \ \forall (\mathbf{p}, T_\mu) \in P \times M^{n_e}.$$

Proof. Let $(T_\mu^*, \mathbf{p}^*)$ be an arbitrary choice of mode trajectory and parameters for the hybrid system. This particular execution of the hybrid system can be represented by the following execution of the dynamic system in (3.95),

$$\left.\begin{aligned} \tilde{\mathbf{A}}(t) &= \mathbf{A}^{(m_i^*)}(t) \\ \tilde{\mathbf{r}}(t) &= \mathbf{B}^{(m_i^*)}(t)\mathbf{p}^* + \mathbf{q}^{(m_i^*)}(t) \end{aligned}\right\} \text{ for } t \in I_i, \ i = 1, \ldots, n_e,$$

with the initial condition $\tilde{\mathbf{x}}(\sigma_1) = \mathbf{E}_0\mathbf{p}^* + \mathbf{k}_0$. Note that this satisfies the constraints in (3.96) – (3.101). Since the choice of $(T_\mu^*, \mathbf{p}^*)$ was arbitrary, any arbitrary execution of the hybrid system can be represented by an equivalent execution of (3.95) subject to (3.96) – (3.101). Applying Corollary 3.31 with the appropriate substitutions, we obtain the desired result. □

Note that Theorem 3.33 can be extended easily for the case when the diagonal entries of $\mathbf{A}^{(m)}$ are given by the hybrid extension to (3.62),

$$a_{ii}^{(m)}(t) = -a_{0i}^{(m)}(t) - \sum_{k \neq i} a_{ki}^{(m)}(t), \ \forall m \in M.$$

We are now in position to present the following algorithm:

Algorithm 3.34 (A3)**.**

1. For $j = 1$ to n_x do:

   (a) For $i = 2$ to n_e do:

i. Initialize $\tilde{\boldsymbol{\lambda}}(\tau_{i-1}) = \mathbf{e}_j$ where the unit vector $\mathbf{e}_j$ is the jth column of the identity matrix of rank n_x.

ii. Take a small step backwards from τ_{i-1} for the adjoint system (3.76) and (3.93) of the relaxed ODE system in Theorem 3.33, and store the initial values of (3.93).

iii. Integrate the adjoint system backwards in time from τ_{i-1} to σ_1 using rigorous state event detection, and store the following triple $(t_{event}, n_d, \mathbf{np})$ at each event, where t_{event} records the event time, n_d records the number of adjoint variables which experience a zero crossing at the event, and $\mathbf{np}$ records the indices of the adjoint variables which have zero crossings.

iv. Reconstruct the matrices $\tilde{\mathbf{A}}^*$ and $\tilde{\mathbf{r}}^*$ for the relaxed ODE system in Theorem 3.33 backwards in time from τ_{i-1} to σ_1.

v. Integrate the following system forward in time from σ_1 to τ_{i-1},

$$\dot{\tilde{\mathbf{x}}}^*(t) = \tilde{\mathbf{A}}^*(t)\tilde{\mathbf{x}}^*(t) + \tilde{\mathbf{r}}^*(t),$$

with the initial condition $\tilde{\mathbf{x}}^*(\sigma_1)$ given by (3.83).

vi. Store $z_{ji}^U := \tilde{x}_j^*(\tau_{i-1})$.

2. Repeat Step 1 with the following changes: $\tilde{\boldsymbol{\lambda}}(\tau_{n_e}) = -\mathbf{e}_j$ in Step 1.a.i, $z_{ji}^L := \tilde{x}_j^*(\tau_{i-1})$ in Step 1.a.vi.

Remark. This algorithm can be easily extended for the case where the diagonal entries of $\tilde{\mathbf{A}}(t)$ are given by (3.62). In that case, $\mathbf{np}$ in Step 1.a.iii stores the indices of the discontinuity functions instead of the adjoint variables.

Consider now Example 3.7. Note that the choice of catalyst corresponds to the choice of the sequence of modes in a linear hybrid system with 3 modes (each mode corresponds to the choice of a different catalyst) and n_e epochs (each epoch corresponds to a section of the reactor), with state continuity at the transitions.

The linear time invariant hybrid system can be written as the following,

$$\dot{\mathbf{x}}(t) = \begin{bmatrix} -(k_1^{(m)} + k_2^{(m)}) & 0 & 0 & 0 & 0 \\ k_2^{(m)} & 0 & 0 & 0 & 0 \\ k_1^{(m)} & 0 & -(k_3^{(m)} + k_4^{(m)}) & 0 & 0 \\ 0 & 0 & k_4^{(m)} & 0 & 0 \\ 0 & 0 & k_3^{(m)} & 0 & 0 \end{bmatrix} \mathbf{x}(t),$$

in which case it is more appropriate to use (3.62) (and hence (3.77) and (3.78) as opposed to (3.93)) as conservation of molar species is automatically enforced while calculating the bounds. In this case, we have the following nonzero bounds on $\tilde{\mathbf{A}}(t)$,

$$1.317 \leq \tilde{a}_{21}(t) \leq 2325,$$

$$2.098 \leq \tilde{a}_{31}(t) \leq 182.3,$$

$$0.033 \leq \tilde{a}_{43}(t) \leq 0.143,$$

$$0.021 \leq \tilde{a}_{53}(t) \leq 1.826.$$

The results of applying Harrison's method to this example will be presented in Section 3.4.4.

3.4.4 A Comparison of the Different Strategies

In this section, we apply the algorithms (A1)($\emptyset$) (Algorithm 3.26), (A2) (Algorithm 3.29) and (A3) (Algorithm 3.34) to the problem of obtaining estimates for the set Z for Example 3.7. We also apply the method of explicit enumeration (EE) to obtain the exact bounds for the set Z for comparison. When physical information from the problem can be used, e.g., conservation of mass, we can add the following additional constraints to Problem 3.28,

$$\sum_{j=1}^{n_x} (\mathbf{s}_{mi})_j = 1000 y_{mi}, \ \forall m \in M, i = 1, \ldots, \alpha, \tag{3.102}$$

noting that because the transition functions are state continuity, the number of auxiliary variables $\mathbf{V}$ and $\mathbf{S}$ can be reduced appropriately. We will label this (A2pi).

Tables 3.1 and 3.2 show the bounds obtained for $\mathbf{z}_8$ and $\mathbf{z}_{15}$ when $n_e = 15$. As can be seen, (A2) produces tighter bounds than (A1($\emptyset$)). When physical insight is employed, it can be seen that the bounds obtained from (A2) with (3.102) produces tighter bounds than using (A2) alone. The reason why (A2) itself does not produce bounds which obey this conservation law is that the exact linearizations of the trilinear (bilinear in this case where transition functions do not depend on the predecessor and successor modes, as in (3.94)) terms in (3.57) are only exact on the set Y^b, and not on the set Y. Hence, we have to enforce the law with (3.102). Note that there is no way to incorporate additional constraints within (A1($\emptyset$)). For further illustration, the upper bounds computed for species W_1 at the beginning of each section when $n_e = 10$ are shown in Table 3.3.

For this example, there are no events detected for (A3) during the backward sweep. It is interesting to note that when the relaxed ODE is transcribed as in Theorem 3.33, Harrison's method automatically chooses the "best" rate constants when minimizing or maximizing a particular species. For example, in order to maximize the formation of the product P, (A3) would pick the highest values of k_1 ($\tilde{a}_{31}$) and k_3 ($\tilde{a}_{53}$) with the lowest values of k_2 ($\tilde{a}_{21}$) and k_4 ($\tilde{a}_{43}$). Intuitively, this makes sense when considering the reaction scheme in Figure 3-1. Unfortunately, the way that the problem is transcribed leads to the algorithm being able to choose different reaction rate constants belonging to different types of catalysts. This results in the bounds being weak as shown.

Table 3.4 shows the bounds obtained for W_1 when the algorithms are trivially extended to calculate the bounds at $l = 1$ for increasing numbers of epochs. It can be seen that the bounds obtained from (A1) and (A2) deteriorate significantly from the exact bounds as n_e increases. When physical insight (3.102) is employed in conjunction with (A2), much tighter bounds are obtained. Also, the bounds described by (A3) are pretty weak compared to that produced by (A2) with (3.102). The reason that they seem time invariant is because the system considered is very stiff, so that

Table 3.1: Bounds for $\mathbf{z}_8$ where $n_e = 15$.

Species	(EE)		(A1)		(A2)	
	$\mathbf{z}_8^L$	$\mathbf{z}_8^U$	$\mathbf{z}_8^L$	$\mathbf{z}_8^U$	$\mathbf{z}_8^L$	$\mathbf{z}_8^U$
A	0.00	203.18	0.00	203.18	0.00	203.18
W_1	307.30	927.18	78.52	3628.49	230.54	1734.62
I	29.08	493.23	29.08	735.13	29.08	493.23
W_2	1.48	12.68	0.76	29.59	1.19	16.16
P	2.98	139.37	0.49	373.51	1.26	190.19

Species	(A2pi)		(A3)	
	$\mathbf{z}_8^L$	$\mathbf{z}_8^U$	$\mathbf{z}_8^L$	$\mathbf{z}_8^U$
A	0.00	203.18	0.00	203.18
W_1	307.30	959.02	7.16	999.10
I	29.08	493.23	0.36	968.42
W_2	1.19	16.16	0.01	63.07
P	1.28	180.97	0.01	561.46

almost all the reactions are completed from the "optimal" choices of the reaction rates by (A3) in a very short length at the beginning of the reactor.

All calculations were performed on an AMD 1.2 GHz, 1 GB RAM machine using CPLEX 7.5 [78] as the LP solver with default settings. All LPs were started cold, and we note that the computational times for (A2) would improve if the LPs were warm started where possible. The computation times for the algorithms are shown in Table 3.5, from which the exponential explosion of (EE) is clear. From these results, it appears that (A2) with physical insight is the best algorithm to use to produce the tightest estimates of the set Z in Example 3.7.

3.5 Branch-and-Cut Algorithm

The branch-and-cut (BC) algorithm proposed in this section has its roots in the Outer Approximation (OA) algorithm, which is a decomposition framework for the solution of convex MINLPs [50, 59]. The extension of OA to nonconvex MINLPs was developed in [82, 83], and hinges on the ability to construct convex relaxations of the objective function and constraints to form a lower bounding convex MINLP. For the nonconvex MINLP in Problem 3.17, we have shown, in the previous sections, how convex relaxations of the objective function and the constraints can be

Table 3.2: Bounds for $\mathbf{z}_{15}$ where $n_e = 15$.

Species	(EE)		(A1)		(A2)	
	$\mathbf{z}_{15}^L$	$\mathbf{z}_{15}^U$	$\mathbf{z}_{15}^L$	$\mathbf{z}_{15}^U$	$\mathbf{z}_{15}^L$	$\mathbf{z}_{15}^U$
A	0.00	41.28	0.00	41.28	0.00	41.28
W_1	369.73	927.18	78.52	4365.72	230.54	2030.39
I	11.60	567.77	11.60	815.88	11.60	567.77
W_2	2.43	27.87	1.08	79.06	1.54	44.77
P	7.34	293.77	0.69	1005.21	1.51	544.39

Species	(A2pi)		(A3)	
	$\mathbf{z}_{15}^L$	$\mathbf{z}_{15}^U$	$\mathbf{z}_{15}^L$	$\mathbf{z}_{15}^U$
A	0.00	41.28	0.00	41.28
W_1	321.03	981.59	7.16	999.10
I	11.60	567.77	0.14	944.32
W_2	1.54	44.77	0.01	122.20
P	1.53	451.06	0.02	801.44

Table 3.3: Upper bound for W_1 ($n_e = 10$).

Section	(EE)	(A1)	(A2)	(A2pi)	(A3)
1	0.00	0.00	0.00	0.00	0.00
2	927.18	927.18	927.18	927.18	999.10
3	927.18	1586.13	927.18	927.18	999.10
4	927.18	2054.45	1161.34	932.50	999.10
5	927.18	2387.29	1245.91	945.55	999.10
6	927.18	2623.83	1356.05	954.30	999.10
7	927.18	2791.95	1401.19	961.80	999.10
8	927.18	2911.43	1461.41	967.78	999.10
9	927.18	2996.34	1482.13	972.70	999.10
10	927.18	3056.69	1516.14	976.73	999.10

Table 3.4: Upper bound for W_1 at $l = 1$.

n_e	(EE)	(A1)	(A2)	(A2pi)	(A3)
5	927.18	1811.88	1094.46	967.02	999.10
10	927.18	3099.58	1523.92	980.04	999.10
15	927.18	4404.00	2052.69	983.43	999.10
20	927.18	5712.63	2605.73	984.89	999.10

Table 3.5: CPU times (s).

n_e	(EE)	(A1)	(A2)	(A2pi)	(A3)
5	0.04	0.04	1.6	2.3	0.83
10	0.4	0.04	8.3	12.8	1.63
15	135	0.04	27.1	45.7	2.46
20	44227	0.04	50.7	86.7	3.25

constructed (using Corollary 3.22 and exact linearization of trilinear terms) to form a lower bounding convex MINLP.

In the context of the algorithm for the global solution of nonconvex MINLPs described in [83], we introduce the following abstraction of a subproblem of Problem 3.17 as the *Primal Problem*:

Problem 3.35 (NLP($\mathbf{Y}^*$))**.**

$$\min_{\mathbf{p}\in P, \mathbf{Z}\in Z} \quad F(\mathbf{p}, \mathbf{Y}^*, \mathbf{Z}) \tag{3.103}$$

$$\text{s.t.} \quad \mathbf{G}(\mathbf{p}, \mathbf{Y}^*, \mathbf{Z}) \leq \mathbf{0}, \tag{3.104}$$

and the corresponding abstraction of a subproblem of the lower bounding convex MINLP as the *Primal Bounding Problem*:

Problem 3.36 (NLPB($\mathbf{Y}^*$))**.**

$$\min_{\mathbf{p}\in P, \mathbf{Z}\in Z} \quad U(\mathbf{p}, \mathbf{Y}^*, \mathbf{Z}; P, Z) \tag{3.105}$$

$$\text{s.t.} \quad \mathbf{H}(\mathbf{p}, \mathbf{Y}^*, \mathbf{Z}; P, Z) \leq \mathbf{0}, \tag{3.106}$$

where $P \subset \mathbb{R}^{n_p}$, $\mathbf{Y}^* \in Y^b \equiv \{0,1\}^{n_m \times n_e} \subset Y \equiv [0,1]^{n_m \times n_e}$, $\mathbf{Z} \in Z \subset \mathbb{R}^{n_x \times n_e}$, U is the convex relaxation constructed for F, and $\mathbf{H}$ is the convex relaxation constructed for $\mathbf{G}$, according to the theory presented in Section 3.3.1.

The primal and primal bounding problems are formed by fixing $\mathbf{Y}^*$ to a particular binary realization. Since (3.32) is always satisfied in the solution of the relaxed Master problem (the relaxation of the equivalent MILP Master problem for the lower bounding convex MINLP), fixing $\mathbf{Y}^*$ corresponds to fixing T_μ^* for some sequence of modes. For problems with state continuity as the transition functions, this implies that $\mathbf{Z}^*$ is also fixed. As a result, the primal and primal bounding problems are simply

nonconvex and convex NLPs for $\mathbf{p} \in P$ respectively. For this class of problems, this is advantageous as it means that the sizes of the primal and primal bounding problems are not affected by the reformulation from Problem 3.4 into Problem 3.17.

The reformulation also introduces a substantial number of additional variables and linear constraints via the exact linearizations (3.56)–(3.57). As these constraints are linear, they are added directly to the relaxed Master problem at the first iteration. For problems with a small number of original constraints (3.31), this could possibly make the cost of solving the relaxed Master problem significant compared to the cost of solving the primal problem. In that case, the application of nonconvex OA would not be attractive.

To mitigate this problem, we propose a BC algorithm based on the BC algorithm for MILPs [14] and the concepts of the primal and primal bounding problems from nonconvex OA that reduces the number of relaxed Master problems solved. As the nonconvex OA algorithm can be viewed as a particular set of heuristics in the generalized BC framework proposed in [81], the following algorithm can be viewed as another set of heuristics within the framework, where the refathoming heuristic of the nonconvex OA algorithm (resolving of the relaxed Master problems) is replaced with the branching heuristic.

Algorithm 3.37 (A4)**.**

1. (***Initialization***) Set $I = J = \mathcal{A}^0 = OAC = \emptyset$, $D = \{0\}$, $LBDRMP^0 = -\infty$, $UBD = UBDPB = +\infty$, $k = 1$, $l = 1$.

2. (***Initial Guess***) Given $\mathbf{Y}^1$ that satisfies (3.32), set $\mathbf{Y}^* = \mathbf{Y}^1$.

3. (***Primal Bounding***) Solve primal bounding problem NLPB($\mathbf{Y}^*$). Then,

   (a) if NLPB($\mathbf{Y}^*$) is feasible, let z_{NLPB} be the optimal solution value. Derive the necessary cuts to be added to the relaxed Master problem as in nonconvex OA, and add them to OAC. If $z_{NLPB} < UBD$, then set $LBDPB^l = z_{NLPB}$, $\mathbf{Y}^l_{PR} = \mathbf{Y}^*$, add node l to J and set $l = l + 1$. If

$z_{NLPB} < UBDPB$ then set $UBDPB = z_{NLPB}$, and delete from I all nodes i where $LBDRMP^i > UBDPB$ and move them into D.

(b) if NLPB($\mathbf{Y}^*$) is infeasible, derive the necessary cuts to be added to the relaxed Master problem as in nonconvex OA, and add them to OAC.

4. (***Construct Relaxed Master Problem***) Construct the relaxed Master problem (RMP) as in nonconvex OA. Let $RMPLP^1$ denote the LP relaxation of RMP.

5. (***Outer Loop***) While D is not empty do:

   (a) Set $I = D$, $D = \emptyset$, $UBDPB = UBD$.

   (b) (***OA Cuts***) For each node $i \in I$, add the cuts accumulated in OAC to $RMPLP^i$. Set $OAC = \emptyset$.

   (c) (***Inner Relaxed Master Loop***) While I is not empty do:

      i. (***Node Selection***) Select and delete a node i from I. Set δ to false.

      ii. (***Dynamic Bounds Tightening***) Apply Algorithm A1($\mathcal{A}^i$) to obtain the set Z. Update the bounds on Z for $RMPLP^i$.

      iii. (***Cut Generation***) Should valid cutting planes be generated? If yes, add the generated cuts to the LP relaxation, $RMPLP^i$.

      iv. (***Lower Bounding***) Solve $RMPLP^i$ to an optimal extreme point. Then,

         A. if $RMPLP^i$ is infeasible, then set $z_{RMPLP} = +\infty$.

         B. if $RMPLP^i$ has an optimal solution, then let z_{RMPLP} be the optimal solution value, and $\mathbf{Y}^*$ be the values of the optimal solution corresponding to the binary variables.

      v. (***Fathoming***)

         A. (***Infeasibility***) If $z_{RMPLP} = +\infty$ then goto vii).

         B. (***Value Dominance of Primal***) If $z_{RMPLP} \geq UBD$ then goto vii).

C. (***Integrality***) If $\mathbf{Y}^* \in Y^b$, then solve the primal bounding problem in Step 3) and goto vii).

D. (***Value Dominance of Primal Bounding***) If $z_{RMPLP} \geq UBDPB$, set δ to true.

vi. (***Branching***) Select from the epochs in $\mathcal{A}^i$ an epoch $\alpha \in \{1, \ldots, n_e\}$ with no active modes to branch on. Create n_m nodes with $\mathcal{A}^k = \mathcal{A}^i \cup (1, \alpha), \mathcal{A}^{k+1} = \mathcal{A}^i \cup (2, \alpha), \ldots, \mathcal{A}^{k+n_m-1} = \mathcal{A}^i \cup (n_m, \alpha)$. Set $LBDRMP^k, \ldots, LBDRMP^{k+n_m-1} = z_{RMPLP}$, and $RMPLP^k, \ldots, RMPLP^{k+n_m-1} = RMPLP^i$. Add nodes $k, \ldots, k + n_m - 1$ to D if δ is true, otherwise add the nodes to I. Set $k = k + n_m$.

vii. Continue.

(d) (***Inner Primal Loop***) While J is not empty do:

i. (***Node Selection***) Select and delete node j from J, where $j \in \arg\min_{l \in J} LBDPB^l$.

ii. (***Primal***) Solve the primal problem NLP($\mathbf{Y}^j_{PR}$) globally, e.g., using Algorithm 2.3. If NLP($\mathbf{Y}^j_{PR}$) is feasible, let z_{NLP} be the optimal solution value, and $\mathbf{p}^j$ be the values of the optimal solution. If $z_{NLP} < UBD$, then

A. set $UBD = z_{NLPB}$, $\mathbf{Y}^*_{PR} = \mathbf{Y}^j_{PR}$, $\mathbf{p}^* = \mathbf{p}^j$.

B. delete from J all nodes j such that $LBDPB^j > UBD$.

C. delete from D all nodes i such that $LBDRMP^i > UBD$.

6. (***Solution***) If $UBD = +\infty$ the problem is infeasible, else the optimal solution is given by $\mathbf{Y}^*_{PR}$, $\mathbf{p}^*$, and the solution value is given by UBD.

Remark. It is straightforward to incorporate absolute and relative tolerances for controlling the accuracy to which the solution is found (adjusting the gap between the lower bounds and the upper bound) by suitably modifying the steps 3.a, 5.c.v, and 5.d.ii.

The proposed algorithm solves for the global solution by alternating finitely between the primal problem, the primal bounding problem, and LP relaxations of the relaxed Master Problem. A flowsheet of the algorithm is shown in Figure 3-15. The inner relaxed Master loop explores the BB tree for the MINLP. Note that the branching step automatically satisfies (3.32), thus exploiting the special structure of the problem, instead of branching on the binary set Y^b. If the solution of the LP relaxation is infeasible or greater than the incumbent solution, the node is fathomed, as the true solution can never be attained at that node or its children. If the solution of the LP relaxation satisfies integrality, the corresponding primal bounding problem is solved, and the upper bound for the inner loop updated if required. Since the primal bounding problem provides a valid and tighter lower bound to the primal problem for each binary realization, $\mathbf{Y}^k$, than that provided by the LP relaxation of the relaxed Master problem, the corresponding primal problem is added to the list of primal problems to solve, J, only when the solution of the primal bounding problem is greater than the incumbent upper bound, UBD. This is important, since the nonconvex primal problem, which requires deterministic global optimization methods to solve, is usually the most expensive subproblem to solve.

On the other hand, if the solution of the LP relaxation is greater than the upper bound for the inner loop ($UBDPB$), the node is branched upon and added to the deferred list of nodes D where they will be added to the list of nodes for the inner loop I upon completion of the inner primal loop. The inner primal loop solves for the primal problems in the list J, and updates the incumbent solution when applicable. If a new incumbent solution is found, the nodes in the deferred list D whose lower bounds are greater than the incumbent solution are fathomed, because the true solution can never be attained in those nodes or their children. And the cycle repeats until the solution is found. Since A1($\mathcal{A}$) provides valid bounds for the LP relaxation of the relaxed Master problem given the mode exclusion set $\mathcal{A}$, the dynamic bounds tightening step is valid. Similarly, the addition of the OA cuts occurs after they have been accumulated from the solution of the primal bounding problems. For proof that this is valid, the reader is directed to [82, 83]. Finally, to show that the algorithm

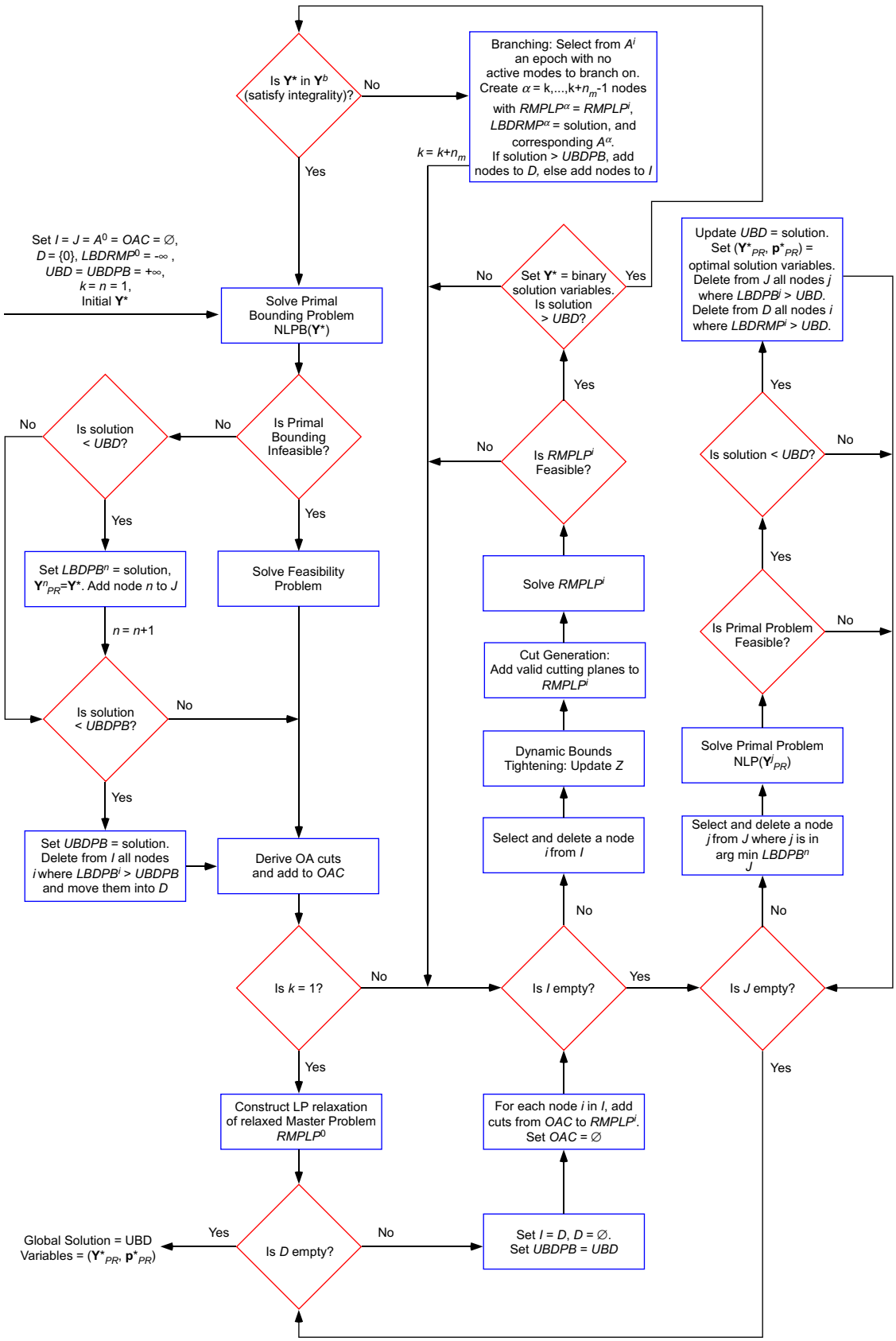


Figure 3-15: Flowsheet of Algorithm 3.37 (branch-and-cut).

finds the true solution and terminates finitely, consider the following

Theorem 3.38. *Assume that the solutions to the LP relaxations of the relaxed Master, the primal bounding and primal problems can be obtained finitely. Then, Algorithm 3.37 terminates with a finite number of iterations, and provides the solution to Problem 3.17.*

Proof. Consider the inner relaxed Master loop. Since we have a finite number of modes and epochs, the BB tree has a finite number of nodes, say n_{max}. Since any solution of the LP relaxation that satisfies integrality is no longer branched upon, at most n_{max} nodes can be generated. By assumption, the solution of the LP relaxation and primal bounding problems are obtained finitely, and thus, the inner relaxed Master loop must terminate finitely. Consider now the inner primal loop. Since there are at most n_{max} primal bounding problems that can be solved from above, there can also only be at most n_{max} primal problems that can be solved. By assumption, the solution to the primal problem terminates finitely, and thus, the inner primal loop must terminate finitely. Finally, consider the outer loop. If a node is deleted from I in the node selection step within the inner relaxed Master loop, it can no longer be added to the deferred node list D. It follows that there are at most n_{max} nodes that can be added to D. Hence, the outer loop (and the algorithm) terminates finitely.

To show that the algorithm provides the solution to Problem 3.17, it suffices to show that the primal problem corresponding to the optimal $\mathbf{Y}^*$ is solved within the inner primal loop. A node is not added to the primal node list J only under any of the following circumstances: (a) the LP relaxation of a parent node is greater than the incumbent upper bound UBD; (b) the LP relaxation of a parent node is infeasible; and (c) the primal bounding solution corresponding to the particular binary realization is greater than the incumbent upper bound UBD. Since the solutions of the LP relaxation and primal bounding problems provide valid lower bounds for the solution of the corresponding primal problem, any node that satisfies any one of the conditions described above cannot contain a primal problem whose solution is better than the incumbent solution. Hence, the node containing the solution to Problem 3.17

must always be added to the primal node list J, and hence, the algorithm provides the solution to Problem 3.17. $\square$

3.5.1 Dynamic Bounds Tightening

There are points within the BC algorithm where difference choices, or algorithmic heuristics, could be made. These do not alter the theoretical convergence of the algorithm, but they may accelerate the convergence for certain classes of problems, hence the term "heuristic". For node selection within the inner relaxed Master loop, a good heuristic would be to choose the node $i \in I$ with the lowest $LBDRMP^i$ (best bound). For branching within the same loop, two common heuristics would be forward (choosing epochs 1 to n_e) and reverse (choosing epochs n_e to 1) chronological order. As will be illustrated later, the choice of branching heuristic can have a significant impact on the solution time of the problem, as will dynamic bounds tightening. The importance of dynamic bounds tightening cannot be understated, especially for problems which have point objective and/or constraints, as these are directly impacted by the bounds on Z; it has been demonstrated, e.g., for the solution of nonconvex NLPs in [117], that the tightening of bounds for convex relaxations can accelerate the solution of problems using deterministic BB frameworks. Similarly, the ability to tighten bounds for each node of the inner relaxed Master loop can lead to a dramatic reduction in solution time. In order for dynamic bounds tightening to be effective, the algorithm for updating the set Z has to be cheap compared to the solution of the other subproblems. In this respect, Algorithm 3.26 is an excellent choice, as its preprocessing step need only be performed once at the root node, and subsequently, for all other nodes, the cost of updating Z can be performed cheaply as a series of function evaluations.

Finally, in the cut generation step, valid cutting planes can be generated for addition to the LP relaxations of the relaxed Master problem. If the problem contains nonconvex point objectives and constraints, the convex relaxations are updated with the information from dynamic bounds tightening. If the problem contains isoperimetric objectives and constraints, then their corresponding convex relaxations can be updated by updating the lower and upper bounding trajectories. The latter requires

integration of the bounding systems which will be expensive, so the choice of whether to generate cutting planes in this case will depend on the problem.

3.6 Examples and Discussion

All calculations in this section were performed on an Intel Pentium 4 3.4 Ghz machine with 1GB RAM running SuSE 9.2 using CPLEX 9.1 [78] as the LP solver. The default settings were used in CPLEX, unless otherwise stated.

Example 3.39. The example considered is Example 3.7.

We can formulate this problem in the form of Problem 3.4 by considering the space-time kinetics of the PFR, and noting that the choice of catalyst corresponds to the choice of the sequence of modes in an LTI hybrid system with 3 modes (each mode corresponds to the choice of a different catalyst) and n_e epochs (each epoch corresponds to a section of the reactor into which catalyst is loaded), with state continuity at the transition. After reformulation into the form of Problem 3.17, and employing exact linearizations for the nonconvex bilinear terms in (3.42), the resulting master problem can be solved as a MILP directly, since the objective function is linear in the point objectives $x_{\mathrm{W}_1}(1), x_{\mathrm{W}_2}(1)$ and $x_{\mathrm{P}}(1)$.

From the point of view of Algorithm 3.37 (A4), this means that the solution of the primal bounding and primal problems for fixed $\mathbf{Y}^*$ becomes the same as the solution of the LP relaxation at that node. The problem has been solved with the following algorithms:

1. (EE) Explicit enumeration of all possible T_μ. This is implemented with a pre-processing stage similar to that in (A4), where the relevant sensitivities are calculated and stored. This eliminates the need for integration to be carried out for each leaf node (fixed T_μ) which would be expensive.

2. (A4DF) Algorithm (A4) with dynamic bounds tightening, forward chronological branching heuristic, best bound node selection.

3. (A4DR) Algorithm (A4) with dynamic bounds tightening, reverse chronological branching heuristic, best bound node selection.

4. (A4NF) Algorithm (A4) with no bounds tightening, forward chronological branching heuristic, best bound node selection, initial Z calculated by (A1($\emptyset$)).

5. (A4NR) Algorithm (A4) with no bounds tightening, reverse chronological branching heuristic, best bound node selection, initial Z calculated by (A1($\emptyset$)).

6. (C1) CPLEX MILP solver with initial Z calculated by (A2) with (3.102).

7. (C2) CPLEX MILP solver with initial Z calculated by (A1($\emptyset$)).

The sensitivity coefficients were calculated using a relative and absolute tolerance for the integrator (DAEPACK [128]) of 1E-8. For the algorithms employing (A4), an initial guess of $T_\mu^* = 1, \ldots, 1$ was used. No initial guess was supplied to the CPLEX solver as it contains advanced heuristics for generating feasible initial guesses in its MILP engine. Because this is a stiff system (which is exacerbated by the possibility of multiple mode changes within a particular T_μ), the following adjustments have been made to improve the conditioning of the problem while solving the LP subproblems / MILPs: sensitivity coefficients with absolute values less than 1E-5 are rounded down to 0; any upper bounds which are less than 1E-3 are set to 1E-3, and any lower bounds which are less than 1E-3 are set to 0. This is particularly important for (C1) and (C2), as the presolve mode (turned on by default) in the MILP engine for CPLEX can potentially cut off the solution when the problem is ill-conditioned. For both algorithms (A4) and CPLEX, the relative and absolute MILP optimality tolerances (gap tolerances) were set to 1E-3.

Figure 3-16 shows a log plot of computational time versus the number of epochs, while Table 3.6 lists the solution times for selected epochs (for column (C1), the time listed is the total solution time, inclusive of the time taken to calculate the bounds with (A2), which is listed in brackets). Table 3.7 lists the optimal mode sequence obtained. The best algorithm to use for this example is (A4DF), which is orders of magnitude better than (EE), when $n_e \geq 17$. The effect of using bounds tightening in

conjunction with the forward branching heuristic is dramatic, especially compared to the other methods, which are 1-2 orders of magnitude worse than (EE). The reason that (EE) does so well here is that each leaf node of the BB tree requires simply a table lookup; in comparison, for the other algorithms, the cost of solving an interior node in the BB tree is that of solving an LP, which is significantly more costly.

The choice of branching heuristic determines how effective dynamic bounds tightening can be. From Figure 3-16, the computational times for (A4NR) and (A4DR) are very close. From Table 3.6, we can see in fact that (A4NR) always performs better than (A4DR). When the branching heuristic is performed with reverse chronological order, the bounds obtained from dynamic bounds tightening provide only incremental improvements. This arises because the problem is very stiff and the greatest changes to the bounds occur in the first epoch. Hence, the same number of nodes is visited for both algorithms and the extra cost incurred by (A4DR) comes in the form of the bounds tightening step when (A1) is called for each node. This also explains why (A4DF) is so effective for this problem. When dynamic bounds tightening is employed with forward chronological branching, there is a big improvement in the bounds for the subproblems after the first branching, and large portions of the BB tree are fathomed.

The effect of calculating tighter bounds for the set Z is illustrated by the computation times of the algorithms (C1) and (C2). For $n_e < 10$, (C2) performs better than (C1) due to the cost of computing bounds with (A2). However, for $n_e \geq 10$, it can be seen that (C1) performs much better than (C2). In addition, from Table 3.6, the cost of computing bounds with (A2) becomes small compared to the solution of the MILP as the number of epochs increases. Comparing the solution times for (C2) with (A4NR) and (A4NF), it can be seen that the MILP engine for CPLEX is much more efficient than the performance of (A4) without bounds tightening. This is not surprising, because CPLEX is a commercial implementation of the BC algorithm for MILPs with better heuristics and the incorporation of various cutting plane methods. Comparing the plots of (C1) with (EE) in Figure 3-16, it can be seen that asymptotically, one would expect that (C1) would outperform (EE) as the number of

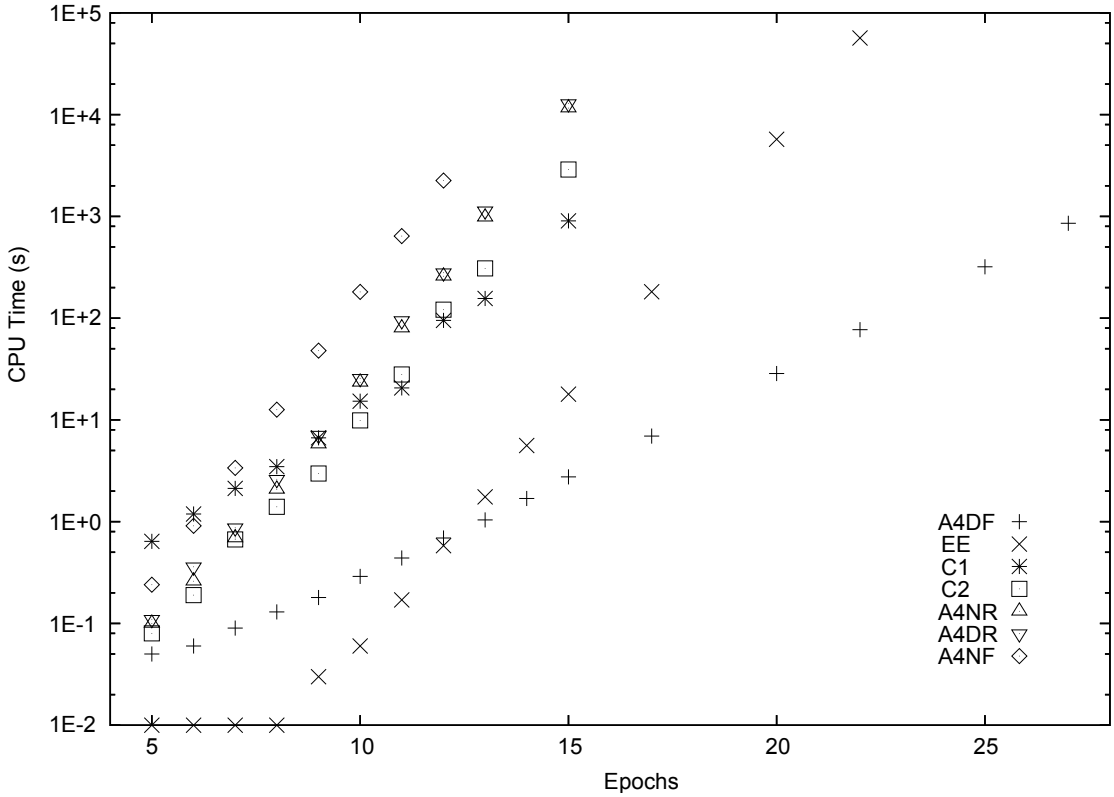


Figure 3-16: Computation times for Example 3.39

epochs increases. However, that would likely take many more epochs and much more computational time than is reasonable to study here.

Example 3.40. Consider the isothermal, well-mixed reactor shown in Figure 3-17. It is desired to design a batch recipe for the production of P given the following production rules. At the beginning of each batch, the reactor contains a large excess volume of solvent (1 m^3). The reactants are fed through their respective valves, and the pumps handling these pure liquid feeds can only be programmed with molar flowrates (kmol d^{-1}) between the following bounds,

$$1 \leq F_{\mathrm{A}}, F_{\mathrm{B}} \leq 5.$$

The reactions exhibit elementary kinetics with the following rate constants (d^{-1}), $k_{1f} = 2$, $k_{1r} = 4$ and $k_2 = 1$. For safety considerations, both reactants cannot be fed into the tank at the same time, so the dynamic behavior of the reactor can be

Table 3.6: Solution times (s) for Example 3.39.

n_e	F	(A4DF)	(EE)	(C1) [(A2)]	(C2)	(A4NR)	(A4DR)	(A4NF)
5	296.6	0.05	0.01	0.64 [0.60]	0.08	0.10	0.11	0.24
8	295.0	0.13	0.01	3.47 [2.35]	1.40	2.08	2.61	12.6
10	296.6	0.29	0.06	15.3 [4.3]	9.90	23.4	25.6	181
12	300.5	0.69	0.58	95.2 [7.3]	121	258	280	2249
15	304.5	2.76	17.9	902 [14]	2888	11600	12800	-
17	306.4	6.93	182	-	-	-	-	-
20	308.6	28.6	5740	-	-	-	-	-
22	309.5	77.0	56600	-	-	-	-	-

Table 3.7: Solution times (s) for Example 3.39.

n_e	Optimal mode sequence
5	1,1,3,3,3
8	1,1,1,3,3,3,3,3
10	1,1,1,1,3,3,3,3,3,3
12	1,1,1,1,2,3,3,3,3,3,3,3
15	1,1,1,1,1,2,3,3,3,3,3,3,3,3,3
17	1,1,1,1,1,1,2,3,3,3,3,3,3,3,3,3,3
20	1,1,1,1,1,1,1,2,3,3,3,3,3,3,3,3,3,3,3,3
22	1,1,1,1,1,1,1,1,2,3,3,3,3,3,3,3,3,3,3,3,3,3

represented by the following 3 modes of action: mode 1 where valve A is open, pump A is on, valve B is closed and pump B is off; mode 2 where valve B is open, pump B is on, valve A is closed, and pump A is off; and mode 3 where both valves are closed and both pumps are off. This is represented by the following hybrid system where N_i represents the number of moles of species i in the reactor:

$$\text{M1}: \begin{cases} \dot{N}_{\text{A}} = 4N_{\text{B}} - 2N_{\text{A}} + F_{\text{A}} \\ \dot{N}_{\text{B}} = 2N_{\text{A}} - 5N_{\text{B}} \\ \dot{N}_{\text{P}} = N_{\text{B}} \end{cases} \quad \text{M2}: \begin{cases} \dot{N}_{\text{A}} = 4N_{\text{B}} - 2N_{\text{A}} \\ \dot{N}_{\text{B}} = 2N_{\text{A}} - 5N_{\text{B}} + F_{\text{B}} \\ \dot{N}_{\text{P}} = N_{\text{B}} \end{cases},$$

$$\text{M3}: \begin{cases} \dot{N}_{\text{A}} = 4N_{\text{B}} - 2N_{\text{A}} \\ \dot{N}_{\text{B}} = 2N_{\text{A}} - 5N_{\text{B}} \\ \dot{N}_{\text{P}} = N_{\text{B}} \end{cases}.$$

The reaction stage has a fixed duration of 1 d, after which the reactions are quenched and the contents of the reactor sent for further processing. Let the fixed time horizon be partitioned into n_e contiguous epochs. The decision variables in this optimization problem are thus the sequence of modes, T_μ, as well as the real valued feed flow rates, $F_{\mathrm{A}}(t)$ and $F_{\mathrm{B}}(t)$. The control profiles for F_{A} and F_{B} are to be piecewise constant with n_e stages. Let y_{mi}, $m \in \{1, 2, 3\}$, $i \in \{1, \ldots, n_e\}$ be the binary decision variables representing T_μ. Due to inventory restrictions, the amount of B used during the operation must not exceed 2 kmol,

$$\sum_{i=1}^{n_e} y_{2i} F_{\mathrm{B}} \leq 2n_e. \tag{3.107}$$

Also, the minimum amount of product produced at the end of this reaction stage must be at least 0.6 kmol,

$$N_{\mathrm{P}}(1) \geq 0.6. \tag{3.108}$$

Due to the need to post process the raw materials A and B, the following cost constraint has to be satisfied,

$$\log(N_{\mathrm{A}}(1) + N_{\mathrm{B}}(1) + 1) \leq 1.2. \tag{3.109}$$

Finally, the desired objective function is to maximize the selectivity of the product P with respect to the raw materials,

$$\max_{T_\mu, F_{\mathrm{A}}, F_{\mathrm{B}}} S = \frac{N_{\mathrm{P}}(1)}{N_{\mathrm{A}}(1) + N_{\mathrm{B}}(1) + 1}. \tag{3.110}$$

For this problem, the lower bounding MINLP constructed is a MILP as all the convex relaxations of the nonconvex objective and constraints are linear. This implies that the solution of the primal bounding problem is the same as the corresponding LP relaxation of the relaxed Master problem that contains the primal bounding problem. Compared to the previous problem, which had large changes in the homogenous part of the ODE system between modes, the changes between modes for this example

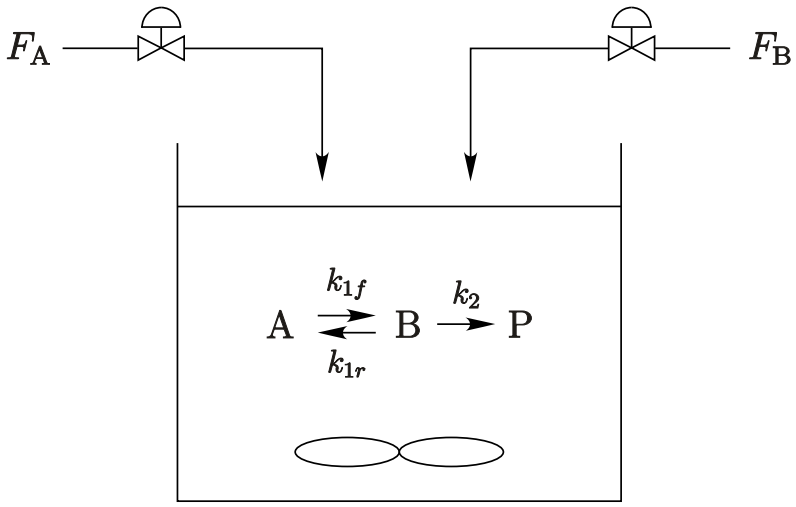


Figure 3-17: Well-mixed tank with reaction kinetics for Example 3.40

are reflected in the forcing terms. Since the problem is non-stiff, the resulting LP relaxations are well conditioned. As before, the sensitivity coefficients were calculated using a relative and absolute tolerance for the integrator of 1E-8.

In addition, because a nonconvex NLP has to be solved to global optimality, the primal subproblem is the most expensive to solve. Algorithm 2.3 (with an absolute and relative tolerance of 1E-3) has been used to solve the primal problems, with SNOPT 6.1 [66] (on default settings) used as the solver for the lower and upper bounding problems. As the problem involves only point objectives and constraints, the parametric sensitivities are calculated once in a preprocessing step for each primal problem, similar to that in (A1). This eliminates the need to call the integrator for each function and derivative call in the upper and lower subproblems, thus reducing the cost of the primal problem. As the number of epochs increases, the total computation time for the problem increases exponentially. This arises due to the following reasons: (a) the number of nodes in the BB tree increases exponentially; and (b) the number of control parameters to solve for in the primal problem increases proportionally with the number of epochs. The problem has been solved with the following algorithms, in addition to (A4DF), (A4DR), (A4NF) and (A4NR) as in Example 3.39 (with relative and absolute tolerances of 1E-3, and an initial guess of $T_\mu = 1, \ldots, 1$).

1. (EES) Explicit enumeration of all possible primal problems in T_μ, starting from $1, \ldots, 1,\ 1, \ldots, 1, 2,$ to $3, \ldots, 3$. This is implemented with an incumbent UBD so that the spatial BB algorithm for each primal problem is started with the incumbent upper bound from the solution of all previous primal problems. This greatly accelerates the performance of explicit enumeration.

2. (EEB) Explicit enumeration with an initial guess of the optimal solution T_μ^*. This is implemented as (EES) with the incumbent UBD set as the optimal solution from the initial guess T_μ^*. This represents the best possible performance for explicit enumeration.

3. (EER) Explicit enumeration with a random sequence of all possible T_μ. This is implemented with a small sample size of 100 randomly generated sequences, to have an idea of the computational time taken compared to (EES) and (EEB). Due to the exponential increase in time, this was only implemented for $n_e \leq 8$.

Table 3.8 shows the optimal solutions obtained, as well as the optimal control profiles for F_A and F_B. The column F_A in Table 3.8 shows the values of F_A for the active modes in the optimal mode sequence which belong to Mode 1, and the same applies for the column F_B. For example, for $n_e = 8$, the optimal mode sequence is 2, 2, 2, 1, 2, 3, 3, 3, and with the profiles given in the Table, this translates to the following recipe: Turn valve F_A off, turn valve F_B on with $F_B = 5$ for 3 epochs. Then, turn F_B off, turn F_A on with $F_A = 1$ for 1 epoch. Then, turn F_A off, turn F_B on with $F_B = 1$ for 1 epoch. Finally, turn F_B off for the remaining 3 epochs.

Table 3.9 shows the computation times for solving the problem with the various algorithms. The column (EER) lists the mean computational time, with the standard deviation in square brackets. Figure 3-18 shows the log plot for selected algorithms. The best algorithm to use for this example is (A4DR), which is significantly better than (EEB). Comparing the solution times for the pairs {(A4DR),(A4NR)} and {(A4DF),(A4NF)} in Table 3.9, it can be seen that dynamic bounds tightening does reduce the solution time appreciably for this example given a particular branching heuristic. The improvement is not as dramatic as the previous example because the

Table 3.8: Optimal solutions for Example 3.40.

n_e	S	Optimal mode sequence	F_{A} profile	F_{B} profile
5	0.238	2,2,1,3,3	{1}	{5,5}
6	0.236	2,2,2,1,3,3	{1}	{5,5,2}
7	0.240	2,2,2,1,3,3,3	{1}	{5,5,4}
8	0.240	2,2,2,1,2,3,3,3	{1}	{5,5,5,1}
9	0.241	2,2,2,2,1,3,3,3,3	{1}	{5,5,5,3}
10	0.243	2,2,2,2,1,3,3,3,3,3	{1}	{5,5,5,5}
11	0.242	2,2,2,2,2,1,3,3,3,3,3	{1}	{5,5,5,5,2}
12	0.243	2,2,2,2,2,1,3,3,3,3,3,3	{1}	{5,5,5,5,4}

problem is not as stiff as the previous one. Thus, the bounds tightening step does not fathom as large a portion of the BB tree compared to the previous example.

The solution times for (EER) are better than (EES) and approach (EEB). This suggests that a random sampling of possible sequences of T_μ would perform better than the ascending ordered sequence that (EES) employs. Table 3.9 also shows the exponential coefficients and R^2 values for the various algorithms when the solution times are regressed to an exponential function. All of the algorithms show a very good fit. As can be seen, the exponential coefficient of (A4DR) is 1.22 which is better than that of (EEB) which is 1.30. For this example, this strongly suggests that as the number of epochs increases, (A4DR) is going to systematically perform better than (EEB).

Table 3.9: Solution times (s) and regression results for Example 3.40.

n_e	(A4DR)	(A4NR)	(A4DF)	(A4NF)
5	1.41	1.30	1.32	1.43
6	5.10	4.60	4.67	5.58
7	17.5	15.6	16.4	19.3
8	59.0	59.0	56.8	74.8
9	193	201	195	268
10	665	677	671	905
11	2180	2300	2240	3220
12	7390	7700	8160	11300
Exp. Coef.	1.2182	1.2431	1.2423	1.2788
R^2 value	0.9999	0.9999	1.0000	0.9998

n_e	(EEB)	(EES)	(EER)[Std. Dev.]
5	2.93	3.70	3.35 [0.02]
6	12.1	14.8	13.4 [0.05]
7	43.8	55.2	47.3 [0.12]
8	167	207	170 [4.98]
9	619	801	-
10	2200	2830	-
11	8050	10200	-
12	27200	34900	-
Exp. Coef.	1.3038	1.3084	-
R^2 value	0.9997	0.9997	-

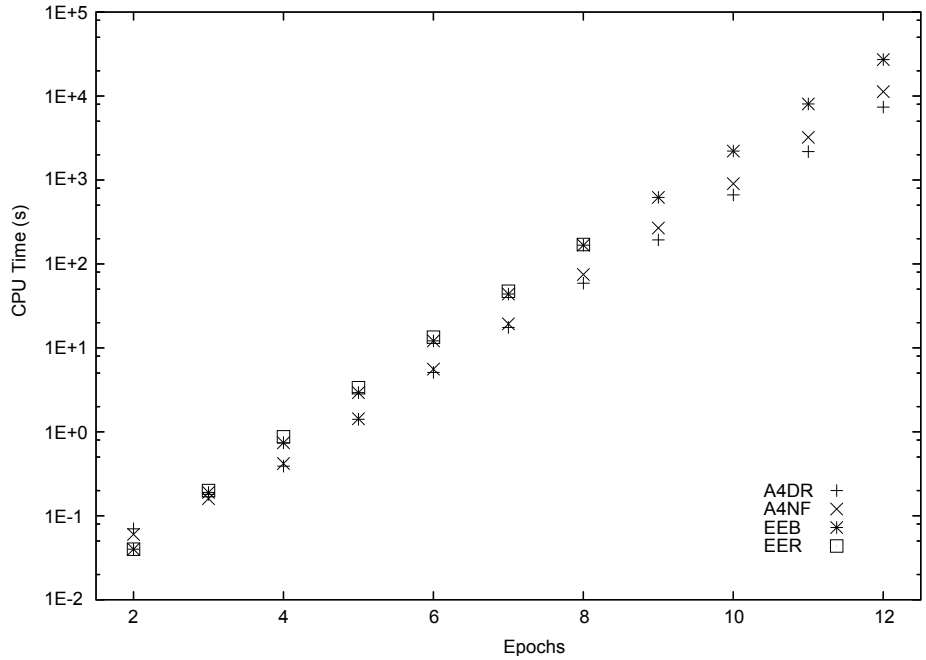


Figure 3-18: Computation times for Example 3.40

Chapter 4

Determining the Optimal Transition Times

In this chapter, we shall examine the class of optimization problems with hybrid systems embedded where the timings of some or all of the transitions are to be determined by the optimization procedure, given a fixed mode sequence. There has been recent research in the hybrid systems community on this class of problems. In [137], the timings of the transitions are parameterized, and the gradients to the local NLP solver are obtained by solving the Hamilton-Jacobi-Bellman (HJB) equations using a dynamic programming approach. This problem (of determining the optimal switching times) constitutes the Stage 1 subproblem of a two stage optimization algorithm described in [138]. One of the methods presented in [138] involves obtaining the gradients of the participating functionals through formulating the co-state equations, and this is further expanded upon by a different group of authors in [51], in which they derive the gradient of the cost functional for an especially simple form (special structure on the costate equations; the problem considered has no controls and has only the switching times as variables). However, this body of research has only focused on obtaining local solutions to the optimization problem (with no guarantees to the global optimality of the transition times), in this respect proving very similar in vein to previous approaches for multi-stage dynamic optimization [100, 134].

We shall present a deterministic global optimization framework for solving such

problems. This class of problems is difficult because it is inherently nonconvex, even if the embedded dynamic system is a LTI parameter dependent ODE, as the following simple single-stage example shows.

Example 4.1. Consider the following problem

$$\min_{p,\tau} \; x(p,\tau)$$

where $x(p,\tau)$ is given by the solution to the following linear system,

$$\begin{aligned} \dot{x} &= -2x + p, \\ x(p,0) &= 1, \\ p \in P = [-4,4], \;\; \tau &\in T = [0,2], \end{aligned}$$

and the time horizon is given by $t \in [0,\tau]$.

Figure 4-1 shows that the objective function is not convex on the set $P \times T$. Since we have a LTV ODE system, in general, we would expect the problem to be nonconvex on T even if we have $\mathbf{p}$ fixed. In order to use a BB algorithm such as Algorithm 2.3 to obtain a global solution of Example 4.1, we need a method to construct rigorous lower bounds for the objective function $F(\mathbf{p},\tau)$ on partitions of the optimization variable set $P \times T$.

Currently, no suitable theory exists for constructing convex relaxations of arbitrary Bolza type objective functionals with embedded multi-stage systems when the transition or switching times are allowed to vary. The approach that we are taking in this chapter is to transform the problem with variable transition times into one with fixed transition times, and then to develop a convexity theory for the time transformed system. The control parametrization enhancing transform (CPET) [85] is a natural transform to use for this purpose. Unfortunately, this transformation comes with an associated difficulty: the right hand sides of the differential equations become multiplied by the enhancing control. Thus, the resulting dynamic system no longer has the special structure exploited by methods specific to linear systems. While this

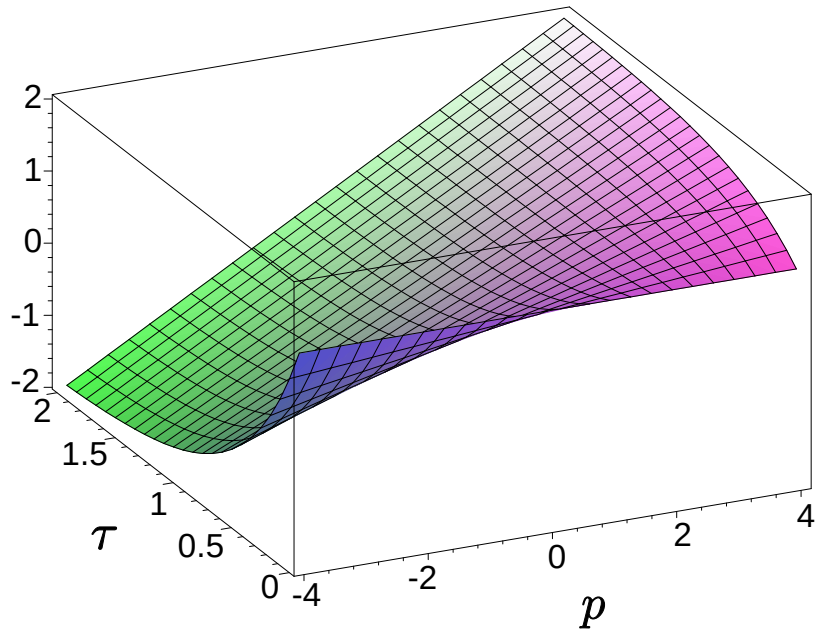


Figure 4-1: Nonconvex objective function for Example 4.1.

is not significant for local optimization, it poses a considerable obstacle for global optimization because global optimization of nonlinear dynamic systems is much harder than that for linear systems. We will thus develop a relaxation theory for the global optimization of general, nonlinear hybrid systems with a fixed sequence of modes and fixed transition times, i.e., nonlinear multi-stage systems with fixed switching times.

This chapter is organized as follows. Section 4.1 presents the general formulation of the problem, and also includes a discussion on nonsmoothness of the problem when the objective function or constraints are evaluated at fixed points in time, which has profound implications on the type of discontinuities permitted in the dynamics of each mode. The time transformation method is introduced in Section 4.2, along with sufficient conditions for the objective function or constraints to be smooth within the control parameterization framework. Section 4.3 presents bounding strategies for time transformed hybrid systems, and includes methods for constructing hybrid bounding systems based on exploiting the properties of the time transformation, and the convex relaxation theory that is developed is presented in Section 4.4. Finally, Section 4.5 contains some examples illustrating the theory that is developed in this chapter.

4.1 Problem Formulation

First, we shall define the linear hybrid system of interest, based on the modeling framework presented in Section 1.1. We shall also introduce additional notation for the durations of the epochs, represented by the vector $\boldsymbol{\delta} = (\delta_1, \ldots, \delta_{n_e})$, where $\delta_i = \tau_i - \sigma_i$ for all $i = 1, \ldots, n_e$. Without loss of generality, we will assume that σ_1 is fixed (see below).

Definition 4.2. Consider the epoch $I_i = [\sigma_i, \tau_i]$ and its corresponding scaled time interval $\hat{I}_i = [\hat{\sigma}_i, \hat{\tau}_i] = [i-1, i]$. A scaled simple discontinuity, scaled point objective or scaled point constraint, occurring at time $t \in I_i$ is one that occurs at a fixed (stationary) point $s \in \hat{I}_i$ such that

$$\frac{s - \hat{\sigma}_i}{\hat{\tau}_i - \hat{\sigma}_i} = s - i + 1 = \frac{t - \sigma_i}{\tau_i - \sigma_i}.$$

It is clear from Definition 4.2 that there is a stationary simple discontinuity (Definition 2.6), point objective or point constraint at s^* in $\hat{I}_i$ iff there is a scaled simple discontinuity, point objective or point constraint at t^* in I_i.

Definition 4.3. The LTV ODE hybrid system of interest is defined by:

1. An index set M for the modes visited along T_μ, $M = \{1, \ldots, n_m\}$, and a $T_\tau = \{I_i\}$ that is allowed to vary. The mode trajectory is known a priori, i.e., $T_\mu = \{m_i^*\}$, where $m_i^* \in M$ is fixed for all $i = 1, \ldots, n_e$.

2. An invariant structure system where the number of continuous state variables is constant between modes, $V = \{\mathbf{x}, \mathbf{u}, \mathbf{p}, \boldsymbol{\delta}, t\}$, where $\mathbf{p} \in P \subset \mathbb{R}^{n_p}$, $\boldsymbol{\delta} \in \Delta \subset \mathbb{R}^{n_e}$, $\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) \in U \subset \mathbb{R}^{n_u}$ for all $(\mathbf{p}, \boldsymbol{\delta}, t) \in P \times \Delta \times I_i$, $i = 1, \ldots, n_e$, and $\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) \in \mathbb{R}^{n_x}$ for all $(\mathbf{p}, \boldsymbol{\delta}, t) \in P \times \Delta \times I_i$, $i = 1, \ldots, n_e$. The optimization parameter sets P and Δ are nondegenerate interval vectors (and hence compact and convex), $P = [\mathbf{p}^L, \mathbf{p}^U]$, $\Delta = [\boldsymbol{\delta}^L, \boldsymbol{\delta}^U]$. The lower bounds on Δ must satisfy the following constraint: $\boldsymbol{\delta}^L \geq \mathbf{0}$, because durations cannot be negative in the modeling framework.

3. The parameterization of the bounded real valued controls,

$$\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) \equiv \mathbf{S}(\boldsymbol{\delta}, t)\mathbf{p} + \mathbf{v}(\boldsymbol{\delta}, t),$$
$$\mathbf{u}^L(t) \leq \mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) \leq \mathbf{u}^U(t), \quad \forall t \in [\sigma_1, \sigma_1 + \sum_{j=1}^{n_e} \delta_j^U],$$

where $\mathbf{u}^L(t)$ and $\mathbf{u}^U(t)$ are known lower and upper bounds on the controls $\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t)$ that define the set U, and $\mathbf{S}(\boldsymbol{\delta}, t)$, $\mathbf{v}(\boldsymbol{\delta}, t)$ are piecewise continuous with a finite number of scaled simple discontinuities for each epoch I_i, and defined at any point of discontinuity.

4. The LTV ODE system for each mode $m_i^* \in M$, which is given by

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t)\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \tilde{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, t)\mathbf{p} + \tilde{\mathbf{C}}^{(m_i^*)}(\boldsymbol{\delta}, t)\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) + \tilde{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, t),$$

where $\mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t)$, $\tilde{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, t)$, $\tilde{\mathbf{C}}^{(m_i^*)}(\boldsymbol{\delta}, t)$, and $\tilde{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, t)$ are piecewise continuous with a finite number of scaled simple discontinuities for each epoch I_i, and defined at any point of discontinuity, for all $m_i^* \in M$. After control parameterization, we have

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t)\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \mathbf{B}^{(m_i^*)}(\boldsymbol{\delta}, t)\mathbf{p} + \mathbf{q}^{(m_i^*)}(\boldsymbol{\delta}, t), \tag{4.1}$$

where $\mathbf{B}^{(m_i^*)}(\boldsymbol{\delta}, t) \equiv \tilde{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, t) + \tilde{\mathbf{C}}(\boldsymbol{\delta}, t)\mathbf{S}(\boldsymbol{\delta}, t)$, and $\mathbf{q}^{(m_i^*)}(\boldsymbol{\delta}, t) \equiv \tilde{\mathbf{C}}(\boldsymbol{\delta}, t)\mathbf{v}(\boldsymbol{\delta}, t) + \tilde{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, t)$ are piecewise continuous with a finite number of scaled simple discontinuities for each epoch I_i, and defined at any point of discontinuity, for all $m_i^* \in M$.

5. The transition conditions for the transitions between epochs I_i and I_{i+1}, $i = 1, \ldots, n_e - 1$, which are variable time events, $L^{(m_i^*)} := (t \geq \tau_i)$, indicating the transition from mode m_i^* in epoch I_i to mode m_{i+1}^* in epoch I_{i+1} at time τ_i.

6. The collection of transition functions, which is given by the following equation,

$$\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \sigma_{i+1}) = \mathbf{D}_i \mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \tau_i) + \mathbf{E}_i \mathbf{p} + \mathbf{J}_i \boldsymbol{\delta} + \mathbf{k}_i, \ \forall i = 1, \ldots, n_e - 1, \tag{4.2}$$

for the transition from mode m_i^* in epoch I_i to mode m_{i+1}^* in epoch I_{i+1} at time τ_i.

7. A given initial condition for mode m_1^*:

$$\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \sigma_1) = \mathbf{E}_0 \mathbf{p} + \mathbf{J}_0 \boldsymbol{\delta} + \mathbf{k}_0.$$

Theorem 4.4. *A solution* $\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t)$*,* $t \in I_i$*,* $i = 1, \ldots, n_e$ *to the LTV ODE hybrid system exists and is unique for each* $(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta$*.*

Proof. Consider any arbitrary $(\mathbf{p}^*, \boldsymbol{\delta}^*) \in P \times \Delta$, and the first epoch I_1. Since $(\mathbf{p}^*, \boldsymbol{\delta}^*)$ is fixed, the form of the LTV ODE system in the first epoch satisfies the nonhomogenous linear system in [42, Chp. 3, pg 74]. Hence, there exists a unique solution of the hybrid system in the first epoch. At the transition to the second epoch, the initial conditions for the second epoch are clearly bounded by (4.2). Thus, the form of the LTV ODE system in the second epoch satisfies the nonhomogenous linear system in [42, Chp. 3, pg 74]. Therefore, a unique solution of the hybrid system in the second epoch exists. By induction, a unique solution of the hybrid system exists for all epochs $i = 1, \ldots, n_e$. Since $(\mathbf{p}^*, \boldsymbol{\delta}^*)$ was arbitrary, we obtain the desired result. □

Remark. It is possible to transcribe a problem in which σ_1 is a decision variable bounded by the interval $[\sigma_1^L, \sigma_1^U]$ into one where the initial time is fixed by prepending an additional mode to the hybrid system:

1. Introduce a new mode m_0^*.

2. Increase the number of durations (and epochs) by one, $\boldsymbol{\delta}^\dagger = (\delta_0, \boldsymbol{\delta})$, where $\delta_0 \in [0, \sigma_1^U - \sigma_1^L]$.

3. The new mode trajectory becomes $T_\mu^\dagger = m_0^*, T_\mu$.

4. The dynamics of the initial mode m_0^* are given by $\dot{\mathbf{x}}(t) = \mathbf{0}$ with initial condition $\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}^\dagger, \sigma_1^L) = \mathbf{E}_0\mathbf{p} + \mathbf{J}_0\boldsymbol{\delta} + \mathbf{k}_0$.

5. The transition condition from mode m_0^* to m_1^* occurs at the time event $L^{(m_0^*)} := (t = \sigma_1^L + \delta_0)$ with state continuity as the transition function, $\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}^\dagger, \sigma_1) = \mathbf{x}(\mathbf{p}, \boldsymbol{\delta}^\dagger, \sigma_1^L + \delta_0)$.

In general, it is very difficult to characterize the exact image of $P \times \Delta$ under the solution of the hybrid system (the implied state bounds first introduced in Section 2.7) when the transition times are varying, thus we will work with relaxations of the image set:

Definition 4.5. Define the following convex sets for all $i = 1, \ldots, n_e$ where $\mathcal{T}_i \equiv [\sigma_1 + \sum_{j=1}^{i} \delta_j^L, \sigma_1 + \sum_{j=1}^{i} \delta_j^U]$. For any fixed $\underline{t} \in \mathcal{T}_i$,

$$X^{(i)}(\underline{t}; P, \Delta) \equiv [\mathbf{x}^L(\underline{t}), \mathbf{x}^U(\underline{t})] \mid \mathbf{x}^L(\underline{t}) \leq \mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \underline{t}) \leq \mathbf{x}^U(\underline{t}),\ \forall (\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta.$$

In addition, $X^{(i)}(P, \Delta) \equiv [\mathbf{x}^L, \mathbf{x}^U] \mid X^{(i)}(\underline{t}; P, \Delta) \subset [\mathbf{x}^L, \mathbf{x}^U],\ \forall \underline{t} \in \mathcal{T}_i$.

We are now in position to present the problem that we are interested in solving.

Problem 4.6. Consider the following problem,

$$\min_{\mathbf{p} \in P, \boldsymbol{\delta} \in \Delta} \ F(\mathbf{p}, \boldsymbol{\delta}) \equiv \sum_{i=1}^{n_e} \left\{ \sum_{j=1}^{n_{\phi i}} \phi_{ij}\Big(\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \alpha_{ij}(\boldsymbol{\delta})), \mathbf{p}, \boldsymbol{\delta}\Big) + \int_{\sigma_i(\boldsymbol{\delta})}^{\tau_i(\boldsymbol{\delta})} f_i\big(\mathbf{x}, \mathbf{p}, \boldsymbol{\delta}, t\big)\, \mathrm{d}t \right\},$$

subject to the following point and isoperimetric constraints,

$$\mathbf{G}(\mathbf{p}, \boldsymbol{\delta}) \equiv \sum_{i=1}^{n_e} \left\{ \sum_{j=1}^{n_{\eta i}} \boldsymbol{\eta}_{ij}\Big(\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \beta_{ij}(\boldsymbol{\delta})), \mathbf{p}, \boldsymbol{\delta}\Big) + \int_{\sigma_i(\boldsymbol{\delta})}^{\tau_i(\boldsymbol{\delta})} \mathbf{g}_i\big(\mathbf{x}, \mathbf{p}, \boldsymbol{\delta}, t\big)\, \mathrm{d}t \right\} \ \leq \ \mathbf{0},$$

where $\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t)$ is given by the solution of the embedded LTV ODE hybrid system in Definition 4.3; f_i and $\mathbf{g}_i$ are piecewise continuous mappings $f_i : X^{(i)}(P, \Delta) \times P \times \Delta \times \mathcal{T}_i \to \mathbb{R}$ and $\mathbf{g}_i : X^{(i)}(P, \Delta) \times P \times \Delta \times \mathcal{T}_i \to \mathbb{R}^{n_c}$ for all $i = 1, \ldots, n_e$, where only a finite number of scaled simple discontinuities are allowed; $n_{\phi i}$ is an arbitrary number

of scaled point objectives in epoch I_i, $\alpha_{ij}(\boldsymbol{\delta}) \in I_i$ such that $\alpha_{ij}(\boldsymbol{\delta}) = \sigma_i + \delta_i(\hat{\alpha}_{ij} - i + 1)$ for some fixed $\hat{\alpha}_{ij} \in \hat{I}_i$, and ϕ_{ij} is a continuous mapping $\phi_{ij} : X^{(i)}(P, \Delta) \times P \times \Delta \to \mathbb{R}$ for all $j = 1, \ldots, n_{\phi i}$ and $i = 1, \ldots, n_e$; and $n_{\eta i}$ is an arbitrary number of scaled point constraints in epoch I_i, $\beta_{ij}(\boldsymbol{\delta}) \in I_i$ such that $\beta_{ij}(\boldsymbol{\delta}) = \sigma_i + \delta_i(\hat{\beta}_{ij} - i + 1)$ for some fixed $\hat{\beta}_{ij} \in \hat{I}_i$, and $\boldsymbol{\eta}_{ij}$ is a continuous mapping $\boldsymbol{\eta}_{ij} : X^{(i)}(P, \Delta) \times P \times \Delta \to \mathbb{R}^{n_c}$ for all $j = 1, \ldots, n_{\eta i}$ and $i = 1, \ldots, n_e$. Additionally, we require that the set $G = \{(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta \mid \mathbf{G}(\mathbf{p}, \boldsymbol{\delta}) \leq \mathbf{0}\}$ is nonempty.

Remark. It is possible to cast the optimization decision variables as the transition times $\boldsymbol{\tau}$ instead of the epoch durations $\boldsymbol{\delta}$. The equivalence between the two is established by the following equations,

$$\sigma_i = \sigma_1 + \sum_{j=1}^{i-1} \delta_j, \ \forall i = 1, \ldots, n_e,$$
$$\tau_i = \sigma_1 + \sum_{j=1}^{i} \delta_j, \ \forall i = 1, \ldots, n_e.$$

However, it is advantageous to work in terms of the epoch durations for the following reasons: (a) it is the natural formulation that facilitates the application of the CPET; and (b) the implicit constraints for feasible simulation trajectories using transition times,

$$\tau_{i-1} \leq \tau_i, \ \forall i = 2, \ldots, n_e$$

need to be added explicitly to the master NLP problem in the control parameterization framework, whereas the same constraints with the duration formulation are handled by the simple bound constraints $\boldsymbol{\delta}^L \geq \mathbf{0}$. This subtle difference is important in the control parametrization framework, as the decision variables passed to the IVP solver have to effect feasible simulations. For the majority of NLP solvers, this is handled much more robustly as simple bound constraints rather than as explicit constraints (i.e., simple bound constraints are satisfied throughout the solution process).

Comparing this problem formulation with that in Sections 2.4 and 3.1, we have the following main differences:

1. The type of discontinuities allowed: the participating functionals and hybrid system now include scaled simple discontinuities rather than stationary simple discontinuities. The following section will illustrate how fixed-time point objectives, stationary simple discontinuities in the integrand of the objective function and stationary simple discontinuities in the dynamics of the hybrid system can cause nonsmoothness in the problem. A discussion of sufficient conditions for the smoothness of the problem is deferred to the next Section.

2. The removal of the functional dependence on the state derivatives, $\dot{\mathbf{x}}$, from the objective and constraint functionals. Again, the reason for this is that the application of bounding techniques for time transformed hybrid systems produces bounds for only the state variables, and not their time derivatives. Note that the time derivatives of any constructed bounding system do not produce rigorous bounds for $\dot{\mathbf{x}}$. This is not a strong concern, because interval extensions of the right hand sides of the differential equations would give valid estimates for the bounds on the state derivatives. However, this is outside the scope of this Chapter and will not be discussed further.

4.1.1 Nonsmooth Examples

Example 4.7 (Fixed point objective)**.** Consider the following problem

$$\min_{\delta_1 \in [0.5, 1.5]} \quad F(\delta_1) \equiv x(\delta_1, 1) + x(\delta_1, 2), \tag{4.3}$$

subject to the following hybrid system

$$\text{Mode 1} \ : \dot{x}(t) = 0, \quad \text{Mode 2} \ : \dot{x}(t) = 1,$$

with $\sigma_1 = 0$, $x(\delta_1, 0) = 0$, $\delta_2 = 2 - \delta_1$, $t \in [0, 2]$, $T_\mu = 1, 2$, the transition condition $L_1^{(1)} := (t = \tau_1)$ and state continuity as the transition function, $x(\delta_1, \sigma_2) = x(\delta_1, \tau_1)$.

The sequence of modes for this hybrid system is fixed, and the optimization parameter is the timing of the transition from the initial mode 1 to the final mode 2.

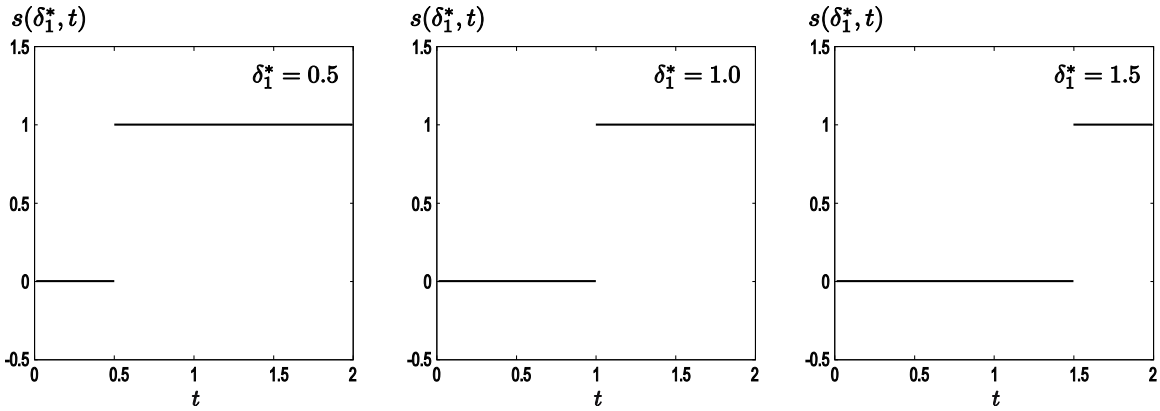


Figure 4-2: Sensitivity trajectories for Example 4.7.

Let the parametric sensitivity for this hybrid system be given by $s \equiv \frac{\partial x}{\partial \delta_1}$. Then, the parametric sensitivity exists, and is given by

$$s(\delta_1^*, t) = \begin{cases} 0, & \text{if } t \in I_1, \\ 1, & \text{if } t \in I_2. \end{cases}$$

for any $\delta_1^* \in [0.5, 1.5]$. Note that just as the continuous state can take on two values at the epoch boundaries, the parametric sensitivity can take on two values at $t = \tau_1 = \sigma_2$, i.e., $s(\delta_1^*, \tau_1) = 0$ and $s(\delta_1^*, \sigma_2) = 1$. The notion of the current epoch makes it clear which value of the parametric sensitivity we are referring to.

Figure 4-2 shows the sensitivity trajectories for various values of δ_1^*. Let $y \equiv x(\delta_1, 1)$ and $z \equiv x(\delta_1, 2)$. Note that the partial derivatives of the objective function, $\frac{\partial F}{\partial y}$ and $\frac{\partial F}{\partial z}$ exist and are continuous. However, the objective function is not continuously differentiable with respect to δ_1 on (0.5,1.5), as can be seen from Figure 4-3, which shows the objective function against δ_1. There is clearly a point of nonsmoothness at $\delta_1 = 1$, and the objective function is given by

$$F(\delta_1) = \begin{cases} 3 - 2\delta_1, & \text{if } \delta_1 < 1, \\ 2 - \delta_1, & \text{if } \delta_1 \geq 1. \end{cases}$$

Hence, for fixed-time point objectives in the *interior* of the time horizon, a simple

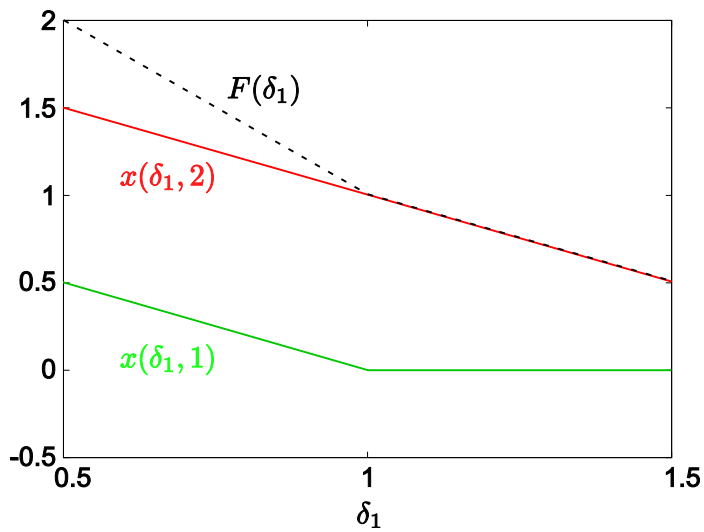


Figure 4-3: Objective function for Example 4.7.

extension of the sufficient conditions proposed in Theorem 2.9 and [62] is not possible. In fact, the source of the nonsmoothness in the objective function arises due to the sequence of modes changing for the fixed-time interior point objectives (e.g., if the objective function for Example 4.7 was $F(\delta_1) = x(\delta_1, 1)$, then the sequence of modes changes at the fixed time $t = 1$).

Example 4.8 (Integrand with simple stationary discontinuity)**.** Consider the following problem

$$\min_{\delta_1 \in [0.5, 1.5]} \quad F(\delta_1) \equiv \int_{\sigma_1}^{\tau_2(\delta_1)} f(x(t))\ \mathrm{d}t,$$

where

$$f(x(t)) = \begin{cases} 0, & \text{if } t < 1, \\ x(t) + 1, & \text{if } t \geq 1, \end{cases}$$

subject to the following hybrid system,

$$\text{Mode 1} \ : \dot{x}(t) = 0,$$

with $\sigma_1 = 0$, $x(\delta_1, 0) = 0$, $\delta_2 = 2 - \delta_1$, $t \in [0, 2]$, $T_\mu = 1, 1$, the transition condition

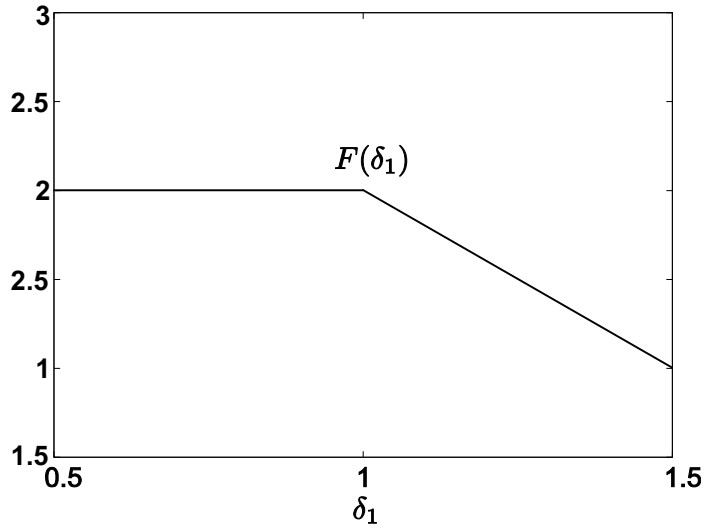


Figure 4-4: Objective function for Example 4.8.

$L_1^{(1)} := (t = \tau_1)$ and the following transition function,

$$x(\delta_1, \sigma_2) = x(\delta_1, \tau_1) + 1.$$

In this example, we have a hybrid system in which there is only one mode, and a unit jump in the value of the state variable is enforced at the transition. The integrand is piecewise continuous with a simple stationary discontinuity. The objective function is not continuously differentiable with respect to δ_1 on (0.5,1.5), as can be seen from Figure 4-4, which shows the objective function against δ_1. There is clearly a point of nonsmoothness at $\delta_1 = 1$, and the objective function is given by

$$F(\delta_1) = \begin{cases} 2, & \text{if } \delta_1 < 1, \\ 4 - 2\delta_1, & \text{if } \delta_1 \geq 1. \end{cases}$$

Hence, even simple stationary discontinuities within the integral type objective function causes nonsmoothness of the objective function. This is somewhat surprising, for one would expect the smoothing effect of the integral to mitigate and remove the effect of a simple stationary discontinuity which is of measure zero. Again, one can view the source of the nonsmoothness in the objective function as arising due to the

sequence of modes changing at the point of the simple stationary discontinuity.

Example 4.9 (Piecewise continuous dynamic system)**.** Consider the following problem

$$\min_{\delta_1 \in [0.5,1.5]} \quad F(\delta_1) \equiv x(\delta_1, 2),$$

subject to the following hybrid system

$$\text{Mode 1} \ : \dot{x}(t) = \begin{cases} 0, & \text{if } t < 1, \\ 1, & \text{if } t \geq 1, \end{cases} \qquad \text{Mode 2} \ : \dot{x}(t) = 0,$$

with $\sigma_1 = 0$, $x(\delta_1, 0) = 0$, $\delta_2 = 2 - \delta_1$, $t \in [0, 2]$, $T_\mu = 1, 2$, the transition condition $L_1^{(1)} := (t = \tau_1)$ and state continuity as the transition function, $x(\delta_1, \sigma_2) = x(\delta_1, \tau_1)$.

The right hand side of the ODE in mode 1 is piecewise continuously differentiable. However, the objective function is not continuously differentiable on δ_1 on (0.5,1.5), as can be seen from Figure 4-5, which shows the objective function against δ_1. There is clearly a point of nonsmoothness at $\delta_1 = 1$. In fact, the analytical expression for the objective function is given by

$$F(\delta_1) = \begin{cases} 0, & \text{if } \delta_1 < 1, \\ \delta_1 - 1, & \text{if } \delta_1 \geq 1. \end{cases}$$

Again, we see the effect of stationary discontinuities, this time in the form of the embedded hybrid system.

4.2 Time Transformation

The CPET (see e.g., [85, 127, 86] for details) is implemented as follows. Consider the original independent variable time (t) in Problem 4.6. We now wish to construct a new time scale in which the varying epoch durations (transition times) are fixed, $s \in [0, n_e]$. The transformation (CPET) from $t \in [\sigma_1, \sigma_1 + \sum_{i=1}^{n_e} \delta_i^U]$ to $s \in [0, n_e]$ is

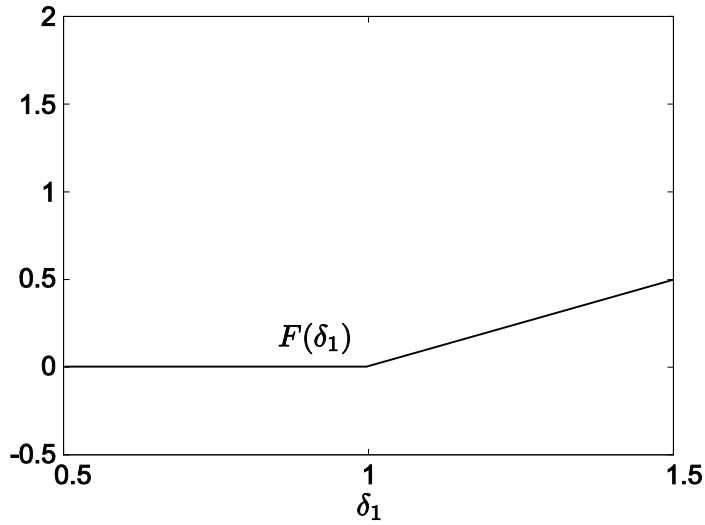


Figure 4-5: Objective function for Example 4.9.

defined by

$$\frac{\mathrm{d}t}{\mathrm{d}s} = v(\boldsymbol{\delta}, s), \quad t(\boldsymbol{\delta}, 0) = \sigma_1, \tag{4.4}$$

where the function $v : \Delta \times [0, n_e] \to \mathbb{R}$ is called the *enhancing control*. It is a piecewise constant function with possible simple discontinuities at the prefixed knots $s = 1, \ldots, n_e - 1$,

$$v(\boldsymbol{\delta}, s) = \sum_{i=1}^{n_e} \delta_i \chi_i(s),$$

where $\chi_i(s)$ is the indicator function defined by

$$\chi_i(s) = \begin{cases} 1 & \text{if } s \in [i-1, i], \\ 0 & \text{otherwise.} \end{cases}$$

Clearly,

$$t(\boldsymbol{\delta}, s) = \sigma_1 + \int_0^s v(\boldsymbol{\delta}, z)\ \mathrm{d}z = \sigma_1 + \delta_i(s - (i-1)) + \sum_{j=1}^{i-1} \delta_j = (s - i + 1)\delta_i + \sigma_i \tag{4.5}$$

for $s \in [i-1, i],\ i = 1, \ldots, n_e$, where the value of the enhancing control on the transformed time interval $(i-1, i)$ corresponds to the value of the duration of epoch I_i in the original time scale. In addition, the scaled simple discontinuities, point objectives

and point constraints in Problem 4.6 become stationary simple discontinuities, point objectives and point constraints in the new time scale, according to Definition 4.2. Finally, let $\mathbf{x}' \equiv \frac{\mathrm{d}\mathbf{x}}{\mathrm{d}s}$. It follows from the CPET that

$$\frac{\mathbf{x}'(\mathbf{p}, \boldsymbol{\delta}, t(\boldsymbol{\delta}, s))}{v(\boldsymbol{\delta}, s)} = \Big(\mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t(\boldsymbol{\delta}, s))\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t(\boldsymbol{\delta}, s)) + \mathbf{B}^{(m_i^*)}(\boldsymbol{\delta}, t(\boldsymbol{\delta}, s))\mathbf{p} + \mathbf{q}^{(m_i^*)}(\boldsymbol{\delta}, t(\boldsymbol{\delta}, s))\Big),$$

where t is an additional differential state variable that has to satisfy (4.4). We can substitute for the explicit form of $t(\boldsymbol{\delta}, s)$ to obtain

$$\hat{\mathbf{x}}'(\mathbf{p}, \boldsymbol{\delta}, s) = v(\boldsymbol{\delta}, s)\Big(\hat{\mathbf{A}}^{(m_i^*)}(\boldsymbol{\delta}, s)\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) + \hat{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, s)\mathbf{p} + \hat{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, s)\Big), \tag{4.6}$$

where $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) \equiv \mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t(\boldsymbol{\delta}, s))$, $\hat{\mathbf{x}}' \equiv \frac{\mathrm{d}\hat{\mathbf{x}}}{\mathrm{d}s}$, $\hat{\mathbf{A}}^{(m_i^*)}(\boldsymbol{\delta}, s) \equiv \mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t(\boldsymbol{\delta}, s))$, $\hat{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, s) \equiv \mathbf{B}^{(m_i^*)}(\boldsymbol{\delta}, t(\boldsymbol{\delta}, s))$, $\hat{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, s) \equiv \mathbf{q}^{(m_i^*)}(\boldsymbol{\delta}, t(\boldsymbol{\delta}, s))$, and $t(\boldsymbol{\delta}, s)$ is given by (4.5).

Consider now any $i \in \{1, \ldots, n_e\}$. It is clear that $v(\boldsymbol{\delta}, s) = \delta_i$ is continuous on $\Delta \times (i-1, i)$, and defined at the points of discontinuity $s = i-1$ and $s = i$. Also, $t(\boldsymbol{\delta}, s)$ is continuous on $\Delta \times [i-1, i]$ from (4.5). Let the epoch I_i be split into a finite number of contiguous intervals (subepochs) where $\mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t)$, $\mathbf{B}^{(m_i^*)}(\boldsymbol{\delta}, t)$ and $\mathbf{q}^{(m_i^*)}(\boldsymbol{\delta}, t)$ are continuous internal to each subepoch.

From Definition 4.2, the scaled simple discontinuities (in t) for $\mathbf{A}^{(m_i^*)}(\boldsymbol{\delta}, t)$, $\mathbf{B}^{(m_i^*)}(\boldsymbol{\delta}, t)$ and $\mathbf{q}^{(m_i^*)}(\boldsymbol{\delta}, t)$ become stationary simple discontinuities (in s) for $\hat{\mathbf{A}}^{(m_i^*)}(\boldsymbol{\delta}, s)$, $\hat{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, s)$ and $\hat{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, s)$. Internal to any arbitrary, transformed subepoch in $\hat{I}_i$, $\hat{\mathbf{A}}^{(m_i^*)}(\boldsymbol{\delta}, s)$, $\hat{\mathbf{B}}^{(m_i^*)}(\boldsymbol{\delta}, s)$ and $\hat{\mathbf{q}}^{(m_i^*)}(\boldsymbol{\delta}, s)$ are continuous (this follows from the fact that the composition of continuous functions is also continuous [116, Theorems 9.15 and 4.7]). The right hand side of (4.6) is thus piecewise continuous in s with a finite number of stationary simple discontinuities, defined at each point of

discontinuity. The objective function and constraints after the CPET are given by

$$\hat{F}(\mathbf{p},\boldsymbol{\delta}) \equiv \sum_{i=1}^{n_e}\left\{\sum_{j=1}^{n_{\phi i}}\phi_{ij}\Big(\hat{\mathbf{x}}(\mathbf{p},\boldsymbol{\delta},\hat{\alpha}_{ij}),\mathbf{p},\boldsymbol{\delta}\Big)+\int_{i-1}^{i} f_i\Big(\hat{\mathbf{x}},\mathbf{p},\boldsymbol{\delta},t(\boldsymbol{\delta},s)\Big)v(\boldsymbol{\delta},s)\ \mathrm{d}s\right\}, \tag{4.7}$$

$$\hat{\mathbf{G}}(\mathbf{p},\boldsymbol{\delta}) \equiv \sum_{i=1}^{n_e}\left\{\sum_{j=1}^{n_{\eta i}}\boldsymbol{\eta}_{ij}\Big(\hat{\mathbf{x}}(\mathbf{p},\boldsymbol{\delta},\hat{\beta}_{ij}),\mathbf{p},\boldsymbol{\delta}\Big)+\int_{i-1}^{i} \mathbf{g}_i\Big(\hat{\mathbf{x}},\mathbf{p},\boldsymbol{\delta},t(\boldsymbol{\delta},s)\Big)v(\boldsymbol{\delta},s)\ \mathrm{d}s\right\}. \tag{4.8}$$

Note that $\hat{\alpha}_{ij}$ and $\hat{\beta}_{ij}$ are no longer a function of $\boldsymbol{\delta}$. Henceforth, we shall use the superscript prime notation to denote the transformed time derivative, i.e., $' \equiv \frac{\mathrm{d}}{\mathrm{d}s}$. We are now able to formally state the transformed hybrid system and problem:

Definition 4.10. The transformed nonlinear hybrid system is given by the following:

1. An index set M for the modes visited along T_μ, $M = \{1,\ldots,n_m\}$, with a fixed $T_\mu = \{m_i^*\}$ and a fixed $T_\tau = \{\hat{I}_i\}$, where $\hat{I}_i = [\hat{\sigma}_i,\hat{\tau}_i] = [i-1,i]$ for all $i = 1,\ldots,n_e$.

2. An invariant structure system where the number of continuous state variables is constant between modes, $V = \{\hat{\mathbf{x}},\mathbf{p},\boldsymbol{\delta},s\}$, where $\mathbf{p}\in P\subset\mathbb{R}^{n_p}$, $\boldsymbol{\delta}\in\Delta\subset\mathbb{R}^{n_e}$, and $\hat{\mathbf{x}}(\mathbf{p},\boldsymbol{\delta},s)\in\mathbb{R}^{n_x}$ for all $(\mathbf{p},\boldsymbol{\delta},s)\in P\times\Delta\times\hat{I}_i$, $i=1,\ldots,n_e$. The optimization parameter sets P and Δ are interval vectors (and hence compact and convex), $P = [\mathbf{p}^L,\mathbf{p}^U]$, $\Delta = [\boldsymbol{\delta}^L,\boldsymbol{\delta}^U]$, where the lower bounds on Δ must satisfy the following constraint: $\boldsymbol{\delta}^L \geq \mathbf{0}$.

3. The nonlinear ODE system for each mode $m_i^*\in M$, which is given by (4.6). This will be represented, for convenience, as the following differential equations,

$$\hat{\mathbf{x}}' = \boldsymbol{\mathcal{F}}^{(m_i^*)}(\hat{\mathbf{x}},\mathbf{p},\boldsymbol{\delta},s), \tag{4.9}$$

   for all $m_i^*\in M$. For each mode $m_i^*\in M$, $\boldsymbol{\mathcal{F}}^{(m_i^*)}$ is piecewise continuous with a finite number of stationary simple discontinuities in s, defined at any point of discontinuity.

4. The transition conditions for the transitions between epochs $\hat{I}_i$ and $\hat{I}_{i+1}$, $i = 1, \ldots, n_e - 1$, which are explicit (fixed) time events, $L^{(m_i^*)} := (s \geq i)$, indicating the transition from mode m_i^* in epoch $\hat{I}_i$ to mode m_{i+1}^* in epoch $\hat{I}_{i+1}$ at time $\hat{\tau}_i$.

5. The collection of transition functions, which is given by the following equation,

$$\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \hat{\sigma}_{i+1}) = \mathbf{D}_i \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \hat{\tau}_i) + \mathbf{E}_i \mathbf{p} + \mathbf{J}_i \boldsymbol{\delta} + \mathbf{k}_i, \ \forall i = 1, \ldots, n_e - 1. \tag{4.10}$$

   for the transition from mode m_i^* in epoch $\hat{I}_i$ to mode m_{i+1}^* in epoch $\hat{I}_{i+1}$ at time $\hat{\tau}_i$.

6. A given initial condition for mode m_1^*:

$$\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, 0) = \mathbf{E}_0 \mathbf{p} + \mathbf{J}_0 \boldsymbol{\delta} + \mathbf{k}_0.$$

The corresponding relaxations for the image set under the solution of the transformed hybrid system are given by the following:

Definition 4.11. Define the following convex sets for all $i = 1, \ldots, n_e$. For any fixed $\underline{s} \in \hat{I}_i$:

$$\hat{X}^{(i)}(\underline{s}; P, \Delta) \equiv [\hat{\mathbf{x}}^L(\underline{s}), \hat{\mathbf{x}}^U(\underline{s})] \mid \hat{\mathbf{x}}^L(\underline{s}) \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \underline{s}) \leq \hat{\mathbf{x}}^U(\underline{s}), \forall (\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta.$$

In addition, $\hat{X}^{(i)}(P, \Delta) \equiv [\hat{\mathbf{x}}^L, \hat{\mathbf{x}}^U] \mid \hat{X}^{(i)}(\underline{s}; P, \Delta) \subset [\hat{\mathbf{x}}^L, \hat{\mathbf{x}}^U], \forall \underline{s} \in \hat{I}_i$.

Problem 4.12. The transformed problem is given by the following,

$$\begin{aligned} \min_{\mathbf{p} \in P, \boldsymbol{\delta} \in \Delta} \quad & \hat{F}(\mathbf{p}, \boldsymbol{\delta}) \\ \text{s.t.} \quad & \hat{\mathbf{G}}(\mathbf{p}, \boldsymbol{\delta}) \leq \mathbf{0}, \end{aligned}$$

where $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s)$ is given by the solution of the embedded nonlinear hybrid system in Definition 4.10; $\hat{F}(\mathbf{p}, \boldsymbol{\delta})$ and $\hat{\mathbf{G}}(\mathbf{p}, \boldsymbol{\delta})$ are given by (4.7) and (4.8) respectively; $\hat{f}_i$ and $\hat{\mathbf{g}}_i$ are piecewise continuous mappings $\hat{f}_i : \hat{X}^{(i)}(P, \Delta) \times P \times \Delta \times \hat{I}_i \to \mathbb{R}$ and

$\hat{\mathbf{g}}_i : \hat{X}^{(i)}(P,\Delta) \times P \times \Delta \times \hat{I}_i \to \mathbb{R}^{n_c}$, for all $i = 1,\ldots,n_e$, with a finite number of stationary simple discontinuities; $n_{\phi i}$ is the number of fixed point objectives in epoch $\hat{I}_i$, $\hat{\alpha}_{ij} \in \hat{I}_i$ and $\hat{\phi}_{ij}$ is a continuous mapping $\hat{\phi}_{ij} : \hat{X}^{(i)}(\hat{\alpha}_{ij}; P,\Delta) \times P \times \Delta \to \mathbb{R}$ for all $j = 1,\ldots,n_{\phi i}$ and $i = 1,\ldots,n_e$; and $n_{\eta i}$ is the number of fixed point constraints in epoch I_i, $\hat{\beta}_{ij} \in \hat{I}_i$ and $\hat{\boldsymbol{\eta}}_{ij}$ is a continuous mapping $\hat{\boldsymbol{\eta}}_{ij} : \hat{X}^{(i)}(\hat{\beta}_{ij}; P,\Delta) \times P \times \Delta \to \mathbb{R}^{n_c}$ for all $j = 1,\ldots,n_{\eta i}$ and $i = 1,\ldots,n_e$. Additionally, we require that the set $\hat{G} = \{(\mathbf{p},\boldsymbol{\delta}) \in P \times \Delta \mid \hat{\mathbf{G}}(\mathbf{p},\boldsymbol{\delta}) \leq \mathbf{0}\}$ is nonempty.

Lemma 4.13. *Consider the hybrid systems defined in Definitions 4.3 and 4.10. Let* $\mathbf{x}$ *be the solution of the hybrid system in Definition 4.3, and* $\hat{\mathbf{x}}$ *be the solution of the hybrid system in Definition 4.10. Then, for any* $(\mathbf{p},\boldsymbol{\delta},s) \in P \times \Delta \times \hat{I}_i$, $\hat{\mathbf{x}}(\mathbf{p},\boldsymbol{\delta},s) = \mathbf{x}(\mathbf{p},\boldsymbol{\delta},t(\boldsymbol{\delta},s))$ *for all* $i \in \{1,\ldots,n_e\}$, *where* $t(\boldsymbol{\delta},s)$ *is given by* (4.5).

Proof. Consider any arbitrary $(\mathbf{p}^*,\boldsymbol{\delta}^*) \in P \times \Delta$, and the first epoch in the transformed time scale, $\hat{I}_1$. From (4.5), $t(\boldsymbol{\delta}^*,\hat{\sigma}_1) = \sigma_1$. Hence, from the initial conditions in Definitions 4.3 and 4.10, $\hat{\mathbf{x}}(\mathbf{p}^*,\boldsymbol{\delta}^*,\hat{\sigma}_1) = \mathbf{x}(\mathbf{p}^*,\boldsymbol{\delta}^*,\sigma_1)$. Integrating (4.6), we obtain

$$\int_0^s \hat{\mathbf{x}}'(z)\ \mathrm{d}z = \int_0^s \Big(\hat{\mathbf{A}}^{(m_1^*)}(\boldsymbol{\delta},z)\hat{\mathbf{x}}(\mathbf{p},\boldsymbol{\delta},z) + \hat{\mathbf{B}}^{(m_1^*)}(\boldsymbol{\delta},z)\mathbf{p} + \hat{\mathbf{q}}^{(m_1^*)}(\boldsymbol{\delta},z)\Big)v(\boldsymbol{\delta},z)\ \mathrm{d}z$$

and with the change of variables $t(\boldsymbol{\delta}^*,s)$ given by (4.5),

$$\int_0^s \hat{\mathbf{x}}'(z)\ \mathrm{d}z = \int_{\sigma_1}^{t(\boldsymbol{\delta}^*,s)} \Big(\mathbf{A}^{(m_1^*)}(w)\mathbf{x}(\mathbf{p},\boldsymbol{\delta},w) + \mathbf{B}^{(m_1^*)}(w)\mathbf{p} + \hat{\mathbf{q}}^{(m_1^*)}(w)\Big)\ \mathrm{d}w,$$

or

$$\hat{\mathbf{x}}(\mathbf{p}^*,\boldsymbol{\delta}^*,s) - \hat{\mathbf{x}}(\mathbf{p}^*,\boldsymbol{\delta}^*,\hat{\sigma}_1) = \mathbf{x}(\mathbf{p}^*,\boldsymbol{\delta}^*,t(\boldsymbol{\delta}^*,s)) - \mathbf{x}(\mathbf{p}^*,\boldsymbol{\delta}^*,\sigma_1).$$

Therefore, for all $s \in \hat{I}_1$, $\hat{\mathbf{x}}(\mathbf{p}^*,\boldsymbol{\delta}^*,s) = \mathbf{x}(\mathbf{p}^*,\boldsymbol{\delta}^*,t(\boldsymbol{\delta}^*,s))$. At the transition to epoch 2, from (4.5), $t(\boldsymbol{\delta}^*,\hat{\tau}_1) = \tau_1$, thus $\hat{\mathbf{x}}(\mathbf{p}^*,\boldsymbol{\delta}^*,\hat{\tau}_1) = \mathbf{x}(\mathbf{p}^*,\boldsymbol{\delta}^*,\tau_1)$. Applying the transition functions in Definitions 4.3 and 4.10, we obtain $\hat{\mathbf{x}}(\mathbf{p}^*,\boldsymbol{\delta}^*,\hat{\sigma}_2) = \mathbf{x}(\mathbf{p}^*,\boldsymbol{\delta}^*,\sigma_2)$. Since the choice of $(\mathbf{p}^*,\boldsymbol{\delta}^*)$ was arbitrary, induction on all epochs gives $\hat{\mathbf{x}}(\mathbf{p},\boldsymbol{\delta},s) = \mathbf{x}(\mathbf{p},\boldsymbol{\delta},t(\boldsymbol{\delta},s))$ for all $i \in \{1,\ldots,n_e\}$, for any $(\mathbf{p},\boldsymbol{\delta},s) \in P \times \Delta \times \hat{I}_i$. $\square$

Remark. A solution, $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s)$, $s \in \hat{I}_i$, $i = 1, \ldots, n_e$, will exist and be unique for all $(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta$. This follows directly from Theorem 4.4 and Lemma 4.13.

Theorem 4.14. *Problem 4.6 has the solution* $(\mathbf{p}^*, \boldsymbol{\delta}^*)$ *iff* $(\mathbf{p}^*, \boldsymbol{\delta}^*)$ *is a solution of Problem 4.12.*

Proof. Consider any arbitrary $(\mathbf{p}^*, \boldsymbol{\delta}^*) \in P \times \Delta$, and any arbitrary epoch $i \in \{1, \ldots, n_e\}$. For any $j \in \{1, \ldots, n_{\phi i}\}$, from Lemma 4.13 and (4.5), we have

$$\phi_{ij}\Big(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\alpha}_{ij}), \mathbf{p}^*, \boldsymbol{\delta}^*\Big) = \phi_{ij}\Big(\mathbf{x}(\mathbf{p}^*, \boldsymbol{\delta}^*, \alpha_{ij}(\boldsymbol{\delta}^*)), \mathbf{p}^*, \boldsymbol{\delta}^*\Big).$$

Similarly,

$$\int_{\hat{\sigma}_i}^{\hat{\tau}_i} f_i\Big(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s), \mathbf{p}^*, \boldsymbol{\delta}^*, t(\boldsymbol{\delta}^*, s)\Big) v(\boldsymbol{\delta}^*, s)\ \mathrm{d}s = \int_{\sigma_i(\boldsymbol{\delta}^*)}^{\tau_i(\boldsymbol{\delta}^*)} f_i\Big(\mathbf{x}(\mathbf{p}^*, \boldsymbol{\delta}^*, t), \mathbf{p}^*, \boldsymbol{\delta}^*, t\Big)\ \mathrm{d}t.$$

Hence, we have $\hat{F}(\mathbf{p}^*, \boldsymbol{\delta}^*) = F(\mathbf{p}^*, \boldsymbol{\delta}^*)$. Applying the same analysis to the constraints, we have $\hat{\mathbf{G}}(\mathbf{p}^*, \boldsymbol{\delta}^*) = \mathbf{G}(\mathbf{p}^*, \boldsymbol{\delta}^*)$. Since $(\mathbf{p}^*, \boldsymbol{\delta}^*)$ was arbitrary, we have shown the equivalence of Problems 4.6 and 4.12. □

Remark. A consequence of the CPET transform is the destruction of the linear structure of the original hybrid system.

The following theorem presents sufficient conditions for the objective function (and analogously the constraints) to be smooth, and will be assumed to hold for the transformed problem.

Theorem 4.15. *Let* $P^o \supset P$, $\Delta^o \supset \Delta$, $\hat{X}^{(i)o}(\hat{\alpha}_{ij}) \supset \hat{X}^{(i)}(\hat{\alpha}_{ij}; P^o, \Delta^o)$ *and* $\hat{X}^{(i)o} \supset \hat{X}^{(i)}(P^o, \Delta^o)$ *be open subsets of* $\mathbb{R}^{n_p}$, $\mathbb{R}^{n_e}$, $\mathbb{R}^{n_x}$ *and* $\mathbb{R}^{n_x}$ *respectively, for all* $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$. *If the following conditions are satisfied, then the objective function* $\hat{F}$ *is continuously differentiable on* $P^o \times \Delta^o$.

C1. $\frac{\partial \phi_{ij}}{\partial \hat{\mathbf{x}}}, \frac{\partial \phi_{ij}}{\partial \mathbf{p}}$ *and* $\frac{\partial \phi_{ij}}{\partial \boldsymbol{\delta}}$ *exist, and are continuous on* $\hat{X}^{(i)o}(\hat{\alpha}_{ij}) \times P^o \times \Delta^o$ *for all* $j = 1, \ldots, n_{\phi i}$, $i = 1, \ldots, n_e$;

C2. $\frac{\partial f_i}{\partial \hat{\mathbf{x}}}, \frac{\partial f_i}{\partial \mathbf{p}}$ *and* $\frac{\partial f_i}{\partial \boldsymbol{\delta}}$ *exist, and are piecewise continuous on* $\hat{X}^{(i)o} \times P^o \times \Delta^o \times \hat{I}_i$ *for all* $i = 1, \ldots, n_e$ *where only a finite number of stationary simple discontinuities in* s *are allowed.*

Proof. Consider an arbitrary epoch $\hat{I}_i$. First, we show that the parametric sensitivities exist and are unique. We have a finite number of stationary discontinuities (in s) in (4.9). Let there be k such discontinuities in $\hat{I}_i$ found at points $s = \zeta_l$, $l = 1, \ldots, k$. Construct a sequence of $k+1$ subepochs $[\xi_1, \zeta_1], \ldots, [\xi_{k+1}, \zeta_{k+1}]$ where $\xi_1 = \hat{\sigma}_i$, $\zeta_{k+1} = \hat{\tau}_i$ and $\xi_{l+1} = \zeta_l$, $l = 1, \ldots, k$. Extend the function $\boldsymbol{\mathcal{F}}^{(m_i^*)}$ to be continuous on all subepochs,

$$\boldsymbol{\mathcal{F}}^{(m_i^*)}(\cdot, \xi_l) \equiv \lim_{s \to \xi_l^+} \boldsymbol{\mathcal{F}}^{(m_i^*)}(\cdot, s), \qquad \boldsymbol{\mathcal{F}}^{(m_i^*)}(\cdot, \zeta_l) \equiv \lim_{s \to \zeta_l^-} \boldsymbol{\mathcal{F}}^{(m_i^*)}(\cdot, s),$$

for $l = 1, \ldots, k+1$ and impose state continuity for each transition between sub-epochs. A hybrid system is thus defined within the epoch $\hat{I}_i$. Now consider an arbitrary subepoch $[\xi_l, \zeta_l]$. From (4.6), it is clear that the partial derivatives $\frac{\partial \boldsymbol{\mathcal{F}}^{(m_i^*)}}{\partial \hat{\mathbf{x}}}$, $\frac{\partial \boldsymbol{\mathcal{F}}^{(m_i^*)}}{\partial \mathbf{p}}$ and $\frac{\partial \boldsymbol{\mathcal{F}}^{(m_i^*)}}{\partial \boldsymbol{\delta}}$ exist and are continuous internal to this subepoch. At the transition ζ_l, it is easy to verify that the remaining assumptions of [63, Thm. 1] are satisfied, which provides the existence and uniqueness result (for an example of this, see the proof of Theorem 2.8). Since the discontinuities are stationary, the transition times in the interior of the epoch are independent of the parameters, and the sensitivities are continuous everywhere interior to the epoch.

Consider now an arbitrary $v \in \{1, \ldots, n_p\}$ and $w \in \{1, \ldots, n_e\}$. For the point objectives, since P^o and Δ^o are open sets, the parametric sensitivities exist and condition C1 holds, the summation terms $\sum_{j=1}^{n_{\phi i}} \frac{\partial \phi_{ij}}{\partial p_v}$ and $\sum_{j=1}^{n_{\phi i}} \frac{\partial \phi_{ij}}{\partial \delta_w}$ exist and are continuous on $P^o \times \Delta^o$ by the chain rule and linearity of the derivative operator. Next, consider the integral objectives. We have a finite number of stationary discontinuities (in s) in the integrand f_i, $\boldsymbol{\mathcal{F}}^{(m_i^*)}$, the parametric sensitivities and the derivatives in condition C2. Let there be $\hat{k}$ such discontinuities found at points $s = \hat{\zeta}_{\hat{l}}$, $\hat{l} = 1, \ldots, \hat{k}$.

Partition the integral into the following:

$$\hat{F}_i(\mathbf{p}, \boldsymbol{\delta}) = \sum_{\hat{l}=0}^{\hat{k}} \hat{F}_{i\hat{l}}(\mathbf{p}, \boldsymbol{\delta}) = \sum_{\hat{l}=0}^{\hat{k}} \int_{\zeta_{\hat{l}}}^{\zeta_{\hat{l}+1}} f_i\big(\hat{\mathbf{x}}, \mathbf{p}, \boldsymbol{\delta}, s\big)\, \mathrm{d}s,$$

where $\zeta_0 = \hat{\sigma}_i$ and $\zeta_{\hat{k}+1} = \hat{\tau}_i$. Now, consider an arbitrary $\hat{l} \in \{1, \ldots, \hat{k}\}$. Extend f_i, $\frac{\partial f_i}{\partial \hat{\mathbf{x}}}$, $\frac{\partial f_i}{\partial \mathbf{p}}$ and $\frac{\partial f_i}{\partial \boldsymbol{\delta}}$ to be continuous on $\hat{X}^{(i)o} \times P^o \times \Delta^o \times [\zeta_{\hat{l}}, \zeta_{\hat{l}+1}]$, and $\frac{\partial \hat{\mathbf{x}}}{\partial \mathbf{p}}$, $\frac{\partial \hat{\mathbf{x}}}{\partial \boldsymbol{\delta}}$, and $\mathbf{x}$ to be continuous on $P^o \times \Delta^o \times [\zeta_{\hat{l}}, \zeta_{\hat{l}+1}]$. At most, these functions are discontinuous at their endpoints in time. Removing these discontinuities does not alter the value of the integral because the endpoints comprise a set of measure zero. As above, applying the chain rule on the partial derivatives $\frac{\partial f_i}{\partial p_v}$ and $\frac{\partial f_i}{\partial \delta_w}$, we obtain continuity of said derivatives on $P^o \times \Delta^o \times [\zeta_{\hat{l}}, \zeta_{\hat{l}+1}]$. These continuity conditions enable us to differentiate under the integral sign [43, Page 308] to obtain

$$\frac{\partial \hat{F}_{i\hat{l}}}{\partial p_v} = \int_{\zeta_{\hat{l}}}^{\zeta_{\hat{l}+1}} \frac{\partial f_i}{\partial p_v}\, \mathrm{d}s, \quad \frac{\partial \hat{F}_{i\hat{l}}}{\partial \delta_w} = \int_{\zeta_{\hat{l}}}^{\zeta_{\hat{l}+1}} \frac{\partial f_i}{\partial \delta_w}\, \mathrm{d}s.$$

We can then apply [120, Proposition 2.1] to yield $\frac{\partial \hat{F}_{i\hat{l}}}{\partial p_v}$ and $\frac{\partial \hat{F}_{i\hat{l}}}{\partial \delta_w}$ continuous on $P^o \times \Delta^o$. Since $\hat{l}$ was arbitrary, $\frac{\partial \hat{F}_i}{\partial p_v}$ and $\frac{\partial \hat{F}_i}{\partial \delta_w}$ are continuous on $P^o \times \Delta^o$ as the sum of continuous functions is continuous. Since i was arbitrary, $\frac{\partial \hat{F}}{\partial p_v}$ and $\frac{\partial \hat{F}}{\partial \delta_w}$ are continuous on $P^o \times \Delta^o$. Since v and w were arbitrary, it follows that $\hat{F}$ is continuously differentiable on $P^o \times \Delta^o$. □

Note that the differentiation with respect to $\boldsymbol{\delta}$ and $\mathbf{p}$ is to develop conditions under which the resulting finite-dimensional problem is continuously differentiable. It is not related to obtaining a solution of the finite-dimensional problem. For example, consider the following problem:

$$\min_{p \in [-1,1], \delta \in [0,2]} x(p, \delta)$$
$$\text{s.t. } x(p, \delta) + 1 \geq 0$$

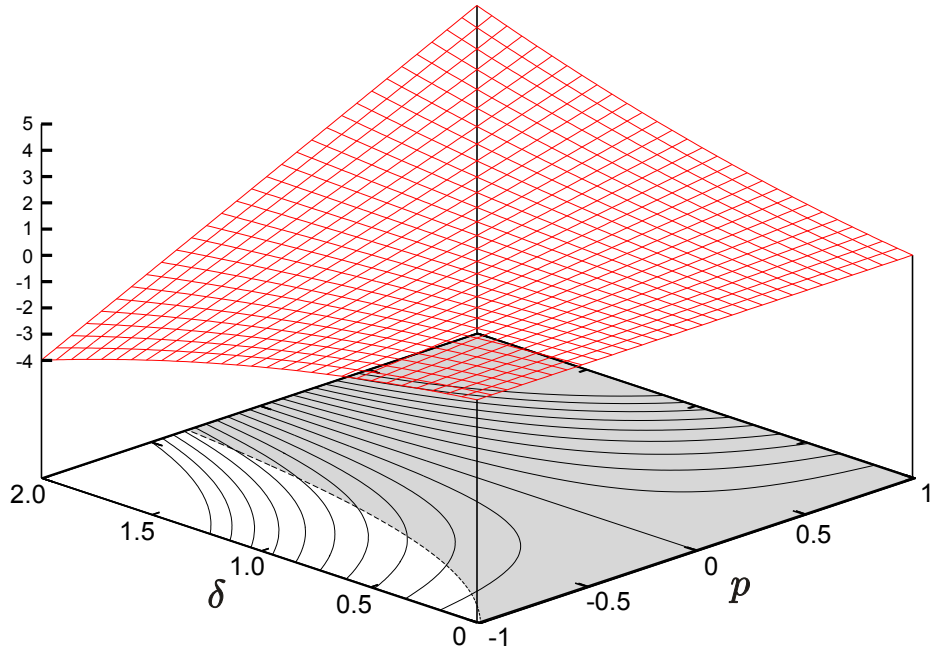


Figure 4-6: Objective function and feasible region.

where $x(p, \delta)$ is given by the solution of the following dynamic system,

$$\dot{x}(p, t) = p,$$
$$x(p, 0) = p,$$

where the time horizon is given by $t \in [0, \delta]$. The objective function and constraint functionals are linear in $x(p, \delta)$. Figure 4-6 shows the objective function surface on the set $[-1, 1] \times [0, 2]$. The solid lines that are projected onto the $p \times \delta$ plane represent contours of the objective function, and the shaded portion represents the feasible space of the problem. The global solution to this problem occurs at $p^* = -0.333$ and $\delta^* = 2.0$. Note that the solution to the problem does not occur on the boundaries of $p \in [-1, 1]$. In general, the solution for such problems will not lie on the boundaries of either $\mathbf{p}$ or $\boldsymbol{\delta}$, because the presence of constraints may cause the feasible region to become nonconvex as illustrated. In addition, when applying

transformation methods, e.g., the CPET, the embedded dynamic system becomes nonlinear.

4.3 Bounding Strategies for Time Transformed Hybrid Systems

In order to solve the transformed Problem 4.12 using a BB framework such as Algorithm 2.3, we have to develop a theory for constructing convex relaxations of the objective and constraint functionals (4.7) and (4.8) subject to the embedded nonlinear hybrid system in Definition 4.10. The steps for constructing such convex relaxations are outlined below:

1. Estimate the implied state bounds, $\hat{X}^{(i)}(\underline{s}; P, \Delta)$ in Definition 4.11.
2. Construct convex and concave relaxations for the states.
3. Apply convex relaxation techniques on subsets of Euclidean spaces to construct the required convex relaxations.

In this section, we shall examine various strategies for estimating the implied state bounds, before we present the convexity theory in the next section.

4.3.1 Nonlinear Differential Inequalities

We shall present some bounding theorems for single-stage nonlinear ODE systems, before extending these to multi-stage nonlinear ODE systems. First, we shall present a very simple bounding theorem for single-stage nonlinear ODE systems:

Consider the following nonlinear ODE system with bounded time varying controls, $\mathbf{u}$,

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}, t), \quad \mathbf{x}(\mathbf{u}(t_0), t_0) = \mathbf{x}_0(\mathbf{u}(t_0)) \tag{4.11}$$

where $\mathbf{u} : [t_0, t_f] \to \mathbb{R}^{n_u}$, $\mathbf{u}(t) \in U(t) \subset \mathbb{R}^{n_u}$, $\forall t \in [t_0, t_f]$, $U(t)$ is a nonempty compact set for each $t \in [t_0, t_f]$, $\mathbf{f}$ is a continuous mapping $\mathbf{f} : X \times U \times [t_0, t_f] \to \mathbb{R}^{n_x}$ that is

bounded, $\mathbf{x}_0$ is a continuous mapping $\mathbf{x}_0 : U(t_0) \to \mathbb{R}^{n_x}$, and the sets X and U are given by the following:

$$X \supset \{\mathbf{x}(\mathbf{u}(t), t) \mid t \in [t_0, t_f], \mathbf{u}(t) \in U(t)\},$$
$$U \supset \{\mathbf{u}(t) \mid t \in [t_0, t_f]\}.$$

As in [121], we will make the following assumptions concerning the solution of (4.11). First, we assume that a solution exists and is unique for each $\mathbf{u}$ such that $\mathbf{u}(t) \in U(t)$ for all $t \in [t_0, t_f]$. This assumption permits the use of regular (weak) inequalities rather than strict inequalities in the results below. This is needed for the application of the theorems below for general, nonlinear, hybrid systems. For the time transformed hybrid system in Definition 4.10, the solution to (4.9) exists and is unique from Theorem 4.4, so this assumption is always satisfied. Next, we assume that (4.11) possesses a solution in the sense of Carathéodory. That is, an admissible solution is one that satisfies (4.11) almost everywhere (a.e.). Therefore, this assumption immediately implies that the derivative of the state variables exists everywhere except possibly on a set of measure zero. By assumption, since a solution exists and is unique for each $\mathbf{u}$ such that $\mathbf{u}(t) \in U(t)$ for all $t \in [t_0, t_f]$, it follows that the solution of (4.11) will be bounded for any such control function $\mathbf{u}$. We can then define the following sets:

Definition 4.16. Let $\mathbf{x}(\mathbf{u}(t), t)$ be a solution of (4.11). For each fixed $\underline{t} \in [t_0, t_f]$ and all $i = 1, \ldots, n_x$, let the set $\mathcal{X}_i^c(\underline{t})$ be represented by the following,

$$\mathcal{X}_i^c(\underline{t}) = \{x_i(\mathbf{u}(\underline{t}), \underline{t}) \mid \mathbf{u}(\underline{t}) \in U(\underline{t})\}.$$

Furthermore, let the set $\mathcal{X}^c(\underline{t})$ be defined pointwise in time, for all $\underline{t} \in [t_0, t_f]$, by

$$\mathcal{X}^c(\underline{t}) = [\mathbf{z}^L, \mathbf{z}^U] \mid z_i^L = \inf \mathcal{X}_i^c(\underline{t}),\ z_i^U = \sup \mathcal{X}_i^c(\underline{t}),\ \forall i = 1, \ldots, n_x.$$

Note that, by definition of the infimum and supremum, the set $\mathcal{X}^c(\underline{t})$ represents the best possible, or exact, bounds on the value of the solution of the nonlinear system

for any fixed time $\underline{t} \in [t_0, t_f]$, i.e., for any $\underline{t} \in [t_0, t_f]$, if

$$\mathbf{v} \leq \mathbf{x}(\mathbf{u}(\underline{t}), \underline{t}) \leq \mathbf{w}, \ \forall \mathbf{u}(\underline{t}) \in U(\underline{t}),$$

then $\mathbf{v} \leq \mathbf{z}^L$ and $\mathbf{z}^U \leq \mathbf{w}$ where $\mathcal{X}^c(\underline{t}) \equiv [\mathbf{z}^L, \mathbf{z}^U]$. In addition, although we may not know what the set $\mathcal{X}^c(\underline{t})$ is, we know that such an exact bounding set exists because the solution of (4.11) is bounded for any control function $\mathbf{u}$ such that $\mathbf{u}(t) \in U(t)$ for all $t \in [t_0, t_f]$ as discussed above.

Theorem 4.17. *Consider the ODE system in* (4.11)*. If the following conditions are satisfied for* $i = 1, \ldots, n_x$

(i) $v_i(t_0) \leq \min_{\mathbf{q} \in U(t_0)} x_i(\mathbf{q}, t_0)$

(ii) $w_i(t_0) \geq \max_{\mathbf{q} \in U(t_0)} x_i(\mathbf{q}, t_0)$

(iii) $\dot{v}_i = \underline{h}_i(t) \leq \min_{\mathbf{z} \in \mathcal{X}^c(t), \mathbf{q} \in U(t)} f_i(\mathbf{z}, \mathbf{q}, t)$

(iv) $\dot{w}_i = \overline{h}_i(t) \geq \max_{\mathbf{z} \in \mathcal{X}^c(t), \mathbf{q} \in U(t)} f_i(\mathbf{z}, \mathbf{q}, t)$

where $\underline{h}_i$ *and* $\overline{h}_i$ *are continuous mappings, then*

$$\mathbf{v}(t) \leq \mathbf{x}(\mathbf{u}(t), t) \leq \mathbf{w}(t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

Proof. Since $U(t_0)$ is a nonempty compact set and $x_i(\mathbf{q}, t_0)$ is continuous on $U(t_0)$ for all $i = 1, \ldots, n_x$, the extrema in conditions (i) and (ii) exist. For each fixed $\underline{t}$, the set $\mathcal{X}^c(\underline{t})$ is nonempty and compact by definition. Also, since $f_i(\cdot, \underline{t})$ is continuous for each fixed $\underline{t}$ for all $i = 1, \ldots, n_x$, the extrema in conditions (iii) and (iv) exist.

We will now consider the lower bounding system, $\mathbf{v}(t)$. By construction, for each $i = 1, \ldots, n_x$ and $\mathbf{u}(t) \in U(t)$, we have

$$\dot{v}_i(t) \leq \min_{\mathbf{z} \in \mathcal{X}^c(t), \mathbf{q} \in U(t)} f_i(\mathbf{z}, \mathbf{q}, t) \leq f_i(\mathbf{x}(\mathbf{u}(t), t), \mathbf{u}(t), t) = \dot{x}_i(t).$$

By the property of the integral [116, Theorem 6.12(b)], we have

$$\int_{t_0}^{t^*} \dot{v}_i \, \mathrm{d}t \leq \int_{t_0}^{t^*} \dot{x}_i \, \mathrm{d}t,$$

for any $t^* \in [t_0, t_f]$. From condition (i), we have

$$v_i(t_0) \leq x_i(\mathbf{q}, t_0)$$

for any $\mathbf{q} \in U(t_0)$. Thus, from the fundamental theorem of calculus,

$$v_i(t) \leq x_i(\mathbf{u}(t), t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

An analogous proof holds for the upper bounding system, $\mathbf{w}(t)$. □

Note that the requirements for $\underline{h}_i$ and $\overline{h}_i$ to be continuous mappings for $i = 1, \ldots, n_x$ are a technical condition needed for the application of the fundamental theorem of calculus. However, these are not strong conditions in the sense that in the practical application of the theorem, one would typically utilize inclusion monotonic and continuous interval extensions to obtain estimates of the minimum and maximum, which would ensure that these functions are indeed continuous. These conditions on the interval extensions are also needed to prove convergence of the bounding technique, which will be presented below. On its own, Theorem 4.17 is hard to apply, because the exact bounds for the nonlinear system given by the set $\mathcal{X}^c(t), \forall t \in [t_0, t_f]$ is difficult to obtain. We have seen that it is possible to obtain the exact state bounds (implied state bounds) when the embedded dynamic system is a LTV ODE system with real valued parameters $\mathbf{p} \in P$, as shown in Chapter 2. However, the same cannot be said when the embedded system is nonlinear with an arbitrary bounded control function, $\mathbf{u}(t) \in U(t)$. On the other hand, if we had some means of obtaining the exact bounds, we wouldn't need Theorem 4.17 in the first place. The following (stronger) result, proved as a corollary of Theorem 4.17, seeks to provide a more practical way of estimating the state bounds without having to know the set $\mathcal{X}^c(t), \forall t \in [t_0, t_f]$ a priori.

Definition 4.18. Let $\mathbf{x}(\mathbf{u}(t),t)$ be a solution of (4.11), and let $x_i(\mathbf{u}(t),t) \in \mathcal{X}_i^d(\mathbf{u}(t),t)$ for each $\mathbf{u}(t) \in U(t)$, $i = 1,\ldots,n_x$, where $\mathcal{X}_i^d(\mathbf{u}(t),t) \subset \mathbb{R}$ is a closed bounding set that is known independently from the solution of (4.11). For each fixed $\underline{t} \in [t_0,t_f]$, let $\alpha_i(\mathbf{q},\underline{t}) = \inf \mathcal{X}_i^d(\mathbf{q},\underline{t})$ and $\beta_i(\mathbf{q},\underline{t}) = \sup \mathcal{X}_i^d(\mathbf{q},\underline{t})$ for each $\mathbf{q} \in U(\underline{t})$, $i = 1,\ldots,n_x$. Furthermore, let the set $\mathcal{X}^d(\underline{t})$ be defined pointwise in time by

$$\mathcal{X}^d(\underline{t}) = [\mathbf{z}^L,\mathbf{z}^U] \mid z_i^L = \inf_{\mathbf{q}\in U(\underline{t})} \alpha_i(\mathbf{q},\underline{t}),\ z_i^U = \sup_{\mathbf{q}\in U(\underline{t})} \beta_i(\mathbf{q},\underline{t}),\ \forall i = 1,\ldots,n_x,$$

where z_i^L and z_i^U are in the extended real number system [116, Definition 1.23].

Corollary 4.19. *Consider the ODE system in* (4.11)*. If the following conditions are satisfied for* $i = 1,\ldots,n_x$

(i) $v_i(t_0) \leq \min_{\mathbf{q}\in U(t_0)} x_i(\mathbf{q},t_0)$

(ii) $w_i(t_0) \geq \max_{\mathbf{q}\in U(t_0)} x_i(\mathbf{q},t_0)$

(iii) $\dot{v}_i = \underline{h}_i(\mathbf{v},\mathbf{w},t) \leq \min_{\mathbf{z}\in\mathcal{X}^d(t)\cap H(t),\mathbf{q}\in U(t)} f_i(\mathbf{z},\mathbf{q},t)$

(iv) $\dot{w}_i = \overline{h}_i(\mathbf{v},\mathbf{w},t) \geq \max_{\mathbf{z}\in\mathcal{X}^d(t)\cap H(t),\mathbf{q}\in U(t)} f_i(\mathbf{z},\mathbf{q},t)$

where $H(t) = \{\mathbf{z} \mid \mathbf{v}(t) \leq \mathbf{z} \leq \mathbf{w}(t)\}$*, and* $\underline{h}_i$ *and* $\overline{h}_i$ *are continuous mappings, then*

$$\mathbf{v}(t) \leq \mathbf{x}(\mathbf{u}(t),t) \leq \mathbf{w}(t),\ \forall \mathbf{u}(t) \in U(t), t \in [t_0,t_f].$$

Proof. First, we apply Theorem 4.17 to the differential system $\mathbf{v}$ and $\mathbf{w}$ that obeys the following conditions, for all $i = 1,\ldots,n_x$,

(i) $v_i(t_0) \leq \min_{\mathbf{q}\in U(t_0)} x_i(\mathbf{q},t_0)$

(ii) $w_i(t_0) \geq \max_{\mathbf{q}\in U(t_0)} x_i(\mathbf{q},t_0)$

(iii) $\dot{v}_i = \underline{h}_i(t) \leq \min_{\mathbf{z}\in\mathcal{X}^c(t),\mathbf{q}\in U(t)} f_i(\mathbf{z},\mathbf{q},t)$

(iv) $\dot{w}_i = \overline{h}_i(t) \geq \max_{\mathbf{z}\in\mathcal{X}^c(t),\mathbf{q}\in U(t)} f_i(\mathbf{z},\mathbf{q},t)$

where $\underline{h}_i$ and $\overline{h}_i$ are continuous mappings, to obtain

$$\mathbf{v}(t) \leq \mathbf{x}(\mathbf{u}(t), t) \leq \mathbf{w}(t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

Now, for each fixed $\underline{t} \in [t_0, t_f]$, let $H(\underline{t}) = \{\mathbf{z} \mid \mathbf{v}(\underline{t}) \leq \mathbf{z} \leq \mathbf{w}(\underline{t})\}$. Then, by Definition 4.16, for each $\underline{t} \in [t_0, t_f]$, we must have

$$\mathcal{X}^c(\underline{t}) \subset \mathcal{X}^d(\underline{t}) \cap H(\underline{t}),$$

because $\mathcal{X}^c(\underline{t})$ is the exact bounding set. Note that the intersection of the nonempty and closed bounding set $\mathcal{X}^d(\underline{t})$ and the nonempty and compact set $H(\underline{t})$ is itself a nonempty (because $\mathcal{X}^c(\underline{t})$ is nonempty) and compact set. Clearly,

$$\min_{\mathbf{z} \in \mathcal{X}^d(\underline{t}) \cap H(\underline{t}), \mathbf{q} \in U(\underline{t})} f_i(\mathbf{z}, \mathbf{q}, \underline{t}) \leq \min_{\mathbf{z} \in \mathcal{X}^c(\underline{t}), \mathbf{q} \in U(\underline{t})} f_i(\mathbf{z}, \mathbf{q}, \underline{t})$$

for any fixed $\underline{t} \in [t_0, t_f]$, $i = 1, \ldots, n_x$. Thus, we can replace conditions (iii) and (iv) above with the following (stronger) conditions,

(v) $\dot{v}_i = \underline{h}_i(\mathbf{v}, \mathbf{w}, t) \leq \min_{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t)} f_i(\mathbf{z}, \mathbf{q}, t)$

(vi) $\dot{w}_i = \overline{h}_i(\mathbf{v}, \mathbf{w}, t) \geq \max_{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t)} f_i(\mathbf{z}, \mathbf{q}, t)$

which will always satisfy the original (weaker) conditions, i.e., any differential system $\mathbf{v}$ and $\mathbf{w}$ that satisfies conditions (i), (ii), (v), (vi) will also satisfy conditions (i), (ii), (iii) and (iv). □

In Definition 4.18, we have defined a known, closed bounding set $\mathcal{X}^d(\underline{t})$ for all $\underline{t} \in [t_0, t_f]$. As explained in [121], differential equations of practical interest to scientists and engineers are derived from physical systems for which more information is known about the behavior of the system dynamics than the information embodied by the differential equations alone. For example, for a system undergoing decay, the state never exceeds its initial condition. In another example, differential equations modeling mass and heat transfer obey conservation principles. The purpose of this bounding

set is thus to accommodate the incorporation of such additional information. Clearly, in order for this information to be useful practically, the set $\mathcal{X}^d(\underline{t})$ must be relatively cheap to obtain for all $\underline{t} \in [t_0, t_f]$.

Now that we have presented the "naive" version of a bounding technique based on differential inequalities, we present a stronger result that is based on the theory developed originally in the field of differential inequalities [135]. To prove the result, we need the following lemma:

Lemma 4.20. *Suppose the vector functions $\boldsymbol{\varphi}(t)$ and $\boldsymbol{\psi}(t)$ are differentiable a.e. in $(t_0, t_f]$. For some index i and fixed $\underline{t} \in (t_0, t_f]$, if $\dot{\varphi}_i(\underline{t}) < \dot{\psi}_i(\underline{t})$ when $\boldsymbol{\varphi}(\underline{t}) \leq \boldsymbol{\psi}(\underline{t})$, $\varphi_i(\underline{t}) = \psi_i(\underline{t})$ then we have precisely one of the following two cases:*

(i) $\boldsymbol{\varphi} < \boldsymbol{\psi}$ *in* $(t_0, t_f]$

(ii) $\boldsymbol{\varphi}(t_{0^+}) < \boldsymbol{\psi}(t_{0^+})$ *does not hold, i.e., there exists an arbitrary small* $\bar{t} \in (t_0, t_f]$ *such that* $\varphi_i(\bar{t}) \geq \psi(\bar{t})$ *for at least one index* i.

Proof. See proof of [135, Lemma 12.I]. □

We are now in position to present a stronger form of Corollary 4.19, which is essentially a weaker form of [121, Corollary 2.6] extended to handle the control functions $\mathbf{u}$ instead of real valued parameters $\mathbf{p}$:

Theorem 4.21. *Consider the ODE system in* (4.11). *If the following conditions are satisfied for* $i = 1, \ldots, n_x$

(i) $v_i(t_0) < \min\limits_{\mathbf{q} \in U(t_0)} x_i(\mathbf{q}, t_0)$

(ii) $w_i(t_0) > \max\limits_{\mathbf{q} \in U(t_0)} x_i(\mathbf{q}, t_0)$

and if $\forall \mathbf{v}(t), \mathbf{w}(t) \in H(t)$

(iii) $\dot{v}_i = \underline{h}_i(\mathbf{v}, \mathbf{w}, t) < \inf\limits_{\substack{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t) \\ z_i = v_i(t)}} f_i(\mathbf{z}, \mathbf{q}, t)$

(iv) $\dot{w}_i = \overline{h}_i(\mathbf{v}, \mathbf{w}, t) > \sup\limits_{\substack{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t) \\ z_i = w_i(t)}} f_i(\mathbf{z}, \mathbf{q}, t)$

where $H(t) = \{\mathbf{z} \mid \mathbf{v}(t) \leq \mathbf{z} \leq \mathbf{w}(t)\}$*, then*

$$\mathbf{v}(t) < \mathbf{x}(\mathbf{u}(t), t) < \mathbf{w}(t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

It is also assumed that the solutions, in the sense of Carathéodory, to the differential systems in $\mathbf{v}$ *and* $\mathbf{w}$ *exist and are unique.*

Proof. The same analysis presented in the proof of Corollary 4.19 regarding the existence of the extrema in conditions (i) and (ii) applies here. Note that the minimum and maximum are not guaranteed to exist in conditions (iii) and (iv) due to the presence of the additional constraint. We will now consider the lower bounding system, $\mathbf{v}(t)$. Condition (iii) ensures that if there exists some $\underline{t} \in (t_0, t_f]$ such that $\mathbf{v}(\underline{t}) \leq \mathbf{x}(\underline{t}) \leq \mathbf{w}(\underline{t})$, $v_i(\underline{t}) = x_i(\underline{t})$ for some index i, the following inequality must hold, for any $\mathbf{u}(\underline{t}) \in U(\underline{t})$:

$$\dot{v}_i(\underline{t}) = \underline{h}_i(\mathbf{v}, \mathbf{w}, \underline{t}) < \min_{\substack{\mathbf{z} \in \mathcal{X}^d(\underline{t}) \cap H(\underline{t}), \mathbf{q} \in U(\underline{t}) \\ z_i = v_i(\underline{t})}} f_i(\mathbf{z}, \mathbf{q}, \underline{t}) \leq f_i(\mathbf{x}(\mathbf{u}(\underline{t}), \underline{t}), \mathbf{u}(\underline{t}), \underline{t}) = \dot{x}_i(\underline{t}).$$

Thus, treating $\mathbf{v}(t)$ as $\boldsymbol{\varphi}(t)$ and $\mathbf{x}(t)$ as $\boldsymbol{\psi}(t)$, the conditions for Lemma 4.20 are satisfied, and so we must have precisely one of the following two cases:

(a) $\mathbf{v} < \mathbf{x}$ in $(t_0, t_f]$

(b) $\mathbf{v}(t_{0^+}) < \mathbf{x}(t_{0^+})$ does not hold.

Clearly, condition (i) excludes case (b), and so case (a) must hold, i.e.,

$$\mathbf{v}(t) < \mathbf{x}(\mathbf{u}(t), t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

An analogous proof holds for the upper bounding system, $\mathbf{w}(t)$. □

Note that by asserting uniqueness of the solution of the differential equations $\forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f]$, the conditions of the above theorem may be relaxed to

(i) $v_i(t_0) \leq \min_{\mathbf{q} \in U(t_0)} x_i(\mathbf{q}, t_0)$,

(ii) $w_i(t_0) \geq \max_{\mathbf{q} \in U(t_0)} x_i(\mathbf{q}, t_0)$,

(iii) $\dot{v}_i = \underline{h}_i(\mathbf{v}, \mathbf{w}, t) \leq \inf_{\substack{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t) \\ z_i = v_i(t)}} f_i(\mathbf{z}, \mathbf{q}, t)$,

(iv) $\dot{w}_i = \overline{h}_i(\mathbf{v}, \mathbf{w}, t) \geq \sup_{\substack{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t) \\ z_i = w_i(t)}} f_i(\mathbf{z}, \mathbf{q}, t)$,

i.e., replacing the strict inequalities with regular inequalities (see [135, Remark 12.X]). Furthermore, by asserting regular inequalities above, the result of the theorem also permits

$$\mathbf{v}(t) \leq \mathbf{x}(\mathbf{u}(t), t) \leq \mathbf{w}(t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

For the remainder of this chapter, we will assume that uniqueness of the constructed bounding differential equations holds $\forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f]$, and so it is understood that reference to Theorem 4.21 refers also to the regular inequalities just described.

As explained in [121], an interesting aspect of formulating the bounding differential equations as constrained optimization problems is that these optimization problems may have no feasible point (and thus the use of inf and sup rather than min and max in conditions (iii) and (iv)). Assume that such an infeasibility occurs in the bounding problem for the ith variable. Such infeasibilities usually arise from the equality constraint on the ith variable. This situation immediately implies that the ith bound lies outside the bounding set $\mathcal{X}^d(t)$. In this case, any finite value for the right-hand side of the differential equation is valid, for any finite instantaneous rate of change in the ith bounding variable ensures that the bound remains outside $\mathcal{X}^d(t)$. In practice, when employing interval techniques to estimate the solution of these optimization problems (as will be described later), the interval computation provides a finite value for the right-hand side of the differential equation, regardless of the feasibility of the optimization problem.

Note that Theorem 4.21 is identical to Corollary 4.19, except for conditions (iii) and (iv), where an additional constraint, $z_i = v_i(t)$ and $z_i = w_i(t)$ is added respectively to the optimization problems. Because of these additional constraints, the same proof for Corollary 4.19 can no longer be applied for Theorem 4.21. However, one can see

that because the optimization problems in Theorem 4.21 are more constrained than those in Corollary 4.19, the bounds obtained in Theorem 4.21 will always be tighter (no worse) than those obtained from Corollary 4.19.

Now, consider the lower bounding system, and any $i \in \{1, \ldots, n_x\}$. The following inequality is always true:

$$\min_{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t)} f_i(\mathbf{z}, \mathbf{q}, t) \leq \inf_{\substack{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t) \\ z_i = v_i(t)}} f_i(\mathbf{z}, \mathbf{q}, t).$$

However, the following inequality is not true for all $\mathbf{u}(t) \in U(t), t \in [t_0, t_f]$:

$$\inf_{\substack{\mathbf{z} \in \mathcal{X}^d(t) \cap H(t), \mathbf{q} \in U(t) \\ z_i = v_i(t)}} f_i(\mathbf{z}, \mathbf{q}, t) \leq f_i(\mathbf{x}(\mathbf{u}(t), t), \mathbf{u}(t), t) = \dot{x}_i(t).$$

This is the reason why the simple proof for Corollary 4.19 will not work, as it is not always guaranteed that $\dot{v}_i(t) \leq \dot{x}_i(t)$ for all $t \in [t_0, t_f]$. Thus, the argument that $\dot{v}_i(t)$ is always less than $\dot{x}_i(t)$ cannot be used. The application of Lemma 4.20 ensures that whenever $v_i(t) = x_i(t)$, then $\dot{v}_i(t)$ is always less than $\dot{x}_i(t)$. Hence, as t increases, v_i can approach x_i, but can never cross it and be greater than x_i because of this condition. In other words, by imposing the constraint $z_i = v_i(t)$, Theorem 4.21 ensures that $v_i(t)$ can never be greater than $x_i(t)$ for any $t \in [t_0, t_f]$, even though $\dot{v}_i(t)$ can possibly be greater than $\dot{x}_i(t)$ when $v_i(t) < x_i(t)$.

Before we extend Theorem 4.21 to multi-stage systems, we will present some examples of its application. Consider the following nonlinear ODE system:

$$\dot{x}_1 = u(x_2 - x_1)$$
$$\dot{x}_2 = ux_1$$

where $t \in [0, 1]$ and $\mathbf{x}(0) = (1, -1)$. We will also assume that we have no prior information about bounds on the state variables, and so $\mathcal{X}^d(t) = \mathbb{R}^2$, $\forall t \in [0, 1]$. All of the examples presented for the above system have been coded using the JACOBIAN Dynamic Modeling and Optimization Software [87] release 2.1A with the default

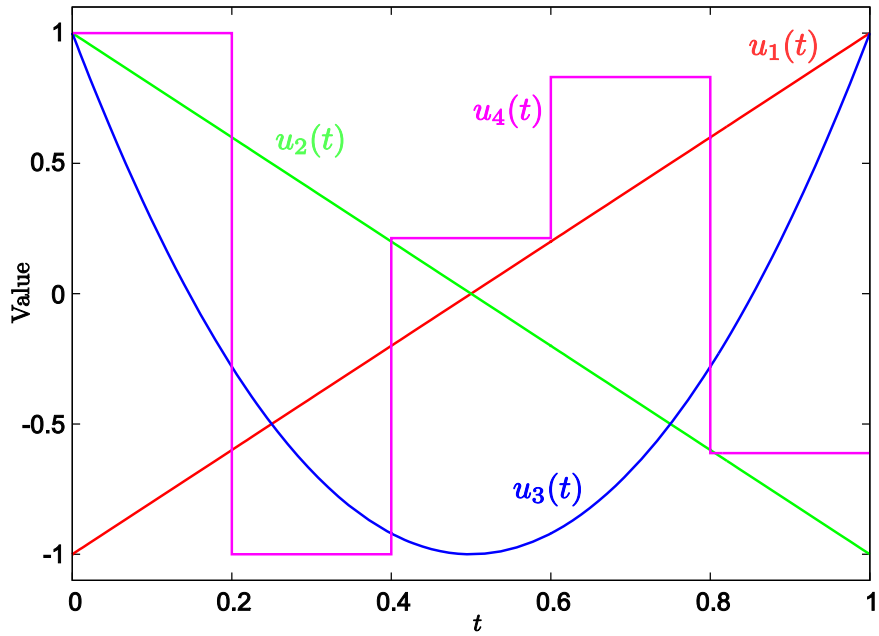


Figure 4-7: Different time varying functions, $u_i(t)$.

options.

First, we consider the case where the bounding set for $U(t)$ is time invariant, i.e., we have $U(t) = [-1, 1],\ \forall t \in [0, 1]$. Figure 4-7 shows 4 different time varying function profiles $u_1(t), u_2(t), u_3(t), u_4(t) \in U(t),\ \forall t \in [0, 1]$:

$$u_1(t) = 2t - 1,$$

$$u_2(t) = 1 - 2t,$$

$$u_3(t) = \frac{2(t-0.5)^2}{0.5^2} - 1,$$

$$u_4(t) = \begin{cases} 1 & \text{if } t \in [0, 0.2) \\ -1 & \text{if } t \in [0.2, 0.4) \\ 0.213 & \text{if } t \in [0.4, 0.6) \\ 0.831 & \text{if } t \in [0.6, 0.8) \\ -0.612 & \text{if } t \in [0.8, 1.0] \end{cases}.$$

Applying Theorem 4.21, and using natural interval extensions to under(over)estimate

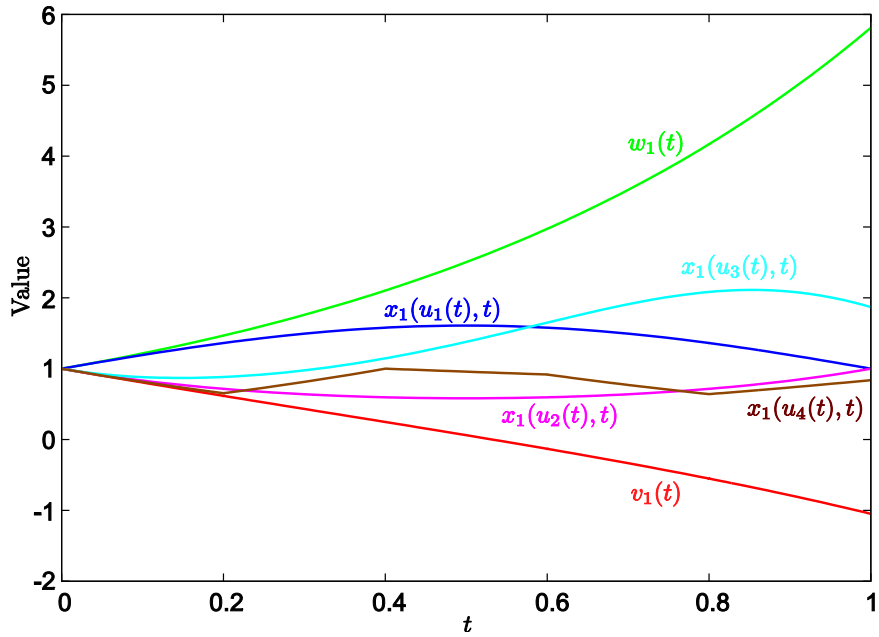


Figure 4-8: Bounds obtained using Theorem 4.21 and state trajectories for $x_1(t)$ for time invariant $U(t)$.

the infimum(supremum), we obtain the following bounding systems,

$$\begin{aligned}
\dot{v}_1 &= \min(-(w_2 - v_1), (w_2 - v_1), -(v_2 - v_1), (v_2 - v_1)),\\
\dot{v}_2 &= \min(-v_1, v_1, -w_1, w_1),\\
\dot{w}_1 &= \max(-(w_2 - w_1), (w_2 - w_1), -(v_2 - w_1), (v_2 - w_1)),\\
\dot{w}_2 &= \max(-v_1, v_1, -w_1, w_1).
\end{aligned}$$

where $\mathbf{v}(0) = \mathbf{w}(0) = (1, -1)$.

Figure 4-8 shows the bounds obtained for $x_1(t)$, as well as the four different state trajectories for $x_1(t)$ using the time varying profiles for $u_1(t), u_2(t), u_3(t)$ and $u_4(t)$ shown in Figure 4-7. It can be seen that the bounds obtained from Theorem 4.21 indeed bound these state trajectories.

Next, we demonstrate the case where the bounding set for the control function

$U(t)$ varies with time. Consider $U(t) = [u^L(t), u^U(t)],\ \forall t \in [0,1]$, where

$$u^L(t) = \frac{2(t-0.5)^2}{0.5^2} + 0.5,$$
$$u^U(t) = -\frac{2(t-0.5)^2}{0.5^2} - 0.5.$$

Figure 4-9 shows 4 different time varying profiles $u_1(t), u_2(t), u_3(t), u_4(t) \in U(t),\ \forall t \in [0,1]$, where the two grey parabolic curves represent $u^L(t)$ and $u^U(t)$:

$$u_1(t) = 4t - 2,$$
$$u_2(t) = 2 - 4t,$$
$$u_3(t) = 0.8\sin(9.5t) + 0.35,$$
$$u_4(t) = \begin{cases} u^L(t) & \text{if } t \in [0, 0.2) \\ u^U(t) & \text{if } t \in [0.2, 0.4) \\ u^L(t) & \text{if } t \in [0.4, 0.6) \\ u^U(t) & \text{if } t \in [0.6, 0.8) \\ u^L(t) & \text{if } t \in [0.8, 1.0] \end{cases}.$$

Applying Theorem 4.21, and using natural interval extensions to under(over)estimate the infimum(supremum), we obtain the following bounding systems,

$$\begin{aligned}
\dot{v}_1 &= \min(u^L(w_2 - v_1), u^U(w_2 - v_1), u^L(v_2 - v_1), u^U(v_2 - v_1)), \\
\dot{v}_2 &= \min(u^L v_1, u^U v_1, u^L w_1, u^U w_1), \\
\dot{w}_1 &= \max(u^L(w_2 - w_1), u^U(w_2 - w_1), u^L(v_2 - w_1), u^U(v_2 - w_1)), \\
\dot{w}_2 &= \max(u^L v_1, u^U v_1, u^L w_1, u^U w_1).
\end{aligned}$$

where $\mathbf{v}(0) = \mathbf{w}(0) = (1, -1)$.

Figure 4-10 shows the bounds obtained for $x_1(t)$, as well as the four different state trajectories for $x_1(t)$ using the time varying function profiles for $u_1(t), u_2(t), u_3(t)$ and $u_4(t)$ shown in Figure 4-9. Again, it is clear that the bounds obtained from Theorem

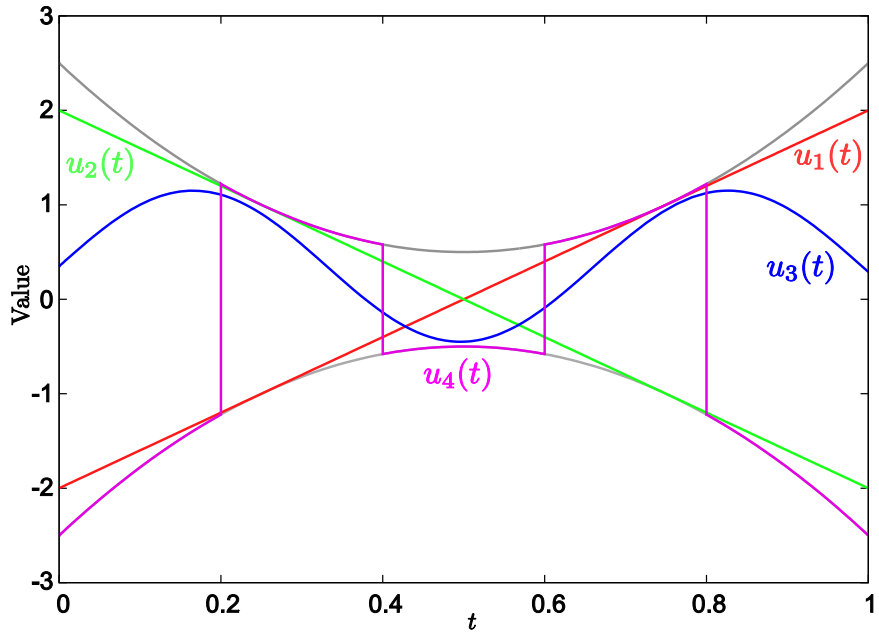


Figure 4-9: Different control functions, $u_i(t)$

4.21 indeed bound these state trajectories.

Note that Theorem 4.21 clearly encompasses systems which have time invariant parameters, i.e., when $\mathbf{u}(t) = \mathbf{p}$, $\forall t \in [t_0, t_f]$ in (4.11). In fact, because the bounds obtained from Theorem 4.21 must take into consideration all possible bounded parameter functions $\mathbf{u}$, the bounds obtained are, in general, weak when systems with time invariant parameters $\mathbf{p}$ are considered. This is illustrated with a simple example in Section 4.3.5 below.

We will now present an extension of Theorem 4.21 to the transformed nonlinear hybrid system in Definition 4.10.

Definition 4.22. Let $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s)$ be the solution of the embedded nonlinear hybrid system in Definition 4.10, and let $\hat{x}_i(\mathbf{p}, \boldsymbol{\delta}, s) \in \hat{\mathcal{X}}_i^{(j)}(\mathbf{p}, \boldsymbol{\delta}, s)$ for each $(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta$, $i = 1, \ldots, n_x$, $j = 1, \ldots, n_e$ where $\hat{\mathcal{X}}_i^{(j)}(\mathbf{p}, \boldsymbol{\delta}, s) \subset \mathbb{R}$ is a bounding set that is known independently. For each fixed $\underline{s} \in \hat{I}_j$, $j = 1, \ldots, n_e$, let $\alpha_i^{(j)}(\mathbf{q}, \mathbf{r}, \underline{s}) = \inf \hat{\mathcal{X}}_i^{(j)}(\mathbf{q}, \mathbf{r}, \underline{s})$ and $\beta_i^{(j)}(\mathbf{q}, \mathbf{r}, \underline{s}) = \sup \hat{\mathcal{X}}_i^{(j)}(\mathbf{q}, \mathbf{r}, \underline{s})$ for each $(\mathbf{q}, \mathbf{r}) \in P \times \Delta$, $i = 1, \ldots, n_x$. Furthermore, let $\hat{\mathcal{X}}^{(j)}(\underline{s})$ be defined pointwise in (transformed) time for each $j = 1, \ldots, n_e$ by

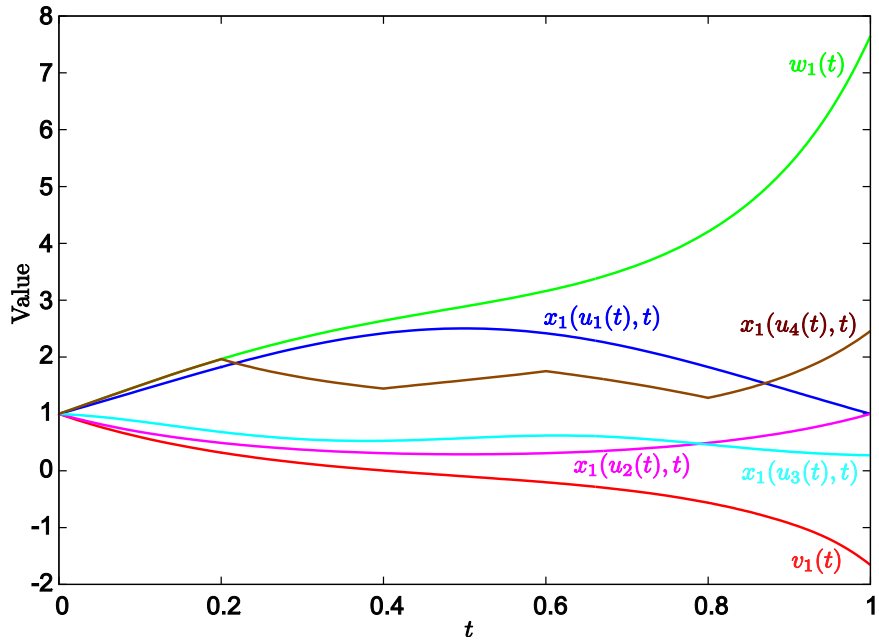


Figure 4-10: Bounds obtained using Theorem 4.21 and state trajectories for $x_1(t)$ for time varying $U(t)$.

$\hat{\mathfrak{X}}^{(j)}(\underline{s}) = [\mathbf{z}^L, \mathbf{z}^U]$ such that

$$z_i^L = \inf_{\mathbf{q}\in P, \mathbf{r}\in\Delta} \alpha_i^{(j)}(\mathbf{q}, \mathbf{r}, \underline{s}), \;\; z_i^U = \sup_{\mathbf{q}\in P, \mathbf{r}\in\Delta} \beta_i^{(j)}(\mathbf{q}, \mathbf{r}, \underline{s}), \;\; \forall i = 1, \ldots, n_x$$

where z_i^L and z_i^U are in the extended real number system.

Corollary 4.23. *Consider the embedded nonlinear hybrid system in Definition 4.10. If the following conditions are satisfied for all $i = 1, \ldots, n_x$ and $j = 1, \ldots, n_e$,*

(i) $v_i(\hat{\sigma}_j) < \min_{\mathbf{q}\in P, \mathbf{r}\in\Delta} \hat{x}_i(\mathbf{q}, \mathbf{r}, \hat{\sigma}_j)$

(ii) $w_i(\hat{\sigma}_j) > \max_{\mathbf{q}\in P, \mathbf{r}\in\Delta} \hat{x}_i(\mathbf{q}, \mathbf{r}, \hat{\sigma}_j)$

and additionally for all $\mathbf{v}(s), \mathbf{w}(s) \in H(s)$, $s \in [i-1, i]$,

(iii) $v_i' = \underline{h}_i^{(m_j^*)}(\mathbf{v}, \mathbf{w}, s; P, \Delta) < \inf_{\substack{\mathbf{z}\in\hat{\mathfrak{X}}^{(j)}(s)\cap H(s), \mathbf{q}\in P, \mathbf{r}\in\Delta \\ z_i = v_i(s)}} \mathcal{F}_i^{(m_j^*)}(\mathbf{z}, \mathbf{q}, \mathbf{r}, s)$

(iv) $w_i' = \overline{h}_i^{(m_j^*)}(\mathbf{v}, \mathbf{w}, s; P, \Delta) > \sup_{\substack{\mathbf{z}\in\hat{\mathfrak{X}}^{(j)}(s)\cap H(s), \mathbf{q}\in P, \mathbf{r}\in\Delta \\ z_i = w_i(s)}} \mathcal{F}_i^{(m_j^*)}(\mathbf{z}, \mathbf{q}, \mathbf{r}, s)$

where $H(s) \equiv \{\mathbf{z} \mid \mathbf{v}(s) \leq \mathbf{z} \leq \mathbf{w}(s)\}$*, then*

$$\mathbf{v}(s) < \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) < \mathbf{w}(s), \ \forall(\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times \hat{I}_i, \ i = 1, \ldots, n_e.$$

It is also assumed that the solutions, in the sense of Carathéodory, to the differential systems in $\mathbf{v}$ *and* $\mathbf{w}$ *exist and are unique, for all* $j = 1, \ldots, n_e$*.*

Proof. From the initial conditions of the hybrid system, Theorem 4.15 (treating $\hat{\mathbf{x}}(\cdot, \hat{\tau}_i)$ as the objective function) and the form of (4.10), $\hat{\mathbf{x}}(\cdot, \hat{\sigma}_i)$ is continuous on $P \times \Delta$ for all $i = 1, \ldots, n_e$. Hence, the extrema in conditions (i) and (ii) exist.

Consider now the first epoch $\hat{I}_1$. From Definition 4.10, $\boldsymbol{\mathcal{F}}^{(m_1^*)}$ is piecewise continuous with a finite number of stationary simple discontinuities in s. Let γ be the number of discontinuities occurring at $\check{\tau}_i \in \hat{I}_1$, $i = 1, \ldots, \gamma$. Then, the first epoch can be further subdivided into $\gamma+1$ contiguous subepochs, for which we have explicit time events at the subepoch boundaries, state continuity at each event, and $\boldsymbol{\mathcal{F}}^{(m_1^*)}$ is continuous for each subepoch. Let the sequence of subepochs be given by $\{\check{I}_j\}, j = 1, \ldots, \gamma + 1$ where $\check{I}_j = [\check{\sigma}_j, \check{\tau}_j]$, $\check{\sigma}_1 = 0$, $\check{\tau}_{\gamma+1} = \hat{\tau}_1$, and $\check{\sigma}_{j+1} = \check{\tau}_j$ for $j = 1, \ldots, \gamma$. Consider now the first subepoch $\check{I}_1$. The initial condition at time $s = 0$ given by Definition 4.10 is clearly continuous on $P \times \Delta$. The form of the nonlinear ODE system in the first subepoch, and conditions (i)–(iv) clearly satisfy the conditions of Theorem 4.21, which gives

$$\mathbf{v}(s) < \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) < \mathbf{w}(s), \tag{4.12}$$

for all $(\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times \check{I}_1$. At the transition $\check{\tau}_1$, state continuity ensures

$$\mathbf{v}(\check{\sigma}_2) = \mathbf{v}(\check{\tau}_1) < \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \check{\sigma}_2) < \mathbf{w}(\check{\tau}_1) = \mathbf{w}(\check{\sigma}_2), \ \forall(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta. \tag{4.13}$$

From Theorem 4.15, $\hat{\mathbf{x}}(\cdot, \check{\sigma}_2)$ is continuous on $P \times \Delta$ (simply treat $\hat{\mathbf{x}}(\cdot, \check{\sigma}_2)$ as the objective function). The form of the nonlinear ODE system in the second subepoch, (4.13) and conditions (iii) and (iv) thus satisfy the conditions of Theorem 4.21, which implies that (4.12) holds for all $(\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times \check{I}_2$. By induction on all subepochs, (4.12) holds for all $(\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times \hat{I}_1$. Consider now the second epoch $\hat{I}_2$. From

Theorem 4.15 and (4.10), $\hat{\mathbf{x}}(\cdot, \hat{\sigma}_2)$ is continuous on $P \times \Delta$. The analysis carried out for the first epoch is thus valid for the second. By induction on all epochs, we have the desired result. $\square$

As with Theorem 4.21, by asserting the uniqueness of the solution of the embedded nonlinear hybrid systems (multi-stage systems), the conditions of Corollary 4.23 may be relaxed by replacing the strict inequalities with regular inequalities, and the result of the corollary also permits

$$\mathbf{v}(s) \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) \leq \mathbf{w}(s), \ \forall (\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times \hat{I}_i, \ i = 1, \ldots, n_e.$$

Again, we will assume that uniqueness of the constructed bounding differential equations holds, and so it is understood that reference to Corollary 4.23 refers also to the regular inequalities just described.

Remark. The bounding set $\hat{\mathcal{X}}_i^{(j)}(\mathbf{p}, \boldsymbol{\delta}, s)$ for variable i and epoch j makes it possible to tighten the implied state bounds obtained when physical insight from the problem in the form of invariants (e.g., conservation laws) and bounds is available.

Corollary 4.23 enables a bounding hybrid system of differential equations to be constructed to obtain the following set for all $i = 1, \ldots, n_e$,

$$\hat{X}^{(i)}(\underline{s}; P, \Delta) \equiv \{\mathbf{z} \mid \mathbf{v}(\underline{s}) \leq \mathbf{z} \leq \mathbf{w}(\underline{s})\}. \tag{4.14}$$

The most difficult aspect of applying the theorem lies in obtaining the extrema in conditions (i) – (iv). As stated in [121], while computing the exact solution to the optimization problems would yield the tightest bounds possible from the theorem, actually solving the optimization problems at each integration step in a numerical integration would typically be a prohibitively expensive task. Hence, in practice, the solutions to the optimization problems are estimated by interval arithmetic [99] pointwise in time. Before we proceed, we will briefly introduce the metric topology for the set of intervals (see [99] for more details). By an *interval* we mean a compact set of real numbers $[x^L, x^U] = \{x \mid x^L \leq x \leq x^U\}$. As in [99], we will not distinguish

between the degenerate interval $[a, a]$ and the real number a. Define the distance $d(X, Y) = \max(|x^L - y^L|, |x^U - y^U|)$ for the intervals $X \equiv [x^L, x^U]$, $Y \equiv [y^L, y^U]$. The absolute value of an interval $X \equiv [x^L, x^U]$ is given by $|X| = \max(|x^L|, |x^U|)$. The vector norm $\|Z\| = \max(|Z_1|, \ldots, |Z_n|)$ is used for interval vectors. An interval valued function $F : Z \to \mathbb{IR}, Z \subset \mathbb{IR}^n$, is said to be *continuous* in the usual $\varepsilon - \delta$ fashion with the metric $d(X, Y)$, where $\mathbb{IR}$ is the set of all intervals. We say that an interval valued function F of the interval variables $X_1, \ldots, X_n$ is *inclusion monotonic* if $Y_i \subset X_i, i = 1, \ldots, n$ implies $F(Y_1, \ldots, Y_n) \subset F(X_1, \ldots, X_n)$. Let f be a real valued function of n real variables $x_1, \ldots, x_n$. By an *interval extension* of f, we mean an interval valued function F of n interval variables $X_1, \ldots, X_n$ with the property $F(x_1, \ldots, x_n) = f(x_1, \ldots, x_n)$ for real arguments, i.e., an interval extension of f is an interval valued function which has real values when the arguments are all real (degenerate intervals) and coincides with f.

Consider now the vector $\mathbf{z} \in \mathbb{R}^n$. We will introduce the following notation: for any fixed $j \in \{1, \ldots, n\}$, let $\mathbf{z}_{k \neq j}$ denote the vector $\tilde{\mathbf{z}} \in \mathbb{R}^{n-1}$ where

$$\tilde{z}_k = \begin{cases} z_k & \text{if } k < j, \\ z_{k+1} & \text{if } k \geq j. \end{cases}$$

For convenience, we will also introduce the following (element wise) maximization and minimization operations: consider the n-dimensional vectors $\mathbf{x}$ and $\mathbf{y}$ whose elements are in the extended real number system. Let the vector valued operation $cmin(\mathbf{x}, \mathbf{y})$ return the n-dimensional vector $\mathbf{z}$ whose elements are in the extended real number system where

$$z_i = \min(x_i, y_i), \ \forall i = 1, \ldots, n.$$

Similarly, let $cmax(\mathbf{x}, \mathbf{y})$ return the n-dimensional vector $\mathbf{z}$ in the extended real number system where

$$z_i = \max(x_i, y_i), \ \forall i = 1, \ldots, n.$$

Corollary 4.24. *Let $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s)$ be the solution of the embedded nonlinear hybrid sys-*

tem in Definition 4.10. Define the following interval valued functions,

$$Y(\hat{\sigma}_1) = [\mathbf{y}^L(\hat{\sigma}_1), \mathbf{y}^U(\hat{\sigma}_1)] = \mathbf{E}_0 P + \mathbf{J}_0 \Delta + \mathbf{k}_0, \tag{4.15}$$

$$\begin{aligned} Y(\hat{\sigma}_{l+1}) = [\mathbf{y}^L(\hat{\sigma}_{l+1}), \mathbf{y}^U(\hat{\sigma}_{l+1})] = \mathbf{D}_l[\mathbf{v}(\hat{\tau}_l), \mathbf{w}(\hat{\tau}_l)] \\ + \mathbf{E}_l P + \mathbf{J}_l \Delta + \mathbf{k}_l, \ \forall l = 1, \ldots, n_e - 1, \end{aligned} \tag{4.16}$$

and let $\Gamma_i^{(m_j^*)}(v_i, Z(i,j,s), P, \Delta, s) = [\boldsymbol{\gamma}_i^{(m_j^*)L}, \boldsymbol{\gamma}_i^{(m_j^*)U}]$ *and* $\Lambda_i^{(m_j^*)}(w_i, Z(i,j,s), P, \Delta, s) = [\boldsymbol{\lambda}_i^{(m_j^*)L}, \boldsymbol{\lambda}_i^{(m_j^*)U}]$ *be inclusion monotonic interval extensions of* $\mathcal{F}_i^{(m_j^*)}(\hat{x}_i, \hat{\mathbf{x}}_{k\neq i}, \mathbf{p}, \boldsymbol{\delta}, s)$ *for all* $i = 1, \ldots, n_x$, $j = 1, \ldots, n_e$, *where*

$$Z(i,j,s) = \{\mathbf{z}_{k\neq i} \mid cmax(\mathbf{v}_{k\neq i}(s), \boldsymbol{\alpha}_{k\neq i}^{(j)}(s)) \leq \mathbf{z}_{k\neq i} \leq cmin(\mathbf{w}_{k\neq i}(s), \boldsymbol{\beta}_{k\neq i}^{(j)}(s))\}$$

and $\hat{X}^{(j)}(s; P, \Delta) = [\boldsymbol{\alpha}^{(j)}(s), \boldsymbol{\beta}^{(j)}(s)]$ *is defined in* (4.14), *and obtained from Corollary 4.23. Then, for all* $i = 1, \ldots, n_x$, $s \in [j-1, j]$ *and* $j = 1, \ldots, n_e$, *the following hybrid system*

$$v_i' = \gamma_i^{(m_j^*)L}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s), \qquad v_i(\hat{\sigma}_j) = y_i^L(\hat{\sigma}_j), \tag{4.17}$$

$$w_i' = \lambda_i^{(m_j^*)U}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s), \qquad w_i(\hat{\sigma}_j) = y_i^U(\hat{\sigma}_j), \tag{4.18}$$

bounds the transformed hybrid system,

$$\mathbf{v}(s) \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) \leq \mathbf{w}(s), \ \forall (\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times \hat{I}_j, \ j = 1, \ldots, n_e.$$

Proof. The rational interval functions (4.15) and (4.16) are inclusion monotonic [99, Page 21]. Together with the inclusion monotonicity of the interval extensions of $\mathcal{F}_j^{(m_i^*)}$, (4.17) and (4.18) thus satisfy conditions (i) – (iv) of Corollary 4.23. □

It is important to note that the implied state bounds are obtained given particular parameter sets P and Δ; in order to ensure convergence of the BB framework, these bounds must converge as P and Δ become degenerate in the limit.

Lemma 4.25. *Let $H(X_1, \ldots, X_n) = [h^L, h^U]$ be an interval valued function, where $X_i = [\mathbf{x}_i^L, \mathbf{x}_i^U]$ are n_{x_i}-dimensional interval vectors for all $i = 1, \ldots, n$. Consider the following real valued functions,*

$$g_1(\mathbf{x}_1^L, \ldots, \mathbf{x}_n^L, \mathbf{x}_1^U, \ldots, \mathbf{x}_n^U) = h^L, \qquad g_2(\mathbf{x}_1^L, \ldots, \mathbf{x}_n^L, \mathbf{x}_1^U, \ldots, \mathbf{x}_n^U) = h^U.$$

If H is continuous on $Y = Y_1 \times \cdots \times Y_n$, where $Y_i = [\mathbf{y}_i^L, \mathbf{y}_i^U] \subset \mathbb{R}^{n_{x_i}}$ for all $i = 1, \ldots, n$, then g_1 and g_2 are continuous on $Y \times Y$, and bounded by a constant M there.

Proof. It suffices to prove the result for g_1 since the proof for g_2 is similar. Let the interval vector $Z = [\mathbf{z}^L, \mathbf{z}^U] = (X_1, \ldots, X_n) = [(\mathbf{x}_1^L, \ldots, \mathbf{x}_n^L), (\mathbf{x}_1^U, \ldots, \mathbf{x}_n^U)]$ and $n_z = \sum_{i=1}^{n} n_{x_i}$. Then, $H(\cdot) \equiv H(Z)$ and $g_1(\cdot) \equiv g_1(\mathbf{z}^L, \mathbf{z}^U)$. Let $H(A) = [h_a^L, h_a^U]$ and $H(B) = [h_b^L, h_b^U]$, where $A = (A_1, \ldots, A_{n_z})$ and $B = (B_1, \ldots, B_{n_z})$, $A_i = [a_i^L, a_i^U], B_i = [b_i^L, b_i^U]$ for all $i = 1, \ldots, n_z$. Consider now any arbitrary $\varepsilon > 0$ and $A \in Y$. Since H is continuous at A, there exists some $\delta > 0$ such that

$$\max(|h_a^L - h_b^L|, |h_a^U - h_b^U|) < \varepsilon \tag{4.19}$$

for any $B \in Y$ when $\max_{1 \le i \le n_z} d(A_i, B_i) < \delta$, or

$$\max\big(\max_{1 \le i \le n_z} |a_i^L - b_i^L|, \max_{1 \le i \le n_z} |a_i^U - b_i^U|\big) < \delta. \tag{4.20}$$

Since (4.19) implies that $|g_1(\mathbf{a}^L, \mathbf{a}^U) - g_1(\mathbf{b}^L, \mathbf{b}^U)| < \varepsilon$, and (4.20) implies that $\|(\mathbf{a}^L, \mathbf{a}^U) - (\mathbf{b}^L, \mathbf{b}^U)\|_\infty < \delta$, we have shown that g_1 is continuous at $(\mathbf{a}^L, \mathbf{a}^U)$. Since the choice of A was arbitrary, g_1 is continuous on $Y \times Y$. Since $Y \times Y$ is a compact set, and g_1 is continuous, the minimum and maximum of g_1 on $Y \times Y$ exists, and so g_1 is bounded. □

Recall the definition of the set $\hat{X}^{(i)}(P, \Delta)$ from Definition 4.11. Let $\hat{X}^{(i)}(P, \Delta) = [\hat{\mathbf{x}}^L, \hat{\mathbf{x}}^U]$. For convenience, for all $i = 1, \ldots, n_e$, $j = 1, \ldots, n_x$, let $\hat{X}_j^{(i)}(P, \Delta)$ denote the jth element of $\hat{X}^{(i)}(P, \Delta)$, i.e., $\hat{X}_j^{(i)}(P, \Delta) = [\hat{x}_j^L, \hat{x}_j^L]$, and let $\hat{X}_{n \neq j}^{(i)}(P, \Delta) =$

$[\hat{\mathbf{x}}^L_{n\neq j}, \hat{\mathbf{x}}^U_{n\neq j}]$.

Theorem 4.26. *Let $\{(P_k, \Delta_k)\}$ be a convergent sequence of interval vectors such that*

$$\lim_{k\to\infty}(P_k, \Delta_k) = (P^*, \Delta^*) \equiv [(\mathbf{p}^*, \boldsymbol{\delta}^*), (\mathbf{p}^*, \boldsymbol{\delta}^*)], \tag{4.21}$$

where $(P^*, \Delta^*) \in P \times \Delta$. Let Corollary 4.24 be used to construct (4.14)*. For all $i = 1, \ldots, n_e$, let the epoch $\hat{I}_i$ be split into a finite number (γ_i) of contiguous subepochs $\check{I}_l = [\check{\sigma}_l, \check{\tau}_l]$, where $\check{\sigma}_1 = i - 1$, $\check{\tau}_{\gamma_i} = i$, and $\check{\sigma}_{l+1} = \check{\tau}_l$ for all $l = 1, \ldots, \gamma_i - 1$. If the interval extensions $\Gamma_j^{(m_i^*)}$ and $\Lambda_j^{(m_i^*)}$ are continuous on $\hat{X}_j^{(i)}(P, \Delta) \times \hat{X}_{n\neq j}^{(i)}(P, \Delta) \times P \times \Delta \times [\check{\sigma}_l, \check{\tau}_l]$ for all $i = 1, \ldots, n_e$, $j = 1, \ldots, n_x$, and $l = 1, \ldots, \gamma_i$, then*

$$\lim_{k\to\infty} \hat{X}_k^{(m_i^*)}(\underline{s}; P_k, \Delta_k) = [\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \underline{s}), \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \underline{s})], \ \forall \underline{s} \in \hat{I}_i, i = 1, \ldots, n_e.$$

Proof. Consider the first subepoch of the first epoch, $\check{I}_1$. By definition, interval extensions have real values when their arguments are all real (degenerate interval vectors). Hence, with the degenerate interval vector (P^*, Δ^*) as argument, the natural interval extension (4.15) becomes $Y(\hat{\sigma}_1) = [\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_1), \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_1)]$. Thus, the initial conditions for the bounding hybrid system becomes

$$\mathbf{v}(\check{\sigma}_1) = \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_1) = \mathbf{w}(\check{\sigma}_1).$$

This implies that the interval vector $Z(j, 1, s)$ defined in Corollary 4.24 is degenerate at $s = \check{\sigma}_1$, which implies

$$\begin{aligned} v_j'(\check{\sigma}_1) &= \gamma_j^{(m_1^*)L}(\mathbf{v}, \mathbf{w}, \mathbf{p}^*, \mathbf{p}^*, \boldsymbol{\delta}^*, \boldsymbol{\delta}^*, \check{\sigma}_1) = \mathcal{F}_j^{(m_1^*)}(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \check{\sigma}_1), \mathbf{p}^*, \boldsymbol{\delta}^*, \check{\sigma}_1), \\ w_j'(\check{\sigma}_1) &= \lambda_j^{(m_1^*)U}(\mathbf{v}, \mathbf{w}, \mathbf{p}^*, \mathbf{p}^*, \boldsymbol{\delta}^*, \boldsymbol{\delta}^*, \check{\sigma}_1) = \mathcal{F}_j^{(m_1^*)}(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \check{\sigma}_1), \mathbf{p}^*, \boldsymbol{\delta}^*, \check{\sigma}_1), \end{aligned}$$

for all $j = 1, \ldots, n_x$. We have thus defined an initial value problem in $\mathbf{v}(s)$ and $\mathbf{w}(s)$. Since the solution trajectory $\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s)$ is unique (see Remark following Lemma 4.13), this implies that the interval vector $Z(j, 1, s)$ is degenerate for all $j = 1, \ldots, n_x$,

$s \in \check{I}_1$, and equal to the value $\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s)$. Hence, (4.17) and (4.18) become

$$\mathbf{v}'(s) = \boldsymbol{\mathcal{F}}^{(m_1^*)}(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s), \mathbf{p}^*, \boldsymbol{\delta}^*, s) = \mathbf{w}'(s), \quad \mathbf{v}(\check{\sigma}_1) = \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_1) = \mathbf{w}(\check{\sigma}_1). \tag{4.22}$$

For convenience, let $\mathbf{z}(s) = (\mathbf{v}(s), \mathbf{w}(s))$ and $\mathbf{y} = (\mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U)$. The system of ODEs in (4.17) and (4.18) can then be expressed as

$$\mathbf{z}' = \mathbf{f}(\mathbf{z}, \mathbf{y}, s),$$

where a solution $\mathbf{z}(\mathbf{y}^*, s)$ exists and is unique for $\mathbf{y}^* = (\mathbf{p}^*, \mathbf{p}^*, \boldsymbol{\delta}^*, \boldsymbol{\delta}^*)$ (because the solution $\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s)$ exists and is unique). Since the interval extensions $\Gamma_j^{(m_1^*)}$ and $\Lambda_j^{(m_1^*)}$ are continuous for all $j = 1, \ldots, n_x$, an application of Lemma 4.25 (treating x_j and s as degenerate intervals) gives $\mathbf{f}$ continuous on $\hat{X}^{(1)}(P, \Delta)^2 \times P^2 \times \Delta^2 \times \check{I}_1$ and bounded by a constant M there. With $\mathbf{z}(\mathbf{y}^*, s)$ as the unique trajectory, we can then apply [42, Chp. 2,Theorem 4.3] to obtain $\mathbf{z}(s) \to (\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s), \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s))$ uniformly over $\check{I}_1$ as $\mathbf{p}^L \to \mathbf{p}^*$, $\mathbf{p}^U \to \mathbf{p}^*$, $\boldsymbol{\delta}^L \to \boldsymbol{\delta}^*$, $\boldsymbol{\delta}^U \to \boldsymbol{\delta}^*$ (or $P_k \to P^*$, $\Delta_k \to \Delta^*$). Consider now the transition at $\check{\tau}_1$. Clearly, state continuity preserves the form of (4.22),

$$\mathbf{v}' = \boldsymbol{\mathcal{F}}^{(m_1^*)}(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s), \mathbf{p}^*, \boldsymbol{\delta}^*, s) = \mathbf{w}', \qquad \mathbf{v}(\check{\sigma}_2) = \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \check{\sigma}_2) = \mathbf{w}(\check{\sigma}_2).$$

We can then perform the same analysis to obtain uniform convergence of the bounds over the second sub-epoch. By induction on all sub-epochs, we obtain uniform convergence over the first epoch. Consider now the transition to the second epoch at $\hat{\tau}_1$. With the degenerate interval vector (P^*, Δ^*) as argument, it is clear from the preceding analysis that the natural interval extension (4.16) becomes $Y(\hat{\sigma}_2) = [\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_2), \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_2)]$. The same analysis made for $\check{I}_1$ in $\hat{I}_1$ thus applies, and (4.22) becomes

$$\mathbf{v}' = \boldsymbol{\mathcal{F}}^{(m_2^*)}(\hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, s), \mathbf{p}^*, \boldsymbol{\delta}^*, s) = \mathbf{w}', \qquad \mathbf{v}(\hat{\sigma}_2) = \hat{\mathbf{x}}(\mathbf{p}^*, \boldsymbol{\delta}^*, \hat{\sigma}_2) = \mathbf{w}(\hat{\sigma}_2).$$

The analysis carried out for the first epoch is thus valid for the second. By induction

on all epochs, we have the desired result. $\square$

Remark. Note that the requirement for $\Gamma_j^{(m_i^*)}$ and $\Lambda_j^{(m_i^*)}$ to be inclusion monotonic and continuous for all $i = 1, \ldots, n_e$ and $j = 1, \ldots, n_x$ is not a strong one. For linear time invariant hybrid systems, it is automatically satisfied since (4.6) becomes a rational function (in the sense of interval analysis [99]). For time varying hybrid systems, inclusion monotonic interval extensions of the time varying matrices in (4.6) can be constructed for most functions in computing provided no division by an interval containing zero occurs (see e.g., [99, Chapter 3 and 4] and [109, Chapter 1]). In addition, since the functions of interest are continuous in each subepoch, the constructed interval extensions will also be continuous (see e.g., [3, Theorem 4 and Corollary 5]).

Before we end this section, we will walk through a procedure for bounding the solution of the transformed hybrid system using Corollary 4.24.

Example 4.27. Consider the following linear hybrid system,

$$\text{Mode 1:} \quad \begin{cases} \dot{x}_1 = 0.5x_1 + x_2 + p_1, \\ \dot{x}_2 = -x_1 + x_2 + p_1, \end{cases}$$

$$\text{Mode 2:} \quad \begin{cases} \dot{x}_1 = x_1 + x_2 - p_2, \\ \dot{x}_2 = -x_1 + p_2, \end{cases}$$

$n_e = 2$, $T_\mu = 1, 2$, $T_\tau = \{I_i\}$ where $I_1 = [0, \delta_1]$, $I_2 = [\delta_1, \delta_1 + \delta_2]$, $P = [0, 1]^2$, $\Delta = [0, 1]^2$, and we have state continuity as the transition functions with initial condition $\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, 0) = (0, 2)$.

Applying the CPET, we obtain the following transformed nonlinear hybrid system,

$$\text{Mode 1:} \quad \begin{cases} \hat{x}_1' = \delta_1\big(0.5\hat{x}_1 + \hat{x}_2 + p_1\big), \\ \hat{x}_2' = \delta_1\big(-\hat{x}_1 + \hat{x}_2 + p_1\big), \end{cases}$$

$$\text{Mode 2:} \quad \begin{cases} \hat{x}_1' = \delta_2\big(\hat{x}_1 + \hat{x}_2 - p_2\big), \\ \hat{x}_2' = \delta_2\big(-\hat{x}_1 + p_2\big), \end{cases}$$

$n_e = 2$, $T_\mu = 1, 2$, $T_\tau = \{\hat{I}_i\}$ where $\hat{I}_1 = [0,1]$, $\hat{I}_2 = [1,2]$, $P = [0,1]^2$, $\Delta = [0,1]^2$, and we have state continuity as the transition functions with initial condition $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, 0) = (0, 2)$.

We now apply Corollary 4.24 to obtain bounds for the transformed hybrid system. For this example, we assume that we do not have additional bounding information, and so the user defined set $\hat{\mathcal{X}}_i^{(j)}(\mathbf{p}, \boldsymbol{\delta}, s)$ is set to $\mathbb{R}$ for all $i = 1, \ldots, n_x$, $j = 1, \ldots, n_e$, $(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta$. From (4.15), $Y(0) = [\mathbf{k}_0, \mathbf{k}_0]$ where $\mathbf{k}_0 = (0, 2)$ since $\mathbf{E}_0$ and $\mathbf{J}_0$ are zero matrices. Expanding the right hand sides of the nonlinear ODEs in mode 1, and taking the natural interval extensions, we obtain the following forms for $\boldsymbol{\Gamma}^{(1)}$ and $\boldsymbol{\Lambda}^{(1)}$ in Corollary 4.24,

$$\begin{aligned}
\Gamma_1^{(1)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_1^L, \delta_1^U] \cdot \big(0.5 v_1(s) + [v_2(s), w_2(s)] + [p_1^L, p_1^U]\big), \\
\Gamma_2^{(1)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_1^L, \delta_1^U] \cdot \big(- [v_1(s), w_1(s)] + v_2(s) + [p_1^L, p_1^U]\big), \\
\Lambda_1^{(1)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_1^L, \delta_1^U] \cdot \big(0.5 w_1(s) + [v_2(s), w_2(s)] + [p_1^L, p_1^U]\big), \\
\Lambda_2^{(1)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_1^L, \delta_1^U] \cdot \big(- [v_1(s), w_1(s)] + w_2(s) + [p_1^L, p_1^U]\big).
\end{aligned}$$

At the transition, state continuity gives $Y(1) = [\mathbf{v}(1), \mathbf{w}(1)]$ for (4.16) since $\mathbf{D}_1$ is the identity matrix, $\mathbf{E}_1$ and $\mathbf{J}_1$ are zero matrices and $\mathbf{k}_1$ is a zero vector. Similarly, expanding and taking the natural interval extensions of right hand sides of the nonlinear ODEs in mode 2, we obtain

$$\begin{aligned}
\Gamma_1^{(2)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_2^L, \delta_2^U] \cdot \big(v_1(s) + [v_2(s), w_2(s)] - [p_2^L, p_2^U]\big), \\
\Gamma_2^{(2)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_2^L, \delta_2^U] \cdot \big(- [v_1(s), w_1(s)] + [p_2^L, p_2^U]\big), \\
\Lambda_1^{(2)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_2^L, \delta_2^U] \cdot \big(w_1(s) + [v_2(s), w_2(s)] - [p_2^L, p_2^U]\big), \\
\Lambda_2^{(2)}(\mathbf{v}, \mathbf{w}, \mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U, s) &= [\delta_2^L, \delta_2^U] \cdot \big(- [v_1(s), w_1(s)] + [p_2^L, p_2^U]\big).
\end{aligned}$$

Applying Corollary 4.24, the following nonlinear hybrid system bounds the trans-

formed hybrid system, $\mathbf{v}(s) \leq \mathbf{x}(s) \leq \mathbf{w}(s)$:

$$\text{Mode 1:} \begin{cases} v_1'(s) &= \min\Big(\delta_1^L(0.5v_1(s)+v_2(s)+p_1^L), \delta_1^L(0.5v_1(s)+w_2(s)+p_1^U), \\ & \qquad \delta_1^U(0.5v_1(s)+v_2(s)+p_1^L), \delta_1^U(0.5v_1(s)+w_2(s)+p_1^U)\Big), \\ v_2'(s) &= \min\Big(\delta_1^L(-w_1(s)+v_2(s)+p_1^L), \delta_1^L(-v_1(s)+v_2(s)+p_1^U), \\ & \qquad \delta_1^U(-w_1(s)+v_2(s)+p_1^L), \delta_1^U(-v_1(s)+v_2(s)+p_1^U)\Big), \\ w_1'(s) &= \max\Big(\delta_1^L(0.5w_1(s)+v_2(s)+p_1^L), \delta_1^L(0.5w_1(s)+w_2(s)+p_1^U), \\ & \qquad \delta_1^U(0.5w_1(s)+v_2(s)+p_1^L), \delta_1^U(0.5w_1(s)+w_2(s)+p_1^U)\Big), \\ w_2'(s) &= \max\Big(\delta_1^L(-w_1(s)+w_2(s)+p_1^L), \delta_1^L(-v_1(s)+w_2(s)+p_1^U), \\ & \qquad \delta_1^U(-w_1(s)+w_2(s)+p_1^L), \delta_1^U(-v_1(s)+w_2(s)+p_1^U)\Big), \end{cases}$$

$$\text{Mode 2:} \begin{cases} v_1'(s) &= \min\Big(\delta_2^L(v_1(s)+v_2(s)-p_2^U), \delta_2^L(v_1(s)+w_2(s)-p_2^L), \\ & \qquad \delta_2^U(v_1(s)+v_2(s)-p_2^U), \delta_2^U(v_1(s)+w_2(s)-p_2^L)\Big), \\ v_2'(s) &= \min\Big(\delta_2^L(-w_1(s)+p_2^L), \delta_2^L(-v_1(s)+p_2^U), \\ & \qquad \delta_2^U(-w_1(s)+p_2^L), \delta_2^U(-v_1(s)+p_2^U)\Big), \\ w_1'(s) &= \max\Big(\delta_2^L(w_1(s)+v_2(s)-p_2^U), \delta_2^L(w_1(s)+w_2(s)-p_2^L), \\ & \qquad \delta_2^U(w_1(s)+v_2(s)-p_2^U), \delta_2^U(w_1(s)+w_2(s)-p_2^L)\Big), \\ w_2'(s) &= \max\Big(\delta_2^L(-w_1(s)+p_2^L), \delta_2^L(-v_1(s)+p_2^U), \\ & \qquad \delta_2^U(-w_1(s)+p_2^L), \delta_2^U(-v_1(s)+p_2^U)\Big), \end{cases}$$

$n_e = 2$, $T_\mu = 1, 2$, $T_\tau = \{\hat{I}_i\}$ where $\hat{I}_1 = [0,1]$, $\hat{I}_2 = [1,2]$, $P = [0,1]^2$, $\Delta = [0,1]^2$, and we have state continuity as the transition functions with initial condition $\mathbf{v}(0) = \mathbf{w}(0) = (0,2)$.

This bounding hybrid system can be integrated efficiently with an integrator that supports the rigorous detection of events, due to the min and max functions in the right hand sides. The following results are obtained using the JACOBIAN Dynamic Modeling and Optimization Software [87] release 2.1A with the default options.

Figure 4-11 shows the bounding trajectories obtained for $P = [0,1]^2$ and $\Delta = [0,1]^2$. To illustrate that the trajectories actually bound the transformed system, 20

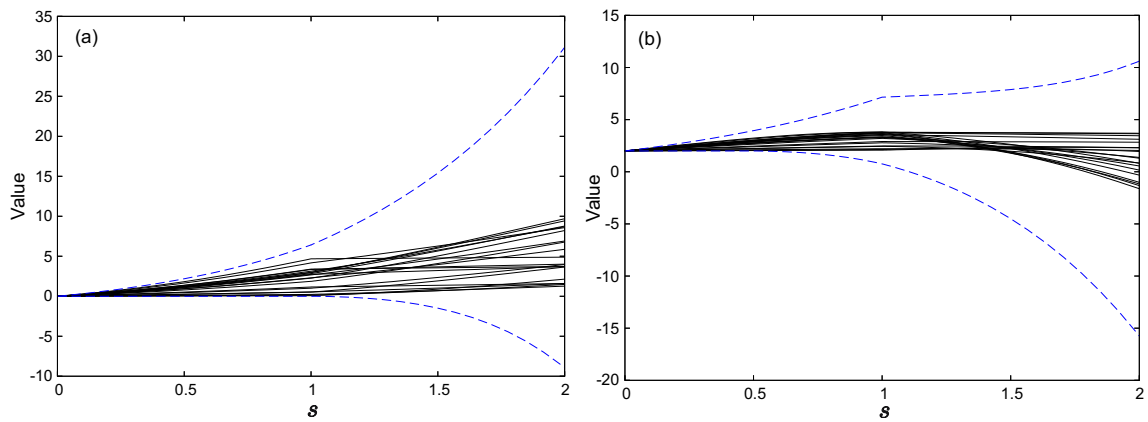


Figure 4-11: Bounding trajectories (dashed lines) and random state trajectories (solid lines) with $P = [0, 1]^2$, $\Delta = [0, 1]^2$ for (a) $\hat{x}_1(s)$, and (b) $\hat{x}_2(s)$.

random points were generated in $P \times \Delta$, and the state trajectories of the transformed system were plotted alongside the bounding trajectories. It can be seen that the bounds indeed enclose the solution of the transformed system, as should be expected on application of Corollary 4.24. Figure 4-12 shows what happens when the bounds on $(\mathbf{p}, \boldsymbol{\delta})$ are changed to $P = [0, 0.25] \times [0.25, 0.5]$, $\Delta = [0.5, 0.75] \times [0.75, 1]$. Again, 20 random points of $(\mathbf{p}, \boldsymbol{\delta})$ were generated in $P \times \Delta$ and plotted together with the new bounding trajectories. Besides bounding the state trajectories, it can be seen from the scales of the vertical axis that the bounding trajectories are closer together than those in Figure 4-11. Finally, Figure 4-13 illustrates the convergence of the bounding trajectories in Theorem 4.26 as P and Δ become degenerate. Note that for the degenerate intervals $P = \Delta = [0.5, 0.5]^2$ (case (f)), the bounding trajectories become the same, i.e., $\mathbf{v}(s) = \mathbf{w}(s) = \hat{\mathbf{x}}(0.5, 0.5, s)$.

4.3.2 Exploiting the Time Transformation

In the previous section, we have developed a method to bound the transformed hybrid system based on the theory of differential inequalities. As previously mentioned, there are no guarantees to the exactness of the generated bounds. In fact, as we shall see, for some simple systems, the bounds that are generated are very weak, and can possess very weak convergence properties.

Thus, there is a need to devise methods for constructing tighter bounds on the

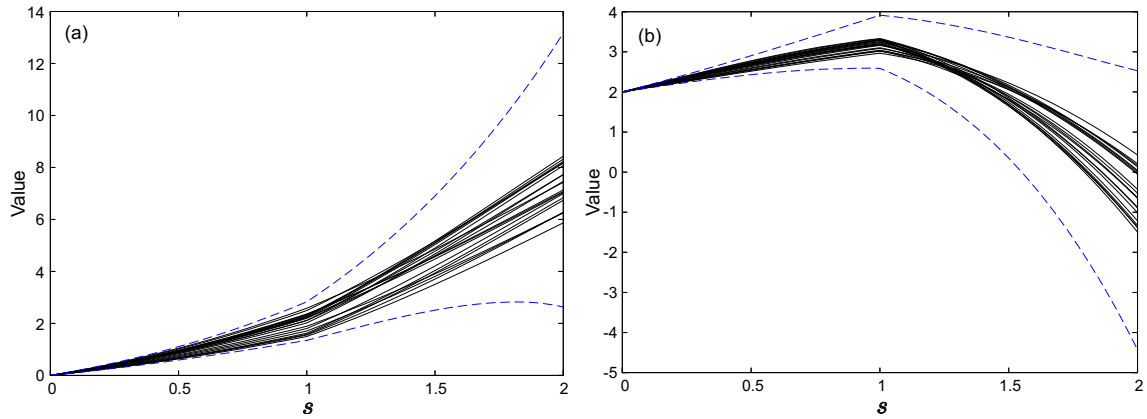


Figure 4-12: Bounding trajectories (dashed lines) and random state trajectories (solid lines) with $P = [0, 0.25] \times [0.25, 0.5]$, $\Delta = [0.5, 0.75] \times [0.75, 1]$ for (a) $\hat{x}_1(s)$, and (b) $\hat{x}_2(s)$.

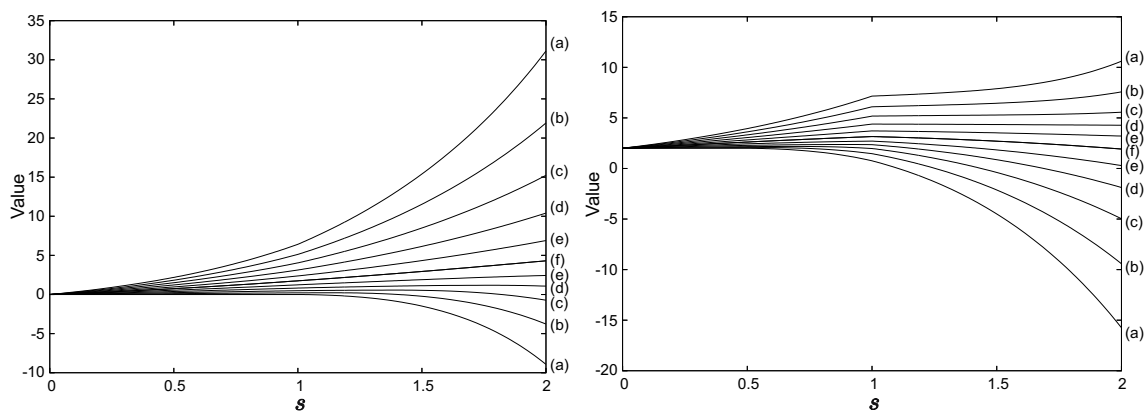


Figure 4-13: Bounding trajectories with $P = \Delta = Z^2$, where Z is given by (a) [0,1], (b) [0.1,0.9], (c) [0.2,0.8], (d) [0.3, 0.7], (e) [0.4, 0.6], and (f) [0.5, 0.5]. The plot on the left is for $\hat{x}_1(s)$, while the one on the right is for $\hat{x}_2(s)$.

state trajectories. As these bounds are utilized in the construction of convex relaxations, the tighter these bounds are, the tighter the relaxations that are constructed, thus accelerating the convergence of the global optimization algorithm. There is an obvious caveat: this is true provided that the cost of obtaining tighter bounds is cheaper than the savings gained from the tighter relaxations. Of course, one can also implement heuristics such as the following hybrid method: use an expensive method for large partitions in the BB tree, but as the partitions shrink, switch to a cheaper method.

In the subsequent sections, we will be presenting new techniques for bounding time transformed dynamic systems, that exploit properties of the time transformation. These methods will not be applicable to the general form of the hybrid system presented in Definition 4.3. Specifically, these methods will be applicable to:

1. Single-stage systems with nonlinear dynamics, in which only stationary simple discontinuities are allowed, as long as a method exists to compute bounds for the original system before the time transformation. In the case of LTV single-stage systems with real-valued parameters, this implies that the exact bounds for the time transformed system can be computed, because the exact bounding trajectories for the original system can be computed.

2. Multi-stage systems with LTI dynamics. The bounds obtained for these systems will have no guarantee of exactness beyond the first stage (epoch).

We shall begin by constructing bounds for a single stage problem whose duration is allowed to vary, and then show how these techniques can be extended to multi-stage systems with LTI dynamics. The impact of these techniques for the single-stage case is that the exact bounding trajectories for a time transformed LTV dynamic system can be obtained.

Consider the following (single-stage) system of ODEs:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t), \qquad \mathbf{x}(\sigma) = \mathbf{x}_0 \tag{4.23}$$

where the time horizon is given by $t \in [\sigma, \tau]$, $\sigma \in \mathbb{R}$ is fixed (constant), $\tau \in [\tau^L, \tau^U] \subset \mathbb{R}$, $\sigma \leq \tau^L \leq \tau^U$, $\mathbf{x}(t) \in X \subset \mathbb{R}^{n_x}$ for all $t \in [\sigma, \tau^U]$, and $\mathbf{f} : X \times [\sigma, \tau^U] \to \mathbb{R}^{n_x}$ is piecewise continuous on $X \times [\sigma, \tau^U]$ where only a finite number of stationary simple discontinuities in t are allowed, and $\mathbf{f}$ is defined at each point of discontinuity. We will assume that a solution exists and is unique for (4.23), at least in the sense of Carothéodory. This assumption automatically implies that the state trajectories $\mathbf{x}(t)$ are continuous in time (see [42, Chp. 2, Theorem 4.2]).

We now apply the CPET to the system described above (from $t \in [\sigma, \tau]$ to $s \in [0, 1]$):

$$t' = \tau - \sigma, \qquad t(\tau, 0) = \sigma. \tag{4.24}$$

Clearly, the solution of (4.24) gives

$$t(\tau, s) = \sigma + s(\tau - \sigma). \tag{4.25}$$

Further, let $\mathbf{y}(\tau, s)$ denote the solution of the transformed system:

$$\mathbf{y}'(\tau, s) = (\tau - \sigma)\mathbf{f}(\mathbf{y}(\tau, s), \sigma + s(\tau - \sigma)), \qquad \mathbf{y}(\tau, 0) = \mathbf{x}_0. \tag{4.26}$$

Lemma 4.28. *Let* $\mathbf{y}$ *be the solution of the transformed system* (4.26)*, and* $\mathbf{x}$ *be the solution of the original system* (4.23)*. Then, for any* $(\tau^*, s^*) \in [\tau^L, \tau^U] \times [0, 1]$*, the transformed solution* $\mathbf{y}(\tau^*, s^*)$ *corresponds to the original solution* $\mathbf{x}(\sigma + s^*(\tau^* - \sigma))$ *of the original system* (4.23)*, i.e.,* $\mathbf{y}(\tau^*, s^*) = \mathbf{x}(\sigma + s^*(\tau^* - \sigma))$*.*

Proof. The lemma is trivial by construction: for any fixed $\tau^* \in [\tau^L, \tau^U]$, the time transformation is simply a change of variables from t to s, where $t = \sigma + s(\tau^* - \sigma)$. Clearly, we have $\mathrm{d}t = (\tau^* - \sigma)\,\mathrm{d}s$. Note that this is the same statement as (4.24) and (4.25). The original ODE system is given by (4.23). Substituting for t (or changing the independent variable), we obtain

$$\mathbf{x}'(\sigma + s(\tau^* - \sigma)) = (\tau^* - \sigma)\mathbf{f}(\mathbf{x}(\sigma + s(\tau^* - \sigma)), \sigma + s(\tau^* - \sigma)).$$

Let $\mathbf{y}(\tau^*, s) \equiv \mathbf{x}(\sigma + s(\tau^* - \sigma))$. Then, the above equation becomes

$$\mathbf{y}'(\tau^*, s) = (\tau^* - \sigma)\mathbf{f}(\mathbf{y}(\tau^*, s), \sigma + s(\tau^* - \sigma)),$$

with initial condition

$$\mathbf{y}(\tau^*, 0) = \mathbf{x}(\sigma) = \mathbf{x}_0.$$

Hence, by construction, $\mathbf{y}(\tau^*, s) = \mathbf{x}(\sigma + s(\tau^* - \sigma)),\ \forall s \in [0, 1]$. □

Remark. Note that the kind of discontinuities allowed in (4.23) are stationary simple ones. Scaled simple discontinuities as defined in Definition 4.2 are not allowed, because then, Lemma 4.28 will not be applicable. This arises because τ would need to be an argument of $\mathbf{f}$ in (4.23), as any scaled simple discontinuities in $\mathbf{f}$ will vary depending on the value of τ. Hence, the value of the state variable $\mathbf{x}(t)$ would no longer only be a function of t, but would also depend on τ as well. In other words, the techniques discussed in this section will only apply when the values of the state trajectories at any specified point in (original) time are the same regardless of the duration of the (original) time horizon.

Lemma 4.29. *Consider the original system* (4.23)*. For any* $t^* \in [\sigma, \tau^U]$*, let* $\mathcal{B}(t^*)$ *denote the set of transformed time points* s^* *such that the transformed solution* $\mathbf{y}(\tau^*, s^*)$ *of* (4.26) *is equal to* $\mathbf{x}(t^*)$ *for some* $\tau^* \in [\tau^L, \tau^U]$*, i.e.,* $\mathbf{y}(\tau^*, s^*) = \mathbf{x}(t^*)$*. Then, the set* $\mathcal{B}(t^*)$ *is given by*

$$\mathcal{B}(t^*) = \{s^* \mid \alpha \leq s^* \leq \beta\},$$

where $\alpha = \frac{t^* - \sigma}{\tau^U - \sigma}$*,* $\beta = \frac{t^* - \sigma}{\gamma - \sigma}$ *and* $\gamma = \max(\tau^L, t^*)$*.*

Proof. Consider any arbitrary $t^* \in [\sigma, \tau^U]$. From Lemma 4.28, any $s^* \in \mathcal{B}(t^*)$ must satisfy (4.25). Hence, to find the lower (upper) bound on s^*, one can pose the following

minimization (maximization) problem:

$$\min_{\tau,s} s$$
$$\text{s.t. } t^* = \sigma + s(\tau - \sigma)$$
$$\tau \in [\tau^L, \tau^U], \quad s \in [0,1].$$

Substituting s with the equality constraint, we obtain the following equivalent problem:

$$\min_{\tau \in [\tau^L, \tau^U]} \frac{t^* - \sigma}{\tau - \sigma}$$
$$\text{s.t. } 0 \leq \frac{t^* - \sigma}{\tau - \sigma} \leq 1.$$

Since $t^* \in [\sigma, \tau^U]$, the constraint is always satisfied for all $\tau \in [\tau^L, \tau^U]$. Clearly, the minimum exists and is attained at τ^U. The corresponding value of s for the original problem is thus $\frac{t^*-\sigma}{\tau^U-\sigma}$. On the other hand, consider the maximization problem. After substituting s with the equality constraint, we obtain the following equivalent problem:

$$\max_{\tau \in [\tau^L, \tau^U]} \frac{t^* - \sigma}{\tau - \sigma}$$
$$\text{s.t. } 0 \leq \frac{t^* - \sigma}{\tau - \sigma} \leq 1.$$

If $t^* \leq \tau^L$, the constraint is always satisfied for all $\tau \in [\tau^L, \tau^U]$. In this case, the maximum exists and is attained at τ^L. The corresponding value of s for the original maximization problem is thus $\frac{t^*-\sigma}{\tau^L-\sigma}$. If $t^* > \tau^L$, then the constraint is only satisfied for $\tau \in [t^*, \tau^U]$. In this case, the maximum exists and is attained at t^*. The corresponding value of s for the original maximization problem is thus $\frac{t^*-\sigma}{t^*-\sigma} = 1$. Next, we consider the case where $\alpha < s^* < \beta$. This implies that

$$\sigma + \frac{t^* - \sigma}{\beta} < \sigma + \frac{t^* - \sigma}{s^*} < \sigma + \frac{t^* - \sigma}{\alpha}$$

or

$$\tau^L \leq \gamma < \tau^* < \tau^U,$$

which implies that τ^* is feasible. Hence, $s^* \in \mathcal{B}(t^*)$. To complete the proof, we have to show that

$$\mathbf{y}(\tau^*, s^*) = \mathbf{x}(t^*), \ \forall s^* \in \mathcal{B}(t^*),$$

which clearly follows from Lemma 4.28. □

Lemma 4.29 can be explained with a geometric illustration. Consider the following system of ODEs,

$$\dot{x}_1 = -x_2, \qquad \dot{x}_2 = x_1, \qquad \mathbf{x}(\sigma = 0) = (1, -1),$$

where $\tau \in [0.5, 2.0]$. Figure 4-14 shows the trajectory of $x_1(t)$ for $t \in [0, 2]$. After applying the time transformation, Figure 4-15 shows the trajectories of $y_1(s)$ for $s \in [0, 1]$, for the values of $\tau = \tau^L = 0.5, \tau = \tau^U = 2.0$ and $\tau = 1.25$. As can be seen, the trajectory for $\tau = \tau^U$ can be thought of as a *squeezing* of the original trajectory in Figure 4-14 to fit in the transformed time scale $s \in [0, 1]$. Consequently, once the trajectory for τ^U has been established, all other trajectories in the range $\tau \in [\tau^L, \tau^U]$ can be thought of as a *stretching* of the trajectory for τ^U. To illustrate Lemma 4.29, consider the point on Figure 4-14 indicated by the dot at $t = 0.228$. The points corresponding to the set $\mathcal{B}(0.228)$ is illustrated by the arrows through the dots on Figure 4-15, where it can be seen that the lower bound on s for $\mathcal{B}(0.228)$ is given by $\tau = 2.0$ at $s = 0.228/2.0 = 0.114$, while the upper bound on s is given by $\tau = \max(0.228, 0.5) = 0.5$ at $s = 0.228/0.5 = 0.456$. The same analysis holds true of the point on Figure 4-14 indicated by the cross at $t = 1.031$. The points corresponding to the set $\mathcal{B}(1.031)$ is illustrated by the arrows through the crosses on Figure 4-15, where the lower bound on s is given by $\tau = 2.0$ at $s = 1.031/2.0 = 0.5155$, while the upper bound is given by $\gamma = \max(1.031, 0.5) = 1.031$ at $s = 1.031/1.031 = 1$.

Although Lemma 4.29 is simple, it provides us with the basic mechanism (and motivation) to prove the algorithms presented in the following sections. The basic

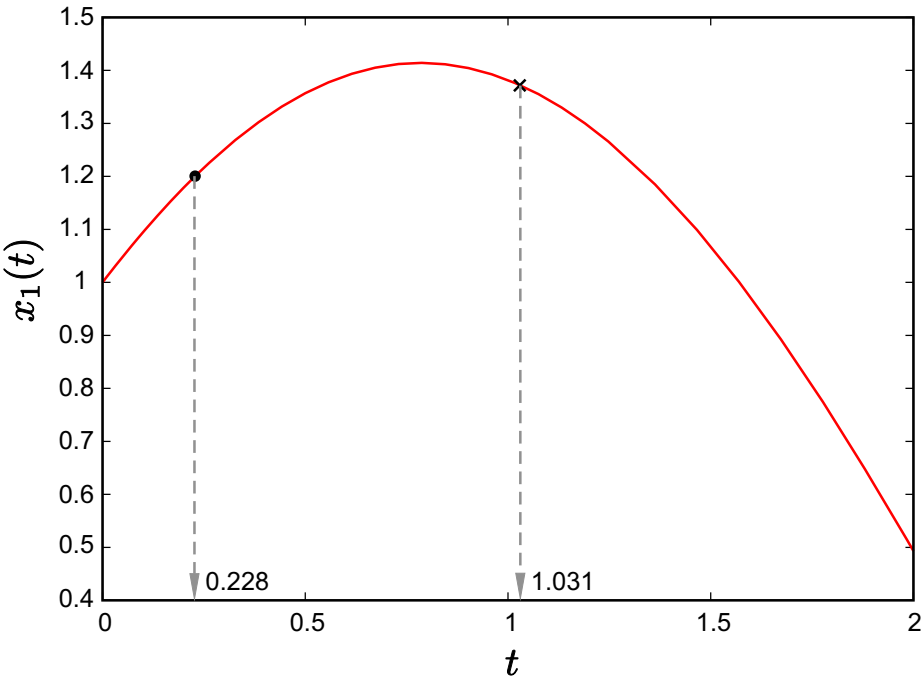


Figure 4-14: Trajectory of $x_1(t)$.

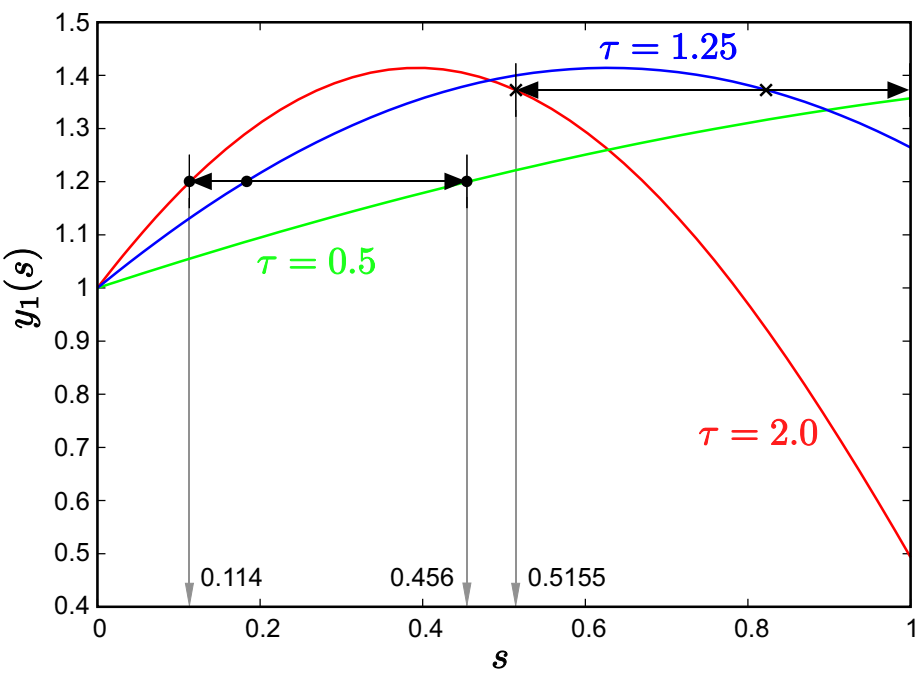


Figure 4-15: Trajectories of $y_1(s)$ for $\tau \in \{0.5, 1.25, 2.0\}$.

idea is as follows: we are going to track the trajectory, in the transformed time scale s, of $\tau = \tau^U$. Consider now any fixed point s^* of this trajectory. Lemma 4.29 assures us that any point of any trajectory in the range $\tau \in [\tau^L, \tau^U]$ that corresponds to the same original solution $\mathbf{x}(\sigma + s^*(\tau^U - \sigma))$ must lie to the right of s^*, and can continue at most to the point of the trajectory with $\tau = \tau^L$, or to the end of the transformed time horizon $s = 1$, whichever occurs first. The proof of the bounding algorithms will then involve showing that the algorithm will bound, at every point in time, the point of the trajectory $\tau = \tau^U$, along with all the associated points of all the other trajectories, which must lie to the right of this original point. Since Lemma 4.28 assures us that any point on any trajectory on the transformed time scale has a corresponding point on the original time scale (and thus, a corresponding point on the trajectory $\tau = \tau^U$), this suffices to show correctness of the bounding algorithms.

4.3.3 Monotonic Bounding Hybrid Systems

In this section, we will describe a bounding strategy that produces monotonically increasing(decreasing) upper(lower) bounds. Consider the following system that tracks $\tau = \tau^U$ in the transformed time scale,

$$\mathbf{u}'(s) = (\tau^U - \sigma)\mathbf{f}(\mathbf{u}(s), \sigma + s(\tau^U - \sigma)), \qquad \mathbf{u}(0) = \mathbf{x}_0.$$

We can then construct the following (lower bounding) hybrid system with state variables $\mathbf{v}(s) \in \mathbb{R}^{n_x}$, where each element of $\mathbf{v}(s)$ is in one of the following modes:

$$\text{Mode 1: } v_i'(s) = u_i'(s), \text{ switch to Mode 2 if } u_i'(s) \geq 0, \tag{4.27}$$

$$\text{Mode 2: } v_i'(s) = 0, \qquad \text{switch to Mode 1 if } u_i(s) \leq v_i(s) - \varepsilon_s, \tag{4.28}$$

for all $i = 1, \ldots, n_x$, $\varepsilon_s > 0$ is some fixed tolerance, $\mathbf{v}(0) = \mathbf{u}(0)$, and the following transition function is enforced at all transitions,

$$v_i(\sigma_{j+1}) = u_i(\tau_j),$$

at the transition time $\tau_j = \sigma_{j+1}$ for the transition between some arbitrary epoch I_j and I_{j+1}, i.e., the value of v_i for the successor mode is set to the value of u_i for the predecessor mode at the transition, for all $i = 1, \ldots, n_x$. Similarly, we can construct the following (upper bounding) hybrid system with state variables $\mathbf{w}(s) \in \mathbb{R}^{n_x}$, where each element of $\mathbf{w}(s)$ is in one of the following modes:

$$\text{Mode 1: } w_i'(s) = u_i'(s), \text{ switch to Mode 2 if } u_i'(s) \leq 0, \tag{4.29}$$

$$\text{Mode 2: } w_i'(s) = 0, \qquad \text{switch to Mode 1 if } u_i(s) \geq w_i(s) + \varepsilon_s, \tag{4.30}$$

for all $i = 1, \ldots, n_x$, $\mathbf{w}(0) = \mathbf{u}(0)$, and the following transition function is enforced at all transitions,

$$w_i(\sigma_{j+1}) = u_i(\tau_j),$$

at the transition time $\tau_j = \sigma_{j+1}$ for the transition between some arbitrary epoch I_j and I_{j+1}, for all $i = 1, \ldots, n_x$. Note that the initial modes for the bounding hybrid systems can be determined for any element $i \in \{1, \ldots, n_x\}$ as follows:

$$v_i \begin{cases} \text{Mode 1 if } u_i'(0) < 0 \\ \text{Mode 2 otherwise.} \end{cases}$$

$$w_i \begin{cases} \text{Mode 1 if } u_i'(0) > 0 \\ \text{Mode 2 otherwise.} \end{cases}$$

Lemma 4.30. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that v_i is in Mode 1 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$v_i(\theta) \leq y_i(\tau, \theta), \ \forall \tau \in [\tau^L, \tau^U]. \tag{4.31}$$

Then,

$$v_i(s) \leq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda]. \tag{4.32}$$

Proof. Since v_i is in Mode 1, $v_i' < 0$ for all $s \in [\theta, \lambda)$, otherwise a transition would have been taken to Mode 2. This clearly implies that $v_i(s)$ is (strictly) monotonically

decreasing in $s \in [\theta, \lambda]$, or, for any fixed $s^* \in [\theta, \lambda]$,

$$v_i(s^*) \leq v_i(s), \ \forall s \in [\theta, s^*]. \tag{4.33}$$

Note that v_i' may be zero at $s = \lambda$, however, since the point comprises a set of measure zero, it does not affect the above result. Assume, for contradiction, that there exists some $(\tau^*, s^*) \in [\tau^L, \tau^U] \times [\theta, \lambda]$ such that

$$v_i(s^*) > y_i(\tau^*, s^*).$$

From Lemma 4.29, there exists some pair (τ^U, α), $\alpha \leq s^*$, such that $y_i(\tau^U, \alpha) = y_i(\tau^*, s^*)$. Consider first the case where $\alpha < \theta$. Since $\alpha < \theta \leq s^*$, Lemma 4.29 ensures that there exists some pair $(\tau^\dagger, \theta)$, $\tau^\dagger \in [\tau^L, \tau^U]$, such that $y_i(\tau^\dagger, \theta) = y_i(\tau^*, s^*)$. This implies

$$y_i(\tau^\dagger, \theta) = y_i(\tau^*, s^*) < v_i(s^*) \leq v_i(\theta) \leq y_i(\tau^\dagger, \theta)$$

where the last two inequalities come from (4.33) and (4.31), which is clearly a contradiction. Consider now the remaining case where $\theta \leq \alpha \leq s^*$. Then,

$$y_i(\tau^U, \alpha) = y_i(\tau^*, s^*) < v_i(s^*) \leq v_i(\alpha),$$

where the last inequality comes from (4.33). Since v_i is in Mode 1, $v_i'(s) = u_i'(s)$ for all $s \in [\theta, \lambda]$. Also, since $v_i(\theta) \leq y_i(\tau^U, \theta) = u_i(\theta)$ from (4.31), this implies that

$$v_i(\alpha) \leq y_i(\tau^U, \alpha).$$

The last two equations clearly form a contradiction. Hence, (4.32) must hold. □

Lemma 4.31. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that w_i is in Mode 1 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$w_i(\theta) \geq y_i(\tau, \theta), \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$w_i(s) \geq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda].$$

Proof. The proof is straightforward, and mirrors that of Lemma 4.30. □

Lemma 4.32. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that v_i is in Mode 2 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$v_i(\theta) \leq y_i(\tau, \theta) + \varepsilon_s, \ \forall \tau \in [\tau^L, \tau^U]. \tag{4.34}$$

Then,

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda]. \tag{4.35}$$

Proof. Since v_i is in Mode 2, $v_i(s) < u_i(s) + \varepsilon_s \ \forall s \in [\theta, \lambda]$, otherwise a transition would have been taken to Mode 1. This implies that

$$v_i(\theta) = v_i(s) < u_i(s) + \varepsilon_s, \ \forall s \in [\theta, \lambda], \tag{4.36}$$

since $v_i'(s) = 0$ for all $s \in [\theta, \lambda]$. Assume, for contradiction, that there exists some $(\tau^*, s^*) \in [\tau^L, \tau^U] \times [\theta, \lambda]$ such that

$$v_i(s^*) > y_i(\tau^*, s^*) + \varepsilon_s.$$

From Lemma 4.29, there exists some pair (τ^U, α), $\alpha \leq s^*$, such that $y_i(\tau^U, \alpha) = y_i(\tau^*, s^*)$. Consider first the case where $\alpha < \theta$. Since $\alpha < \theta \leq s^*$, Lemma 4.29 ensures that there exists some pair $(\tau^\dagger, \theta)$, $\tau^\dagger \in [\tau^L, \tau^U]$, such that $y_i(\tau^\dagger, \theta) = y_i(\tau^*, s^*)$. This implies

$$y_i(\tau^\dagger, \theta) = y_i(\tau^*, s^*) < v_i(s^*) - \varepsilon_s = v_i(\theta) - \varepsilon_s \leq y_i(\tau^\dagger, \theta)$$

where the second equality comes from (4.36) and the second inequality from (4.34), which is clearly a contradiction. Consider now the remaining case where $\theta \leq \alpha \leq s^*$. Then,

$$y_i(\tau^U, \alpha) = y_i(\tau^*, s^*) < v_i(s^*) - \varepsilon_s = v_i(\alpha) - \varepsilon_s < u_i(\alpha) = y_i(\tau^U, \alpha),$$

where the second inequality comes from (4.36), which is clearly a contradiction. Hence, (4.35) must hold. □

Lemma 4.33. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that w_i is in Mode 2 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$w_i(\theta) \geq y_i(\tau, \theta) - \varepsilon_s, \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$w_i(s) \geq y_i(\tau, s) - \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda].$$

Proof. The proof is straightforward and mirrors that of Lemma 4.32. □

We are now in position to present the bounding theorem:

Theorem 4.34. *Assume that there are a finite number of events (transitions taken) for the bounding hybrid systems* (4.27)*,* (4.28)*,* (4.29) *and* (4.30)*. Then, they bound the transformed system* (4.26) *for all $\tau \in [\tau^L, \tau^U]$, i.e.,*

$$\mathbf{v}(s) - \boldsymbol{\varepsilon}_s \leq \mathbf{y}(\tau, s) \leq \mathbf{w}(s) + \boldsymbol{\varepsilon}_s, \quad \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1],$$

where $\boldsymbol{\varepsilon}_s = (\varepsilon_s, \ldots, \varepsilon_s)$.

Proof. Consider any arbitrary element $i \in \{1, \ldots, n_x\}$. Consider next the case of the lower bounding hybrid system, $\mathbf{v}$. Suppose that v_i starts in Mode 1 at $s = 0$. Say that a transition occurs to Mode 2 at some time $s_1 \in [0, 1]$. Since $v_i(0) = u_i(0)$, and $v_i(0) = y_i(\tau, 0)$ $\forall \tau \in [\tau^L, \tau^U]$, we can apply Lemma 4.30 to obtain

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, s_1].$$

Let s_1^- and s_1^+ represent the time s_1 at the epoch boundary for the predecessor and successor mode respectively. Since $v_i(0) = u_i(0)$ and $v_i'(s) = u_i'(s)$ for all $s \in [0, s_1]$, we have $v_i(s_1^-) = u_i(s_1^-)$. At $s = s_1$, we are now in Mode 2. From the transition function, we have $v_i(s_1^+) = u_i(s_1^-) = v_i(s_1^-)$, which implies $v_i(s_1^+) \leq y_i(\tau, s_1^+) + \varepsilon_s$ for

all $\tau \in [\tau^L, \tau^U]$ from the equation above. Suppose that a transition occurs to Mode 1 at some time $s_2 \in [s_1, 1]$. We can then apply Lemma 4.32 to obtain

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [s_1, s_2].$$

As above, let s_2^- and s_2^+ represent the time s_2 at the epoch boundary for the predecessor and successor mode respectively. At s_2, the transition occurs when the transition condition $u_i(s) \leq v_i(s) - \varepsilon_s$ is satisfied, so we have $v_i(s_2^-) - \varepsilon_s = u_i(s_2^-)$. From the transition function, we have $v_i(s_2^+) = u_i(s_1^-)$, which implies that $v_i(s_2^+) \leq y_i(\tau, s_2^+), \ \forall \tau \in [\tau^L, \tau^U]$, from the above equation. Thus, we can repeat the procedure, and apply Lemma 4.30 to the epoch with Mode 1 starting at $s = s_2$. By assumption, there can only be a finite number of transitions within $s \in [0, 1]$. Thus, by finite mathematical induction, we obtain

$$v_i(s) - \varepsilon_s \leq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

Next, we consider the remaining case where v_i starts in Mode 2 at $s = 0$. We can apply the same analysis presented above to arrive at the same result after finite mathematical induction.

Consider now the case of the upper bounding hybrid system, $w_i(s)$. It is straightforward to show that a similar induction argument as that presented above holds, with Lemma 4.31 and 4.33 used in place of Lemma 4.30 and 4.32, to obtain

$$w_i(s) + \varepsilon_s \geq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

Since the choice of i was arbitrary, we have

$$\mathbf{v}(s) - \boldsymbol{\varepsilon}_s \leq \mathbf{y}(\tau, s) \leq \mathbf{w}(s) + \boldsymbol{\varepsilon}_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

□

We will now discuss the steps taken to implement Theorem 4.34. Monotonic

bounding is easily implemented on any integrator that is capable of robustly and reliably detecting zero event points. Here, we will illustrate how this is done using DAEPACK and DSL48SE [128]. First, the system $\mathbf{u}(s)$ is constructed from $\mathbf{y}(\tau, s)$. Next, the lower and upper bounding systems $\mathbf{v}(s)$ and $\mathbf{w}(s)$ are constructed using the IF-THEN-ELSE conditional statements in FORTRAN to represent the transition conditions. This is described by the following:

Pseudo Code Block: Monotonic res0 File

```
do i=1,nx
   ! lower bounding system
   rhs = udot(i)
   if (udot(i) .ge. 0d0) then
      rhs = 0d0
   elseif (u(i) .gt. v(i) - eps) then
      rhs = 0d0
   endif
   vdot(i) = rhs

   ! upper bounding system
   rhs = udot(i)
   if (udot(i) .le. 0d0) then
      rhs = 0d0
   elseif (u(i) .lt. w(i) + eps) then
      rhs = 0d0
   endif
   wdot(i) = rhs
enddo
```

Here, `udot(i)` $\equiv u_i'$, `vdot(i)` $\equiv v_i'$, `wdot(i)` $\equiv w_i'$, `u(i)` $\equiv u_i(s)$, `v(i)` $\equiv v_i(s)$, `w(i)` $\equiv w_i(s)$, and `eps` $\equiv \varepsilon_s$. Note that we have made use of the special structure of the bounding hybrid systems to write the above residual file. No additional modifications have to be made to the event detection algorithm, other than the simple

implementation of the transition functions after each event has been detected. To see that the above residual file is correct, consider the ith element of lower bounding system, v_i. First, we set $v_i' = u_i'$, i.e., the default mode for v_i is Mode 1. Next, the conditional statements force $v_i' = 0$, i.e., they describe conditions under which v_i is in Mode 2. The first `if` condition is the transition condition from Mode 1 to Mode 2. The second `elseif` condition simply describes the condition for which v_i stays in Mode 2, i.e., the negation of the transition condition from Mode 2 to Mode 1.

For the examples presented in Section 4.3.5, ε_s was set to the value of the absolute tolerance of the integrator. Besides preventing chattering or Zeno behavior between the modes of the bounding hybrid system, it also helps to prevent spurious events from being detected in the way that the code has been set up, because the integrator (`DSL48SE`) calculates the trajectories of $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ within some specified tolerances. While any of the bounding trajectories for element i are in Mode 1, they track the value of $u_i(s)$. However, due to the presence of (possible) events and their consistent re-initialization calculations, these trajectories will not be exactly the same numerically for every epoch in which the active mode is 1. Thus, having $\varepsilon_s > 0$ helps to prevent these spurious events from being detected.

Before we end this section, we shall extend Theorem 4.34 to handle parameter dependent ODEs and multi-stage LTI systems. Consider the single-stage ODE system in (4.11). We will make the following assumption: there exists some method to bound the states, $\mathbf{x}(\mathbf{u}(t), t)$, provided by the following (possibly hybrid) bounding system,

$$\dot{\mathbf{v}} = \underline{\mathbf{h}}(\mathbf{v}, \mathbf{w}, \mathbf{z}, t), \qquad \mathbf{v}(t_0) = \mathbf{v}_0, \tag{4.37}$$

$$\dot{\mathbf{w}} = \overline{\mathbf{h}}(\mathbf{v}, \mathbf{w}, \mathbf{z}, t), \qquad \mathbf{w}(t_0) = \mathbf{w}_0, \tag{4.38}$$

$$\dot{\mathbf{z}} = \tilde{\mathbf{h}}(\mathbf{z}, t), \qquad \mathbf{z}(t_0) = \mathbf{z}_0, \tag{4.39}$$

where $\mathbf{v}(t), \mathbf{w}(t) \in \mathbb{R}^{n_x}$ and $\mathbf{z}(t) \in \mathbb{R}^{n_z}$ for all $t \in [t_0, t_f]$, $n_z \geq 0$ is the size of the auxiliary state variables $\mathbf{z}(t)$, such that

$$\mathbf{v}(t) \leq \mathbf{x}(\mathbf{u}(t), t) \leq \mathbf{w}(t), \ \forall \mathbf{u}(t) \in U(t), t \in [t_0, t_f].$$

In addition, we will assume that $\underline{\mathbf{h}}$, $\overline{\mathbf{h}}$ and $\tilde{\mathbf{h}}$ are piecewise continuous on their respective domains, where only a finite number of stationary simple discontinuities are allowed. We will also assume that a solution exists and is unique for the bounding system. In addition, we will assume that the state trajectories $\mathbf{x}(t)$ are continuous in time, although they may be nonsmooth (note that if the bounding systems are hybrid, this assumption implies that state continuity holds for all transitions).

Note that the inclusion of the auxiliary variables $\mathbf{z}(t)$ encompasses the use of time invariant parameters in the bounding systems, e.g., to incorporate $\mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U$ in Corollary 4.24 into the framework above, one can simply force $\mathbf{z}(t) = (\mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U)$ for all $t \in [t_0, t_f]$ by setting $\tilde{\mathbf{h}} = \mathbf{0}$ and setting $\mathbf{z}_0 = (\mathbf{p}^L, \mathbf{p}^U, \boldsymbol{\delta}^L, \boldsymbol{\delta}^U)$.

Let us now assume that we wish to apply the time transformation to (4.11). Without loss of generality, we will set $\sigma = t_0$ and $\tau^U = t_f$. Then, the form of the bounding system (4.37), (4.38) and (4.39) can be cast in the form of (4.23) by considering $\tilde{\mathbf{x}}(t) = (\mathbf{v}(t), \mathbf{w}(t), \mathbf{z}(t))$. Hence, we can apply Theorem 4.34 to obtain the following bounds,

$$\tilde{\mathbf{v}}(s) - \boldsymbol{\varepsilon}_s \leq \tilde{\mathbf{y}}(\tau, s) \leq \tilde{\mathbf{w}}(s) + \boldsymbol{\varepsilon}_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1],$$

where $\tilde{\mathbf{v}}(s)$ and $\tilde{\mathbf{w}}(s)$ are the lower and upper bounds for the transformed system respectively. Clearly, in terms of implementation, we only need to track the lower bounds for $\tilde{x}_1, \ldots, \tilde{x}_{n_x}$ and the upper bounds for $\tilde{x}_{n_x+1}, \ldots, \tilde{x}_{2n_x}$. Let $\hat{\mathbf{v}}(\tau, s)$, $\hat{\mathbf{w}}(\tau, s)$ and $\mathbf{y}(\mathbf{u}(\sigma + s(\tau - \sigma)), \tau, s)$ represent the transformed solution of $\mathbf{v}(t)$, $\mathbf{w}(t)$ and $\mathbf{x}(\mathbf{u}(t), t)$ respectively under the CPET. Then, it follows from the CPET, Lemma 4.28 and Theorem 4.34 that the following holds,

$$\begin{aligned} \tilde{\mathbf{v}}(s) - \boldsymbol{\varepsilon}_s \leq \hat{\mathbf{v}}(\tau, s) \leq \mathbf{y}(\mathbf{u}(\sigma + s(\tau - \sigma)), \tau, s) \leq \hat{\mathbf{w}}(\tau, s) \leq \tilde{\mathbf{w}}(s) + \boldsymbol{\varepsilon}_s, \\ \forall \mathbf{u}(\sigma + s(\tau - \sigma)) \in U(\sigma + s(\tau - \sigma)), \tau \in [\tau^L, \tau^U], s \in [0, 1], \end{aligned} \quad (4.40)$$

which shows that we have obtained rigorous bounds for the transformed system of (4.11).

It is clear from the form of Corollary 4.24 that it satisfies the assumptions made for the bounding system (4.37), (4.38) and (4.39). We will now consider the special case where we have a single-stage LTV ODE system with a finite number of stationary simple discontinuities. First, we note that we have a method of calculating the implied state bounds for LTV ODE system with time invariant, real valued parameters (see Section 2.7). Let the system be given by

$$\dot{\mathbf{x}} = \mathbf{A}(t)\mathbf{x} + \mathbf{B}(t)\mathbf{p} + \mathbf{q}(t), \qquad \mathbf{x}(\sigma) = \mathbf{E}_0\mathbf{p} + \mathbf{k}_0.$$

Then, the implied state bounds are given by the single-stage version of (2.13),

$$x_i^L(t) = n_i(t) + \sum_{j=1}^{n_p} \lambda_{ij}^L(t), \qquad x_i^U(t) = n_i(t) + \sum_{j=1}^{n_p} \lambda_{ij}^U(t),$$

for all $i = 1, \ldots, n_x$ where $\lambda_{ij}^L(t)$ and $\lambda_{ij}^U(t)$ are given by the following for all $j = 1, \ldots, n_p$,

$$\lambda_{ij}^L(t) = \begin{cases} m_{ij}(t)p_j^L \text{ if } m_{ij}(t) \geq 0, \\ m_{ij}(t)p_j^U \text{ otherwise,} \end{cases} \qquad \lambda_{ij}^U(t) = \begin{cases} m_{ij}(t)p_j^U \text{ if } m_{ij}(t) \geq 0, \\ m_{ij}(t)p_j^L \text{ otherwise,} \end{cases}$$

and $\mathbf{M}(t)$ and $\mathbf{n}(t)$ are given by the solution to the following ODE system,

$$\dot{\mathbf{M}} = \mathbf{A}(t)\mathbf{M} + \mathbf{B}(t) \qquad \mathbf{M}(\sigma) = \mathbf{E}_0, \tag{4.41}$$

$$\dot{\mathbf{n}} = \mathbf{A}(t)\mathbf{n} + \mathbf{q}(t) \qquad \mathbf{n}(\sigma) = \mathbf{k}_0. \tag{4.42}$$

Note the reformulation of the min and max in (2.13) into a reversible transition condition, which should be treated as described in Chapter 1. The implied state bounds are continuous (see [120, Proposition 4.1]). We can clearly transform the algebraic equations above by differentiating with respect to t into the following equivalent (hybrid)

differential equations,

$$\dot{x}_i^L(t) = \dot{n}_i(t) + \sum_{j=1}^{n_p} \hat{\lambda}_{ij}^L(t), \qquad \dot{x}_i^U(t) = \dot{n}_i(t) + \sum_{j=1}^{n_p} \hat{\lambda}_{ij}^U(t), \tag{4.43}$$

for all $i = 1, \ldots, n_x$ where $\hat{\lambda}_{ij}^L(t)$ and $\hat{\lambda}_{ij}^U(t)$ are given by the following for all $j = 1, \ldots, n_p$,

$$\hat{\lambda}_{ij}^L(t) = \begin{cases} \dot{m}_{ij}(t) p_j^L \text{ if } m_{ij}(t) \geq 0, \\ \dot{m}_{ij}(t) p_j^U \text{ otherwise,} \end{cases} \qquad \hat{\lambda}_{ij}^U(t) = \begin{cases} \dot{m}_{ij}(t) p_j^U \text{ if } m_{ij}(t) \geq 0, \\ \dot{m}_{ij}(t) p_j^L \text{ otherwise,} \end{cases}$$

with state continuity holding for each transition, and the initial condition

$$x_i^L(\sigma) = k_{0i} + \sum_{j=1}^{n_p} \tilde{\lambda}_{ij}^L, \qquad x_i^U(\sigma) = k_{0i} + \sum_{j=1}^{n_p} \tilde{\lambda}_{ij}^U,$$

for all $i = 1, \ldots, n_x$ where $\tilde{\lambda}_{ij}^L$ and $\tilde{\lambda}_{ij}^U$ are given by the following for all $j = 1, \ldots, n_p$,

$$\tilde{\lambda}_{ij}^L = \begin{cases} e_{0ij} p_j^L \text{ if } e_{0ij} \geq 0, \\ e_{0ij} p_j^U \text{ otherwise,} \end{cases} \qquad \tilde{\lambda}_{ij}^U = \begin{cases} e_{0ij} p_j^U \text{ if } e_{0ij} \geq 0, \\ e_{0ij} p_j^L \text{ otherwise,} \end{cases}$$

for all $i = 1, \ldots, n_x$. We have thus formulated the implied state bounds in the form of the bounding system (4.37), (4.38) and (4.39), since we can set $\mathbf{v}(t) = \mathbf{x}^L(t)$, $\mathbf{w}(t) = \mathbf{x}^U(t)$, and $\mathbf{z}(t) = (\mathbf{M}_1(t), \ldots, \mathbf{M}_{n_x}(t), \mathbf{n}(t))$ where $\mathbf{M}_i(t)$ is the i-th column of $\mathbf{M}(t)$. Thus, the analysis performed above is valid, and (4.40) holds.

Finally, we consider the case where we have multi-stage LTI systems.

Definition 4.35. Consider the hybrid system defined in Definition 4.3 with points 3 and 4 replaced by the following,

3. The parameterization of the bounded real valued controls,

$$\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{S}^{(m_i^*)}\mathbf{p} + \mathbf{W}^{(m_i^*)}\boldsymbol{\delta} + \mathbf{v}^{(m_i^*)},$$
$$\mathbf{u}^L(t) \leq \mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) \leq \mathbf{u}^U(t), \quad \forall t \in [\sigma_1, \sigma_1 + \sum_{j=1}^{n_e} \delta_j^U],$$

where $\mathbf{u}^L(t)$ and $\mathbf{u}^U(t)$ are known lower and upper bounds on the controls $\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t)$, and $\mathbf{S}^{(m_i^*)}$, $\mathbf{W}^{(m_i^*)}$ and $\mathbf{v}^{(m_i^*)}$ are known for all $i = 1, \ldots, n_e$.

4. The LTV ODE system for each mode $m_i^* \in M$, which is given by

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(m_i^*)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \tilde{\mathbf{B}}^{(m_i^*)}\mathbf{p} + \tilde{\mathbf{C}}^{(m_i^*)}\boldsymbol{\delta} + \tilde{\mathbf{D}}^{(m_i^*)}\mathbf{u}(\mathbf{p}, \boldsymbol{\delta}, t) + \tilde{\mathbf{q}}^{(m_i^*)},$$

where $\mathbf{A}^{(m_i^*)}$, $\tilde{\mathbf{B}}^{(m_i^*)}$, $\tilde{\mathbf{C}}^{(m_i^*)}$, $\tilde{\mathbf{D}}^{(m_i^*)}$ and $\tilde{\mathbf{q}}^{(m_i^*)}$ are known for all $i = 1, \ldots, n_e$. After control parameterization, we have

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(m_i^*)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \mathbf{B}^{(m_i^*)}\mathbf{p} + \mathbf{C}^{(m_i^*)}\boldsymbol{\delta} + \mathbf{q}^{(m_i^*)}, \tag{4.44}$$

where $\mathbf{B}^{(m_i^*)} \equiv \tilde{\mathbf{B}}^{(m_i^*)} + \tilde{\mathbf{D}}^{(m_i^*)}\mathbf{S}^{(m_i^*)}$, $\mathbf{C}^{(m_i^*)} \equiv \tilde{\mathbf{C}}^{(m_i^*)} + \tilde{\mathbf{D}}^{(m_i^*)}\mathbf{W}^{(m_i^*)}$ and $\mathbf{q}^{(m_i^*)} \equiv \tilde{\mathbf{D}}^{(m_i^*)}\mathbf{v}^{(m_i^*)} + \tilde{\mathbf{q}}^{(m_i^*)}$ are known for all $i = 1, \ldots, n_e$.

Note that this formulation clearly supports the use of piecewise constant control profiles in the control parameterization framework, where there is a constant control element in each epoch (simply introduce a parameter p_i for each desired constant control element). It is straightforward to incorporate more than one control element in an epoch by splitting the epoch into more epochs as needed, enforcing state continuity at the newly introduced transitions, and adding more optimization parameters as needed. It is also possible to handle or "simulate" scaled discontinuities within the dynamics by increasing the number of modes, epochs and optimization variables suitably. For example, consider the following scenario. We have a hybrid system with 3 epochs, 3 modes $M = \{1, 2, 3\}$, $T_\mu = 1, 2, 3$ and the dynamics for the LTI

system in epoch 2 is given by

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(2)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \mathbf{B}^{(2)}\mathbf{p} + \mathbf{C}^{(2)}\boldsymbol{\delta} + \mathbf{q}^{(2)}.$$

Suppose we wished to incorporate a scaled discontinuity in (the middle of) epoch 2, such that for $t \leq \sigma_2 + (\tau_2 - \sigma_2)/2$, the dynamics are given by

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(\alpha)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \mathbf{B}^{(\alpha)}\mathbf{p} + \mathbf{C}^{(\alpha)}\boldsymbol{\delta} + \mathbf{q}^{(\alpha)},$$

while for $t > \sigma_2 + (\tau_2 - \sigma_2)/2$, the dynamics are given by

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(\beta)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \mathbf{B}^{(\beta)}\mathbf{p} + \mathbf{C}^{(\beta)}\boldsymbol{\delta} + \mathbf{q}^{(\beta)}.$$

Then, we could reformulate the original hybrid system into the following equivalent hybrid system with 4 epochs, 4 modes $M = \{1, \alpha, \beta, 3\}$, $T_\mu = 1, \alpha, \beta, 3$, state continuity for the transition between mode α and β, and the following linear constraint

$$\delta_2 = \delta_3.$$

Clearly, the lower and upper bounds for the durations of epochs 2 and 3 of the reformulated hybrid system would be given by $\delta_{2o}^L/2$ and $\delta_{\delta_{2o}}^U/2$ respectively, where δ_{2o}^L and δ_{2o}^U are the lower and upper bounds respectively for the second epoch of the original hybrid system.

We will now present the algorithm for computing bounds for the transformed system (under the CPET) of the multi-stage hybrid system in Definition 4.35. The idea is to compute the bounds for the first (transformed) epoch, estimate the bounds for the initial conditions for the second (transformed) epoch, treat the initial conditions as parameters for the second (transformed) epoch, and repeat. This is very similar in structure as Algorithm 3.26 presented in Section 3.4.1. It should be clear from the description of the algorithm and the discussion above, that the algorithm does indeed produce rigorous bounds for the hybrid system in Definition 4.35.

Algorithm 4.36.

1. (***First epoch***) Extract the exact bounding system $\dot{\mathbf{x}}^L$, $\dot{\mathbf{x}}^U$, $\dot{\mathbf{M}}$ and $\dot{\mathbf{n}}$ as described above in (4.41), (4.42) and (4.43) for the LTI dynamic system in the first epoch given by

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, t) = \mathbf{A}^{(m_1^*)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, t) + \mathbf{B}^{(m_1^*)}\mathbf{p} + \mathbf{C}^{(m_1^*)}\boldsymbol{\delta} + \mathbf{q}^{(m_1^*)},$$
$$\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \sigma) = \mathbf{E}_0\mathbf{p} + \mathbf{J}_0\boldsymbol{\delta} + \mathbf{k}_0,$$

   where the real valued parameters are given by $(\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta$.

2. Apply Theorem 4.34 with $\sigma = 0$, $\tau^L = \delta_1^L$, $\tau^U = \delta_1^U$ to the extracted bounding system, where $\mathbf{v}(t) = \mathbf{x}^L(t)$, $\mathbf{w}(t) = \mathbf{x}^U(t)$, and $\mathbf{z}(t) = (\mathbf{M}_1(t), \ldots, \mathbf{M}_{n_x}(t), \mathbf{n}(t))$ where $\mathbf{M}_i(t)$ is the i-th column of $\mathbf{M}(t)$ for the system (4.37), (4.38) and (4.39), to obtain the following form of (4.40),

$$\tilde{\mathbf{v}}(s) - \boldsymbol{\varepsilon}_s \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) \leq \tilde{\mathbf{w}}(s) + \boldsymbol{\varepsilon}_s, \ \forall(\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times [0, 1],$$

   where $\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s)$ is the solution of the transformed hybrid system under the CPET, and $\tilde{\mathbf{v}}(s) - \boldsymbol{\varepsilon}_s$ and $\tilde{\mathbf{w}}(s) + \boldsymbol{\varepsilon}_s$ are the generated lower and upper bounds respectively.

3. (***Subsequent epochs***) For $i = 2$ to n_e do:

   (a) Calculate the following interval vector $[\boldsymbol{\theta}, \boldsymbol{\lambda}]$ from the natural interval extension of (4.2),

$$[\boldsymbol{\theta}, \boldsymbol{\lambda}] = \mathbf{D}_i[\tilde{\mathbf{v}}(i-1), \tilde{\mathbf{w}}(i-1)] + \mathbf{E}_i[\mathbf{p}^L, \mathbf{p}^U] + \mathbf{J}_i[\boldsymbol{\delta}^L, \boldsymbol{\delta}^U] + \mathbf{k}_i.$$

   (b) Augment the LTI dynamic system in epoch I_i by introducing the auxiliary real valued parameters $\boldsymbol{\zeta} \in [\boldsymbol{\theta}, \boldsymbol{\lambda}] \subset \mathbb{R}^{n_x}$ to serve as the initial conditions

for the epoch,

$$\dot{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \boldsymbol{\zeta}, t) = \mathbf{A}^{(m_i^*)}\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \boldsymbol{\zeta}, t) + \mathbf{B}^{(m_i^*)}\mathbf{p} + \mathbf{C}^{(m_i^*)}\boldsymbol{\delta} + \mathbf{q}^{(m_i^*)},$$
$$\mathbf{x}(\mathbf{p}, \boldsymbol{\delta}, \boldsymbol{\zeta}, \sigma) = \boldsymbol{\zeta}.$$

Extract the exact bounding system $\dot{\mathbf{x}}^L$, $\dot{\mathbf{x}}^U$, $\dot{\mathbf{M}}$ and $\dot{\mathbf{n}}$ as described above in (4.41), (4.42) and (4.43) for this augmented LTI dynamic system where the real valued parameters are given by $(\mathbf{p}, \boldsymbol{\delta}, \boldsymbol{\zeta}) \in P \times \Delta \times [\boldsymbol{\theta}, \boldsymbol{\lambda}]$.

(c) Apply Theorem 4.34 with $\sigma = 0$, $\tau^L = \delta_1^L$, $\tau^U = \delta_1^U$ to the extracted bounding system, where $\mathbf{v}(t) = \mathbf{x}^L(t)$, $\mathbf{w}(t) = \mathbf{x}^U(t)$, and $\mathbf{z}(t) = (\mathbf{M}_1(t), \ldots, \mathbf{M}_{n_x}(t), \mathbf{n}(t))$ where $\mathbf{M}_i(t)$ is the i-th column of $\mathbf{M}(t)$ for the system (4.37), (4.38) and (4.39), to obtain the following form of (4.40),

$$\tilde{\mathbf{v}}(s) - \boldsymbol{\varepsilon}_s \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, s) \leq \tilde{\mathbf{w}}(s) + \boldsymbol{\varepsilon}_s, \ \forall (\mathbf{p}, \boldsymbol{\delta}, s) \in P \times \Delta \times [i-1, i].$$

Note that Steps 2 and 3(c) in Algorithm 4.36 work because we are considering LTI dynamic systems. The form of (4.41) and (4.42) becomes time invariant,

$$\dot{\mathbf{M}} = \mathbf{A}^{(m_i^*)}\mathbf{M} + \mathbf{B}^{(m_i^*)} \qquad \mathbf{M}(\sigma = 0) = \mathbf{E}_0,$$
$$\dot{\mathbf{n}} = \mathbf{A}^{(m_i^*)}\mathbf{n} + \mathbf{q}^{(m_i^*)} \qquad \mathbf{n}(\sigma = 0) = \mathbf{k}_0,$$

which allows the bounding system of each epoch to be extracted without having to take into account the durations of previous epochs, i.e., the above system can be integrated under Theorem 4.34 with $\sigma = 0$, $\tau^L = \delta_i^L$, and $\tau^U = \delta_i^U$ because the differential equations are time invariant. The algorithm cannot be easily extended to the LTV case because it is not trivial to extract a suitable bounding system for the application of Theorem 4.34 when the form of (4.41) and (4.42) is a function of time (and thus depends on the durations of any previous epochs).

4.3.4 Tight Bounding Hybrid Systems

In this section, we will describe a bounding strategy that produces exact bounds, within some $\varepsilon_s > 0$ tolerance, for the system (4.23) under the CPET. Consider the following system that tracks $\tau = \tau^U$ in the transformed time scale,

$$\mathbf{u}'(s) = (\tau^U - \sigma)\mathbf{f}(\mathbf{u}(s), \sigma + s(\tau^U - \sigma)), \qquad \mathbf{u}(0) = \mathbf{x}_0, \tag{4.45}$$

and the following system that tracks $\tau = \tau^L$ in the transformed time scale,

$$\mathbf{q}'(s) = (\tau^L - \sigma)\mathbf{f}(\mathbf{q}(s), \sigma + s(\tau^L - \sigma)), \qquad \mathbf{q}(0) = \mathbf{x}_0. \tag{4.46}$$

Next, we introduce the following definition of zero event points:

Definition 4.37 (Zero Event Points)**.** Consider the function $g : \mathbb{R} \to \mathbb{R}$. We say that $s^* \in \mathbb{R}$ is an **ascending** zero event point of g at s^* if there exists some $\varepsilon > 0$ such that

$$g(s) < 0 \text{ for all } s \in (s^*, s^* + \varepsilon), \text{ and } g(s) \geq 0 \text{ for all } s \in (s^* - \varepsilon, s^*).$$

Similarly, we say that $s^* \in \mathbb{R}$ is a **descending** zero event point of g at s^* if there exists some $\varepsilon > 0$ such that

$$g(s) > 0 \text{ for all } s \in (s^*, s^* + \varepsilon), \text{ and } g(s) \leq 0 \text{ for all } s \in (s^* - \varepsilon, s^*).$$

Note that these zero event points are essentially the points at which the function g first touches the zero axis, and they essentially represent the times at which the events occur for the bounding hybrid systems defined below.

We can then introduce the following sets, for all $i \in \{1, \ldots, n_x\}$,

$$\begin{aligned}
\mathcal{A}_i^U &\equiv \{(s,j) \mid \ s \text{ is the } j\text{th ascending zero event point of } g = u_i'\}, \\
\mathcal{A}_i^L &\equiv \{(s,j) \mid \ s \text{ is the } j\text{th ascending zero event point of } g = q_i'\}, \\
\mathcal{B}_i^U &\equiv \{(s,j) \mid \ s \text{ is the } j\text{th descending zero event point of } g = u_i'\}, \\
\mathcal{B}_i^L &\equiv \{(s,j) \mid \ s \text{ is the } j\text{th descending zero event point of } g = q_i'\}.
\end{aligned}$$

The index j in the pair (s, j) imposes order for the zero event points. For example, suppose that there are 3 descending zero event points for $g = u_2'$ at the points $s = 0.2, 0.3$ and 0.55. Then, the set $\mathcal{A}_2^U = \{(0.2, 1), (0.3, 2), (0.55, 3)\}$. Similar to the assumption that there are a finite number of events for the bounding hybrid systems in Theorem 4.34, we will assume that there are a finite number of zero event points for u_i' and q_i' for all $i \in \{1, \ldots, n_x\}$ over the finite time horizon of interest, $s \in [0, 1]$.

Based on the sets above, we can define the following sets, for all $i \in \{1, \ldots, n_x\}$, $s \in [0, 1]$,

$$\begin{aligned}
\mathcal{C}_i^U(s) &\equiv \{(s^*, j) \mid (s^*, j) \in \mathcal{A}_i^U \text{ for any } j, s^* \leq s\}, \\
\mathcal{C}_i^L(s) &\equiv \{(s^*, j) \mid (s^*, j) \in \mathcal{A}_i^L \text{ for any } j, s^* \leq s\}, \\
\mathcal{D}_i^U(s) &\equiv \{(s^*, j) \mid (s^*, j) \in \mathcal{B}_i^U \text{ for any } j, s^* \leq s\}, \\
\mathcal{D}_i^L(s) &\equiv \{(s^*, j) \mid (s^*, j) \in \mathcal{B}_i^L \text{ for any } j, s^* \leq s\}, \\
\mathcal{G}_i(s) &\equiv \{(s, j) \mid (s, j) \in \mathcal{C}_i^U(s), (s^*, j) \notin \mathcal{C}_i^L(s) \text{ for any } s^*\}, \\
\mathcal{H}_i(s) &\equiv \{(s, j) \mid (s, j) \in \mathcal{D}_i^U(s), (s^*, j) \notin \mathcal{D}_i^L(s) \text{ for any } s^*\}.
\end{aligned}$$

For the lower bounding system, for any $i \in \{1, \ldots, n_x\}$, $s^* \in [0, 1]$, consider the following optimization problem,

$$\begin{aligned}
&\inf_{s \in [0,1], j \in \mathbb{Z}} u_i(s) \\
&\text{s.t. } (s, j) \in \mathcal{G}_i(s^*).
\end{aligned} \tag{P1a}$$

Let $\eta_i(s^*)$ be the solution value, and $Z^L(s^*)$ be the set $\arg\inf$ of the problem. Furthermore, let $\gamma_i^U(s^*)$ be the solution value of the following problem,

$$\begin{aligned} &\inf_{j\in\mathbb{Z}} j \\ &\text{s.t. } (s,j) \in Z^L(s^*). \end{aligned} \tag{P1b}$$

If the set $\mathcal{G}_i(s^*)$ is empty, then the minimum to (P1a) and minimum to (P1b) does not exist, $\eta_i(s^*)$ is set to $+\infty$, and $\gamma_i^U(s^*)$ is set to $+\infty$. Otherwise, since it is assumed that there is a finite number of zero event points, the set $\mathcal{G}_i(s^*)$ has finite cardinality, and the extrema will exist for (P1a) and (P1b).

Similarly, for the upper bounding system, we introduce the following optimization problem, for any $i \in \{1,\ldots,n_x\}$, $s^* \in [0,1]$,

$$\begin{aligned} &\sup_{s\in[0,1],j\in\mathbb{Z}} u_i(s) \\ &\text{s.t. } (s,j) \in \mathcal{H}_i(s^*). \end{aligned} \tag{P2a}$$

Let $\mu_i(s^*)$ be the solution value, and $Z^U(s^*)$ be the set $\arg\sup$ of the problem. Furthermore, let $\rho_i^U(s^*)$ be the solution value of the following problem,

$$\begin{aligned} &\inf_{j\in\mathbb{Z}} j \\ &\text{s.t. } (s,j) \in Z^U(s^*). \end{aligned} \tag{P2b}$$

If the set $\mathcal{H}_i(s^*)$ is empty, then the maximum to (P2a) and minimum to (P2b) does not exist, $\mu_i(s^*)$ is set to $-\infty$, and $\rho_i^U(s^*)$ is set to $+\infty$. Otherwise, since it is assumed that there is a finite number of zero event points, the set $\mathcal{H}_i(s^*)$ has finite cardinality, and the extrema will exist for (P2a) and (P2b).

Finally, for any $i \in \{1,\ldots,n_x\}$, $s^* \in [0,1]$, consider the following optimization

problem,

$$\begin{aligned} &\sup_{s\in[0,1],j\in\mathbb{Z}} \quad j \\ &\text{s.t. } (s,j)\in\mathcal{C}_i^L(s^*). \end{aligned} \tag{P3}$$

Let $\gamma_i^L(s^*)$ be the solution value of (P3). If the set $\mathcal{C}_i^L(s^*)$ is empty, then the maximum to (P3) does not exist and $\gamma_i^L(s^*)$ is set to $-\infty$. Otherwise, since it is assumed that there is a finite number of zero event points, the set $\mathcal{C}_i^L(s^*)$ has finite cardinality, and the maximum exists for (P3). Similarly, we have the corresponding optimization problem,

$$\begin{aligned} &\sup_{s\in[0,1],j\in\mathbb{Z}} \quad j \\ &\text{s.t. } (s,j)\in\mathcal{D}_i^L(s^*). \end{aligned} \tag{P4}$$

Let $\rho_i^L(s^*)$ be the solution value of (P4). If the set $\mathcal{D}_i^L(s^*)$ is empty, then the maximum to (P4) does not exist and $\rho_i^L(s^*)$ is set to $-\infty$. Otherwise, since it is assumed that there is a finite number of zero event points, the set $\mathcal{D}_i^L(s^*)$ has finite cardinality, and the maximum exists for (P4).

To illustrate these sets and trajectories, consider the example given above where $\mathcal{A}_2^U = \{(0.2,1),(0.3,2),(0.55,3)\}$, and $u_2(0.15)=15$, $u_2(0.35)=5$ and $u_2(0.78)=10$. Suppose that $\tau^U-\sigma = 2(\tau^L-\sigma)$. Then, we have $\mathcal{A}_2^L=\{(0.4,1),(0.6,2)\}$. The sets $\mathcal{C}_2^L(s)$, $\mathcal{C}_2^U(s)$, $\mathcal{G}_2(s)$, and the trajectories $\eta_2(s)$, $\gamma_2^U(s)$, $\gamma_2^L(s)$ are then given by the following,

$$\mathcal{C}_2^U(s) = \begin{cases} \emptyset & \text{for } s\in[0,0.2] \\ \{(0.2,1)\} & \text{for } s\in[0.2,0.3] \\ \{(0.2,1),(0.3,2)\} & \text{for } s\in[0.3,0.55] \\ \{(0.2,1),(0.3,2),(0.55,3)\} & \text{for } s\in[0.55,1] \end{cases}$$

$$\mathcal{C}_2^L(s) = \begin{cases} \emptyset & \text{for } s \in [0, 0.4] \\ \{(0.4, 1)\} & \text{for } s \in [0.4, 0.6] \\ \{(0.4, 1), (0.6, 2)\} & \text{for } s \in [0.6, 1] \end{cases}$$

$$\mathcal{G}_2(s) = \begin{cases} \emptyset & \text{for } s \in [0, 0.2] \\ \{(0.2, 1)\} & \text{for } s \in [0.2, 0.3] \\ \{(0.2, 1), (0.3, 2)\} & \text{for } s \in [0.3, 0.4] \\ \{(0.3, 2)\} & \text{for } s \in [0.4, 0.55] \\ \{(0.3, 2), (0.55, 3)\} & \text{for } s \in [0.55, 0.6] \\ \{(0.55, 3)\} & \text{for } s \in [0.6, 1] \end{cases}$$

$$\eta_2(s) = \begin{cases} \emptyset & \text{for } s \in [0, 0.2] \\ 15 & \text{for } s \in [0.2, 0.4] \\ 5 & \text{for } s \in [0.4, 0.55] \\ 10 & \text{for } s \in [0.55, 1] \end{cases} \qquad \gamma_2^U(s) = \begin{cases} +\infty & \text{for } s \in [0, 0.2] \\ 1 & \text{for } s \in [0.2, 0.4] \\ 2 & \text{for } s \in [0.4, 0.55] \\ 3 & \text{for } s \in [0.55, 1] \end{cases}$$

$$\gamma_2^L(s) = \begin{cases} -\infty & \text{for } s \in [0, 0.4] \\ 1 & \text{for } s \in [0.4, 0.6] \\ 2 & \text{for } s \in [0.6, 1] \end{cases} \cdot$$

Note that we have used the convention of closed intervals for each epoch (the bounding systems are hybrid systems, as will be seen later). These sets and (hybrid) trajectories look complicated, but they are easy to compute and store, and only depend upon the ability to rigorously detect the zero event points of u_i' and q_i' for all $i \in \{1, \ldots, n_x\}$. Once each zero event point has been detected, the sets and trajectories can be constructed appropriately.

We are now in position to construct the following (lower bounding) hybrid system with state variables $\mathbf{v}(s) \in \mathbb{R}^{n_x}$, where each element of $\mathbf{v}(s)$ is in one of the following

modes:

$$\text{Mode 1: } v_i'(s) = u_i'(s), \text{ switch to Mode 2 if } u_i'(s) \geq 0, \tag{4.47}$$

$$\text{Mode 2: } v_i'(s) = 0, \quad \begin{cases} \text{switch to Mode 1 if } u_i(s) \leq v_i(s) - \varepsilon_s, \\ \text{switch to Mode 3 if } q_i'(s) \geq 0 \text{ and } \gamma_i^U(s_-^\dagger) = \gamma_i^L(s_+^\dagger), \end{cases} \tag{4.48}$$

$$\text{Mode 3: } v_i'(s) = q_i'(s), \begin{cases} \text{switch to Mode 1 if } u_i(s) \leq v_i(s) - \varepsilon_s, \\ \text{switch to Mode 2 if } \eta_i(s) \leq v_i(s), \end{cases} \tag{4.49}$$

for all $i = 1, \ldots, n_x$, $\mathbf{v}(0) = \mathbf{u}(0)$, the following transition function is enforced for all transitions to the successor mode 1, i.e., for the transition from Mode 2 to Mode 1 and the transition from Mode 3 to Mode 1,

$$v_i(\sigma_{j+1}) = u_i(\tau_j),$$

at the transition time $\tau_j = \sigma_{j+1}$ for the transition between some arbitrary epoch I_j and I_{j+1}, for all $i = 1, \ldots, n_x$, state continuity is enforced at all other transitions, and the condition $\gamma_i^U(s_-^\dagger) = \gamma_i^L(s_+^\dagger)$ is explained by the following: $s^\dagger$ is the transition time when the condition $q_i'(s) \geq 0$ becomes true, $\gamma_i^U(s_-^\dagger) = \lim_{s \uparrow s^\dagger} \gamma_i^U(s)$, and $\gamma_i^L(s_+^\dagger) = \lim_{s \downarrow s^\dagger} \gamma_i^L(s)$. Essentially, the transition from Mode 2 to Mode 3 is taken if the transition condition $q_i'(s) \geq 0$ becomes true, and the value of $\gamma_i^U(s)$ at the end of Mode 2 (the predecessor mode) is equal to the value of $\gamma_i^L(s)$ at the beginning of Mode 3 (the successor mode). This ensures that the correct zero event point of $q_i(s)$ is followed that corresponds to the zero event point of $u_i(s)$ that started the incumbent Mode 2. Note that the condition $\gamma_i^U(s_-^\dagger) = \gamma_i^L(s_+^\dagger)$ should not be treated as a conventional transition condition by the integrator, i.e., it does not need to become a discontinuity function to be tracked by the integrator; instead, it should be thought of as a supplementary condition that is checked after event detection of the transition condition $q_i'(s) \geq 0$. This is illustrated by the algorithm presented later in this section.

Similarly, we can construct the following (upper bounding) hybrid system with state variables $\mathbf{w}(s) \in \mathbb{R}^{n_x}$, where each element of $\mathbf{w}(s)$ is in one of the following modes:

$$\text{Mode 1: } w_i'(s) = u_i'(s), \text{ switch to Mode 2 if } u_i'(s) \leq 0, \tag{4.50}$$

$$\text{Mode 2: } w_i'(s) = 0, \quad \begin{cases} \text{switch to Mode 1 if } u_i(s) \geq w_i(s) + \varepsilon_s, \\ \text{switch to Mode 3 if } q_i'(s) \leq 0 \text{ and } \rho_i^U(s_-^\dagger) = \rho_i^L(s_+^\dagger), \end{cases} \tag{4.51}$$

$$\text{Mode 3: } w_i'(s) = q_i'(s), \begin{cases} \text{switch to Mode 1 if } u_i(s) \geq w_i(s) + \varepsilon_s, \\ \text{switch to Mode 2 if } \mu_i(s) \geq w_i(s), \end{cases} \tag{4.52}$$

for all $i = 1, \ldots, n_x$, $\mathbf{w}(0) = \mathbf{u}(0)$, the following transition function is enforced for all transitions to the successor mode 1, i.e., for the transition from Mode 2 to Mode 1 and the transition from Mode 3 to Mode 1,

$$w_i(\sigma_{j+1}) = u_i(\tau_j),$$

at the transition time $\tau_j = \sigma_{j+1}$ for the transition between some arbitrary epoch I_j and I_{j+1}, for all $i = 1, \ldots, n_x$, state continuity is enforced at all other transitions, and the condition $\rho_i^U(s_-^\dagger) = \rho_i^L(s_+^\dagger)$ is explained by the following: $s^\dagger$ is the transition time when the transition condition $q_i'(s) \leq 0$ becomes true, $\rho_i^U(s_-^\dagger) = \lim_{s \uparrow s^\dagger} \rho_i^U(s)$, and $\rho_i^L(s_+^\dagger) = \lim_{s \downarrow s^\dagger} \rho_i^L(s)$. The same comments made about the condition $\gamma_i^U(s_-^\dagger) = \gamma_i^L(s_+^\dagger)$ above applies here. Note that the initial modes for the bounding hybrid systems can be determined for any element $i \in \{1, \ldots, n_x\}$ as follows:

$$v_i \begin{cases} \text{Mode 1 if } u_i'(0) < 0 \\ \text{Mode 3 otherwise,} \end{cases}$$

$$w_i \begin{cases} \text{Mode 1 if } u_i'(0) > 0 \\ \text{Mode 3 otherwise.} \end{cases}$$

Also note that in both hybrid systems, there are two possible pending transitions in Modes 2 and 3, for each element i. To define a deterministic execution of the hybrid system, if the earliest transition times for both pending transitions are the same, then preference is given to the transition to Mode 1, i.e., the transition to Mode 1 has priority over those to other modes. This precedence relation defines a deterministic execution of the bounding hybrid systems.

Lemma 4.38. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that v_i is in Mode 1 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$v_i(\theta) \leq y_i(\tau, \theta), \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$v_i(s) \leq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda].$$

Proof. The proof is identical to that of Lemma 4.30. □

Lemma 4.39. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that w_i is in Mode 1 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$w_i(\theta) \geq y_i(\tau, \theta), \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$w_i(s) \geq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda].$$

Proof. The proof is identical to that of Lemma 4.31. □

Lemma 4.40. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that v_i is in Mode 2 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$v_i(\theta) \leq y_i(\tau, \theta) + \varepsilon_s, \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda]. \tag{4.53}$$

Proof. The proof for Lemma 4.32 applies here, since the same transition to Mode 1 is pending. The presence of the other pending transition to Mode 3 may reduce the time spent in the current Mode 2, but will not change the desired result (4.53). □

Lemma 4.41. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that w_i is in Mode 2 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$w_i(\theta) \geq y_i(\tau, \theta) - \varepsilon_s, \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$w_i(s) \geq y_i(\tau, s) - \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda]. \tag{4.54}$$

Proof. The proof for Lemma 4.33 applies here, since the same transition to Mode 1 is pending. The presence of the other pending transition to Mode 3 may reduce the time spent in the current Mode 2, but will not change the desired result (4.54). □

Lemma 4.42. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that v_i is in Mode 3 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$v_i(\theta) \leq y_i(\tau, \theta) + \varepsilon_s, \ \forall \tau \in [\tau^L, \tau^U]. \tag{4.55}$$

Then,

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda]. \tag{4.56}$$

Proof. From (4.55), $v_i(\theta) \leq y_i(\tau^L, \theta) + \varepsilon_s = q_i(\theta) + \varepsilon_s$. Since the dynamics are given by Mode 3, $v_i'(s) = q_i'(s)$ for all $s \in [\theta, \lambda]$, and thus $v_i(s) \leq q_i(s) + \varepsilon_s = y_i(\tau^L, s) + \varepsilon_s \ \forall s \in [\theta, \lambda]$. Assume, for contradiction, that there exists some $(\tau^*, s^*) \in [\tau^L, \tau^U] \times [\theta, \lambda]$ such that

$$v_i(s^*) > y_i(\tau^*, s^*) + \varepsilon_s.$$

From Lemma 4.29, there exists some pair (τ^U, α), $\alpha \leq s^*$, such that $y_i(\tau^U, \alpha) = u_i(\alpha) = y_i(\tau^*, s^*)$. Also, there exists some pair (τ^U, β), $\beta \leq s^*$, such that $y_i(\tau^U, \beta) = u_i(\beta) = y_i(\tau^L, s^*) = q_i(s^*)$. Since $q_i(s^*) \geq v_i(s^*) - \varepsilon_s > y_i(\tau^*, s^*)$, we have $\tau^* > \tau^L$. Let $y_i(\tau^*, s^\dagger) = y_i(\tau^L, s^*)$, where $\alpha \leq s^\dagger < s^*$. Thus, it follows from Lemma 4.29 that $\beta < \alpha$ since $s^\dagger < s^*$.

Now, consider $y_i(\tau^U, s^*) = u_i(s^*)$. We must have $u_i(s^*) > v_i(s^*) - \varepsilon_s$, for if $u_i(s^*) \leq v_i(s^*) - \varepsilon_s$, then a transition to Mode 1 is taken. We thus have the following points that lie on the trajectory of $u_i(s)$: $u_i(\alpha)$, $u_i(\beta)$ and $u_i(s^*)$, where $\beta < \alpha < s^*$, $u_i(\beta) > u_i(\alpha) < u_i(s^*)$. Consider the interval (β, α). Since $u_i(s)$ is continuous by assumption, in order for $u_i(\beta) > u_i(\alpha)$, there must exist some non-degenerate interval within (β, α) such that $u_i'(s) < 0$ (for otherwise u_i will be monotonically increasing). Similarly, consider the interval (α, s^*). Since $u_i(s)$ is continuous by assumption, in order for $u_i(\alpha) < u_i(s^*)$, there must exist some non-degenerate interval within (α, s^*) such that $u_i'(s) > 0$. Thus, there must exist an ascending zero event point for u_i', $\hat{s}$, within the interval (β, s^*) such that $u_i(\hat{s}) \leq u_i(\alpha)$, for $u_i(s)$ is continuous. Clearly, this implies that the value of $\eta_i(\hat{s}) \leq u_i(\hat{s}) \leq u_i(\alpha)$. Since $\hat{s} > \beta$, this implies that the corresponding zero event point for q_i' must occur at some time $s^\ddagger > s^*$. Hence, we must have $\eta_i(s^*) \leq \eta_i(\hat{s}) \leq u_i(\alpha) = y_i(\tau^*, s^*) < v_i(s^*) - \varepsilon_s$, which is a contradiction, since $\eta_i(s^*) < v_i(s^*) - \varepsilon_s$ means that a transition to Mode 2 would have occurred. Thus, (4.56) must hold. □

Lemma 4.43. *Consider any arbitrary $i \in \{1, \ldots, n_x\}$. Suppose that w_i is in Mode 2 for the transformed time interval $s \in [\theta, \lambda]$, where $0 \leq \theta \leq \lambda \leq 1$, and the following condition holds*

$$w_i(\theta) \geq y_i(\tau, \theta), \ \forall \tau \in [\tau^L, \tau^U].$$

Then,

$$w_i(s) \geq y_i(\tau, s), \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [\theta, \lambda].$$

Proof. The proof is straightforward and mirrors that of Lemma 4.42. □

We are now in position to present the bounding theorem:

Theorem 4.44. *Assume that there are a finite number of events (transitions taken) for the bounding hybrid systems* (4.47), (4.48), (4.49), (4.50), (4.51) *and* (4.52). *Then, they bound the transformed system* (4.26) *for all $\tau \in [\tau^L, \tau^U]$, i.e.,*

$$\mathbf{v}(s) - \boldsymbol{\varepsilon}_s \leq \mathbf{y}(\tau, s) \leq \mathbf{w}(s) + \boldsymbol{\varepsilon}_s, \ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

Proof. Consider any arbitrary element $i \in \{1, \ldots, n_x\}$. Consider next the case of the lower bounding hybrid system, $\mathbf{v}(s)$. Suppose that $v_i(s)$ starts in Mode 1 at $s = 0$. Suppose further that a transition occurs to Mode 2 at some time $s_1 \in [0, 1]$. Since $v_i(0) = u_i(0)$, and $v_i(0) = y_i(\tau, 0)\ \forall \tau \in [\tau^L, \tau^U]$, we can apply Lemma 4.38 to obtain

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s,\ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, s_1].$$

Let s_1^- and s_1^+ represent the time s_1 at the epoch boundary for the predecessor and successor mode respectively. Since $v_i(0) = u_i(0)$ and $v_i'(s) = u_i'(s)$ for all $s \in [0, s_1]$, we have $v_i(s_1^-) = u_i(s_1^-)$. At $s = s_1$, we are now in Mode 2. From the transition function which is state continuity, we have $v_i(s_1^+) = v_i(s_1^-)$, which implies $v_i(s_1^+) \leq y_i(\tau, s_1^+) + \varepsilon_s$ for all $\tau \in [\tau^L, \tau^U]$ from the equation above. Suppose that a transition occurs to either Mode 1 or Mode 3 at some time $s_2 \in [s_1, 1]$. We can then apply Lemma 4.40 to obtain

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s,\ \forall (\tau, s) \in [\tau^L, \tau^U] \times [s_1, s_2].$$

As above, let s_2^- and s_2^+ represent the time s_2 at the epoch boundary for the predecessor and successor mode respectively. First, we consider the case where the transition is to Mode 1. This transition occurs when the transition condition $u_i(s) \leq v_i(s) - \varepsilon_s$ is satisfied, so we have $v_i(s_2^-) - \varepsilon_s = u_i(s_2^-)$. From the transition function, we have $v_i(s_2^+) = u_i(s_2^-)$, which implies that $v_i(s_2^+) \leq y_i(\tau, s_2^+),\ \forall \tau \in [\tau^L, \tau^U]$, from the above equation. Thus, we can repeat the procedure, and apply Lemma 4.38 to the epoch with Mode 1 starting at $s = s_2$.

Next, we consider the case where the transition is to Mode 3. This transition occurs with state continuity as the transition function, and so we have $v_i(s_2^+) = v_i(s_2^-)$, which implies $v_i(s_2^+) \leq y_i(\tau, s_2^+) + \varepsilon_s$ for all $\tau \in [\tau^L, \tau^U]$ from the equation above. Now, suppose that a transition occurs to either Mode 1 or Mode 2 at some time $s_3 \in [s_2, 1]$. We can then apply Lemma 4.42 to obtain

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s,\ \forall (\tau, s) \in [\tau^L, \tau^U] \times [s_2, s_3].$$

First, we consider the case where the transition is to Mode 1. This transition occurs when the transition condition $u_i(s) \leq v_i(s) - \varepsilon_s$ is satisfied, so we have $v_i(s_3^-) - \varepsilon_s = u_i(s_3^-)$. From the transition function, we have $v_i(s_3^+) = u_i(s_3^-)$, which implies that $v_i(s_3^+) \leq y_i(\tau, s_3^+),\ \forall \tau \in [\tau^L, \tau^U]$, from the above equation. Thus, we can repeat the procedure, and apply Lemma 4.38 to the epoch with Mode 1 starting at $s = s_3$.

Next, we consider the case where the transition is to Mode 2. This transition occurs with state continuity as the transition function, and so we have $v_i(s_3^+) = v_i(s_3^-)$, which implies $v_i(s_3^+) \leq y_i(\tau, s_3^+) + \varepsilon_s$ for all $\tau \in [\tau^L, \tau^U]$ from the equation above. Thus, we can repeat the procedure, and apply Lemma 4.40 to the epoch with Mode 2 starting at $s = s_3$.

By assumption, there can only be a finite number of transitions within $s \in [0, 1]$. Thus, by finite mathematical induction, we obtain

$$v_i(s) \leq y_i(\tau, s) + \varepsilon_s,\ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

Next, we consider the remaining case where $v_i(s)$ starts in Mode 3 at $s = 0$. We can apply the same analysis presented above to arrive at the same result after finite mathematical induction.

Consider now the case of the upper bounding hybrid system, $w_i(s)$. It is straightforward to show that a similar induction argument as that presented above holds, with Lemmas 4.39, 4.41 and 4.43 used in place of Lemmas 4.38, 4.40 and 4.42, to obtain

$$w_i(s) \geq y_i(\tau, s) - \varepsilon_s,\ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

Since the choice of i was arbitrary, we have

$$\mathbf{v}(s) - \boldsymbol{\varepsilon}_s \leq \mathbf{y}(\tau, s) \leq \mathbf{w}(s) + \boldsymbol{\varepsilon}_s,\ \forall (\tau, s) \in [\tau^L, \tau^U] \times [0, 1].$$

□

Theorem 4.45. *Consider Theorem 4.44. The bounds obtained from the bounding*

hybrid systems are exact in the following sense: for any $i \in \{1, \ldots, n_x\}$, $s^* \in [0, 1]$,

$$y_i(\tau^\dagger, s^*) \leq v_i(s^*), \forall \tau \in [\tau^L, \tau^U]$$
$$w_i(s^*) \leq y_i(\hat{\tau}, s^*), \forall \tau \in [\tau^L, \tau^U]$$

for some $\tau^\dagger, \hat{\tau} \in [\tau^L, \tau^U]$.

Proof. Consider the lower bounding hybrid system for any $i \in \{1, \ldots, n_x\}$, $v_i(s)$. For any $s \in [0, 1]$, v_i is in either one of Modes 1, 2 or 3. Consider now any epoch for which $v_i(s)$ is in Mode 1. Let this epoch be $[\theta, \lambda] \subset [0, 1]$. If this is the first epoch, then $v_i(\theta) = y_i(\tau^U, \theta) = u_i(\theta)$ by construction. Otherwise, the transition to the epoch can either be from Mode 2 or Mode 3, with the transition function setting $v_i(\theta) = u_i(\theta)$. Thus, in both cases, we have $v_i(\theta) = y_i(\tau^U, \theta) = u_i(\theta)$. For the epoch under consideration, the dynamics of v_i is given by $v_i'(s) = u_i'(s)$. Hence, we have shown that whenever $v_i(s)$ is in Mode 1, it is equivalent to $y_i(\tau^U, s)$.

Now, consider any epoch for which v_i is in Mode 2, and let this epoch be $[\theta, \lambda] \subset [0, 1]$. The transition to this mode can be either from Mode 1 or Mode 3. Suppose that the predecessor mode is Mode 1. At the transition, state continuity is preserved, and so $v_i(\theta) = u_i(\theta)$. Since the dynamics of v_i in Mode 2 is given by $v_i'(s) = 0$, we have $v_i(s) = v_i(\theta) = u_i(\theta)$ for all $s \in [\theta, \lambda]$. Consider now the other case where the predecessor mode is Mode 3. At the transition, state continuity is preserved, and so $v_i(\theta) = \eta_i(\theta)$, where $\eta_i(\theta) = u_i(\alpha)$ for some $\alpha \leq \theta$, by construction of $\eta_i(s)$. Since the dynamics of v_i in Mode 2 is given by $v_i'(s) = 0$, we have $v_i(s) = v_i(\theta) = u_i(\alpha)$ for all $s \in [\theta, \lambda]$. In either case, the value of $v_i(s)$ in the epoch $[\theta, \lambda]$ is equal to the value of $u_i(\beta)$ where $\beta \leq \theta$. From Lemma 4.29, there exists some $\tau \in [\tau^L, \tau^U]$ such that $y_i(\tau, s) = y_i(\tau^U, \beta) = y_i(\tau^L, \gamma)$ for any $s \in [\beta, \gamma]$. The transition from Mode 2 at $s = \lambda$ can be to either Mode 1 or Mode 3. By construction, the transition condition from Mode 2 to Mode 3, $q_i'(s) \leq 0$ and $\gamma_i^U(s_-^\dagger) = \gamma_i^L(s_+^\dagger)$, ensures that $\lambda \leq \gamma$. The transition condition to Mode 1 is given by $u_i(s) \leq v_i(s) - \varepsilon_s$, hence either $u_i(s^*) = y_i(\tau^U, s^*) \leq v_i(s^*)$ or there exists some $\tau \in [\tau^L, \tau^U]$ such that $v_i(s^*) = y_i(\tau, s^*)$ for any $s^* \in [\theta, \lambda]$.

Finally, consider any epoch for which v_i is in Mode 3, and let this epoch be $[\theta, \lambda] \subset [0, 1]$. If this is the first epoch, then $v_i(\theta) = y_i(\tau^L, \theta) = q_i(\theta)$ by construction. Otherwise, the transition to the epoch must be from Mode 2. At the transition, state continuity is preserved, so $v_i(\theta) = u_i(\beta)$ for some $\beta \leq \theta$. By construction, the transition condition from Mode 2 to Mode 3, $q_i'(s) \leq 0$ and $\gamma_i^U(s_-^\dagger) = \gamma_i^L(s_+^\dagger)$, ensures that θ is the zero event point for $y_i(\tau^L, \theta)$ corresponding to the same zero event point for $y_i(\tau^U, \beta) = u_i(\beta)$. Thus, we have $v_i(\theta) = y_i(\tau^L, \theta) = q_i(\theta)$. For the epoch under consideration, the dynamics of v_i is given by $v_i'(s) = q_i'(s)$. Hence, we have shown that whenever v_i is in Mode 3, it is equivalent to $y_i(\tau^L, s)$.

We have thus shown that for any $i \in \{1, \ldots, n_x\}$, $s^* \in [0, 1]$,

$$y_i(\tau^\dagger, s^*) \leq v_i(s^*), \ \forall \tau \in [\tau^L, \tau^U],$$

for some $\tau^\dagger \in [\tau^L, \tau^U]$. The same analysis can be applied for the upper bounding hybrid system to obtain the desired result. □

Clearly, the above theorem shows that the bounds $\mathbf{v}(s) - \boldsymbol{\varepsilon}_s$ and $\mathbf{w}(s) + \boldsymbol{\varepsilon}_s$ are at most $\boldsymbol{\varepsilon}_s$ away from the exact bounds. Theorem 4.44 is slightly more complicated to set up within the `DAEPACK` and `DSL48SE` framework, compared to Theorem 4.34. The presence of the transition conditions to Mode 3, and the form of the conditions (which requires the construction of the various sets and trajectories presented above) necessitates some additional post-processing following the detection of each event.

First, the systems $\mathbf{u}(s)$ and $\mathbf{q}(s)$ are constructed from $\mathbf{y}(\tau, s)$. Next, the lower and upper bounding systems $\mathbf{v}(s)$ and $\mathbf{w}(s)$ are constructed in the `FORTRAN` residual file for `DAEPACK` with the appropriate transition conditions. This is illustrated by the following pseudo code:

Pseudo Code Block: Exact res0 File

```
do i=1,nx
  rhs(i,1) = udot(i)
  rhs(i,2) = 0.0
  rhs(i,3) = qdot(i)
```

```
if (udot(i) .le. 0.0) then
  phi(i) = 1
else
  phi(i) = 0
endif

if (qdot(i) .le. 0.0) then
  psi(i) = 1
else
  psi(i) = 0
endif

if (u(i) .le. v(i) - eps) then
  theta(i) = 1
else
  theta(i) = 0
endif

if (u(i) .ge. w(i) + eps) then
  lambda(i) = 1
else
  lambda(i) = 0
endif

if (v(i) .ge. eta(i)) then
  alpha(i) = 1
else
  alpha(i) = 0
endif
```

```
      if (w(i) .le. mu(i)) then
        beta(i) = 1
      else
        beta(i) = 0
      endif

      vdot(i) = rhs(i,modeL(i))
      wdot(i) = rhs(i,modeU(i))
    enddo
```

Here, `udot(i)` $\equiv u_i'$, `qdot(i)` $\equiv q_i'$, `vdot(i)` $\equiv v_i'$, `wdot(i)` $\equiv w_i'$, `u(i)` $\equiv u_i(s)$, `q(i)` $\equiv q_i(s)$, `v(i)` $\equiv v_i(s)$, `w(i)` $\equiv w_i(s)$, `phi(i)` $\equiv \phi_i(s)$, `psi(i)` $\equiv \psi_i(s)$, `theta(i)` $\equiv \theta_i(s)$, `lambda(i)` $\equiv \lambda_i(s)$, `alpha(i)` $\equiv \alpha_i(s)$, `beta(i)` $\equiv \beta_i(s)$, `eta(i)` $\equiv \eta_i(s)$, `mu(i)` $\equiv \mu_i(s)$, `modeL(i)` $\equiv m_i^L(s)$, `modeU(i)` $\equiv m_i^U(s)$ and `eps` $\equiv \varepsilon_s$. The value of ε_s is the same as that described above for the implementation of Theorem 4.34. The role of the `if-then-else` equations are to define the transition conditions for event detection.

The algorithm for integrating the exact bounding hybrid systems is described by the following:

Algorithm 4.46.

1. (***Initialization***) Determine the starting modes m_i^L and m_i^U based on the values of $u_i'(0)$, for all $i = 1, \ldots, n_x$. Perform a consistent initialization calculation for $\mathbf{q}(0), \mathbf{u}(0), \mathbf{v}(0), \mathbf{w}(0)$ and their derivatives. Set $\eta_i = +\infty$ and $\mu_i = -\infty$ for all $i = 1, \ldots, n_x$. Store the values of ϕ_i^S, ψ_i^S, by calling the `res0` file with ϕ_i^S, ψ_i^S, in place of ϕ_i, ψ_i, for all $i = 1, \ldots, n_x$. Set $s = 0$. Set the values of the counters $\gamma_i^L = \gamma_i^C = \rho_i^L = \rho_i^C = 0$, $\gamma_i^U = \gamma_i^S = \rho_i^U = \rho_i^S = -\infty$ for all $i = 1, \ldots, n_x$. Initialize the (dynamically allocated) arrays v_i^S and w_i^S to be empty for all $i = 1, \ldots, n_x$.

2. (***Integration***) Integrate the hybrid system until an event has been detected, or until $s = 1$. In the latter case, terminate.

3. (***Event Detection***) Let the polished time event be s^*. Set $s = s^*$. Perform a consistent re-initialization calculation for $\mathbf{q}(s^*), \mathbf{u}(s^*), \mathbf{v}(s^*), \mathbf{w}(s^*)$ and their derivatives. Update the values of ϕ_i, ψ_i, θ_i, λ_i for all $i = 1, \ldots, n_x$ by calling the `res0` file.

4. (***Update Zero Event Counters***) For $i = 1, \ldots, n_x$ do:

   (a) if $(\psi_i^S \neq \psi_i)$

   if $(\psi_i = 1)$ set $\rho_i^L = \rho_i^L + 1$ else set $\gamma_i^L = \gamma_i^L + 1$ endif

   endif

   (b) if $(\phi_i^S \neq \phi_i)$

   if $(\phi_i = 1)$ set $\rho_i^C = \rho_i^C + 1$, and $w_i^S(\rho_i^C) = u_i(s^*)$

   else set $\gamma_i^C = \gamma_i^C + 1$, and $v_i^S(\gamma_i^C) = u_i(s^*)$ endif

   endif

5. (***Update $\boldsymbol{\eta}$ and $\boldsymbol{\mu}$***) For $i = 1, \ldots, n_x$ do:

   (a) Set $dz = +\infty$, $iz = -\infty$.

   (b) if $(\gamma_i^C > 0)$ and $(\gamma_i^C \neq \gamma_i^L)$

   for $j = \gamma_i^L + 1, \ldots, \gamma_i^C$ do:

   if $(v_i^S(j) \leq dz)$ set $dz = v_i^S(j)$, and $iz = j$ endif

   endif

   (c) Set $\eta_i := dz$ and $\gamma_i^S := iz$.

   (d) Set $dz = -\infty$, $iz = -\infty$.

   (e) if $(\rho_i^C > 0)$ and $(\rho_i^C \neq \rho_i^L)$

   for $j = \rho_i^L + 1, \ldots, \rho_i^C$ do:

   if $(w_i^S(j) \geq dz)$ set $dz = w_i^S(j)$, and $iz = j$ endif

   endif

   (f) Set $\mu_i := dz$ and $\rho_i^S := iz$.

6. (***Update $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$***) Update the values of α_i, β_i for all $i = 1, \ldots, n_x$ by calling the `res0` file.

7. (***Determine transitions***) For $i = 1, \ldots, n_x$ do:

   (a) if $(m_i^L = 2)$

      i. if $(\psi_i \neq \psi_i^S)$ and $(\psi_i = 0)$ and $(\gamma_i^U = \gamma_i^L)$ set $m_i^L := 3$ endif

      ii. if $(\theta_i = 1)$ set $m_i^L := 1$ endif

   else if $(m_i^L = 3)$

      i. if $(\alpha_i = 1)$ set $m_i^L := 2$ endif

      ii. if $(\theta_i = 1)$

         if $(u_i' \geq 0)$ set $m_i^L := 2$ else set $m_i^L := 1$ endif

      endif

   endif

   (b) if $(m_i^L = 1)$ and $(\phi_i \neq \phi_i^S)$ and $(\phi_i = 0)$ set $m_i^L := 2$ endif

   (c) if $(m_i^U = 2)$

      i. if $(\psi_i \neq \psi_i^S)$ and $(\psi_i = 1)$ and $(\rho_i^U = \rho_i^L)$ set $m_i^U := 3$ endif

      ii. if $(\lambda_i = 1)$ set $m_i^U := 1$ endif

   else if $(m_i^U = 3)$

      i. if $(\beta_i = 1)$ set $m_i^U := 2$ endif

      ii. if $(\lambda_i = 1)$

         if $(u_i' \leq 0)$ set $m_i^U := 2$ else set $m_i^U := 1$ endif

      endif

   endif

   (d) if $(m_i^U = 1)$ and $(\phi_i \neq \phi_i^S)$ and $(\phi_i = 1)$ set $m_i^U := 2$ endif

8. (***Transition functions***) For $i = 1, \ldots, n_x$ do:

   (a) if $(m_i^L = 1)$ set $v_i(s^*) := u_i(s^*)$ endif

(b) if ($m_i^U = 1$) set $w_i(s^*) := u_i(s^*)$ endif

9. (***Update storage and counters***) Set $\gamma_i^U := \gamma_i^S$, $\rho_i^U := \rho_i^S$, $\phi_i^S := \phi_i$, $\psi_i^S := \psi_i$ for all $i = 1, \ldots, n_x$. Goto Step 2.

Algorithm 4.46 will provide the exact bounds for Theorem 4.44 subject to numerical tolerance in the algorithms for integration and event detection (discontinuity handling). Note that step 7. in Algorithm 4.46 contains provisions for handling multiple (and instantaneous) transitions after event detection. This is needed because there is the possibility that multiple transitions are triggered (these could be in any time order) in the event time polishing step. This is particularly relevant for the algorithm, because the determination of the wrong mode after event detection will produce qualitatively invalid bounds.

Consider the lower or upper bounding system. For any $i \in \{1, \ldots, n_x\}$, there are 8 possible scenarios for multiple transitions. Consider the possible mode sequences 2, 1, 2 and 3, 1, 2 where there is an instantaneous transition (or an epoch with a very short duration such that it appears to be instantaneous after event polishing) from Mode 1 to Mode 2. This scenario is accounted for by placing the test for Mode 1 after the tests for Modes 2 and 3 in step 7(b). This detects any instantaneous events from Mode 1 to Mode 2 that occur during event polishing. In addition, for the sequence 3, 1, 2, an additional test on u_i' is added in steps 7.a and 7.c to ensure that the correct instantaneous transition to Mode 2 is taken depending on the value of u_i' at the transition. Such a condition is not needed for the sequence 2, 1, 2 because the first transition from Mode 2 to Mode 1 requires that the aforementioned test on u_i' is always satisfied.

Consider now the possible mode sequences 3, 2, 1 and 2, 3, 1 where there is an instantaneous transition (or an epoch with a very short duration such that it appears to be instantaneous after event polishing) from Mode 2 to Mode 1 and Mode 3 to Mode 1 respectively. This scenario is accounted for by the precedence relations stating that Mode 1 has priority over the other modes in the event that multiple transitions become true at the same time, and implemented in step 7.a.ii.

Consider next the possible mode sequences 2, 3, 2 and 3, 2, 3 where there is an instantaneous transition (or an epoch with a very short duration such that it appears to be instantaneous after event polishing) from Mode 3 to Mode 2 and Mode 2 to Mode 3 respectively. This scenario is accounted for by construction; because the zero event counters are updated in step 4 before the transitions are determined in step 7, the second mode in the sequences (Modes 3 and 2 respectively) effectively gets ignored during step 7, which is the desired behavior.

Finally, consider the possible mode sequences 1, 2, 1 and 1, 2, 3 where there is an instantaneous transition (or an epoch with a very short duration such that it appears to be instantaneous after event polishing) from Mode 2 to Mode 1 and Mode 2 to Mode 3 respectively. This situation does not happen unless (i) the interval $[\tau^L, \tau^U]$ for (4.23) is degenerate, or (ii) the width of the interval $[\tau^L, \tau^U]$ for (4.23) is so small that the duration of Mode 2 is so small such that it occurs within the event polishing time step. In the former case, the algorithm cannot be expected to perform correctly due to numerical integration and event handling tolerances; however, the algorithm is redundant for a problem with a degenerate interval $[\tau^L, \tau^U]$ in the first place. In the latter case, the algorithm will fail, again due to numerical integration and event handling tolerances. This can be mitigated by reducing said tolerances. We can calculate the smallest possible duration in the latter case as follows: let the earliest zero event point (excluding possible points at $s = 0$) of u_i' for all $i = 1, \ldots, n_x$ occur at time s^U where $s^U > 0$. From Lemma 4.28, the point in (original) time that this occurs at is given by $t^* = \sigma + s^U(\tau^U - \sigma)$. The corresponding zero event point for q_i' is then given by $t^* = \sigma + s^L(\tau^L - \sigma)$, thus the minimum duration for Mode 2 is given by $s^L - s^U = s^U(\tau^U - \tau^L)/(\tau^L - \sigma)$ if $\tau^L > \sigma$ (this scenario poses no problem for the algorithm if $\tau^L = \sigma$). The algorithm will very likely produce incorrect results if this value of $s^U(\tau^U - \tau^L)/(\tau^L - \sigma)$ is less than the integration and event detection tolerances (in general, a rough recommendation would be to ensure that the said tolerances are at least an order of magnitude smaller than this minimum duration quantity). In any practical implementation of Algorithm 4.46, this condition can be detected, and the user alerted to the possible failure of the algorithm. It is worth

noting that Algorithm 4.46 will not have any problems with this scenario given the ability to integrate and detect events with infinite precision. However, as we will illustrate in the examples below, Algorithm 4.46 can produce reliable results even with finite precision algorithms. In addition, for any practical implementation of Algorithm 4.46, it is easy to detect whether u_i or q_i crosses the calculated bounds of v_i and w_i for all $i = 1, \ldots, n_x$, and inform the user of failure should that occur.

Clearly, the extension of Algorithm 4.46 to LTI multi-stage systems is simply given by Alrogithm 4.36 with Theorem 4.34 replaced with Theorem 4.44.

4.3.5 A Comparison of the Different Strategies

In this section, we will illustrate the application of Corollary 4.24, Theorem 4.34 and Algorithm 4.46 to a few illustrative examples. The integrator used in this section was `DAEPACK` with `DSL48SE` with the value of the absolute and relative tolerance of the integrator and ε_s set to 10^{-8}. Consider first the following illustrative example from the previous section (and used in Figures 4-14 and 4-15):

$$\dot{x}_1 = -x_2, \qquad \dot{x}_2 = x_1, \qquad \mathbf{x}(\sigma = 0) = (1, -1),$$

where $\tau \in [\tau^L, \tau^U]$. The transformed system then becomes

$$y_1' = -\tau y_2, \qquad y_2' = \tau y_1, \qquad \mathbf{y}(\tau, \sigma = 0) = (1, -1),$$

where $\tau \in [\tau^L, \tau^U]$ and $s \in [0, 1]$. If we were to apply Corollary 4.24 derived from the theory of differential inequalities to bound the transformed system, with $\hat{\mathcal{X}}_1(s) = \hat{\mathcal{X}}_2(s) = \mathbb{R}$ for all $s \in [0, 1]$, we would obtain the following nonlinear bounding system,

$$\begin{aligned}
\tilde{v}_1' &= \min(-\tau^L \tilde{v}_2, -\tau^U \tilde{v}_2, -\tau^L \tilde{w}_2, -\tau^U \tilde{w}_2), \\
\tilde{v}_2' &= \min(\tau^L \tilde{v}_1, \tau^U \tilde{v}_1, \tau^L \tilde{w}_1, \tau^U \tilde{w}_1), \\
\tilde{w}_1' &= \max(-\tau^L \tilde{v}_2, -\tau^U \tilde{v}_2, -\tau^L \tilde{w}_2, -\tau^U \tilde{w}_2), \\
\tilde{w}_2' &= \max(\tau^L \tilde{v}_1, \tau^U \tilde{v}_1, \tau^L \tilde{w}_1, \tau^U \tilde{w}_1),
\end{aligned}$$

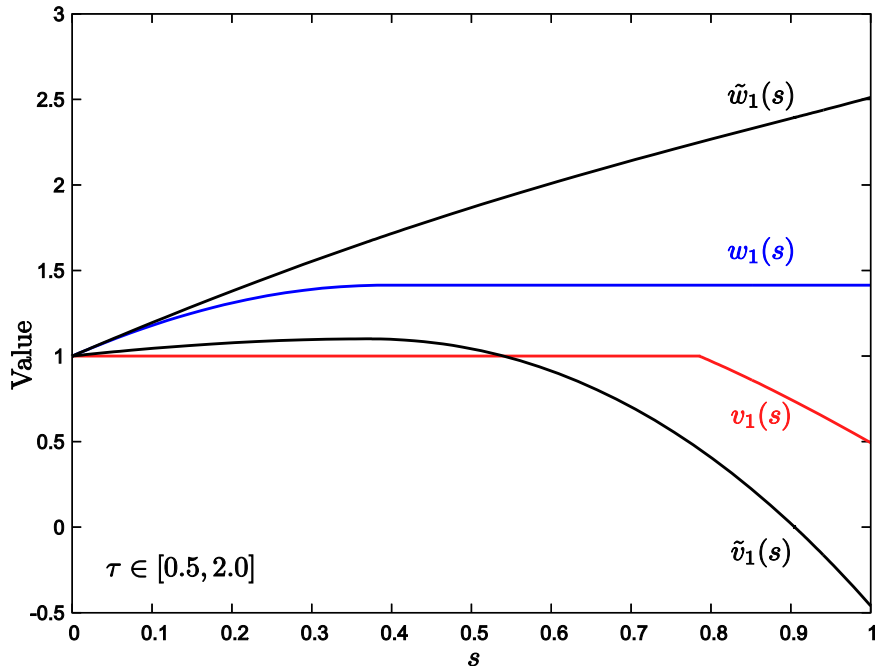


Figure 4-16: Comparison of monotonic hybrid bounds versus nonlinear bounds for $y_1(\tau, s)$.

$\tilde{\mathbf{v}}(0) = \tilde{\mathbf{w}}(0) = (-1, 1)$. These nonlinear bounds will bound the transformed system,

$$\tilde{\mathbf{v}}(s) \leq \mathbf{y}(\tau, s) \leq \tilde{\mathbf{w}}(s), \ \forall \tau \in [\tau^L, \tau^U], s \in [0, 1].$$

Applying Theorem 4.34, we can also construct our monotonic bounding hybrid systems, such that

$$\mathbf{v}(s) - \boldsymbol{\varepsilon}_s \leq \mathbf{y}(\tau, s) \leq \mathbf{w}(s) + \boldsymbol{\varepsilon}_s, \ \forall \tau \in [\tau^L, \tau^U], s \in [0, 1].$$

Figure 4-16 shows the comparison between the two bounds, for $[\tau^L, \tau^U] = [0.5, 2.0]$. As can be seen, the upper bound from the monotonic hybrid bounds is tighter than the nonlinear bounds, whereas the lower bound is tighter for a portion of the transformed time horizon. To illustrate that the monotonic hybrid bounds actually bound, Figure 4-17 shows the bounds with 20 random trajectories of $\tau \in [0.5, 2]$.

Next, we examine how the situation changes when τ^U is increased to 20. Figure 4-18 shows the monotonic hybrid bounds with 20 random trajectories of $\tau \in [0.5, 20]$.

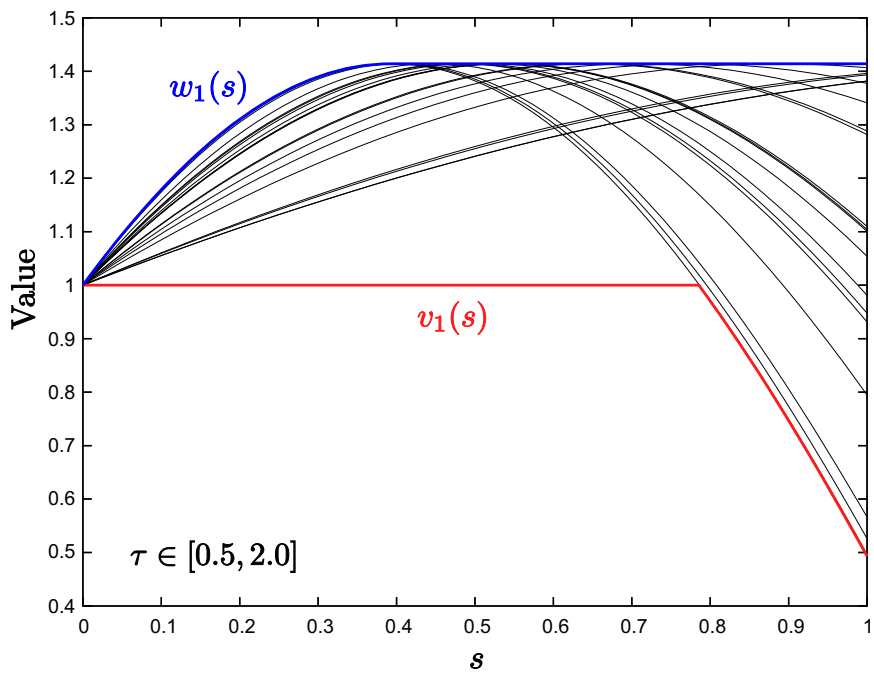


Figure 4-17: Monotonic hybrid bounds for $y_1(\tau, s)$ with 20 random trajectories of $\tau \in [0.5, 2]$.

Note that these monotonic hybrid bounds are really quite tight. For comparison, Figure 4-19 shows the nonlinear bounds for $\tau \in [0.5, 20]$. Note that the bounds obviously explode.

If no additional insight is given to bound the system, eventually, as τ^U increases, there reaches a point where the integrator gives up and fails because of the explosion of the bounds. This explosion of the bounds arises due to the fact that the differential inequalities from Corollary 4.23 do not provide tight bounds for general nonlinear systems. Note that the use of the natural interval extensions in applying Corollary 4.24 provides the exact minimum and maximum for conditions (i), (ii), (iii) and (iv) in Corollary 4.23 for this example, thus the loose bounds arise solely due to the nature of differential inequalities. This effect should not be confused with the *wrapping effect* associated with the use of interval methods (often utilizing Taylor series expansions) for the enclosure of the solution of ordinary differential equations, which was first coined by Moore [98].

For comparison, suppose that we knew, a priori, that $\mathbf{y}(\tau, s) \in [-1.5, 1.5]$ $\forall \tau \in [\tau^L, \tau^U], s \in [0, 1]$, we can then set $\hat{\mathfrak{X}}^{(1)}(s) = \hat{\mathfrak{X}}^{(2)}(s) = [-1.5, 1.5]$ for all $s \in [0, 1]$.

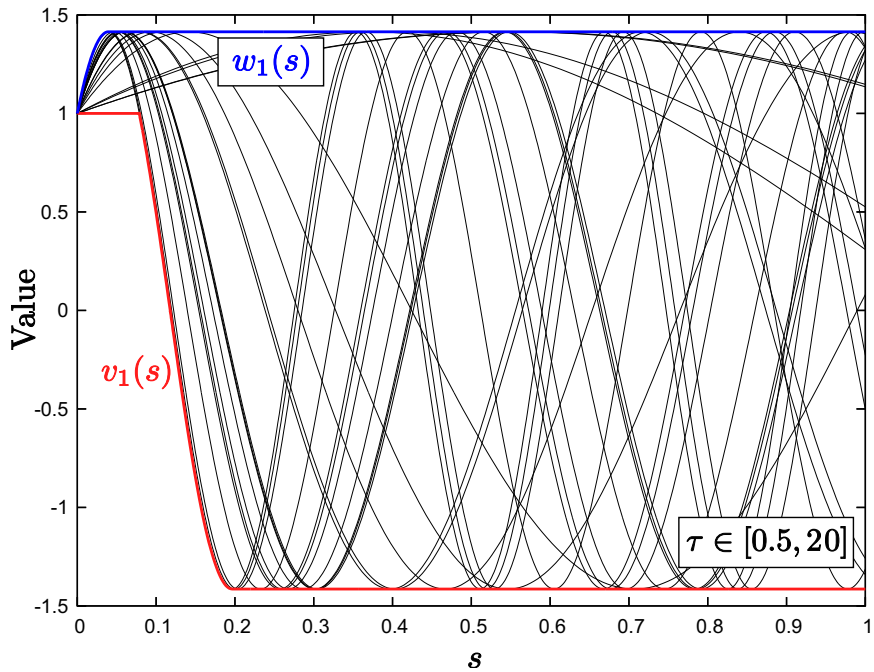


Figure 4-18: Monotonic hybrid bounds for $y_1(\tau, s)$ with 20 random trajectories of $\tau \in [0.5, 20]$.

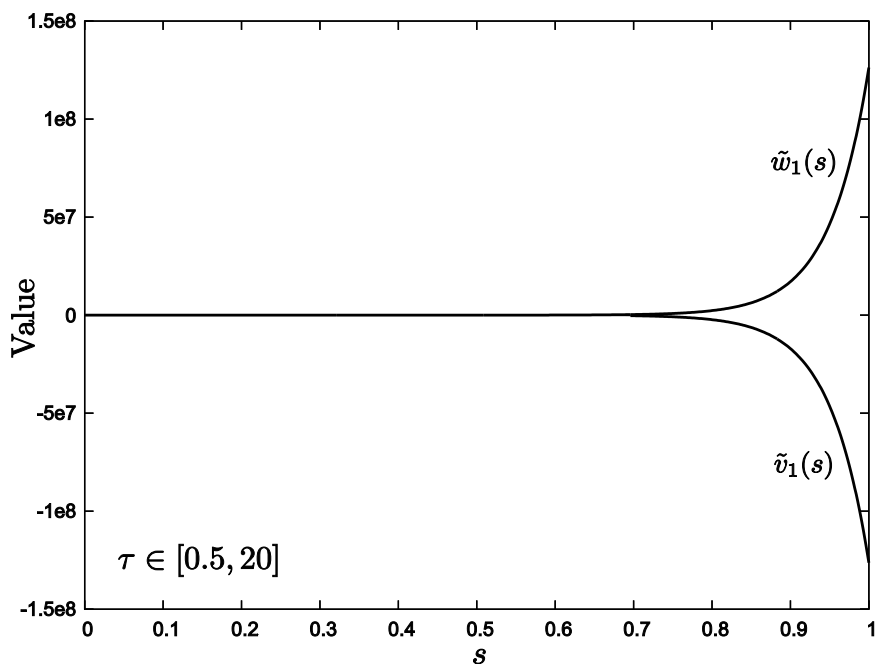


Figure 4-19: Nonlinear bounds for $\tau \in [0.5, 20]$.

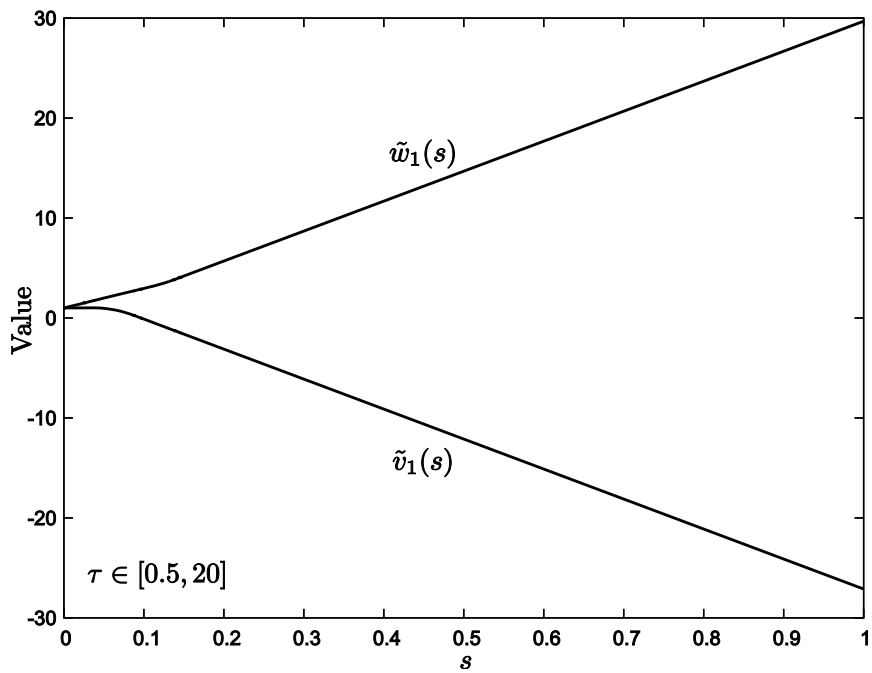


Figure 4-20: Nonlinear bounds with a priori bounding set information for $\tau \in [0.5, 20]$.

Figure 4-20 shows the nonlinear bounds obtained when this information is given. As can be seen, the additional information prevents the bounds from exploding (exponentially), however, these bounds are still quite loose, especially compared to the monotonic hybrid bounds shown in Figure 4-18. Of course, in practice, one would cut off the lower and upper bounds at -1.5 and 1.5 respectively due to information from the bounding set obtained a priori. Figure 4-20 illustrates that the bounds produced from Corollary 4.24 do not improve on what was already known for most of the (transformed) time horizon $s \in [0, 1]$.

It is worth noting that although the monotonic hybrid bounding technique from Theorem 4.34 provides valid bounds for the transformed system, it does not possess the same theoretical convergence properties as the nonlinear bounds from Corollary 4.24 (proof of convergence was shown in Theorem 4.26). In particular, since it does not utilize any information from τ^L, the bounds shown in Figure 4-18 will remain the same, when τ^L is changed, whether it is 0.5 or 19.9999. In other words, it would not be applicable for use in global optimization algorithms, since the bounds do not actually converge to a single trajectory when the partition on τ approaches degeneracy. Due

to this reason, we will no longer consider the use of Theorem 4.34 for the rest of this thesis. However, it is worth noting that these monotonic hybrid bounds are relatively easy to set up, and relatively cheap to obtain, so they can actually serve as the bounding set $\hat{\mathcal{X}}^{(i)}(s)$ for the nonlinear bounding systems in Corollary 4.24 in the absence of bounding sets known a priori. It is also interesting to note that the final values of the monotonic bounds at $s = 1$ also provide the minimum and maximum values that the state trajectories of the original system have attained respectively.

It is interesting to observe how the nonlinear bounds with information from the supplied bounding set would behave when the interval $[\tau^L, \tau^U]$ is small. Figure 4-21 shows the bounds obtained for $\tau \in [19.9999, 20]$. This is a very small interval (or partition, within the BB framework) for τ. Note that the bounds still diverge appreciably when $s > 0.6$. This implies that even with the invariant sets $\hat{\mathcal{X}}_1(s)$ and $\hat{\mathcal{X}}_2(s)$, a global optimization procedure employing these nonlinear bounds will likely be doomed when the original parameter set is $\tau \in [0.5, 20]$, because the convergence rate of these nonlinear bounds requires a partitioning of the parameter set beyond a reasonable integration tolerance (and maybe even machine precision). Clearly, there is a need for a better bounding procedure.

We will now consider applying Algorithm 4.46 to construct the exact bounding hybrid systems, such that

$$\mathbf{v}(s) - \boldsymbol{\varepsilon}_s \leq \mathbf{y}(\tau, s) \leq \mathbf{w}(s) + \boldsymbol{\varepsilon}_s, \ \forall \tau \in [\tau^L, \tau^U], s \in [0, 1].$$

Figure 4-22 shows the exact bounding hybrid systems together with the same 20 random trajectories found in Figure 4-17, for the range $\tau \in [0.5, 2.0]$. As can be seen, the exact bounds for $y_1(\tau, s)$ are obtained. Figure 4-23 shows the exact bounds obtained for the range $\tau \in [18.0, 20.0]$. Clearly, since the system considered is LTI, these bounds obtained from Algorithm 4.46 are the tightest possible bounds within $\boldsymbol{\varepsilon}_s$ (as proved in Theorem 4.45).

To further illustrate the application of Algorithm 4.46, consider the following time

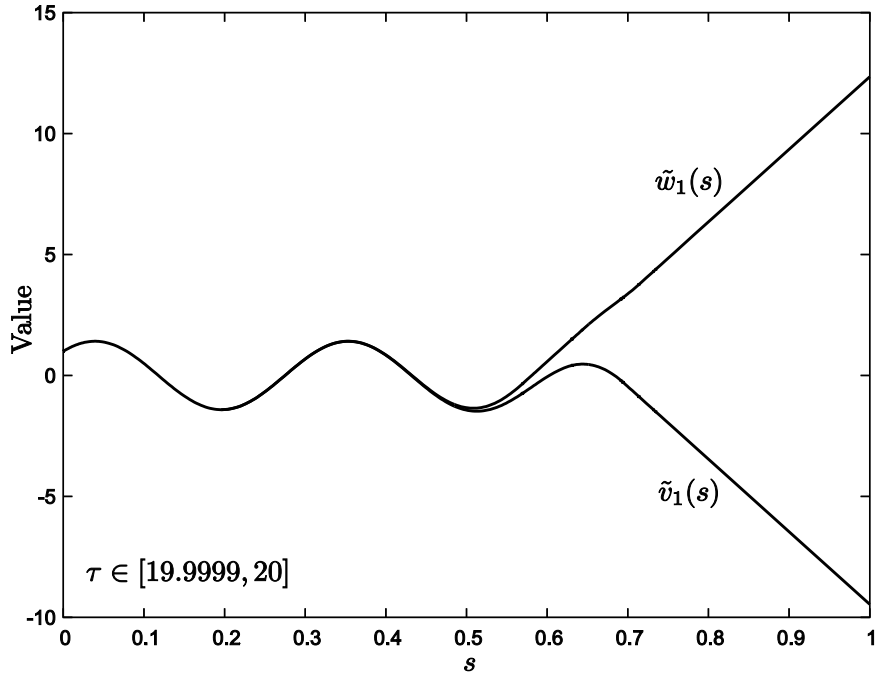


Figure 4-21: Nonlinear bounds with a priori bounding set information for $\tau \in [19.9999, 20]$.

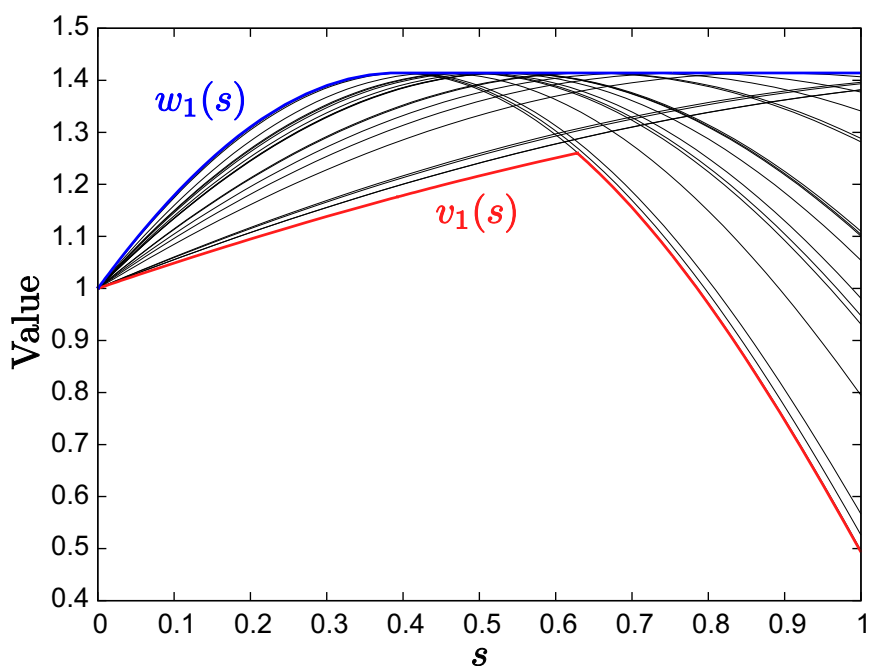


Figure 4-22: Tight hybrid bounds for $y_1(\tau, s)$ with 20 random trajectories of $\tau \in [0.5, 2]$.

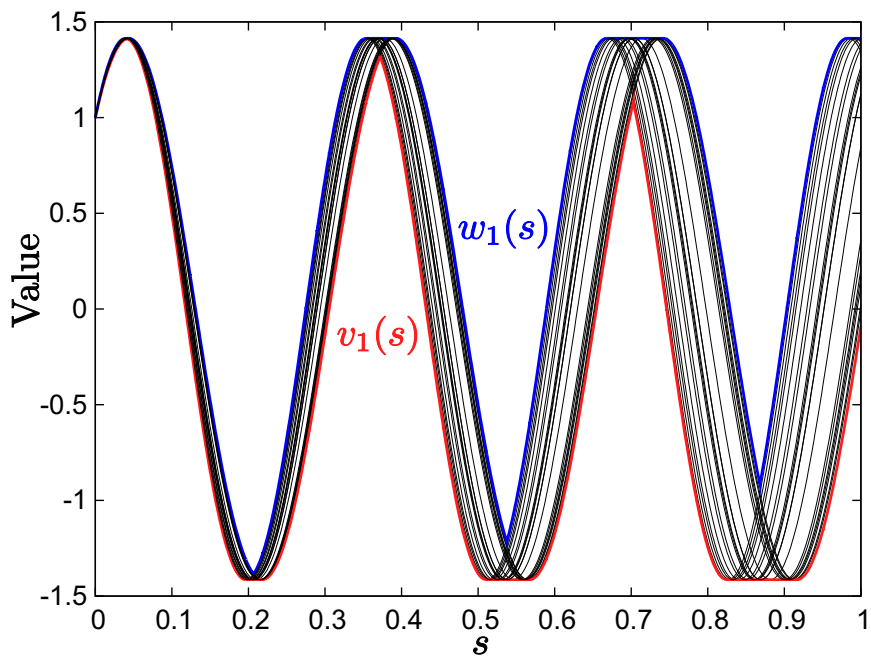


Figure 4-23: Tight hybrid bounds for $y_1(\tau, s)$ with 20 random trajectories of $\tau \in [18.0, 20.0]$.

varying ODE,

$$\dot{x} = \sin(50t)x + \cos(3t) + 0.1, \tag{4.57}$$

where $x(0) = -1$, $\tau \in [10, 12]$. In this case, (4.25) gives $t = \tau s$ since $\sigma = 0$. The transformed systems are then given by

$$q' = 10\sin(500s)q + 10\cos(30s) + 1,$$
$$u' = 12\sin(600s)u + 12\cos(36s) + 1.2,$$

where $q(0) = u(0) = -1$. Figure 4-24 shows the trajectories of the exact bounding hybrid systems, together with $q(s)$ and $u(s)$, while Figure 4-25 show the same trajectories zoomed in on a portion of the plot. This is to illustrate the interactions between the zero event points for q' and u', and their influence on the mode sequences of v and w. To test the validity of Algorithm 4.46, Figure 4-26 and 4-27 show the exact bounding trajectories together with 20 random trajectories for $\tau \in [10, 12]$. As can be seen, v and w are indeed valid bounds.

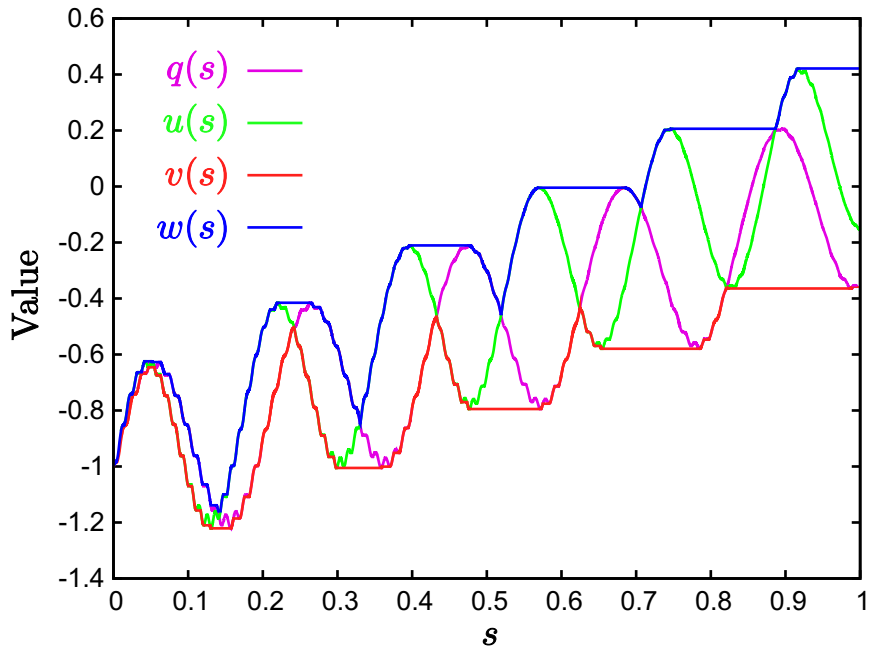


Figure 4-24: Tight hybrid bounds with $y(\tau^L, s) = q(s)$ and $y(\tau^U, s) = u(s)$ for $\tau \in [10, 12]$.

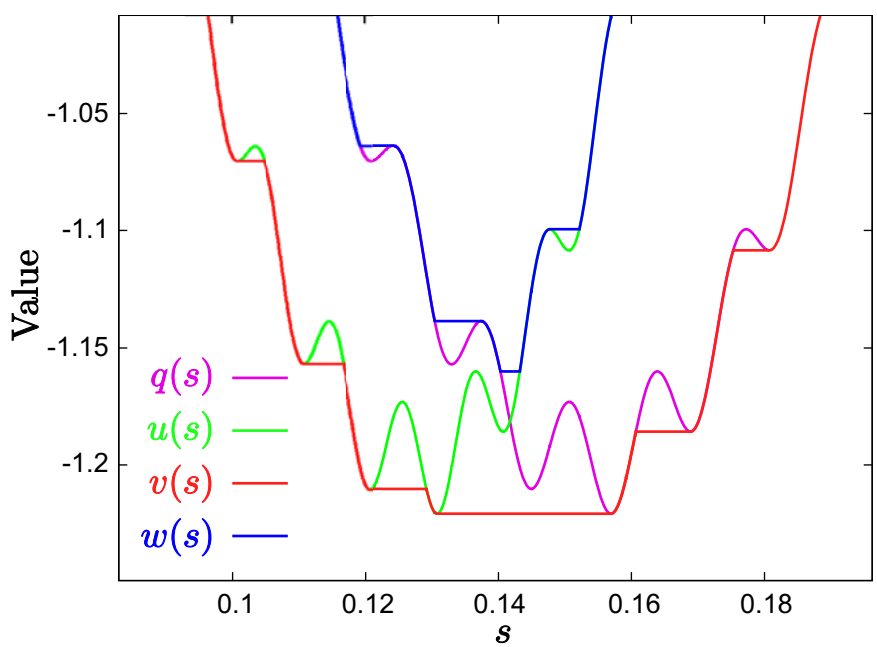


Figure 4-25: Zoomed in portion of Figure 4-24.

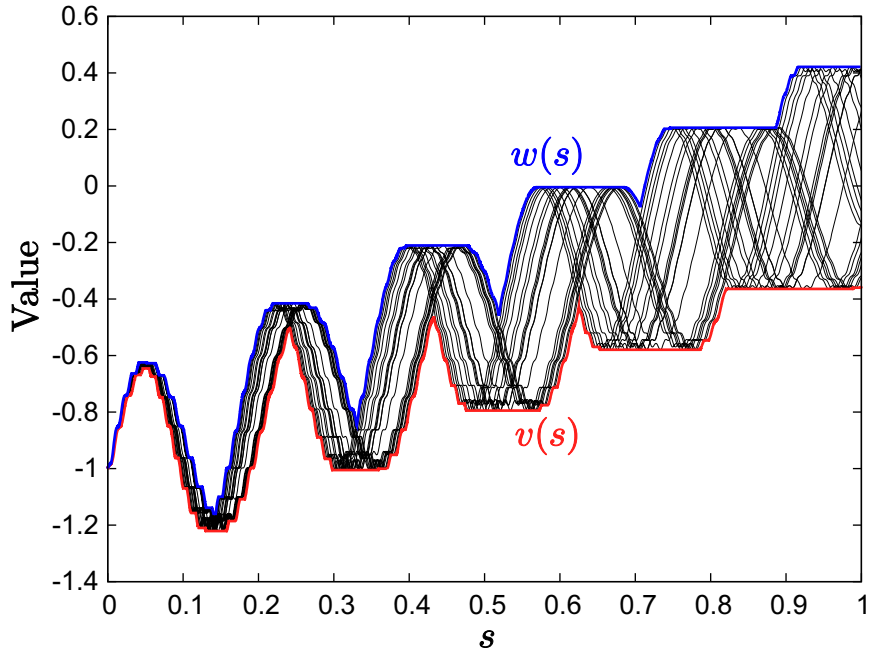


Figure 4-26: Tight hybrid bounds with with 20 random trajectories for $\tau \in [10, 12]$.

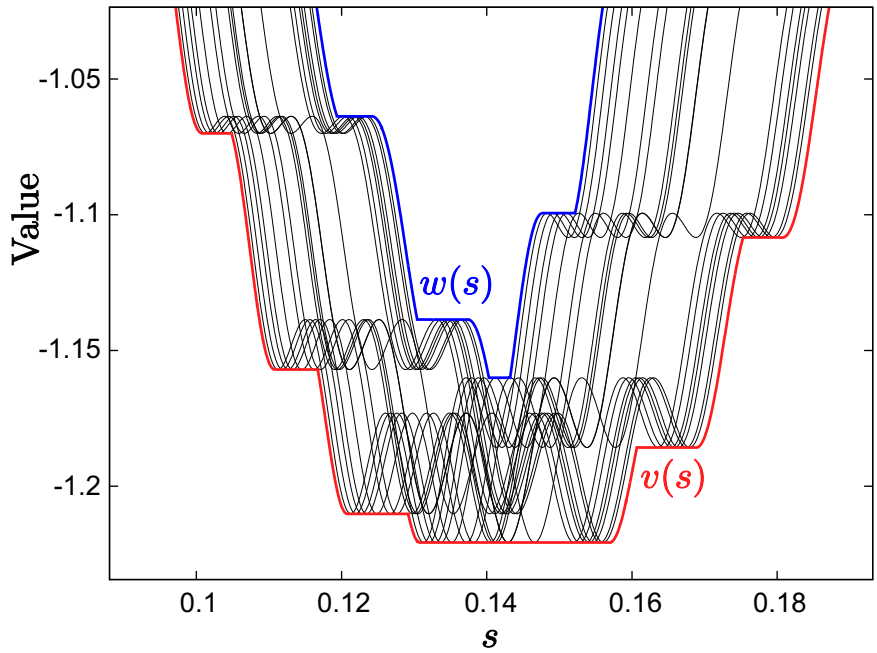


Figure 4-27: Zoomed in portion of Figure 4-26.

For comparison, we will also apply Corollary 4.24 to the system in (4.57). The transformed system (after CPET) is given by

$$y' = \tau(\sin(50\tau s)y + \cos(5\tau s) + 0.1), \tag{4.58}$$

where $y(\tau, 0) = -1$ and $\tau \in [10, 12]$. Let the inclusion monotonic interval extension of $\sin([500s, 600s])$ and $\cos([30s, 36s])$ be given by $[\alpha^L(s), \alpha^U(s)]$ and $[\beta^L(s), \beta^U(s)]$ respectively, where

$$\sin([500s, 600s]) \subset [\alpha^L(s), \alpha^U(s)],$$
$$\cos([30s, 36s]) \subset [\beta^L(s), \beta^U(s)],$$

for all $s \in [0, 1]$. Then, the nonlinear bounding system is given by the following,

$$\begin{aligned}\tilde{v}' = \min(&10(\min(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{v}) + \beta^L(s) + 0.1),\\ &10(\max(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{v}) + \beta^U(s) + 0.1),\\ &12(\min(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{v}) + \beta^L(s) + 0.1),\\ &12(\max(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{v}) + \beta^U(s) + 0.1))\end{aligned}$$

$$\begin{aligned}\tilde{w}' = \max(&10(\min(\alpha^L(s)\tilde{w}, \alpha^U(s)\tilde{v}) + \beta^L(s) + 0.1),\\ &10(\max(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{w}) + \beta^U(s) + 0.1),\\ &12(\min(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{w}) + \beta^L(s) + 0.1),\\ &12(\max(\alpha^L(s)\tilde{v}, \alpha^U(s)\tilde{w}) + \beta^U(s) + 0.1))\end{aligned}$$

where $\tilde{v}(0) = \tilde{w}(0) = -1$. To integrate this nonlinear system, the interval extensions of sin and cos were calculated using the `INTLIB` library [80], and implemented using `DAEPACK` and `DSL48S` in black box mode. Figure 4-28 shows the bounds obtained for the transformed system (4.58). It can be seen that the bounds are orders of magnitude worse than the exact bounds provided by Algorithm 4.46. Figure 4-29 shows the bounds provided by the two strategies together on a zoomed portion of the

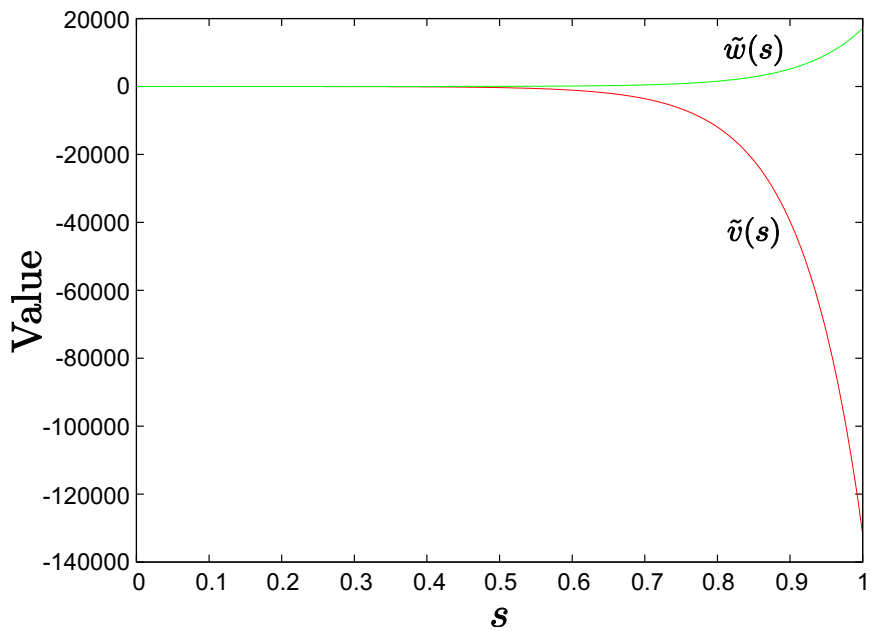


Figure 4-28: Nonlinear bounds for (4.58).

transformed time scale.

Finally, consider the following LTI multi-stage system,

$$\text{Mode 1:} \begin{cases} \dot{x}_1 = -x_2, \\ \dot{x}_2 = x_1, \end{cases} \qquad \text{Mode 2:} \begin{cases} \dot{x}_1 = x_1 - x_2 + 1, \\ \dot{x}_2 = -x_1 + x_2 - 1, \end{cases} \tag{4.59}$$

where $\mathbf{x}(0) = (1, -1)$, the mode sequence is fixed and given by $T_\mu = 1, 2, 1$, the transition functions are given by state continuity, and $\Delta = [10, 12] \times [1, 2] \times [10, 12]$. After CPET and applying Corollary 4.24, the nonlinear bounding systems are given by the following,

$$\text{Mode 1:} \begin{cases} \tilde{v}_1' = \min(-\delta_i^L v_2, -\delta_i^U v_2, -\delta_i^L w_2, -\delta_i^U w_2) \\ \tilde{v}_2' = \min(\delta_i^L v_1, \delta_i^U v_1, \delta_i^L w_1, \delta_i^U w_1) \\ \tilde{w}_1' = \max(-\delta_i^L v_2, -\delta_i^U v_2, -\delta_i^L w_2, -\delta_i^U w_2) \\ \tilde{w}_2' = \max(\delta_i^L v_1, \delta_i^U v_1, \delta_i^L w_1, \delta_i^U w_1) \end{cases}, i = 1, 3,$$

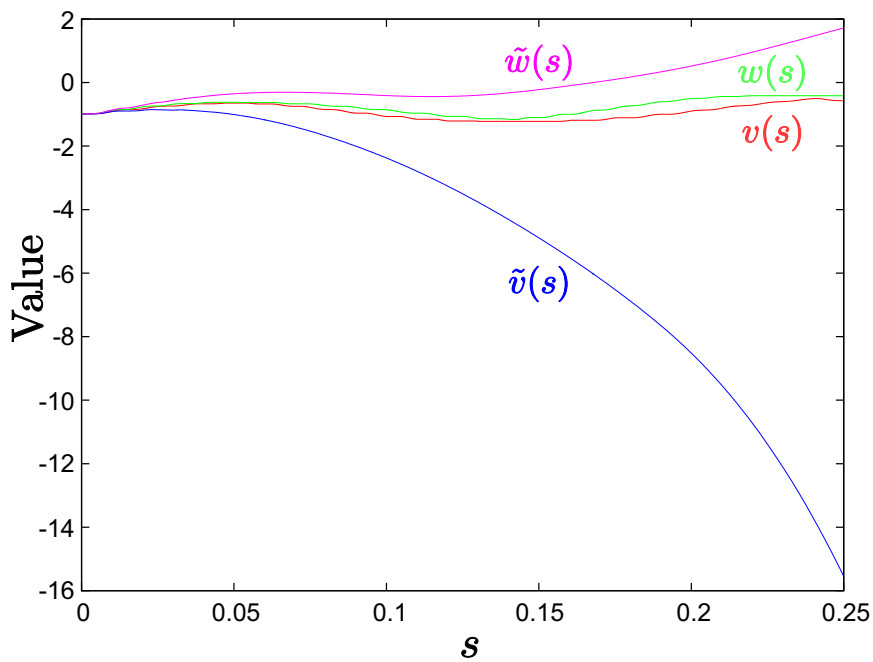


Figure 4-29: Comparison of nonlinear and exact hybrid bounding strategies.

$$\text{Mode 2:} \begin{cases} \tilde{v}_1' = \begin{array}{l} \min(\delta_2^L(v_1 - w_2 + 1), \delta_2^L(v_1 - v_2 + 1), \\ \qquad \delta_2^U(v_1 - w_2 + 1), \delta_2^U(v_1 - v_2 + 1)) \end{array} \\ \tilde{v}_2' = \begin{array}{l} \min(\delta_2^L(v_2 - w_1 - 1), \delta_2^L(v_2 - v_1 - 1), \\ \qquad \delta_2^U(v_2 - w_1 - 1), \delta_2^U(v_2 - v_1 - 1)) \end{array} \\ \tilde{w}_1' = \begin{array}{l} \max(\delta_2^L(w_1 - w_2 + 1), \delta_2^L(w_1 - v_2 + 1), \\ \qquad \delta_2^U(w_1 - w_2 + 1), \delta_2^U(w_1 - v_2 + 1)) \end{array} \\ \tilde{w}_2' = \begin{array}{l} \max(\delta_2^L(w_2 - w_1 - 1), \delta_2^L(w_2 - v_1 - 1), \\ \qquad \delta_2^U(w_2 - w_1 - 1), \delta_2^U(w_2 - v_1 - 1)) \end{array} \end{cases},$$

where $\tilde{\mathbf{v}}(0) = \tilde{\mathbf{w}}(0) = (1, -1)$, the mode sequence is fixed and given by $T_\mu = 1, 2, 1$, the transition functions are given by state continuity, and $\Delta = [10, 12] \times [1, 2] \times [10, 12]$. Figure 4-30 shows the nonlinear bounds for y_1 of (4.59) after the CPET. It can be seen that the bounds explode, as expected. On the other hand, Figure 4-31 shows the bounds for y_1 of (4.59) after the CPET, obtained by applying Algorithm 4.36 with Theorem 4.44 (labeled as $v_1(s)$ and $w_1(s)$), together with 20 random trajectories of the transformed system of (4.59) in Δ. As can be seen, the bounds obtained are orders of magnitude tighter than those shown in Figure 4-30.

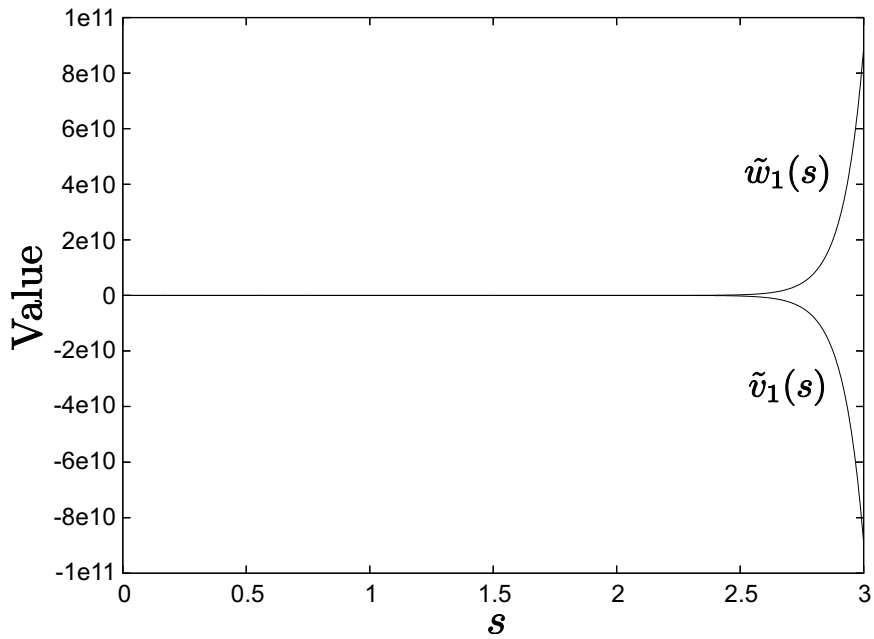


Figure 4-30: Nonlinear bounds for element y_1 of transformed system of (4.59).

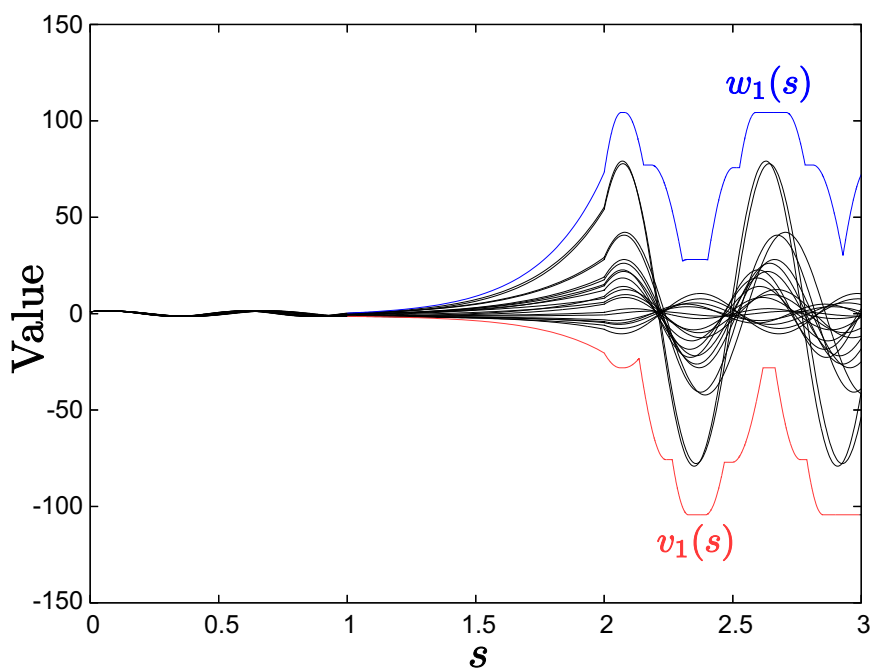


Figure 4-31: Bounds for element y_1 of transformed system of (4.59) using Algorithm 4.36 with Theorem 4.44, and 20 random trajectories of $\boldsymbol{\delta} \in \Delta$.

4.4 Constructing Convex Relaxations

In this section, we will present the theory required for constructing convex relaxations of Problem 4.12. This theory is an extension of that developed in [118, Chapter 6] and [121] for (single-stage) nonlinear dynamic systems. The main idea behind the theorems presented below will consist of breaking down the multi-stage hybrid system into contiguous intervals in time, verifying that the hypotheses of the theorems in [118, Chapter 6] and [121] hold for each of these intervals, and applying the theorems sequentially for each interval via finite induction.

The ultimate goal of this section is condensed into constructing a convex relaxation for the objective function (4.7), subject to the transformed nonlinear hybrid system. The exact same theory is applied for the point and isoperimetric constraints in (4.8). The ability to construct convex relaxations for the objective function and constraints then enables a convex relaxation of the problem to be solved. Finally, it is shown that the constructed convex relaxations possess the same consistent bounding properties of the convex relaxation techniques used in their construction, so that their incorporation into a BB framework such as Algorithm 2.3 leads to an infinitely convergent algorithm [77]. This implies ε global optimality within a finite number of iterations. We will now show how convex and concave relaxations for the states of the transformed hybrid system in Definition 4.10 can be constructed.

Definition 4.47. Consider the following functions, $f : Z \times P \times \Delta \times S \to \mathbb{R}$ and $\mathbf{z} : S \to Z$, where $Z \subset \mathbb{R}^{n_x}$, $P \subset \mathbb{R}^{n_p}$, $\Delta \subset \mathbb{R}^{n_e}$, $S \subset \mathbb{R}$ and $f(\cdot, \underline{s})$ is differentiable on some suitable open set containing $Z \times P \times \Delta$ for each $\underline{s} \in S$. Define the function $\mathcal{L}_f|_{\boldsymbol{\zeta}^*(s)} : Z \times P \times \Delta \times S \to \mathbb{R}$ to be a *linearization* of f at the point $\boldsymbol{\zeta}^*(s) = (\mathbf{z}^*(s), \mathbf{p}^*, \boldsymbol{\delta}^*)$ where $(\mathbf{z}^*(s), \mathbf{p}^*, \boldsymbol{\delta}^*) \in Z \times P \times \Delta$, and given by the following:

$$\mathcal{L}_f|_{\boldsymbol{\zeta}^*(s)}(\mathbf{z}, \mathbf{p}, \boldsymbol{\delta}, s) = f(\mathbf{z}^*, \mathbf{p}^*, \boldsymbol{\delta}^*, s) + \sum_{k=1}^{n_x} \left.\frac{\partial f}{\partial z_k}\right|_{(\boldsymbol{\zeta}^*(s),s)} \Big(z_k(s) - z_k^*(s)\Big) + \sum_{k=1}^{n_p} \left.\frac{\partial f}{\partial p_k}\right|_{(\boldsymbol{\zeta}^*(s),s)} \big(p_k - p_k^*\big) + \sum_{k=1}^{n_p} \left.\frac{\partial f}{\partial \delta_k}\right|_{(\boldsymbol{\zeta}^*(s),s)} \big(\delta_k - \delta_k^*\big).$$

Theorem 4.48. *For $i = 1, \ldots, n_e$ and $j = 1, \ldots, n_x$, define the functions $u_j^{(m_i^*)}(\cdot, \underline{s}) : \hat{X}^{(i)}(s; P, \Delta) \times P \times \Delta \to \mathbb{R}$ and $o_j^{(m_i^*)}(\cdot, \underline{s}) : \hat{X}^{(i)}(s; P, \Delta) \times P \times \Delta \to \mathbb{R}$ for each fixed $\underline{s} \in \hat{I}_i$. Let the following conditions be satisfied for all $i = 1, \ldots, n_e$, $j = 1, \ldots, n_x$, and each fixed $\underline{s} \in \hat{I}_i$,*

1. *$u_j^{(m_i^*)}(\cdot, \underline{s})$ is a convex underestimator and $o_j^{(m_i^*)}(\cdot, \underline{s})$ is a concave overestimator for $\mathcal{F}_j^{(m_i^*)}(\cdot, \underline{s})$ on $\hat{X}^{(i)}(\underline{s}; P, \Delta) \times P \times \Delta$,*

2. *$u_j^{(m_i^*)}(\cdot, \underline{s})$ and $o_j^{(m_i^*)}(\cdot, \underline{s})$ are differentiable on some suitable open set containing $\hat{X}^{(i)}(s; P, \Delta) \times P \times \Delta$ along some reference trajectory $\boldsymbol{\zeta}^*(s) = (\mathbf{z}^*(s), \mathbf{p}^*, \boldsymbol{\delta}^*) \in \hat{X}^{(i)}(s; P, \Delta) \times P \times \Delta$,*

and the following hybrid system be constructed,

$$c_j' = h_{c,j}^{(m_i^*)}(\mathbf{c}, \mathbf{C}, \mathbf{p}, \boldsymbol{\delta}, s) = \inf_{\substack{\mathbf{z} \in \mathcal{C}(\mathbf{p}, \boldsymbol{\delta}, s) \\ z_j = c_j(s)}} \mathcal{L}_{u_j^{(m_i^*)}} \Big|_{(\boldsymbol{\zeta}^*(s), s)} (\mathbf{z}, \mathbf{p}, \boldsymbol{\delta}, s), \;\; s \in [i-1, i],$$

$$C_j' = h_{C,j}^{(m_i^*)}(\mathbf{c}, \mathbf{C}, \mathbf{p}, \boldsymbol{\delta}, s) = \sup_{\substack{\mathbf{z} \in \mathcal{C}(\mathbf{p}, \boldsymbol{\delta}, s) \\ z_j = C_j(s)}} \mathcal{L}_{o_j^{(m_i^*)}} \Big|_{(\boldsymbol{\zeta}^*(s), s)} (\mathbf{z}, \mathbf{p}, \boldsymbol{\delta}, s), \;\; s \in [i-1, i],$$

with initial conditions

$$\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, 0) = \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, 0) = \mathbf{E}_0 \mathbf{p} + \mathbf{J}_0 \boldsymbol{\delta} + \mathbf{k}_0, \tag{4.60}$$

and transition functions given by the following interval equation,

$$[\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \hat{\sigma}_{l+1}), \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \hat{\sigma}_{l+1})] = \mathbf{D}_l [\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \hat{\tau}_l), \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \hat{\tau}_l)] + \mathbf{E}_l \mathbf{p} + \mathbf{J}_l \boldsymbol{\delta} + \mathbf{k}_l, \tag{4.61}$$

for $l = 1, \ldots, n_e - 1$, where $\mathcal{C}(\mathbf{p}, \boldsymbol{\delta}, s) = \{\mathbf{z} \mid \mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, s) \leq \mathbf{z} \leq \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, s)\}$. Then, for each fixed $\underline{s} \in \hat{I}_i$, $\mathbf{c}(\cdot, \underline{s})$ is a convex underestimator and $\mathbf{C}(\cdot, \underline{s})$ is a concave overestimator for $\hat{\mathbf{x}}(\cdot, \underline{s})$ on $P \times \Delta$, for all $i = 1, \ldots, n_e$.

Proof. We proceed as in the proof of Theorem 4.21 by subdividing the epochs into contiguous subepochs. Consider now the first subepoch $\check{I}_1$. The initial condition given by (4.60) is clearly affine on $P \times \Delta$ and satisfies $\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, 0) \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, 0) \leq \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, 0)$.

The conditions for [118, Theorem 6.16] are thus satisfied, and applying said theorem, $\mathbf{c}(\cdot, \underline{s})$ is a convex underestimator and $\mathbf{C}(\cdot, \underline{s})$ is a concave overestimator for $\hat{\mathbf{x}}(\cdot, \underline{s})$ for each fixed $\underline{s} \in \check{I}_1$. At the transition $\check{\tau}_1$, state continuity of the hybrid system gives $\hat{x}(\mathbf{p}, \boldsymbol{\delta}, \check{\sigma}_2) = \hat{x}(\mathbf{p}, \boldsymbol{\delta}, \check{\tau}_1)$, which implies that

$$\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \check{\sigma}_2) = \mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \check{\tau}_1) \leq \hat{x}(\mathbf{p}, \boldsymbol{\delta}, \check{\sigma}_2) \leq \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \check{\tau}_1) = \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \check{\sigma}_2), \ \forall (\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta.$$

From [118, Theorem 6.16], we know that $\mathbf{c}(\cdot, \check{\sigma}_2)$ and $\mathbf{C}(\cdot, \check{\sigma}_2)$ are affine in $(\mathbf{p}, \boldsymbol{\delta})$. The conditions for [118, Theorem 6.16] are thus satisfied for the second subepoch. By induction on all subepochs, the desired result holds for each fixed $\underline{s} \in \hat{I}_1$. Consider now the second epoch $\hat{I}_2$. From (4.61),

$$\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \hat{\sigma}_2) \leq \hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \hat{\sigma}_2) \leq \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \hat{\sigma}_2) \forall (\mathbf{p}, \boldsymbol{\delta}) \in P \times \Delta,$$

where $\mathbf{c}(\cdot, \hat{\sigma}_2)$ and $\mathbf{C}(\cdot, \hat{\sigma}_2)$ are clearly affine in $(\mathbf{p}, \boldsymbol{\delta})$. The conditions for [118, Theorem 6.16] are thus satisfied for the second epoch, and by induction on all epochs, we obtain the desired result. □

Note that the infima and suprema in Theorem 4.48 are attained at the vertices of the set $\mathcal{C}(\mathbf{p}, \boldsymbol{\delta}, s)$ due to the properties of the linearizations, and are easily computed, see [118, Theorem 6.16]. The next theorem demonstrates the convergence properties of the convex relaxations constructed using the relaxation techniques presented in this section.

Theorem 4.49. *Consider the following convex relaxation of* (4.7),

$$\hat{U}(\mathbf{p}, \boldsymbol{\delta}; P, \Delta) = \sum_{i=1}^{n_e} \left\{ \sum_{j=1}^{n_{\phi i}} \hat{\psi}_{ij} \Big(\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \hat{\alpha}_{ij}), \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \hat{\alpha}_{ij}), \mathbf{p}, \boldsymbol{\delta}; \hat{X}^{(i)}(\hat{\alpha}_{ij}; P, \Delta), P, \Delta \Big) + \right.$$
$$\left. \int_{i-1}^{i} \hat{u}_i \Big(\mathbf{c}, \mathbf{C}, \mathbf{p}, \boldsymbol{\delta}, s; \hat{X}^{(i)}(s; P, \Delta), P, \Delta \Big) \mathrm{d}s \right\}, \quad (4.62)$$

where $\hat{\psi}_{ij}$ and $\hat{u}_i$ are constructed using any relaxation technique that possesses a consistent bounding operation [77, Definition IV.4, pg. 128], the convex and concave

relaxations for the state and derivatives are constructed using Theorem 4.48, and the estimation of the state bounds constructed using Corollary 4.24. If the interval vector (P_k, Δ_k) in any partition on $P \times \Delta$ approaches degeneracy $(P^*, \Delta^*) = ([\mathbf{p}^*, \mathbf{p}^*], [\boldsymbol{\delta}^*, \boldsymbol{\delta}^*]) \in P \times \Delta$, then the lower bound on this partition $\hat{U}(\mathbf{p}, \boldsymbol{\delta}; P_k, \Delta_k)$ converges pointwise to the objective function value $\hat{F}(\mathbf{p}^*, \boldsymbol{\delta}^*)$ in this same partition.

Proof. Choose any arbitrary partition and any fixed $\underline{s}$ in any epoch $\hat{I}_i$. From Corollary 4.24, as $(P_k, \Delta_k) \to (P^*, \Delta^*)$, the interval vector $\hat{X}_k^{(m_i^*)}(\underline{s}; P_k, \Delta_k)$ approaches the degenerate value of $\hat{\mathbf{x}}^*(\mathbf{p}^*, \boldsymbol{\delta}^*, \underline{s})$. To be valid, the convex and concave overestimators ($u_j^{(m_i^*)}$ and $o_j^{(m_i^*)}$) from Theorem 4.48 must themselves possess a consistent bounding operation. Hence, as $\hat{X}_k^{(m_i^*)}(\underline{s}; P_k, \Delta_k) \times P_k \times \Delta_k$ shrinks to degeneracy, $u_j^{(m_i^*)}(\cdot, \underline{s}) \uparrow \mathcal{F}_j^{(m_i^*)}(\cdot, \underline{s})$ and $o_j^{(m_i^*)}(\cdot, \underline{s}) \downarrow \mathcal{F}_j^{(m_i^*)}(\cdot, \underline{s})$ for $j = 1, \ldots, n_x$. The right hand sides of the equations defining c_i' and C_i' are linearizations on $u_j^{(m_i^*)}$ and $o_j^{(m_i^*)}$ respectively. Since $u_j^{(m_i^*)}(\cdot, \underline{s})$ and $o_j^{(m_i^*)}(\cdot, \underline{s})$ are each approaching $\mathcal{F}_j^{(m_i^*)}(\cdot, \underline{s})$, $h_{c,j}^{(m_i^*)}(\cdot, \underline{s}) \uparrow \mathcal{F}_j^{(m_i^*)}(\cdot, \underline{s})$ and $h_{C,j}^{(m_i^*)}(\cdot, \underline{s}) \downarrow \mathcal{F}_j^{(m_i^*)}(\cdot, \underline{s})$ because the linearization approaches the value of the function it approximates at the point of linearization. Thus, as $k \to \infty$,

$$\hat{\psi}_{ij}\Big(\mathbf{c}(\mathbf{p}, \boldsymbol{\delta}, \hat{\alpha}_{ij}), \mathbf{C}(\mathbf{p}, \boldsymbol{\delta}, \hat{\alpha}_{ij}), \mathbf{p}, \boldsymbol{\delta}; \hat{X}^{(i)}(\hat{\alpha}_{ij}; P_k, \Delta_k), P_k, \Delta_k\Big) \uparrow \phi_{ij}\Big(\hat{\mathbf{x}}(\mathbf{p}, \boldsymbol{\delta}, \hat{\alpha}_{ij}), \mathbf{p}, \boldsymbol{\delta}\Big), \tag{4.63}$$

for all $j = 1, \ldots, n_{\phi i}$, and $\hat{u}_i\Big(\mathbf{c}, \mathbf{C}, \boldsymbol{\delta}, \underline{s}; \hat{X}^{(i)}(s; P_k, \Delta_k), \mathbf{p}, P_k, \Delta_k\Big) \uparrow f_i(\hat{\mathbf{x}}, \mathbf{p}, \boldsymbol{\delta}, \underline{s})$, for all $\underline{s} \in \hat{I}_i$, where the convergence arises because the convex relaxations $\hat{\psi}_{ij}$ and $\hat{u}_i$ possess consistent bounding operations as (P_k, Δ_k) approaches degeneracy. Because $\underline{s}$ is fixed arbitrarily, the convergence for the integrand is true for all $\underline{s} \in \hat{I}_i$. An application of the monotone convergence theorem [116, Theorem 11.28] for the integral term then gives

$$\lim_{k \to \infty} \int_{i-1}^{i} \hat{u}_i\Big(\mathbf{c}, \mathbf{C}, \mathbf{p}, \boldsymbol{\delta}, s; \hat{X}^{(i)}(s; P_k, \Delta_k), P_k, \Delta_k\Big)\, \mathrm{d}s = \int_{i-1}^{i} f_i\Big(\hat{\mathbf{x}}, \mathbf{p}^*, \boldsymbol{\delta}^*, s\Big)\, \mathrm{d}s. \tag{4.64}$$

Since the partition and epoch was arbitrarily chosen, (4.63) and (4.64) imply

$$\lim_{k \to \infty} \hat{U}(\mathbf{p}, \boldsymbol{\delta}; P_k, \Delta_k) = \hat{F}(\mathbf{p}^*, \boldsymbol{\delta}^*)$$

which is the desired result. $\square$

4.5 Examples and Discussion

All of the results in this section were obtained using a Pentium 4 3.4 GHz machine with 1 GB RAM running SuSE Linux 9.2.

Example 4.50. Consider Example 3.7. It has now been determined that catalyst 1, 2 and 3 will be loaded in that order into the reactor. The optimization problem is now to determine the optimal lengths of the 3 catalyst sections to maximize the same objective function as in Example 3.7.

Clearly, we now have a problem where the mode sequence is fixed, but the transition times are allowed to vary. The optimization problem is to determine the optimal transition times (the independent variable in the example above is the length of the reactor). After applying the CPET, the transformed problem is given by

$$\min_{\boldsymbol{\delta}\in\Delta} \; 0.01\hat{x}_2(\boldsymbol{\delta},3) + 0.1\hat{x}_4(\boldsymbol{\delta},3) - \hat{x}_5(\boldsymbol{\delta},3),$$

subject to the following point constraint,

$$\delta_1 + \delta_2 + \delta_3 = 1, \tag{4.65}$$

where $\hat{\mathbf{x}}(\boldsymbol{\delta}, s)$ is given by the solution of the following nonlinear hybrid system,

$$\text{Mode } i\text{:} \left\{ \begin{array}{rcl} \hat{x}_1'(s) & = & \delta_i\Big(-(k_1^{(i)} + k_2^{(i)})\hat{x}_1(\boldsymbol{\delta}, s)\Big) \\ \hat{x}_2'(s) & = & \delta_i\Big(k_2^{(i)}\hat{x}_1(\boldsymbol{\delta}, s)\Big) \\ \hat{x}_3'(s) & = & \delta_i\Big(k_1^{(i)}\hat{x}_1(\boldsymbol{\delta}, s) - (k_3^{(i)} + k_4^{(i)})\hat{x}_3(\boldsymbol{\delta}, s)\Big) \\ \hat{x}_4'(s) & = & \delta_i\Big(k_4^{(i)}\hat{x}_3(\boldsymbol{\delta}, s)\Big) \\ \hat{x}_5'(s) & = & \delta_i\Big(k_3^{(i)}\hat{x}_3(\boldsymbol{\delta}, s)\Big) \end{array} \right\} i = 1, 2, 3,$$

$n_e = 3$, $T_\mu = 1, 2, 3$, $T_\tau = \{\hat{I}_i\}$ where $\hat{I}_i = [i-1, i]$ for $i = 1, 2, 3$, $\Delta \equiv [0, 1]^3$, $k_j^{(i)}$ is

the rate constant k_j for catalyst i in Figure 3-1, and we have state continuity as the transition functions with initial condition $\hat{\mathbf{x}}(\boldsymbol{\delta}, 0) = (1000, 0, 0, 0, 0)$.

The convex relaxations for this example are constructed directly using Theorem 4.48 as the original objective function comprises an affine function of the state variables at the final time. Since the right hand sides of the nonlinear hybrid system exhibit a bilinear structure, the convex and concave relaxations in Theorem 4.48 can be calculated from the convex envelope of a bilinear term [54]. However, since the convex envelope is composed of two intersecting hyperplanes and thus not continuously differentiable everywhere, there is no guarantee that condition C2 will be satisfied for a particular choice of a reference trajectory. Fortunately, it is clear that the condition can be relaxed to accommodate the nonsmoothness in the intersection of the two hyperplanes by constructing the linearizations using any subgradient at the point of nonsmoothness. In practice, we have implemented a heuristic that either chooses one or the other hyperplane (which are both valid convex relaxations and supply valid subgradients), where the effects of any possible chattering in the numerical integration can be mitigated, see [118, Chapter 7].

It is also possible to remove a degree of freedom, δ_3, from the optimization problem by substituting it with $1-\delta_1-\delta_2$ and eliminating the constraint (4.65) from the problem. This is attractive because it reduces the dimension of the parameter space in the branch-and-bound framework. The numerical implementation used for solving this problem is as follows: the convex relaxations constructed using Theorem 4.48 and the natural interval extensions of Corollary 4.24 were generated automatically based on an operator-overloading approach using C++; the local dynamic optimizations were performed using the code DYNO [58], which implements the control parametrization approach; and the branch-and-bound framework used was libBandB 3.2 [119]. Using a reference trajectory of $(\hat{\mathbf{x}}, \boldsymbol{\delta})_k = (\hat{\mathbf{x}}^L, \boldsymbol{\delta}^L)_k$, a relative tolerance for libBandB of 10^{-3}, relative and absolute tolerances for the numerical integrator in DYNO of 10^{-7}, and an optimality tolerance for the NLP solver in DYNO of 10^{-5}, an optimal solution value of 314.2 was obtained, at the point $\boldsymbol{\delta}^* = (0.3626, 0.0196, 0.6178)$. There was a total of 483 nodes visited in the branch-and-bound tree, with a total CPU time

of 315s. For comparison, if the problem just involved two sections with the mode sequence $T_\mu = 1, 3$, an optimal solution value of 296.9 was obtained, at the point $\boldsymbol{\delta}^* = (0.4181, 0.5819)$, with a total of 17 nodes and a total CPU time of 4.5s.

Comparing these results to those obtained in Example 3.39 in Section 3.6 when the mode sequence was allowed to vary with fixed transition times, algorithm (BCDF) was used to solve the problem with 27 epochs. The solution obtained was $F = 311.3$ with a total CPU time of 857s. Eventually, as n_e increases, the solution obtained using the varying mode sequence approach is expected to asymptotically approach the solution of $F = 314.2$ obtained here. However, that would likely take up much more computational resources. Hence, for this stiff example, these results suggest that better solutions can be found faster by considering a continuous time formulation with varying event times on a coarse event grid as opposed to a discrete time formulation on a very fine uniform event grid.

Example 4.51. Consider the following problem,

$$\begin{aligned} &\min_{\delta \in [0.5, 20]} x_1(\delta) \\ &\text{s.t. } \delta \leq x_1(\delta), \end{aligned}$$

where $\mathbf{x}(t)$ is given by the solution of the following LTI ODE system,

$$\begin{aligned} \dot{x}_1 &= x_1 - x_2 + 0.1, \\ \dot{x}_2 &= x_1 + 1.0, \end{aligned}$$

where $\mathbf{x}(0) = (1, -1)$, and $t \in [0, \delta]$.

Note that we have what seems to be a simple, reasonable optimization problem to solve in that it only involves a single decision variable, and a well scaled LTI ODE system. However, this problem contains multiple local minima due to the point constraint. To solve it globally, we apply the CPET to obtain the following, equivalent

problem,

$$\min_{\delta \in [0.5,20]} \hat{x}_1(1)$$
$$\text{s.t. } \delta \leq \hat{x}_1(1),$$

where $\hat{\mathbf{x}}(s)$ is given by the solution of the following nonlinear ODE system,

$$\hat{x}_1' = \delta(\hat{x}_1 - \hat{x}_2 + 0.1),$$
$$\hat{x}_2' = \delta(\hat{x}_1 + 1.0),$$

where $\hat{\mathbf{x}}(0) = (1, -1)$, and $s \in [0, 1]$.

First, we shall try to solve the transformed problem utilizing Theorem 4.48 with bounds obtained from Corollary 4.24. We note that in deriving the linearizations of Theorem 4.48, we can exploit the bilinear structure of the transformed problem above. We know that the convex envelope of a bilinear term vw for $v^L \leq v \leq v^U$ and $w^L \leq w \leq w^U$ is given by the following,

$$vw \geq vw^U + v^U w - v^U w^U, \tag{4.66}$$
$$vw \geq vw^L + v^L w - v^L w^L, \tag{4.67}$$
$$vw \leq vw^L + v^U w - v^U w^L, \tag{4.68}$$
$$vw \leq vw^U + v^L w - v^L w^U. \tag{4.69}$$

Thus, the linearization for $\hat{x}_1$ can be obtained by substituting $v \equiv \delta$ and $w \equiv \hat{x}_1 - \hat{x}_2 + 0.1$ into the formulas above. For this problem, we have chosen the underestimating linearization to be (4.67) and the overestimating linearization to be (4.69). The integrator used was `DAEPACK` and `DSL48SE` with an absolute and relative tolerance of 10^{-8}, and an in-house implementation of the BB algorithm (Algorithm 2.3) was implemented in `C++`. The relative tolerance of the BB algorithm was set to 10^{-3}, while the absolute tolerance was not used. The NLP solver used was SNOPT 6.1 [66] with the default settings, and an optimality tolerance of 10^{-6}. The finite difference option

was used for this example to estimate the required Jacobian, rather than using the sensitivity option from `DSL48SE` because finite differences solved each subproblem in the BB algorithm faster in this example involving one optimization decision variable.

For this example, the bounds obtained from Corollary 4.24 explode. This causes many problems for the NLP solver. For the first group of nodes (up to 140) of the BB tree employing a best bound node selection heuristic, the NLP solver returns an error message saying that the problem is either infeasible or badly scaled. To mitigate this problem, whenever this error occurs, we have set the lower bound to -10^{20} and set the lower bounding problem to be feasible (so that the algorithm is forced to branch on that partition corresponding to the node). The first lower bound is obtained after 140 nodes with the value of -1.9×10^{11}. This is an extremely bad lower bound for this problem. For this problem, the BB algorithm does not terminate with the global solution. This is because we terminate the BB algorithm when the width of a node being explored is less than the optimality tolerance of the NLP solver, 10^{-6}. This occurs after 28594 nodes, 2630 CPU seconds, with an incumbent lower bound of -1.5×10^5 and a local solution of 6.25. This implies that the BB algorithm will not converge with the tolerances used. What is particularly discouraging about this example is that it only involves a single decision variable to be branched upon for the BB algorithm.

Next, we examine the effect of using the exact bounds for Theorem 4.48 from Theorem 4.44. The cost per node increases significantly when using this method, because of the cost of obtaining the exact bounds. For this example, it is particularly expensive because the cost of integration is the major component of the cost for the NLP subproblems. What is somewhat surprising is that although the exact bounds improve the values of the lower bounds, the values of the lower bounds are still very weak, suggesting that the convex relaxations constructed from Theorem 4.48 can still be weak even with the exact state bounds. For example, after 2000 nodes, the incumbent lower bound from before was -2.6×10^7 using 190 CPU seconds, while after 2000 nodes with the exact bounds, the incumbent lower bound was -1.1×10^6 using 3240 CPU seconds. The trade off is better lower bounds with more time spent

per node. The algorithm was terminated after an hour of CPU time, with the BB algorithm not having converged, and an incumbent upper bound of 9.68.

Finally, we note that the exact bounds from Theorem 4.44 can be used as convex relaxations, $\mathbf{c}$ and $\mathbf{C}$, in place of Theorem 4.48. This way, the lower bounding problem will no longer require a call to the NLP solver, but simply an integration of the exact bounding hybrid system. Implementing this, the BB algorithm converges in 95 nodes with the global solution of $\delta = \hat{x}_1(1) = 1.8975$, in 21 CPU seconds. The incumbent lower bound after the root node was -3.34×10^4. Clearly, this is orders of magnitude better than the aforementioned methods. This example illustrates the utility of the developed bounding theory, at least for solving single-stage problems with a varying time horizon to global optimality. The further application of these bounding techniques will be very interesting for future work.

Chapter 5

Conclusions and Future Work

Chapter 1 discusses the literature to date on the modeling, simulation, sensitivity analysis and optimization of hybrid systems. A clear and concise framework, based on the concept of the hybrid automaton, is presented for the modeling and analysis of hybrid systems. The importance of defining a deterministic execution of the hybrid system is discussed, with emphasis on the semantics of a transition condition to define a unique transition time, as well as a unique successor mode. The latter is accomplished through the introduction of precedence rules for pending transitions. It is shown that transversality of the discontinuity functions is not sufficient to define a well-behaved execution of a hybrid system. Instead, it is proposed that a well-behaved execution be defined as one in which the parametric sensitivities of the hybrid system exist and are unique. An important difficulty is highlighted regarding the modeling of reversible discontinuities within the current modeling frameworks. While this difficulty can be mitigated, to some extent, by introducing appropriate transversality conditions, its satisfactory resolution requires much more work and inspiration.

Simulation techniques for the robust simulation of hybrid systems are discussed, including the importance of rigorous state event location and issues regarding the consistent reinitialization at events. In particular, a recently developed method for the automatic determination of natural transition functions for LTI DAEs is highlighted. There remains a need to develop a corresponding theory for LTV and nonlinear DAEs.

Another area that needs to be addressed is the development of better practically implementable algorithms for the diagnosis and reformulation of high index DAEs.

The theory for the parametric sensitivity analysis of hybrid systems is also presented, together with efficient ways of calculating these sensitivities robustly and correctly. In particular, the sensitivity trajectories have to be handled carefully at transitions, and this reiterates the point that rigorous detection of all state events in strict time order is pivotal. The potential of a more efficient method for calculating derivative information in certain situations through the use of adjoints is noted. A control parameterization approach to the open loop dynamic optimization of hybrid systems is also discussed. Such an approach hinges on the existence and uniqueness of the parametric sensitivities of the embedded hybrid system, that is, it requires all executions of the hybrid system to be well-behaved for all values of the optimization decision variables. A classification of problems is proposed as a general guide to which gradient based optimization techniques can be brought to bear, based on whether the sequence of modes of the hybrid system changes depending on the values of the optimization decision variables. Unfortunately, the most intriguing class of problems are those in which the sequence of modes does change in the parameter space.

In Chapter 2, the deterministic, global optimization algorithms branch-and-bound and nonconvex outer approximation are introduced. A method to construct convex relaxations for general, nonlinear Bolza type functions subject to an embedded linear hybrid system with explicit time events is presented. The method relies on existing methods to construct convex underestimators of factorable and twice differentiable functions on compact Euclidean sets. A major requirement of the method is to obtain bounding trajectories of the embedded hybrid system. It is shown how the implied state bounds, which are also the tightest possible bounds, of the embedded hybrid system can be obtained. The constructed convex relaxations are shown to possess consistent bounding properties, and hence, a branch-and-bound algorithm employing these relaxations will be infinitely convergent. Additionally, it is shown that the parametric sensitivities of the embedded hybrid system under consideration exist. Sufficient conditions for the smoothness of the objective function in the parameter

set are also presented, which allows efficient, gradient based optimization algorithms to be utilized within the branch-and-bound framework to obtain an ε-optimal estimate for the global solution to the optimization problem.

The problem of obtaining the optimal mode sequence for a continuous time linear time varying hybrid system with fixed time transitions is considered in Chapter 3. Dynamic programming approaches are discussed, but ultimately abandoned due to the need to discretize the continuous state space (and the inevitable curse of dimensionality), as well as difficulties in handling (possibly nonconvex) constraints. Instead, binary decision variables are introduced to represent the mode sequence, and a superstructure of the hybrid system is proposed, where the linearity of the embedded dynamic system is retained by introducing auxiliary continuous variables, and shifting the nonlinearities and nonconvexities into additional constraints. This exploitation of the linear structure of the embedded hybrid system is important, because nonlinear approaches can suffer from very weak relaxations, as shown in Chapter 4.

An important component of constructing a convex relaxation of the reformulated problem lies in the estimation of bounds for the hybrid system at the epoch boundaries, which bound the auxiliary variables acting as the initial conditions for the decomposed system. Various bounding strategies are presented for this purpose. It is shown that a simple and efficient decomposition approach based on calculating the exact state bounds for the subproblems of the hybrid superstructure produces weak bounds which deteriorate as the number of epochs increases. A novel algorithm is proposed based on the solution of families of MILPs as relaxed LPs. This algorithm is able to incorporate physical insight as additional constraints in the LPs, and produces significantly tighter bounds than the decomposition algorithm.

The convexity theory developed in Chapter 2 is applied to construct convex relaxations for the reformulated problem that can handle arbitrary point and isoperimetric constraints. The resulting nonconvex MINLP with its convex relaxation is then solved using a branch-and-cut algorithm with a novel dynamic bounds tightening heuristic. This heuristic utilizes the simple decomposition approach based on calculating the implied state bounds for the subproblems of the hybrid superstructure mentioned

above, which ultimately proves extremely useful due to its efficiency. It is illustrated through examples that dynamic bounds tightening can be very effective in accelerating the convergence of the proposed algorithm, which allows it to systematically outperform explicit enumeration of all possible mode sequences.

In Chapter 4, the global optimization problem with continuous time linear hybrid systems embedded has been considered where the embedded systems have varying time transitions and a fixed mode sequence. This problem is shown to be inherently nonconvex, and it is shown that special care has to be taken with respect to the type of discontinuities allowed in the problem for the resulting optimization problem to be smooth. The control parametrization enhancing transform has been utilized to transform the problem into a global optimization problem with nonlinear hybrid systems embedded where the transitions are now fixed in time. Sufficient conditions have been proposed for these problems to be smooth in the control parametrization framework. A method of constructing convex relaxations for the transformed problem has been developed that is shown to be convergent within a branch-and-bound framework. A very important requirement for utilizing this convexity theory for nonlinear hybrid systems is the ability to construct bounding trajectories for the states of the nonlinear hybrid system. The theory of differential inequalities is utilized for this purpose, where it is possible for additional information about the system to be exploited in the form of bounding sets which are known independently from the solution of the hybrid system. However, it is shown, with simple examples, that this theory might not produce satisfactory bounds when the differential system is non-quasimonotone. An algorithm is proposed to exploit the time transformation, that guarantees the exactness of the bounding trajectories for single stage linear time varying systems. It is shown that this proposed algorithm can solve simple problems for which the method employing differential inequalities fails.

While it is hoped that the material in this thesis will make a significant impact upon the field of global optimization of hybrid systems, there remains a lot of exciting and significant opportunities for future research. Obviously, an immediate area of research is to consider optimization problems where both the sequence of modes and

transition times are allowed to vary. In addition, it is very desirable for methods to be developed for the optimization of general, nonlinear hybrid systems, and even hybrid systems with differential algebraic equations, which requires the formal development of the required convexity theory for Bolza type functions with differential algebraic equations embedded. In particular, there exists a tremendous scope of research on providing better bounding techniques, and developing methods to construct convex relaxations of Bolza type functions with nonlinear systems embedded, as the current methods are not entirely satisfactory. These very challenging problems will likely require some inspiration, and the exploitation of problem structure wherever possible.

Bibliography

[1] C. S. Adjiman, C. S. Dallwig, C. A. Floudas, and A. Neumaier. A global optimization method, αBB, for general twice-differentiable constrained NLPs - I. Theoretical advances. *Computers & Chemical Engineering*, 22(9):1137–1158, 1998.

[2] C. S. Adjiman, I. P. Androulakis, and C. A. Floudas. Global optimization of mixed-integer nonlinear problems. *AICHE Journal*, 46(9):1769–1797, 2000.

[3] G. Alefeld and J. Herzberger. *Introduction to Interval Computations*. Academic Press, New York, 1983.

[4] R. Alur, C. Courcoubetis, T. A. Henzinger, and P. Ho. Hybrid automata: An algorithmic approach to the specification and verification of hybrid systems. In [69], pages 209–229. 1993.

[5] R. Alur, C. Courcoubetis, N. Halbwachs, T. A. Henzinger, P.-H. Ho, X. Nicollin, A. Olivero, J. Sifakis, and S. Yovine. The algorithmic analysis of hybrid systems. *Theoretical Computer Science*, 138:3–34, 1995.

[6] R. Alur, T. A. Henzinger, and E. D. Sontag, editors. *Hybrid Systems III*, volume 1066 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1996.

[7] M. Andersson. *Object-Oriented Modeling and Simulation of Hybrid Systems*. PhD thesis, Lund Institute of Technology, Sweden, 1994.

[8] P. Antsaklis, W. Kohn, and A. Nerode, editors. *Hybrid Systems II*, volume 999 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1995.

[9] P. Antsaklis, W. Kohn, A. Nerode, and S. Sastry, editors. *Hybrid Systems IV*, volume 1273 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1997.

[10] P. Antsaklis, W. Kohn, M. Lemmon, A. Nerode, and S. Sastry, editors. *Hybrid Systems V*, volume 1567 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1999.

[11] U. M. Ascher and L. R. Petzold. *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia, 1998.

[12] M. P. Avraam, N. Shah, and C. C. Pantelides. Modelling and optimisation of general hybrid systems in the continuous time domain. *Computers & Chemical Engineering*, 22(S):S221–S228, 1998.

[13] A. Back, J. Guckenheimer, and M. Myers. A dynamical simulation facility for hybrid systems. In [69], pages 255–267. 1993.

[14] E. Balas, S. Ceria, and G. Cornuejols. Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. *Management Science*, 42:1229–1246, 1996.

[15] P. I. Barton. *The Modelling and Simulation of Combined Discrete/Continuous Processes*. PhD thesis, University of London, 1992.

[16] P. I. Barton and S. Galán. Linear DAEs with nonsmooth forcing. Technical report, Department of Chemical Engineering, MIT, available from *http://yoric.mit.edu/reports.html*, 26 Jan 2000.

[17] P. I. Barton and C. K. Lee. Modeling, simulation, sensitivity analysis and optimization of hybrid systems. *ACM Transactions on Modeling and Computer Simulation*, 12(4):1–34, 2002.

[18] P. I. Barton and C. C. Pantelides. Modeling of combined discrete/continuous processes. *AICHE Journal*, 40:966–979, 1994.

[19] P. I. Barton, J. R. Banga, and S. Galán. Optimization of hybrid discrete/continuous dynamic systems. *Computers & Chemical Engineering*, 24 (9-10):2171–2182, 2000.

[20] Paul I. Barton. *Mixed-Integer and Nonconvex Optimization*. Massachusetts Institute of Technology, Cambridge, USA, 2006.

[21] M. S. Bazaraa, H. D. Sherali, and C. M. Shetty. *Nonlinear Programming. Theory and Algorithms*. John Wiley & Sons, Inc., Canada, second edition, 1993.

[22] E. F. Beckenbach and R. Bellman. *Inequalities*. Springer-Verlag, New York, 1965.

[23] R. E. Bellman and R. E. Kalaba. *Dynamic Programming and Modern Control Theory*. Academic Press, New York, 1965.

[24] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics and constraints. *Automatica*, 35(3):407–427, 1999.

[25] A. Bemporad, G. Ferrari-Trecate, and M. Morari. Observability and controllability of piecewise affine and hybrid systems. *IEEE Transactions on Automatic Control*, 45(10):1864–1876, October 2000.

[26] A. Bemporad, D. Corona, A. Giua, and C. Seatzu. Optimal state-feedback quadratic regulation of linear hybrid automata. In *Proceedings of the IFAC Conference on Analysis and Design of Hybrid Systems*, pages 407–412, 2003.

[27] D. Bertsimas and J. N. Tsitsiklis. *Introduction to Linear Optimization*. Athena Scientific, Massachusetts, 1997.

[28] M. S. Branicky and S. E. Mattsson. Simulation of hybrid systems. In [9], pages 31–56. 1997.

[29] M. S. Branicky and S. K. Mitter. Algorithms for optimal hybrid control. In *Proceedings of the 34th IEEE Conference on Decision and Control*, volume 3, pages 2661–2666, 1995.

[30] M. S. Branicky, V. S. Borkar, and S. K. Mitter. A unified framework for hybrid control: Model and optimal control theory. *IEEE Transactions on Automatic Control*, 43(1):31–45, 1998.

[31] R. W. Brankin, I. Gladwell, and L. F. Shampine. RKSUITE: A suite of Runge-Kutta codes for the initial value problem for ODEs. Softreport 92-S1, Department of Mathematics, Southern Methodist University, Dallas, Texas, U.S.A, 1992.

[32] K. E. Brenan, S. L. Campbell, and L. R. Petzold. *Numerical Solution of Initial Value Problems in Differential-Algebraic Equations*. North-Holland, New York, 1989.

[33] P. N. Brown, G. D. Byrne, and A. C. Hindmarsh. VODE: A variable-coefficient ODE solver. *SIAM Journal on Scientific and Statistical Computing*, 10(5): 1038–1051, 1989.

[34] L. Brüll and U. Pallaske. On differential algebraic equations with discontinuities. *Zeitschrift für angewandte Mathematik und Physik*, 43:319–327, 1992.

[35] A. E. Bryson and Y. Ho. *Applied Optimal Control*. Hemisphere, New York, 1975.

[36] S. L. Campbell and C. W. Gear. The index of general nonlinear DAEs. *Numerische Mathematik*, 72(2):173–196, 1995.

[37] M. Caracotsios and W. E. Stewart. Sensitivity analysis of initial value problems with mixed ODEs and algebraic constraints. *Computers & Chemical Engineering*, 9:359–365, 1985.

[38] C. G. Cassandras, D. L. Pepyne, and Y. Wardi. Optimal control of a class of hybrid systems. *IEEE Transactions on Automatic Control*, 46(3):398–415, March 2001.

[39] F. E. Cellier. *Combined Continuous/Discrete System Simulation by Use of Digital Computers: Techniques and Tools*. PhD thesis, Swiss Federal Institute of Technology, Zurich, 1979.

[40] F. E. Cellier. Combined continuous/discrete simulation applications, techniques and tools. In J. Wilson, J. Henriken, and S. Roberts, editors, *Proceedings of the 1986 Winter Simulation Conference*, pages 24–33. 1986.

[41] J. A. Clabaugh, J. E. Tolsma, and P. I. Barton. ABACUSS II: Advanced modeling environment and embedded simulator. *http://yoric.mit.edu/abacuss2/abacuss2.html*, 1999.

[42] E. A. Coddington and N. Levinson. *Theory of Ordinary Differential Equations*. McGraw Hill, 1955.

[43] L. J. Corwin and R. H. Szczarba. *Multivariable Calculus*. Marcel Dekker, Inc., New York, 1982.

[44] A. Crema. The multiparametric 0-1-integer linear programming problem: A unified approach. *European Journal of Operational Research*, 139:511–520, 2002.

[45] R. David and H. Alla. On hybrid Petri nets. *Discrete Event Dynamic Systems*, 11:9–40, 2001.

[46] V. D. Dimitriadis, N. Shah, and C. C. Pantelides. Modeling and safety verification of discrete/continuous processing systems. *AICHE Journal*, 43(4): 1041–1059, April 1997.

[47] S. E. Dreyfus and A. M. Law. *The Art and Theory of Dynamic Programming*. Academic Press, London, 1977.

[48] I. S. Duff and J. K. Reid. MA48, a FORTRAN code for direct solution of sparse unsymmetric linear systems of equations. Technical report, RAL-93-072; Rutherford Appleton Laboratory, Oxon, UK, 1993.

[49] I. S. Duff, A. M. Erisman, and J. K. Reid. *Direct Methods for Sparse Matrices*. Clarendon Press, Oxford, 1986.

[50] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36: 307–339, 1986.

[51] M. Egerstedt, Y. Wardi, and F. Delmotte. Optimal control of switching times in switched dynamical systems. In *Proceedings of the 42th IEEE Conference on Decision and Control*, pages 2138–2143, 2003.

[52] H. Elmqvist, F. E. Cellier, and M. Otter. Object-oriented modeling of hybrid systems. In *Proceedings of ESS'93, European Simulation Symposium*, pages xxxi–xli, 1993.

[53] D. A. Fahrland. Combined discrete event continuous systems simulation. *Simulation*, 14:61–72, 1970.

[54] J. E. Falk and R. M. Soland. An algorithm for separable nonconvex programming problems. *Management Science*, 15(9):550–569, 1969.

[55] W. F. Feehery. *Dynamic Optimization with Path Constraints*. PhD thesis, Massachusetts Institute of Technology, Cambridge, 1998.

[56] W. F. Feehery and P. I. Barton. Dynamic optimization with equality path constraints. *Industrial & Engineering Chemistry Research*, 38(6):2350–2363, 1999.

[57] W. F. Feehery, J. E. Tolsma, and P. I. Barton. Efficient sensitivity analysis of large-scale differential-algebraic systems. *Applied Numerical Mathematics*, 25 (1):41–54, 1997.

[58] M. Fikar and M. A. Latifi. User's guide for Fortran dynamic optimisation code DYNO. Technical report, Nancy, France: LSGC-CNRS and Bratislava, Slovak Republic: Slovak Technical University Bratislava, 2002.

[59] R. Fletcher and S. Leyffer. Solving mixed integer nonlinear programs by outer approximation. *Mathematical Programming*, 66:327–349, 1994.

[60] C. A. Floudas. *Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications*. Oxford University Press, Oxford, 1995.

[61] P. M. Frank. *Introduction to System Sensitivity Theory*. Academic Press, New York, 1978.

[62] S. Galán and P. I. Barton. Dynamic optimization of hybrid systems. *Computers & Chemical Engineering*, 22(Suppl.):S183–S190, 1998.

[63] S. Galán, W. F. Feehery, and P. I. Barton. Parametric sensitivity functions for hybrid discrete/continuous systems. *Applied Numerical Mathematics*, 31 (1):17–48, 1999.

[64] E. P. Gatzke, J. E. Tolsma, and P. I. Barton. Construction of convex function relaxations using automated code generation techniques. *Optimization & Engineering*, 3:305–326, 2002.

[65] C. W. Gear. *Numerical Initial Value Problems in Ordinary Differential Equations*. Prentice-Hall, Englewood Cliffs, NJ, 1973.

[66] P. E. Gill, W. Murray, and M. A. Saunders. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Journal on Optimization*, 12:979–1006, 2002.

[67] F. Glover. Improved linear integer programming formulations of nonlinear integer problems. *Management Science*, 22(4):455–460, 1975.

[68] V. Gopal and L. T. Biegler. A successive linear programming approach for initialization and reinitialization after discontinuities of differential-algebraic equations. *SIAM Journal on Scientific Computing*, 20(2):447–467, 1999.

[69] R. L. Grossman, A. Nerode, A. P. Ravn, and H. Rischel, editors. *Hybrid Systems*, volume 736 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 1993.

[70] G. W. Harrison. Dynamic models with uncertain parameters. In X. J. R. Avula, editor, *Proceedings of the First International Conference on Mathematical Modeling*, volume 1, pages 295–304, University of Missouri, Rolla, 1977.

[71] G. W. Harrison. Compartmental models with uncertain flow rates. *Mathematical Biosciences*, 43:131–139, 1979.

[72] J. L. Hay and A. W. J. Griffin. Simulation of discontinuous dynamical systems. In L. Dekker, G. Savastano, and G. C. Vansteenkiste, editors, *Simulation of Systems '79*. North-Holland, 1980.

[73] S. Hedlund and A. Rantzer. Optimal control of hybrid systems. In *Proceedings of the 38th IEEE Conference on Decision and Control*, volume 4, pages 3972–3977, 1999.

[74] A. C. Hindmarsh. ODEPACK, a systematized collection of ODE solvers. In R. S. Stepleman, editor, *Scientific Computing*, pages 55–64. North-Holland, Amsterdam, 1983.

[75] I. A. Hiskens and M. A. Pai. Trajectory sensitivity analysis of hybrid systems. *IEEE Transactions on Circuits and Systems-I*, 47(2):204–220, 2000.

[76] Y. C. Ho and X. R. Cao. *Perturbation Analysis of Discrete Event Dynamic Systems*. Kluwer Academic Publishers, Boston, 1991.

[77] R. Horst and H. Tuy. *Global Optimization*. Springer-Verlag, Berlin, 3rd edition, 1996.

[78] ILOG Inc. ILOG CPLEX 9.1 documentation, 2005. URL `http://www.ilog.com/products/cplex/`.

[79] K. H. Johansson, M. Egerstedt, J. Lygeros, and S. Sastry. On the regularization of Zeno hybrid automata. *Systems & Control Letters*, 38:141–150, 1999.

[80] R. B. Kearfott, M. Dawande, K. Du, and C. Hu. Algorithm 737: INTLIB: A portable Fortran 77 interval standard-function library. *ACM Transactions on Mathematical Software*, 20(4):447–459, 1994.

[81] P. Kesavan and P. I. Barton. Generalized branch-and-cut framework for mixed-integer nonlinear optimization problems. *Computers & Chemical Engineering*, 24(2-7):1361–1366, 2000.

[82] P. Kesavan and P. I. Barton. Decomposition algorithms for nonconvex mixed-integer nonlinear programs. *AIChE Symposium Series*, 96(323):458–461, 2000.

[83] P. Kesavan, R. J. Allgor, E. P. Gatzke, and P. I. Barton. Outer approximation algorithms for separable nonconvex mixed-integer nonlinear programs. *Mathematical Programming Series A*, 100(3):517–535, 2004.

[84] V. Lakshmikantham and S. Leela. *Differential and Integral Inequalities*, volume 1. Academic, New York, 1969.

[85] H. W. J. Lee, K. L. Teo, V. Rehbock, and L. S. Jennings. Control parametrization enhancing technique for time optimal control problems. *Dynamic Systems and Applications*, 6:243–262, 1997.

[86] H. W. J. Lee, K. L. Teo, V. Rehbock, and L. S. Jennings. Control parametrization enhancing technique for optimal discrete-valued control problems. *Automatica*, 35:1401–1407, 1999.

[87] Numerica Technology LLC. *JACOBIAN User Guide*. Numerica Technology, Cambridge, MA, 2005. www.numericatech.com.

[88] J. Logsdon and L. T. Biegler. Accurate solution of differential-algebraic optimization problems. *Industrial & Engineering Chemistry Research*, 28:1628–1639, 1989.

[89] J. Lu, L. Liao, A. Nerode, and J. H. Taylor. Optimal control of systems with continuous and discrete states. In *Proceedings of the 32nd IEEE Conference on Decision and Control*, volume 3, pages 2292–2297, 1993.

[90] J. Lygeros, K. H. Johansson, S. N. Simić, J. Zhang, and S. S. Sastry. Dynamical properties of hybrid automata. *IEEE Transactions on Automatic Control*, 48 (1):2–17, January 2003.

[91] C. Majer, W. Marquardt, and E. D. Gilles. Reinitialization of DAEs after discontinuities. *Computers & Chemical Engineering*, 19(suppl.):S507–S512, 1995.

[92] J. Malmborg and B. Bernhardsson. Control and simulation of hybrid systems. *Nonlinear Analysis: Theory, Methods & Applications*, 30(1):337–347, 1997.

[93] T. Maly and L. R. Petzold. Numerical methods and software for sensitivity analysis of differential-algebraic systems. *Applied Numerical Mathematics*, 20: 57–79, 1996.

[94] S. E. Mattsson. On modelling and differential/algebraic systems. *Simulation*, 52:24–32, 1989.

[95] S. E. Mattsson and G. Söderlind. Index reduction in differential-algebraic equations using dummy derivatives. *SIAM Journal on Scientific Computing*, 14(3): 677–692, 1993.

[96] G. P. McCormick. Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems. *Mathematical Programming*, 10:147–175, 1976.

[97] A. Mitsos. *Man-Portable Power Generation Devices: Product Design and Supporting Algorithms*. PhD thesis, MIT, Cambridge, MA, USA, 2006. Available from http://yoric.mit.edu/reports.html.

[98] R. E. Moore. *Interval Analysis*. Prentice-Hall, Englewood Cliffs, N.J., 1966.

[99] R. E. Moore. *Methods and Applications of Interval Analysis.* SIAM, Philadelphia, 1979.

[100] K. R. Morison and R. W. H. Sargent. Optimization of multistage processes described by differential-algebraic equations. In J. P. Hennart, editor, *Numerical Analysis: Proceedings of the 4th IIMAS Workshop*, volume 1230 of *Lecture Notes in Mathematics*, pages 86–102. Springer-Verlag, Berlin, 1986.

[101] P. J. Mosterman. *Hybrid Dynamic Systems: A Hybrid Bond Graph Modeling Paradigm and its Application in Diagnosis.* PhD thesis, Vanderbilt University, Tennessee, 1997.

[102] P. J. Mosterman. An overview of hybrid simulation phenomena and their support by simulation packages. In F. W. Vaandrager and J. H. Van Schuppen, editors, *Hybrid Systems: Computation and Control*, volume 1569 of *Lecture Notes in Computer Science*, pages 165–177. Springer-Verlag, Berlin, 1999.

[103] G. L. Nemhauser and L. A. Wolsey. *Integer and Combinatorial Optimization.* Wiley, New York, 1988.

[104] T. I. Oren. Software for the simulation of combined continuous and discrete systems: A state-of-the-art review. *Simulation*, 28:33–45, 1977.

[105] M. Otter, H. Elmqvist, and S. E. Mattsson. Hybrid modeling in Modelica based on synchronous data flow principle. In *Proceedings of the 1999 IEEE Symposium on Computer-Aided Control System Design, CACSD'99*, pages 151–157. IEEE Control Systems Society, 1999.

[106] C. C. Pantelides. The consistent initialization of differential-algebraic systems. *SIAM Journal on Scientific and Statistical Computing*, 9(2):213–231, 1988.

[107] T. Park and P. I. Barton. State event location in differential-algebraic models. *ACM Transactions on Modeling and Computer Simulation*, 6(2):137–165, 1996.

[108] L. R. Petzold. A description of DASSL: A differential/algebraic system solver. In R. S. Stepleman, editor, *Scientific Computing*, pages 65–68. North-Holland, Amsterdam, 1983.

[109] H. Ratschek and J. Rokne. *Computer methods for the range of functions*. Ellis Horwood Limited, England, 1984.

[110] G. Reißig. Differential-algebraic equations and impasse points. *IEEE Transactions on Circuits and Systems-I*, 43:122–133, 1996.

[111] G. Reißig, W. S. Martinson, and P. I. Barton. Differential-algebraic equations of index 1 may have an arbitrarily high structural index. *SIAM Journal on Scientific Computing*, 21(6):1987–1990, 2000.

[112] G. Reißig, H. Boche, and P. I. Barton. On inconsistent initial conditions for linear time-invariant differential-algebraic equations. *IEEE Transactions on Circuits and Systems-I*, 49(11):1646–1648, 2002.

[113] R. T. Rockafellar. *Convex Analysis*. Princeton University Press, New Jersey, 1970.

[114] E. N. Rozenvasser. General sensitivity equations of discontinuous systems. *Automation and Remote Control*, pages 400–404, 1967.

[115] A. I. Ruban. Sensitivity coefficients for discontinuous dynamic systems. *Journal of Computer and System Sciences International*, 36(4):536–542, 1997.

[116] W. Rudin. *Principles of Mathematical Analysis*. McGraw-Hill Inc., New York, third edition, 1976.

[117] H. S. Ryoo and N. V. Sahinidis. A branch-and-reduce approach to global optimization. *Journal of Global Optimization*, 8(2):107–139, 1996.

[118] A. B. Singer. *Global Dynamic Optimization*. PhD thesis, MIT, Cambridge, MA, USA, 2004. Available from http://yoric.mit.edu/reports.html.

[119] A. B. Singer. LibBandB.a version 3.2 manual. Technical report, Massachusetts Institute of Technology, Cambridge, MA, 2004.

[120] A. B. Singer and P. I. Barton. Global solution of optimization problems with parameter-embedded linear dynamic systems. *Journal of Optimization Theory and Applications*, 121(3):613–646, 2004.

[121] A. B. Singer and P. I. Barton. Bounding the solutions of parameter dependent nonlinear ordinary differential equations. *SIAM Journal on Scientific Computing*, 27(6):2167–2182, 2006.

[122] A. B. Singer, J. W. Taylor, P. I. Barton, and W. H. Green. Global dynamic optimization for parameter estimation in chemical kinetics. *Journal of Physical Chemistry A*, 110(3):971–976, 2006.

[123] J. C. Strauss, D. C. Augustin, M. S. Fineberg, B. B. Johnson, R. N. Linebarger, and F. J. Sanson. The SCi continuous system simulation language (CSSL). *Simulation*, 9:281–303, 1967.

[124] L. Tavernini. Differential automata and their discrete simulators. *Nonlinear Analysis, Theory, Methods & Applications*, 11(6):665–683, 1987.

[125] M. Tawarmalani and N. V. Sahinidis. *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Algorithms, Software, and Applications*. Nonconvex Optimization And Its Applications. Kluwer Academic Publishers, Dordrecht, 2002.

[126] K. Teo, G. Goh, and K. Wong. *A Unified Computational Approach to Optimal Control Problems*. Pitman Monographs and Surveys in Pure and Applied Mathematics. Wiley, New York, 1991.

[127] K. L. Teo, L. S. Jennings, H. W. J. Lee, and V. Rehbock. The control parametrization enhancing transform for constrained optimal control problems. *Journal of the Australian Mathematical Society Series B*, 40:314–335, 1999.

[128] J. E. Tolsma and P. I. Barton. DAEPACK: An open modeling environment for legacy models. *Industrial & Engineering Chemistry Research*, 39(6):1826–1839, 2000.

[129] J. E. Tolsma and P. I. Barton. Hidden discontinuities and parametric sensitivity calculations. *SIAM Journal on Scientific Computing*, 23(6):1861–1874, 2002.

[130] J. E. Tolsma and P. I. Barton. DAEPACK: A symbolic and numeric library for open modeling. *http://yoric.mit.edu/daepack/daepack.html*, 1999.

[131] C. Tomlin, G. J. Pappas, and S. Sastry. Conflict resolution for air traffic management: A case study in multi-agent hybrid systems. *IEEE Transactions on Automatic Control*, 43(4):509–521, April 1998.

[132] C. Tomlin, J. Lygeros, and S. Sastry. A game theoretic approach to controller design for hybrid systems. *Proceedings of the IEEE*, 88(7):949–970, July 2000.

[133] V. I. Utkin. *Sliding Modes in Control and Optimization*. Springer-Verlag, Berlin, 1992.

[134] V. S. Vassiliadis, R. W. H. Sargent, and C. C. Pantelides. Solution of a class of multistage dynamic optimization problems. 1. Problems without path constraints. *Industrial & Engineering Chemistry Research*, 33:2111–2122, 1994.

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

[136] H. S. Witsenhausen. A class of hybrid-state continuous-time dynamic systems. *IEEE Transactions on Automatic Control*, 11(2):161–167, April 1966.

[137] X. Xu and P. J. Antsaklis. An approach for solving general switched linear quadratic optimal control problems. In *Proceedings of the 40th IEEE Conference on Decision and Control*, pages 2478–2483, 2001.

[138] X. Xu and P. J. Antsaklis. Optimal control of switched systems based on parameterization of the switching instants. *IEEE Transactions on Automatic Control*, 49(1):2–16, January 2004.

[139] J. Zhang, K. H. Johansson, J. Lygeros, and S. Sastry. Zeno hybrid systems. *International Journal of Robust and Nonlinear Control*, 11(5):435–451, 2001.