

# Reachability Analysis and Deterministic Global Optimization of DAE Models

Joseph K. Scott and Paul I. Barton

**Abstract** This article provides a tutorial overview of recent progress in numerical methods for the reachability analysis and deterministic global optimization of DAE models. These problems are highly interrelated, and are global problems in the sense that they concern the parametric solutions of a DAE model over a potentially large range of model parameters, rather than locally about a single value. Many techniques are available for computing such global information for functions known in closed-form (i.e., factorable functions). Two of the simplest and most flexible techniques are interval arithmetic and McCormick's relaxation technique. The methods reviewed herein extend these techniques to functions defined as the solutions of DAE models, which are notably non-factorable. In doing so, we repeatedly exploit the idea that the factorable representations of the DAE governing equations, combined with insights from dynamical systems theory, can be used to infer global information about the DAE solutions. This concept is first used to derive methods for computing interval bounds and more general convex enclosures of the solutions of DAE models over a range of model parameters. Subsequently, these enclosures are employed in a branch-and-bound algorithm for deterministic global dynamic optimization with DAE's embedded. The article closes with an illustrative case study in parameter estimation and some prospects for future work.

**Keywords** Convex relaxations • Dynamic optimization • Global optimization • Interval methods • Reachability analysis

**MSC:** 93B03, 90C26, 34A09, 34A40, 34A60, 49J15, 49M37

---

J.K. Scott (✉)

Department of Chemical and Biomolecular Engineering, Clemson University, Clemson, SC 29609, USA

e-mail: [jks9@clemson.edu](mailto:jks9@clemson.edu)

P.I. Barton

Department of Chemical Engineering, MIT, Cambridge, MA 02139, USA

e-mail: [pib@mit.edu](mailto:pib@mit.edu)

© Springer International Publishing Switzerland 2015

A. Ilchmann, T. Reis (eds.), *Surveys in Differential-Algebraic Equations III*, Differential-Algebraic Equations Forum, DOI 10.1007/978-3-319-22428-2\_2

## 1 Introduction

Systems of differential-algebraic equations (DAEs) are used to model an incredible variety of dynamic phenomena. In the chemical process industry in particular, the numerical simulation of detailed DAE models has become a cornerstone of many core activities including process development, economic optimization, control system design, and safety analysis.

The purpose of this article is to review recent progress in numerical methods for the reachability analysis and deterministic global optimization of DAE models. These problems are highly interrelated, and are *global* problems in the sense that they concern the parametric solutions of a DAE model over a potentially large range of model parameters, rather than locally about a single value. This global information makes it possible to address a number of challenging problems in the design and control of chemical processes, thus motivating significant development of these techniques within the chemical engineering community over the past decade.

Given a DAE model, the *reachable set* at time  $t$  is defined to be the set of all states that can be reached by a solution of the DAEs at time  $t$  given initial conditions and model parameters in some specified sets. Reachability analysis generally refers to the characterization of this set. In modern process control, the computation of approximations or enclosures of reachable sets is an active area of research and finds quite extensive application. Such computations have been used, for example, for state estimation from online measurements in chemical and biological processes [54, 63, 64], feedback controller synthesis [46, 61], robust model predictive control [38, 39], and fault detection for chemical processes [33, 43]. Reachable sets also provide a means to quantify the effects of uncertainties in model parameters or inputs. A particular example of interest comes from models of chemical reaction kinetics, where the rate parameters are often only known to within an order of magnitude or worse [72, 88]. Since these models are nearly always nonlinear, the effects of such uncertainty on the model solution can be extremely difficult to infer. Reachable set enclosures have been applied in this context for uncertain chemical kinetics models [76, 88], ecology models [26, 42], and biological systems [54, 64].

A large variety of methods have been developed for computing enclosures of the reachable sets of dynamic systems. However, the vast majority of these methods apply only to systems of explicit ordinary differential equations (ODEs), rather than to DAEs, and often under further simplifying assumptions. For linear ODEs, enclosures are typically computed in the form of ellipsoids [37, 73] or polytopes [3, 14]. Many of these methods have been extended to treat nonlinear ODEs using local linearizations with a rigorous bound on the approximation error [4, 13]. However, a more efficient and widely used approach for nonlinear systems is to compute time-varying interval enclosures of the reachable set, either through interval Taylor series methods [57, 58] and their refinements [9, 31, 42], or through the solution of differential inequalities [26, 64, 76, 80, 87].

Two methods have been proposed for extending interval bounding methods for ODEs to the case of semi-explicit index-one DAEs [65, 78, 79]. In both methods, the addition of implicit algebraic equations is addressed through the use of interval Newton methods [59]. The methods differ in the treatment of the differential states, which is done using interval Taylor series methods in [65] and using differential inequalities in [78, 79]. In this article, we review the essential concepts and results from [78, 79], keeping in mind that the treatment of the algebraic equations is conceptually similar to that in [65]. Two further approaches that we do not review here can be found in [16, 28]. The method in [28] applies to implicit ODEs and could potentially be extended to treat semi-explicit DAEs. The method in [16] extends so-called level set methods for ODEs [51] to the case of semi-explicit index-one DAEs. However, methods of this type are designed to provide an accurate approximation of the reachable set, rather than a rigorous enclosure of it, and are therefore inappropriate for the applications of primary interest here.

Recently, it has been shown that generic convex enclosures of the reachable set can be characterized by convex and concave relaxations of the state variables with respect to the model parameters [75]. These relaxations are customarily used for solving global dynamic optimization problems, as discussed below. In the case of nonlinear ODEs, numerous methods for computing state relaxations have been developed over the past decade, exclusively within the chemical engineering community [62, 70, 71, 77, 82, 84, 87]. These methods require interval enclosures as input, and can therefore be thought of as refinements of these enclosures. Empirically, convex and concave state relaxations are known to provide tighter enclosures than the underlying interval methods, and have superior convergence behavior. Of the methods above, those in [82, 84] have been extended to semi-explicit index-one DAEs in [74, 81]. The main concepts of these approaches are reviewed herein.

The second problem addressed in this article is the global solution of dynamic optimization problems constrained by DAE models. Specifically, these are optimization problems in which the decision variables appear as parameters or control inputs in a DAE model, and the solution of this model in turn appears in the objective and constraints of the optimization problem. If control inputs are present among the decision variables, these problems are also commonly called optimal control problems. Dynamic optimization problems arise in a wide variety of applications. In the chemical process industry, dynamic optimization techniques are routinely used to locate optimal process designs, operating conditions, and control actions. For example, open-loop control of batch processes can be formulated as a dynamic optimization problem and has been widely studied in this context, particularly with application to high-value added industries such as specialty chemicals, pharmaceuticals, and bioprocessing [10, 47, 90]. Dynamic optimization problems also arise when considering processes with periodic dynamic behavior, such as pressure swing adsorption and simulated moving bed chromatography [19, 34]. Even for processes that are nominally operated at steady-state, several important problems require dynamic optimization, including the determination of optimal start-up and shut-down procedures [8, 12] and optimal policies for changeover from one product to

another [25]. A more fundamental application is the problem of estimating unknown parameters in a dynamic model from a given set of data [40, 55, 88]. Here, the model parameters are the decision variables, and the optimization algorithm finds those parameters which minimize the deviation of the model prediction from the measured data. This problem is extremely important, for example, for the determination of chemical reaction mechanisms from kinetic data [55, 88].

Solving dynamic optimization problems to local optimality is a very mature technology and can be done efficiently even for large and complex DAE models. The methods used in most modern codes can be classified as either *sequential* or *simultaneous*. In both approaches, any control inputs among the decision variables are first discretized through a procedure called control parameterization [93]. In the simultaneous approach, the DAEs themselves are also discretized, typically by collocation on finite elements [11, 17, 94]. This provides a representation of the state and control functions in terms of finitely many real parameters, so that the resulting optimization problem is a standard nonlinear program (NLP) on a Euclidean space with a large system of equality constraints approximating the original DAEs. This procedure makes standard methods in nonlinear programming applicable in principle, but typically generates very large-scale NLPs. On the other hand, the resulting NLPs are also highly structured, and the development of specialized interior point algorithms over the past several years has made this approach attractive for many problem classes [11].

In contrast, the sequential approach makes use of state-of-the-art dynamic simulation software to embed the DAE solution in the evaluation of the objective and constraint functions. This again leads to a standard NLP, with the caveat that the objective and constraint functions are not known explicitly as functions of the decision variables, but rather are evaluated by numerical solution of the embedded DAEs. The primary advantage of this scheme is that the resulting NLP is potentially much smaller than the NLP generated through the simultaneous approach because no decisions are introduced through discretization. On the other hand, the objective and constraint functions of this NLP, as well as their derivatives, can be costly to evaluate and have limited accuracy due to the embedded simulation. Nonetheless, this approach has become quite powerful due to the development of efficient and robust methods for numerical integration and sensitivity analysis of DAE systems [6, 24, 27, 49].

In general, local dynamic optimization algorithms require much less computational time than global algorithms. Unfortunately, local algorithms can only guarantee globally optimal solutions under restrictive convexity assumptions which are often violated in practical applications. For example, it has been shown that dynamic optimization problems arising in chemical engineering applications are very commonly nonconvex and exhibit multiple suboptimal local minima, especially when nonlinear models of chemical reaction kinetics are involved [45, 55]. The search for global solutions is well motivated in many such applications. One need only consider the problem of maximizing the profitability of a process. Clearly, a significant economic penalty may be incurred by designing and operating such a process according to a suboptimal local solution [85]. However, other

applications pose more serious problems. In parameter estimation problems, one is often interested in determining whether a model, equipped with its best fit parameter estimates, is consistent with measured data according to a statistical significance test. However, if only locally optimal parameter estimates are available, any conclusions drawn from such an analysis are dubious [52, 88].

One approach for solving dynamic optimization problems globally is the so-called dynamic programming approach, based on Bellman's principle of optimality [7]. However, this requires the solution of a boundary value problem in PDEs that becomes intractable for systems with many states [93]. Thus, more versatile algorithms have been developed as global extensions of the sequential and simultaneous approaches discussed above. Since both of these methods involve a reformulation of the original dynamic optimization problem to a standard NLP on a Euclidean space, the main idea is to combine these reformulations with the spatial branch-and-bound (B&B) global optimization framework for NLPs [67]. In the case of the simultaneous approach, this is conceptually straightforward [15, 21]. However, spatial B&B has worst-case exponential scaling in the number of decisions, which is problematic for the large-scale NLPs generated by the simultaneous approach. As a result, recent efforts in global dynamic optimization have primarily focused on the sequential approach. However, for this approach the application of spatial B&B is challenging. In particular, the objective and constraint functions in the resulting NLP are not known explicitly, but rather are defined implicitly through the solution of the embedded dynamic system, and this fact precludes the use of standard lower bounding procedures in the spatial B&B algorithm (see Sect. 7.1). In fact, establishing a valid lower bounding procedure for the sequential formulation requires a method for computing an enclosure of the reachable set, thus establishing the fundamental connection between global dynamic optimization and reachability analysis.

The first method for overcoming this problem was proposed in [20] using a lower bounding procedure based on a dynamic extension of the  $\alpha$ BB method [1] known as  $\beta$ BB. However, the validity of the procedure depends on a user specified parameter, which must exceed a threshold value that is not known in general. This procedure was first made generally valid in [62] using an interval-based reachable set enclosure method. Notably, this method applies only to embedded ODE systems. In [86, 89], an alternative method was proposed based on the use of convex and concave relaxations of the state variables in the underlying reachability computation. Lin and Stadtherr proposed a further method in [41] using a sophisticated variant of the interval Taylor series reachable set enclosure methods discussed above [42]. Developments in this area are ongoing and have been primarily focused on improved state relaxation techniques for use in the spatial B&B algorithm [30, 70, 71, 82, 84]. Recently, the state relaxation methods in [82, 84] have been extended to treat semi-explicit index-one DAEs in [74, 81], making it possible to solve dynamic optimization problems with DAEs embedded to guaranteed global optimality for the first time. The essential features of this optimization algorithm are reviewed herein.

The purpose of this article is to give a self-contained review of the major concepts and tools necessary to address reachability analysis and global optimization problems for DAE systems. The problems considered here are stated formally in Sect. 2. In Sect. 3, the basic tools for computing global information about a given function, such as interval bounds and convex and concave relaxations, are presented for functions known explicitly in closed-form. The essential concepts here are the *factorable representation* of a function and its *natural interval* and *McCormick extensions*. These concepts are applied in Sect. 4 to obtain methods for bounding and relaxing the parametric solutions of systems of nonlinear algebraic equations. This is a direct prerequisite for treating differential-algebraic systems in subsequent sections, and is also the first example of a reoccurring theme: global information can be obtained for non-factorable functions that are defined as the solutions of equation systems with factorable governing equations. In Sects. 5 and 6, this theme is taken further to provide bounds and relaxations of the parametric solutions of DAEs. With these reachability algorithms as the enabling technology, global dynamic optimization is considered in Sect. 7. A numerical case study is presented in Sect. 8, and concluding remarks are given in Sect. 9.

Finally, we note that this review is intended as a tutorial overview of the specific results of several recent papers by the authors, rather than a high level survey of a large body of literature. This is primarily because the field is nascent, and with few exceptions noted above, the material reviewed here is the only available approach. At the same time, alternative approaches for problems with ODEs that seem likely to have fruitful extensions to DAEs are based on such similar foundational concepts that this review should serve as a useful introductory reference for them as well. In their original published form, the results and methods reviewed herein are presented in a highly technical manner, and are distributed among several papers in such a way that their synthesis into complete algorithms is difficult to appreciate. Thus, our intent is to provide a clear but thorough introduction that will encourage and direct further research into reachability and global optimization problems with nonlinear DAE models.

## 2 Problem Formulation

### 2.1 Notation

In the remainder of this article, vector quantities are denoted in bold, while scalar quantities are written without emphasis. For any  $\mathbf{v} \in \mathbb{R}^n$ , the standard  $p$ -norms are denoted by  $\|\mathbf{v}\|_p = (\sum_{i=1}^n |v_i|^p)^{1/p}$ ,  $1 \leq p < \infty$ , and  $\|\mathbf{v}\|_\infty = \max_i |v_i|$ . For  $\mathbf{v}, \mathbf{w}, \mathbf{u} \in \mathbb{R}^n$ , the order relations  $\mathbf{v} \leq \mathbf{w}$  and  $\mathbf{v} < \mathbf{w}$  denote that these relations hold component-wise. Similarly,  $\min(\mathbf{v}, \mathbf{w})$  and  $\max(\mathbf{v}, \mathbf{w})$  denote the vectors with components  $\min(v_i, w_i)$  and  $\max(v_i, w_i)$ , respectively, and  $\text{mid}(\mathbf{v}, \mathbf{w}, \mathbf{u})$  denotes the vector where each component is the middle value of  $v_i$ ,  $w_i$ , and  $u_i$ . Let  $C^k(D, \mathbb{R}^m)$

denote the set of  $k$ -times continuously differentiable functions from  $D$  into  $\mathbb{R}^m$ . For  $D_s \subset \mathbb{R}^{n_s}$ ,  $D_r \subset \mathbb{R}^{n_r}$ , and  $\ell \in C^k(D_s \times D_r, \mathbb{R}^m)$ , let  $\frac{\partial \ell}{\partial \mathbf{r}}(\hat{\mathbf{s}}, \hat{\mathbf{r}})$  denote the Jacobian matrix of  $\ell(\hat{\mathbf{s}}, \cdot)$  at  $\hat{\mathbf{r}} \in D_r$ .

## 2.2 Semi-explicit Index-One DAEs

This article considers exclusively the initial value problem in semi-explicit index-one DAEs

$$\left. \begin{aligned} \dot{\mathbf{x}}(t, \mathbf{p}) &= \mathbf{f}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \\ \mathbf{0} &= \mathbf{g}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \end{aligned} \right\}, \quad (2.1a)$$

$$\mathbf{x}(t_0, \mathbf{p}) = \mathbf{x}_0(\mathbf{p}), \quad (2.1b)$$

where  $t$  is the independent variable,  $\mathbf{p}$  is the vector of problem parameters,  $\dot{\mathbf{x}}(t, \mathbf{p})$  denotes the derivative of  $\mathbf{x}(\cdot, \mathbf{p})$  at  $t$ ,  $t_0$  is the initial time, and  $\mathbf{x}_0$  specifies the parametric initial conditions. It is assumed that  $\mathbf{f} \in C^1(D_t \times D_p \times D_x \times D_y, \mathbb{R}^{n_x})$ ,  $\mathbf{g} \in C^1(D_t \times D_p \times D_x \times D_y, \mathbb{R}^{n_y})$ , and  $\mathbf{x}_0 \in C^1(D_p, D_x)$ , where  $D_t \subset \mathbb{R}$ ,  $D_p \subset \mathbb{R}^{n_p}$ ,  $D_x \subset \mathbb{R}^{n_x}$ , and  $D_y \subset \mathbb{R}^{n_y}$  are open sets. A function  $(\mathbf{x}, \mathbf{y}) \in C^1(I \times P, D_x) \times C^1(I \times P, D_y)$  is called a *solution of (2.1)* on  $I \times P \subset D_t \times D_p$  if (2.1) holds for all  $(t, \mathbf{p}) \in I \times P$ .

The special form of the DAEs (2.1) is called the *semi-explicit* form. In addition to considering this special form, we will further restrict our attention to index-one, or *regular* solutions (although it is customary to refer to a DAE system as index-one or high-index, the differential index is actually a property of solutions). A solution  $(\mathbf{x}, \mathbf{y}) \in C^1(I \times P, D_x) \times C^1(I \times P, D_y)$  is called an index-one solution, or a regular solution, if

$$\det \frac{\partial \mathbf{g}}{\partial \mathbf{y}}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \neq 0, \quad \forall (t, \mathbf{p}) \in I \times P. \quad (2.2)$$

Note that, for any regular solution of (2.1) on  $I \times P$ , a single differentiation of the algebraic equations  $\mathbf{g}$  gives the underlying ODEs

$$\dot{\mathbf{x}}(t, \mathbf{p}) = \mathbf{f}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})), \quad (2.3)$$

$$\dot{\mathbf{y}}(t, \mathbf{p}) = - \left( \frac{\partial \mathbf{g}}{\partial \mathbf{y}} \right)^{-1} \left( \frac{\partial \mathbf{g}}{\partial \mathbf{x}} \mathbf{f}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) + \frac{\partial \mathbf{g}}{\partial t} \right), \quad (2.4)$$

for all  $(t, \mathbf{p}) \in I \times P$ , where all derivatives of  $\mathbf{g}$  are evaluated at  $(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}))$ .

The DAEs (2.1) have the following existence and uniqueness properties (see Theorems 4.13 and 4.18 in [36]):

- Given any point  $(t_0, \hat{\mathbf{p}}, \hat{\mathbf{x}}_0, \hat{\mathbf{y}}_0)$  such that  $\mathbf{x}_0(\hat{\mathbf{p}}) = \hat{\mathbf{x}}_0$ ,  $\mathbf{g}(t_0, \hat{\mathbf{p}}, \hat{\mathbf{x}}_0, \hat{\mathbf{y}}_0) = \mathbf{0}$ , and  $\det \frac{\partial \mathbf{g}}{\partial \mathbf{y}}(t_0, \hat{\mathbf{p}}, \hat{\mathbf{x}}_0, \hat{\mathbf{y}}_0) \neq 0$ , there exists a regular solution  $(\mathbf{x}, \mathbf{y})$  of (2.1) defined on some sufficiently small open ball around  $(t_0, \hat{\mathbf{p}})$  and satisfying  $(\mathbf{x}(t_0, \hat{\mathbf{p}}), \mathbf{y}(t_0, \hat{\mathbf{p}})) = (\hat{\mathbf{x}}_0, \hat{\mathbf{y}}_0)$ .
- If  $(\mathbf{x}, \mathbf{y})$  and  $(\mathbf{x}^*, \mathbf{y}^*)$  are solutions of (2.1) on  $I \times P$ ,  $(\mathbf{x}, \mathbf{y})$  is regular,  $P$  is connected, and there exists  $\hat{\mathbf{p}} \in P$  such that  $\mathbf{y}(t_0, \hat{\mathbf{p}}) = \mathbf{y}^*(t_0, \hat{\mathbf{p}})$ , then  $\mathbf{x}(t, \mathbf{p}) = \mathbf{x}^*(t, \mathbf{p})$  and  $\mathbf{y}(t, \mathbf{p}) = \mathbf{y}^*(t, \mathbf{p})$ ,  $\forall (t, \mathbf{p}) \in I \times P$ .

In order to guarantee that (2.1) has a unique regular solution, the conditions above suggest that we must add a further specification on the initial condition of the form  $\mathbf{y}(t_0, \hat{\mathbf{p}}) = \hat{\mathbf{y}}_0$ . Indeed, without this condition, it is possible for (2.1) to have multiple solutions, even if both solutions are regular. The regularity only implies that any two such solutions must be completely isolated from one another (i.e., they cannot both satisfy  $\mathbf{y}(t_0, \hat{\mathbf{p}}) = \hat{\mathbf{y}}_0$ ). The following example illustrates the need for this condition.

*Example 2.1* Let  $I \equiv [0, \delta] \subset D_t = \mathbb{R}$ ,  $D_p = \emptyset$ ,  $D_x = D_y = \mathbb{R}$ , and define  $g(t, z_x, z_y) = z_y^2 - z_x$ . With fixed initial condition  $x_0 = 1$  at  $t_0 = 0$ , there are two possible values for  $y(t_0)$  satisfying  $g(t_0, x(t_0), y(t_0)) = 0$ ;  $y(t_0) = 1$  and  $y(t_0) = -1$ . Letting  $f(t, z_x, z_y) = 1$ , clearly  $x(t) = 1 + t$  satisfies  $\dot{x}(t) = 1 = f(t, x(t), y(t))$  for any  $y : I \rightarrow \mathbb{R}$ . However, both  $y(t) = \sqrt{1+t}$  and  $y(t) = -\sqrt{1+t}$  result in  $g(t, x(t), y(t)) = (y(t))^2 - x(t) = 0$ . In particular,  $y(t) = \sqrt{1+t}$  satisfies (2.1) with the additional specification  $y(t_0) = 1$ , while  $y(t) = -\sqrt{1+t}$  satisfies (2.1) along with the additional specification  $y(t_0) = -1$ .

### 2.3 Reachable Set Enclosures

This article considers two types of enclosures of the reachable set of (2.1). The first type of enclosure, discussed in Sect. 5, is an interval enclosure described by component-wise upper and lower bounds on each state. In particular, let  $I = [t_0, t_f]$ , let  $P$  be a compact, convex set, and let  $(\mathbf{x}, \mathbf{y})$  be a regular solution of (2.1) on  $I \times P$ . Then, our objective is to compute functions  $\mathbf{x}^L, \mathbf{x}^U : I \rightarrow \mathbb{R}^{n_x}$  and  $\mathbf{y}^L, \mathbf{y}^U : I \rightarrow \mathbb{R}^{n_y}$  such that

$$\mathbf{x}^L(t) \leq \mathbf{x}(t, \mathbf{p}) \leq \mathbf{x}^U(t) \quad \text{and} \quad \mathbf{y}^L(t) \leq \mathbf{y}(t, \mathbf{p}) \leq \mathbf{y}^U(t), \quad \forall (t, \mathbf{p}) \in I \times P. \quad (2.5)$$

These functions are referred to as *state bounds* for the solution  $(\mathbf{x}, \mathbf{y})$ . An interesting feature of the methods discussed is that the existence of a unique regular solution satisfying the given initial data need not be assumed; it can be verified computationally by the bounding method.

The second type of reachable set enclosure, discussed in Sect. 6, is characterized by convex and concave relaxations. Recall that, for  $D \subset \mathbb{R}^n$  convex, a vector function  $\mathbf{w} : D \rightarrow \mathbb{R}^m$  is called convex if each component is convex; i.e.,

$$\mathbf{w}(\lambda \mathbf{z}_1 + (1 - \lambda) \mathbf{z}_2) \leq \lambda \mathbf{w}(\mathbf{z}_1) + (1 - \lambda) \mathbf{w}(\mathbf{z}_2), \quad \forall (\lambda, \mathbf{z}_1, \mathbf{z}_2) \in [0, 1] \times D \times D,$$

and it is called concave if the opposite (weak) inequality holds. For arbitrary  $\mathbf{w} : D \rightarrow \mathbb{R}^m$  with  $D$  convex, the functions  $\mathbf{w}^{cv}, \mathbf{w}^{cc} : D \rightarrow \mathbb{R}^m$  are called *convex and concave relaxations for  $\mathbf{w}$  on  $D$* , respectively, if  $\mathbf{w}^{cv}$  is convex on  $D$ ,  $\mathbf{w}^{cc}$  is concave on  $D$ , and

$$\mathbf{w}^{cv}(\mathbf{z}) \leq \mathbf{w}(\mathbf{z}) \leq \mathbf{w}^{cc}(\mathbf{z}), \quad \forall \mathbf{z} \in D. \quad (2.6)$$

Returning to (2.1), given a regular solution  $(\mathbf{x}, \mathbf{y})$  on  $I \times P$ , our objective is to compute functions  $\mathbf{x}^{cv}, \mathbf{x}^{cc} : I \times P \rightarrow \mathbb{R}^{n_x}$  and  $\mathbf{y}^{cv}, \mathbf{y}^{cc} : I \times P \rightarrow \mathbb{R}^{n_y}$  such that

$$\mathbf{x}^{cv}(t, \mathbf{p}) \leq \mathbf{x}(t, \mathbf{p}) \leq \mathbf{x}^{cc}(t, \mathbf{p}) \quad \text{and} \quad \mathbf{y}^{cv}(t, \mathbf{p}) \leq \mathbf{y}(t, \mathbf{p}) \leq \mathbf{y}^{cc}(t, \mathbf{p}), \quad (2.7)$$

for all  $(t, \mathbf{p}) \in I \times P$ , and, for each  $t \in I$ ,  $\mathbf{x}^{cv}(t, \cdot)$  and  $\mathbf{y}^{cv}(t, \cdot)$  are convex on  $P$ , and  $\mathbf{x}^{cc}(t, \cdot)$  and  $\mathbf{y}^{cc}(t, \cdot)$  are concave on  $P$ . These functions are called *state relaxations for  $(\mathbf{x}, \mathbf{y})$  on  $I \times P$* . In Sect. 6, it will be shown that these relaxations describe a convex enclosure of the reachable set that can be outer-approximated by a polytope of any desired complexity [75]. State relaxations often provide a tighter enclosure of the reachable set than do state bounds. Moreover, state relaxations are better suited for use in a global dynamic optimization algorithm, as will be shown in Sect. 7.

## 2.4 Global Dynamic Optimization

The second problem considered in this article is the global solution of the dynamic optimization problem

$$\begin{aligned} \min_{\mathbf{p} \in P} \quad & \phi(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) \\ \text{s.t.} \quad & \eta(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) \leq \mathbf{0}. \end{aligned} \quad (2.8)$$

Above,  $\phi$  and  $\eta$  are continuous functions of the form  $(\phi, \eta) : D_p \times D_x \times D_y \rightarrow \mathbb{R} \times \mathbb{R}^{n_c}$ , and  $(\mathbf{x}, \mathbf{y})$  satisfies (2.1) on  $I \times P$ , along with the additional initial condition

$$\mathbf{y}(t_0, \hat{\mathbf{p}}) = \hat{\mathbf{y}}_0, \quad (2.9)$$

where  $\hat{\mathbf{p}} \in P$  and  $\hat{\mathbf{y}}_0 \in D_y$  satisfy  $\mathbf{g}(t_0, \hat{\mathbf{p}}, \mathbf{x}_0(\hat{\mathbf{p}}), \hat{\mathbf{y}}_0) = \mathbf{0}$  and  $\det \frac{\partial \mathbf{g}}{\partial \mathbf{y}}(t_0, \hat{\mathbf{p}}, \mathbf{x}_0(\hat{\mathbf{p}}), \hat{\mathbf{y}}_0) \neq 0$ . Note that  $\mathbf{x}$  and  $\mathbf{y}$  are well-defined functions of  $(t, \mathbf{p}) \in I \times P$  only if the solution of (2.1) with (2.9) exists and is unique for all  $(t, \mathbf{p}) \in I \times P$ . As discussed in Sect. 2.2, the condition (2.9) ensures local existence. For (2.8) to be well-posed, it is assumed that a solution exists on all of  $I \times P$ . Given this assumption, (2.9) ensures uniqueness on  $I \times P$ . In most applications, there is a consistent initial condition of interest, so that the specification (2.9) is easily made. On the other hand, if one is interested in an optimization problem that considers all possible solutions of (2.1), then some additional method will be required for exhaustively enumerating such solutions. No such method has yet been proposed in the literature.

As discussed in Sect. 1, there are many algorithms available for solving (2.8) to local optimality. In contrast, our concern here is with algorithms that are guaranteed to terminate finitely with an  $\epsilon$ -global optimum  $\mathbf{p}^*$ . In particular, for a user specified  $\epsilon > 0$ ,  $\mathbf{p}^*$  satisfies  $\boldsymbol{\eta}(\mathbf{p}^*, \mathbf{x}(t_f, \mathbf{p}^*), \mathbf{y}(t_f, \mathbf{p}^*)) \leq \mathbf{0}$  and

$$\phi(\mathbf{p}^*, \mathbf{x}(t_f, \mathbf{p}^*), \mathbf{y}(t_f, \mathbf{p}^*)) \leq \phi(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) + \epsilon, \quad (2.10)$$

for all  $\mathbf{p} \in P$  such that  $\boldsymbol{\eta}(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) \leq \mathbf{0}$ .

Note that (2.8) considers only optimization problems with a real vector of decision variables  $\mathbf{p}$ . Although many dynamic optimization problems take this form (e.g., parameter estimation), there are also many problems of practical interest involving continuous control inputs  $u(t)$  as decisions. For such problems, the methods discussed here can be used only after applying *control parameterization*. Control parameterization refers to the approximation of the control inputs by functions that can be characterized by a finite number of real parameters (e.g., polynomials, piece-wise affine controls, etc.). In most cases, control parameterization can be done with minimal error, and is common practice in state-of-the-art dynamic optimization algorithms [93]. Nonetheless, the reader should note that the guarantee of global optimality provided by the method discussed herein applies to the parameterized problem (2.8), and any inferences about the original problem are subject to the parameterization error. A method for overcoming this limitation in the case of ODE embedded problems has recently been proposed in [30].

Without yet considering the details of solving (2.8) globally, it is possible to intuitively appreciate the relationship between (2.8) and reachability analysis. In short, a global solver must provide a guarantee that the located solution  $\mathbf{p}^*$  minimizes the function  $\phi(\cdot, \mathbf{x}(t_f, \cdot), \mathbf{y}(t_f, \cdot))$  among all feasible  $\mathbf{p} \in P$ . Clearly such a guarantee cannot be made unless one has characterized all possible values of  $\mathbf{x}(t_f, \cdot)$  and  $\mathbf{y}(t_f, \cdot)$  that can be obtained with  $\mathbf{p} \in P$ , which is exactly the reachability problem discussed in the previous section.

The problem (2.8) admits several generalizations that we have omitted for notational convenience:

1. Integral terms can be included in the objective and constraints; i.e.,

$$\begin{aligned} \min_{\mathbf{p} \in P} \quad & \phi(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) + \int_{t_0}^{t_f} \psi(s, \mathbf{p}, \mathbf{x}(s, \mathbf{p}), \mathbf{y}(s, \mathbf{p})) ds \\ \text{s.t.} \quad & \eta(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) + \int_{t_0}^{t_f} \ell(s, \mathbf{p}, \mathbf{x}(s, \mathbf{p}), \mathbf{y}(s, \mathbf{p})) ds \leq \mathbf{0}. \end{aligned} \quad (2.11)$$

Problems of this type can always be recast in the form of (2.8) by introducing quadrature variables as described in [74].

2. The objective and constraints may contain values of the states at multiple time points  $t_0, \dots, t_m \in I$ :

$$\begin{aligned} \min_{\mathbf{p} \in P} \quad & \phi(\mathbf{p}, \mathbf{x}(t_0, \mathbf{p}), \dots, \mathbf{x}(t_m, \mathbf{p}), \mathbf{y}(t_0, \mathbf{p}), \dots, \mathbf{y}(t_m, \mathbf{p})) \\ \text{s.t.} \quad & \eta(\mathbf{p}, \mathbf{x}(t_0, \mathbf{p}), \dots, \mathbf{x}(t_m, \mathbf{p}), \mathbf{y}(t_0, \mathbf{p}), \dots, \mathbf{y}(t_m, \mathbf{p})) \leq \mathbf{0}. \end{aligned} \quad (2.12)$$

The algorithm for solving (2.8) presented in Sect. 7 is easily extended to this case. The restriction to final time terms only simplifies the notation.

### 3 Factorable Functions, Interval Arithmetic, and McCormick Relaxations

Computing interval bounds and/or convex and concave relaxations of a given function requires global information about that function. In general, local characterizations of a function, such as its value or its derivative at a point, are not enough. Rather, one requires information about the behavior of the function on the entire domain of interest. An essential tool in this regard is the so-called *factorable representation* of a function [50, 56]. Indeed, the primary complication in computing the state bounds and state relaxations defined in Sect. 2.3 is that the parametric solutions  $\mathbf{x}(t, \cdot)$  and  $\mathbf{y}(t, \cdot)$  are not known in closed-form, but rather are evaluated by numerical integration. Because of this, they have no known factorable representation to work from. Nonetheless, factorable representations, as well as the methods for computing bounds and relaxations of them, will be central to the state bounding and relaxation methods for DAEs presented in the following sections. Therefore, these concepts are developed in detail in this section.

Informally, a function is called *factorable* if it can be written as a finite sequence of simple operations, including basic arithmetic operations as well as intrinsic functions available on a computer. For example, the function

$$h(s_1, s_2) = 10s_1 + s_2 e^{s_2} \quad (3.1)$$

is factorable because it can be evaluated for any  $(s_1, s_2) \in \mathbb{R}^2$  by executing the following sequence of simple computations:

$$\begin{aligned}
 v_1(s_1, s_2) &= s_1, \\
 v_2(s_1, s_2) &= s_2, \\
 v_3(s_1, s_2) &= 10v_1(s_1, s_2), \\
 v_4(s_1, s_2) &= \exp(v_2(s_1, s_2)), \\
 v_5(s_1, s_2) &= v_2(s_1, s_2) \times v_4(s_1, s_2), \\
 v_6(s_1, s_2) &= v_3(s_1, s_2) + v_5(s_1, s_2), \\
 h(s_1, s_2) &= v_6(s_1, s_2).
 \end{aligned}$$

Each of the intermediates  $v_i$  is called a factor, and the factorable representation is the sequence  $v_1, \dots, v_6$ . In essence, any function written explicitly in computer code is factorable.

To formalize the notion of a factorable function, we must first define the set of operations that will be permissible in the sequence of computations defining such functions. This set will contain binary addition, binary multiplication, and elements of a *library of univariate functions*  $u : B \subset \mathbb{R} \rightarrow \mathbb{R}$ , denoted by  $\mathcal{L}$ . The elements of  $\mathcal{L}$  will be used to represent functions such as  $\sqrt{s}$ ,  $s^n$ ,  $e^s$ ,  $\sin s$ , etc. Furthermore,  $\mathcal{L}$  should include the negative and reciprocal functions  $-s$  and  $1/s$ , so that subtraction and division can be achieved by combination with binary addition and multiplication. Any univariate function of interest may be included in  $\mathcal{L}$ , provided that certain information related to bounding and relaxing  $u$  is available (see Assumptions 3.1–3.2). This information has been collected for a large number of common univariate operations in [83].

**Definition 3.1** A function  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$  is *factorable* if it can be expressed in terms of a finite number of factors  $v_1, \dots, v_q : D_s \rightarrow \mathbb{R}$  such that  $v_i(\mathbf{s}) = s_i$  for  $i = 1, \dots, n$ ,  $v_k(\mathbf{s})$  is defined for each  $n < k \leq q$  as either

- (a)  $v_k(\mathbf{s}) = v_i(\mathbf{s}) + v_j(\mathbf{s})$ , with  $i, j < k$ , or
- (b)  $v_k(\mathbf{s}) = v_i(\mathbf{s})v_j(\mathbf{s})$ , with  $i, j < k$ , or
- (c)  $v_k(\mathbf{s}) = u_k(v_i(\mathbf{s}))$ , with  $i < k$ ,  $u_k \in \mathcal{L}$ ,

and  $\mathbf{h}(\mathbf{s}) = (v_{i(1)}(\mathbf{s}), \dots, v_{i(m)}(\mathbf{s}))$  for some indices  $i(1), \dots, i(m) \in \{1, \dots, q\}$ .

### 3.1 Interval Arithmetic

For  $a, b \in \mathbb{R}$ ,  $a \leq b$ , define the *interval*  $[a, b]$  as the compact, connected set  $\{x \in \mathbb{R} : a \leq x \leq b\}$ . The set of all nonempty intervals is denoted  $\mathbb{IR}$ . Intervals are denoted by capital letters,  $S \in \mathbb{IR}$ . Since  $S$  is a subset of  $\mathbb{R}$ , the notation  $s \in S$  is

well-defined. The set of  $n$ -dimensional interval vectors is denoted  $\mathbb{IR}^n$ . In particular,  $S \in \mathbb{IR}^n$  has elements  $S_i \in \mathbb{IR}$ ,  $i = 1, \dots, n$ . Every  $S \in \mathbb{IR}^n$  can be regarded as a subset of  $\mathbb{R}^n$  defined by the Cartesian product  $S_1 \times \dots \times S_n$ , so that  $\mathbf{s} \in \mathbb{R}^n$  satisfies  $\mathbf{s} \in S$  if  $s_i \in S_i$ ,  $i = 1, \dots, n$ . The set of  $n \times m$  interval matrices is denoted  $\mathbb{IR}^{n \times m}$  and defined analogously to  $\mathbb{IR}^n$ ;  $A \in \mathbb{IR}^{n \times m}$  has elements  $A_{ij} \in \mathbb{IR}$ , for all  $i \in \{1, \dots, n\}$  and  $j \in \{1, \dots, m\}$ , and, for any  $\mathbf{A} \in \mathbb{R}^{n \times m}$  with elements  $a_{ij}$ ,  $\mathbf{A} \in A$  if  $a_{ij} \in A_{ij}$  for all indices  $i$  and  $j$ . For any  $D \subset \mathbb{R}^n$ , let  $\mathbb{I}D$  denote the set  $\{S \in \mathbb{IR}^n : S \subset D\}$ . This notation is also used for  $D \subset \mathbb{R}^{n \times m}$ .

If  $\mathbf{v}, \mathbf{w} \in \mathbb{R}^n$  and  $\mathbf{v} \leq \mathbf{w}$ , then  $[\mathbf{v}, \mathbf{w}]$  denotes the  $n$ -dimensional interval  $[v_1, w_1] \times \dots \times [v_n, w_n]$ . Moreover, for any  $S \in \mathbb{IR}$ , the notation  $\mathbf{s}^L, \mathbf{s}^U \in \mathbb{R}^n$  will be commonly used to denote the vectors such that  $S = [\mathbf{s}^L, \mathbf{s}^U]$ . The notation  $m(S)$  denotes the *midpoint* of  $S$ ,  $m(S) \equiv 0.5(\mathbf{s}^L + \mathbf{s}^U)$ , and  $w(S)$  denotes the *width* of  $S$ ,  $w(S) \equiv \mathbf{s}^U - \mathbf{s}^L$ . For  $A \in \mathbb{IR}^{n \times m}$ ,  $m(A)$  and  $w(A)$  are real-valued matrices defined analogously. For any  $\mathbf{s} \in \mathbb{R}^n$ , the singleton  $[\mathbf{s}, \mathbf{s}]$  is called a *degenerate* interval.

The central task in interval analysis is to compute an interval which encloses the range of a given function [56].

**Definition 3.2** Let  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$ . An interval mapping  $H : \mathbb{I}D_s \rightarrow \mathbb{IR}^m$  is an *inclusion function* for  $\mathbf{h}$  if  $\mathbf{h}(S) \subset H(S)$ ,  $\forall S \in \mathbb{I}D_s$ , where  $\mathbf{h}(S)$  is the image of  $S$  under  $\mathbf{h}$ .

Typically, inclusion functions are derived from a simpler object known as an *interval extension*. The function  $H$  is an *interval extension* of  $\mathbf{h}$  if, for every  $\mathbf{s} \in D_s$ ,  $H([\mathbf{s}, \mathbf{s}]) = [\mathbf{h}(\mathbf{s}), \mathbf{h}(\mathbf{s})]$ . It is *inclusion monotonic* if

$$S_1 \subset S_2 \implies H(S_1) \subset H(S_2), \quad \forall S_1, S_2 \in \mathbb{I}D_s. \quad (3.2)$$

The following theorem is a central result in interval analysis:

**Theorem 3.1** Let  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$ . If  $H : \mathbb{I}D_s \rightarrow \mathbb{IR}^m$  is an inclusion monotonic interval extension of  $\mathbf{h}$ , then it is an inclusion function for  $\mathbf{h}$ .

Inclusion monotonic interval extensions for binary addition and multiplication are given by the formulas

$$S + Q = [s^L + q^L, s^U + q^U], \quad (3.3)$$

$$SQ = [\min(s^L q^L, s^L q^U, s^U q^L, s^U q^U), \max(s^L q^L, s^L q^U, s^U q^L, s^U q^U)], \quad (3.4)$$

where  $S = [s^L, s^U]$  and  $Q = [q^L, q^U]$ . Moreover, formulas are readily available for many common univariate functions [56, 59]. We make the following assumption throughout:

**Assumption 3.1** For every  $u \in \mathcal{L}$ ,  $u : B \subset \mathbb{R} \rightarrow \mathbb{R}$ , an inclusion monotonic interval extension  $[u] : \mathbb{I}B \rightarrow \mathbb{IR}$  is available and can be evaluated computationally.

For any factorable function  $\mathbf{h}$ , one can compute a particular interval extension called the *natural interval extension* and denoted by  $[\mathbf{h}] : \mathbb{I}D_s \rightarrow \mathbb{IR}^m$  (the notations

$[\mathbf{h}]^L(S)$  and  $[\mathbf{h}]^U(S)$  are used to denote the lower and upper bounds of  $[\mathbf{h}](S)$ , respectively). The natural interval extension is computed by recursively applying the known interval extensions of the factors of  $\mathbf{h}$ . That is, each operation in the definition of  $\mathbf{h}$  (Definition 3.1) is replaced by its interval counterpart. For example, if  $h$  is defined by (3.1), then

$$[h](S_1, S_2) = 10S_1 + S_2[\exp](S_2), \quad (3.5)$$

$$= [10s_1^L, 10s_1^U] + [s_2^L, s_2^U][e^{s_2^L}, e^{s_2^U}], \quad (3.6)$$

$$= [10s_1^L, 10s_1^U] + [\min(s_2^L e^{s_2^L}, s_2^L e^{s_2^U}, s_2^U e^{s_2^L}, s_2^U e^{s_2^U}), \quad (3.7)$$

$$\max(s_2^L e^{s_2^L}, s_2^L e^{s_2^U}, s_2^U e^{s_2^L}, s_2^U e^{s_2^U})],$$

$$= [10s_1^L + \min(s_2^L e^{s_2^L}, s_2^L e^{s_2^U}, s_2^U e^{s_2^L}, s_2^U e^{s_2^U}), \quad (3.8)$$

$$10s_1^U + \max(s_2^L e^{s_2^L}, s_2^L e^{s_2^U}, s_2^U e^{s_2^L}, s_2^U e^{s_2^U})]. \quad (3.9)$$

The natural interval extension of a factorable function is inclusion monotonic, and therefore defines an inclusion function. The reader is referred to [56, 59] for further details on interval analysis.

### 3.2 McCormick Relaxations

McCormick's relaxation technique [50] is a method for computing convex and concave relaxations of a given factorable function  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$ . Given an interval  $S = [s^L, s^U] \subset D_s$  and a point  $\mathbf{s} \in S$ , the method computes three quantities associated with each factor  $v_k$  in Definition 3.1:  $V_k(S)$ ,  $v_k^{cv}(S, \mathbf{s})$ , and  $v_k^{cc}(S, \mathbf{s})$ .  $V_k(S)$  is an interval bound for  $v_k$  on  $S$  computed using interval arithmetic. The remaining numbers  $v_k^{cv}(S, \mathbf{s})$  and  $v_k^{cc}(S, \mathbf{s})$  are the values of convex and concave relaxations of  $v_k$  on  $S$ , respectively, evaluated at  $\mathbf{s}$ . For every  $k \leq n$ , we have  $v_k = s_k$ , so these quantities can be trivially assigned as  $(V_k(S), v_k^{cv}(S, \mathbf{s}), v_k^{cc}(S, \mathbf{s})) = (S_k, s_k, s_k)$ . For every successive factor  $v_k$ , relaxations and bounds are computed recursively using known rules based on the definition of  $v_k$  in Definition 3.1. For example, if  $v_k(\mathbf{s}) = v_i(\mathbf{s}) + v_j(\mathbf{s})$ , we assign

$$V_k(S) = V_i(S) + V_j(S), \quad (3.10)$$

$$v_k^{cv}(S, \mathbf{s}) = \max(V_i^L(S), v_i^{cv}(S, \mathbf{s})) + \max(V_j^L(S), v_j^{cv}(S, \mathbf{s})), \quad (3.11)$$

$$v_k^{cc}(S, \mathbf{s}) = \min(V_i^U(S), v_i^{cc}(S, \mathbf{s})) + \min(V_j^U(S), v_j^{cc}(S, \mathbf{s})). \quad (3.12)$$

It is straightforward to see that  $v_k(\mathbf{s}) \in [v_k^{cv}(S, \mathbf{s}), v_k^{cc}(S, \mathbf{s})]$ ,  $\forall \mathbf{s} \in S$ , and convexity of  $v_k^{cv}(S, \cdot)$  (resp. concavity of  $v_k^{cc}(S, \cdot)$ ) follows from the well-known results that

the minimum (resp. maximum) of two convex (resp. concave) functions is convex (resp. concave) and the sum of two convex (resp. concave) functions is convex (resp. concave). Rules for multiplication and composition with many common univariate functions are readily available [83]. Noting that  $\mathbf{h}(\mathbf{s}) = (v_{i(1)}(\mathbf{s}), \dots, v_{i(m)}(\mathbf{s}))$ , this procedure provides convex and concave relaxations for  $\mathbf{h}$  on  $S$  through the definitions  $h_j^{cv}(\mathbf{s}) = v_{i(j)}^{cv}(S, \mathbf{s})$  and  $h_j^{cc}(\mathbf{s}) = v_{i(j)}^{cc}(S, \mathbf{s})$ ,  $j = 1, \dots, m$ .

On account of the initialization  $(V_k(S), v_k^{cv}(S, \mathbf{s}), v_k^{cc}(S, \mathbf{s})) = (S_k, s_k, s_k)$ ,  $k \leq n$ , McCormick's relaxation procedure can be thought of as a mapping of the form  $(S, \mathbf{s}) \mapsto (H(S), \mathbf{h}^{cv}(\mathbf{s}), \mathbf{h}^{cc}(\mathbf{s}))$ . However, in [83], it was observed that generalizing this initialization to  $(V_k(S), v_k^{cv}(S, \mathbf{s}), v_k^{cc}(S, \mathbf{s})) = (S_k, s_k^{cv}, s_k^{cc})$ ,  $k \leq n$ , where the inputs satisfy  $S \cap [s^{cv}, s^{cc}] \neq \emptyset$ , results in a generalized mapping  $(S, \mathbf{s}^{cv}, \mathbf{s}^{cc}) \mapsto (H(S), \mathbf{u}(\mathbf{s}^{cv}, \mathbf{s}^{cc}), \mathbf{o}(\mathbf{s}^{cv}, \mathbf{s}^{cc}))$  with an interesting interpretation:  $\mathbf{u}$  and  $\mathbf{o}$  are *composite relaxations* for  $\mathbf{h}$  on  $S$ .

**Definition 3.3** Let  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$  and let  $S \subset D_s$  be convex. Two functions  $\mathbf{u}, \mathbf{o} : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}^m$  are called convex and concave composite relaxations for  $\mathbf{h}$  on  $S$ , respectively, if the following condition holds: For any convex set  $P \subset \mathbb{R}^q$  and any functions  $\mathbf{s}, \mathbf{s}^{cv}, \mathbf{s}^{cc} : P \rightarrow \mathbb{R}^n$  such that  $\mathbf{s}(\mathbf{p}) \in S$ ,  $\forall \mathbf{p} \in P$ , and  $\mathbf{s}^{cv}$  and  $\mathbf{s}^{cc}$  are, respectively, convex and concave relaxations of  $\mathbf{s}$  on  $P$ , convex and concave relaxations of the composite mapping  $\mathbf{h}(\mathbf{s}(\cdot))$  are given by  $\mathbf{u}(\mathbf{s}^{cv}(\cdot), \mathbf{s}^{cc}(\cdot))$  and  $\mathbf{o}(\mathbf{s}^{cv}(\cdot), \mathbf{s}^{cc}(\cdot))$ , respectively.

Composite relaxations are central to the relaxation theory for DAEs presented in Sect. 6. Given a function  $\psi : P \rightarrow \mathbb{R}^m$  defined by  $\psi(\mathbf{p}) \equiv \mathbf{h}(\mathbf{s}(\mathbf{p}))$  with  $\mathbf{s} : P \rightarrow D_s$ , composite relaxations of  $\mathbf{h}$  provide a means to compute relaxations for  $\psi$  on  $P$  given bounds and relaxations for  $\mathbf{s}$  on  $P$ . Note that this requires factorability of  $\mathbf{h}$ , but not of  $\psi$ . The notion of composite relaxations can also be used to treat the case where  $\mathbf{h} : P \times D_s \rightarrow \mathbb{R}^m$  and  $\psi(\mathbf{p}) \equiv \mathbf{h}(\mathbf{p}, \mathbf{s}(\mathbf{p}))$ . By initializing the factors corresponding to the arguments  $\mathbf{p}$  as in the standard relaxation procedure and using the generalized initialization for the factors corresponding to  $\mathbf{s}$ , one obtains a mapping of the form  $(S, \mathbf{p}, \mathbf{s}^{cv}, \mathbf{s}^{cc}) \mapsto (H(S), \mathbf{u}(\mathbf{p}, \mathbf{s}^{cv}, \mathbf{s}^{cc}), \mathbf{o}(\mathbf{p}, \mathbf{s}^{cv}, \mathbf{s}^{cc}))$  with the obvious properties.

Since McCormick's relaxation technique is based on the recursive application of relaxation rules for a set of basic operations, it is convenient to denote the procedure explicitly for a given function, e.g., for (3.1) as

$$\{h\}(\mathcal{S}_1, \mathcal{S}_2) = 10\mathcal{S}_1 + \mathcal{S}_2\{\exp\}(\mathcal{S}_2). \quad (3.13)$$

Formalizing this type of expression requires a more abstract development of McCormick's procedure and is intimately related to the concept of composite relaxations. To begin, we define the space of *McCormick objects*

$$\mathbb{MR}^n \equiv \{\mathcal{S} = (S^B, S^C) \in \mathbb{IR}^n \times \mathbb{IR}^n : S^B \cap S^C \neq \emptyset\}. \quad (3.14)$$

Elements of  $\mathbb{MR}^n$  are denoted by script capitals. For any  $\mathcal{S} \in \mathbb{MR}^n$ , the notations  $S^B, S^C \in \mathbb{IR}^n$ , and  $\mathbf{s}^L, \mathbf{s}^U, \mathbf{s}^{cv}, \mathbf{s}^{cc} \in \mathbb{R}^n$  will commonly be used to denote the intervals and vectors satisfying  $\mathcal{S} = (S^B, S^C) = ([\mathbf{s}^L, \mathbf{s}^U], [\mathbf{s}^{cv}, \mathbf{s}^{cc}])$ . As with intervals, the set

$\text{MR}^{n \times m}$  can be defined analogously to  $\text{MR}^n$ ;  $\mathcal{A} \in \text{MR}^{n \times m}$  has elements  $\mathcal{A}_{ij} \in \text{MR}$ , for all  $i \in \{1, \dots, n\}$  and  $j \in \{1, \dots, m\}$ . For any  $D \subset \mathbb{R}^n$ , let  $\text{MD}$  denote the set  $\{\mathcal{S} \in \text{MR}^n : S^B \subset D\}$ . This notation is also used for  $D \subset \mathbb{R}^{n \times m}$ .

Now, consider again a factorable function  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$ . Using the notation above, the McCormick relaxation of  $\mathbf{h}$  can be formalized as a mapping of the form  $\{\mathbf{h}\} : \text{MD}_s \rightarrow \text{MR}^m$ , as suggested by (3.13). This mapping is a *relaxation function* for  $\mathbf{h}$ .

**Definition 3.4** Let  $\mathbf{h} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$  and let  $S \subset D_s$  be convex. A function  $\mathcal{H} : \text{MD}_s \rightarrow \text{MR}^m$  is a relaxation function for  $\mathbf{h}$  if, for every  $\mathcal{S} = (S, [\mathbf{s}^{cv}, \mathbf{s}^{cc}])$ ,  $\mathcal{H}(\mathcal{S}) = (H(S), [\mathbf{u}(S, \mathbf{s}^{cv}, \mathbf{s}^{cc}), \mathbf{o}(S, \mathbf{s}^{cv}, \mathbf{s}^{cc})])$ , where  $H$  is an inclusion function for  $\mathbf{h}$  and  $\mathbf{u}(S, \cdot, \cdot)$  and  $\mathbf{o}(S, \cdot, \cdot)$  are, respectively, convex and concave composite relaxations for  $\mathbf{h}$  on  $S$ .

To construct  $\{\mathbf{h}\}$ , McCormick's procedure makes use of relaxation functions for the basic operations  $+$ ,  $\times$ , and  $u \in \mathcal{L}$  appearing in the factorable representation of  $\mathbf{h}$ . For example, the relaxation function for binary addition is defined as

$$\mathcal{X} + \mathcal{Y} = (X^B + Y^B, (X^C \cap X^B) + (Y^C \cap Y^B)), \quad (3.15)$$

which agrees with (3.10). A relaxation function for binary multiplication can be similarly defined [74], and relaxation functions for many common univariate functions and are compiled in [74, 83]. We make the following assumption throughout.

**Assumption 3.2** For every  $u \in \mathcal{L}$ ,  $u : B \subset \mathbb{R} \rightarrow \mathbb{R}$ , a relaxation function  $\{u\} : \text{MB} \rightarrow \text{MR}$  is available and can be evaluated computationally.

Naturally,  $\{\mathbf{h}\}$  is constructed by applying the relaxation functions of the basic operations  $+$ ,  $\times$ , and  $u \in \mathcal{L}$  sequentially according to the factorable representation of  $\mathbf{h}$ . Thus,  $\{\mathbf{h}\}$  is a particular relaxation function for  $\mathbf{h}$  called the *natural McCormick extension of  $\mathbf{h}$* . For example, the natural McCormick extension of (3.1) is given by (3.13), which is now well-defined. The fact that this procedure defines a relaxation function for  $\mathbf{h}$  is a consequence of the following basic composition result.

**Theorem 3.2** Let  $\mathbf{v} : D_s \subset \mathbb{R}^n \rightarrow \mathbb{R}^m$  and  $o : B \subset \mathbb{R}^m \rightarrow \mathbb{R}$  have relaxation functions  $\mathcal{V} : \text{MD}_s \rightarrow \text{MR}^m$  and  $\mathcal{O} : \text{MB} \rightarrow \text{MR}$ , respectively. If  $\mathbf{v}(D_s) \subset B$  and  $\mathcal{V}(\text{MD}_s) \subset \text{MB}$ , then  $\mathcal{O} \circ \mathcal{V}$  is a relaxation function for  $o \circ \mathbf{v}$ .

*Remark 3.1* For simplicity of exposition, the definition of a relaxation function here is slightly different than the original definition in [74]. For a function  $\mathcal{H} : \text{MD}_s \rightarrow \text{MR}^m$ , it can be shown that if  $\mathcal{H}$  is a relaxation function as defined here, then it is also a relaxation function as defined in [74]. The converse holds provided that  $\mathcal{H}$  satisfies an inclusion monotonicity property (Definition 2.4.13, [74]). Accordingly, the composition result in [74] (Lemmas 2.4.15 and 2.4.17) require inclusion monotonicity, while Theorem 3.2 here is a direct consequence of Definitions 3.3 and 3.4 above.

For any  $\mathcal{S} = (S, [\mathbf{s}^{cv}, \mathbf{s}^{cc}]) \in \mathbb{M}D_s$ , the natural McCormick extension of  $\mathbf{h}$  evaluates to  $\{\mathbf{h}\}(\mathcal{S}) = ([\mathbf{h}](S), [\mathbf{u}(S, \mathbf{s}^{cv}, \mathbf{s}^{cc}), \mathbf{o}(S, \mathbf{s}^{cv}, \mathbf{s}^{cc})])$ , where  $[\mathbf{h}]$  is the natural interval extension of  $\mathbf{h}$  and  $\mathbf{u}(S, \cdot, \cdot)$  and  $\mathbf{o}(S, \cdot, \cdot)$  are convex and concave composite relaxations for  $\mathbf{h}$  on  $S$ , respectively. Convex and concave relaxations are recovered from  $\{\mathbf{h}\}$  by simply considering arguments of the form  $\mathcal{S} = (S, [\mathbf{s}, \mathbf{s}])$  for  $\mathbf{s} \in S$ . This is equivalent to the initialization  $(V_k(S), v_k^{cv}(S_k, \mathbf{s}), v_k^{cc}(S, \mathbf{s})) = (S_k, s_k, s_k)$ ,  $k \leq n$ , discussed above. For any such  $\mathcal{S}$ , we have  $\{\mathbf{h}\}(\mathcal{S}) = ([\mathbf{h}](S), [\mathbf{u}(S, \mathbf{s}, \mathbf{s}), \mathbf{o}(S, \mathbf{s}, \mathbf{s})])$ , and convex and concave relaxations of  $\mathbf{h}$  on  $S$  are given by the definitions  $\mathbf{h}^{cv}(\mathbf{s}) = \mathbf{u}(S, \mathbf{s}, \mathbf{s})$  and  $\mathbf{h}^{cc}(\mathbf{s}) = \mathbf{o}(S, \mathbf{s}, \mathbf{s})$ , respectively. This follows from Definition 3.3 after choosing  $P \equiv S$  and observing that the identity mapping  $P = S \ni \mathbf{s} \mapsto \mathbf{s} \in S$  is both a convex and a concave relaxation of itself on  $P$ . The reader is referred to [53, 74, 83] for a comprehensive treatment of McCormick relaxations.

### 3.3 Implementation

Several computational tools are available for performing interval and McCormick arithmetic operations. Of course, any function written explicitly in computer code is factorable, and its factorable representation can be generated automatically by parsing this code. Thus, it is possible to use code generation to create additional subroutines that compute the natural interval and McCormick extensions of such a function by replacing the standard arithmetic operations with interval and McCormick operations, respectively. A more flexible way to achieve this is through so-called operator overloading, which is available in most object oriented programming languages (e.g., C++). In this scheme, one can create interval and McCormick objects as new variable types with defined rules for the standard arithmetic operations. Then, the existing code for a factorable function can be executed with these objects to obtain the natural interval and McCormick extensions. A number of libraries defining interval objects and their associated arithmetic are freely available. For example, the Boost interval library in C++ ([http://www.boost.org/doc/libs/1\\_55\\_0/libs/numeric/interval/doc/interval.htm](http://www.boost.org/doc/libs/1_55_0/libs/numeric/interval/doc/interval.htm)) and the MATLAB toolbox INTLAB (<http://www.ti3.tu-harburg.de/~rump/intlab/>). Similarly, relaxation functions can be computed using the McCormick arithmetic library MC++ (<http://www3.imperial.ac.uk/people/b.chachuat/research>).

## 4 Bounds and Relaxations for Implicit Functions

In this section, procedures are derived for computing interval bounds and convex and concave relaxations for functions defined implicitly as the solutions of systems of algebraic equations. This is a fundamental step required for treating DAEs in the

following two sections. In general, we consider the nonlinear algebraic equations

$$\ell(\mathbf{s}, \mathbf{r}) = \mathbf{0}, \quad (4.1)$$

where  $\ell \in C^k(D_s \times D_r, \mathbb{R}^n)$ ,  $k \geq 1$ , and  $D_s \subset \mathbb{R}^{n_s}$  and  $D_r \subset \mathbb{R}^n$  are open sets. Given an interval  $S \subset D_s$ , our primary interest is in the case where (4.1) defines a unique implicit function  $\mathbf{h} : S \rightarrow \mathbb{R}^n$  such that  $\ell(\mathbf{s}, \mathbf{h}(\mathbf{s})) = \mathbf{0}$ ,  $\forall \mathbf{s} \in S$ . In this case, bounds and relaxations for  $\mathbf{h}$  on  $S$  are well-defined. More generally, we can define inclusion and relaxation functions for  $\mathbf{h}$  on  $S$  of the form  $H : \mathbb{I}S \rightarrow \mathbb{I}\mathbb{R}^n$  and  $\mathcal{H} : \mathbb{M}S \rightarrow \mathbb{M}\mathbb{R}^n$ , respectively.

In doing so, the key challenge that must be addressed is that  $\mathbf{h}$  is implicitly defined via (4.1), and hence no factorable representation of  $\mathbf{h}$  is available in general. Of course, this implies that the methods of Sect. 3 cannot be applied directly. Instead, the key idea here is to use the factorable representation of  $\ell$ , and hence the ability to compute inclusion and relaxation functions for  $\ell$ , to infer the required information about  $\mathbf{h}$  through (4.1). This approach originated with the development of so-called *interval Newton methods* [59], and has recently been extended to compute relaxations in [91]. To follow this approach here, the following basic assumption is required.

**Assumption 4.1** *The functions  $\ell$  and  $\frac{\partial \ell}{\partial \mathbf{r}}$  are factorable and have natural interval and McCormick extensions  $[\ell] : \mathbb{I}D_s \times \mathbb{I}D_r \rightarrow \mathbb{I}\mathbb{R}^n$ ,  $[\frac{\partial \ell}{\partial \mathbf{r}}] : \mathbb{I}D_s \times \mathbb{I}D_r \rightarrow \mathbb{I}\mathbb{R}^{n \times n}$ ,  $\{\ell\} : \mathbb{M}D_s \times \mathbb{M}D_r \rightarrow \mathbb{M}\mathbb{R}^n$ , and  $\{\frac{\partial \ell}{\partial \mathbf{r}}\} : \mathbb{M}D_s \times \mathbb{M}D_r \rightarrow \mathbb{M}\mathbb{R}^{n \times n}$ .*

*Remark 4.1* Although any factorable function  $\mathbf{w} : D \rightarrow \mathbb{R}^n$  has natural interval and McCormick extensions, the content of the above assumption is that these are defined on all of  $\mathbb{I}D$  and  $\mathbb{M}D$ , respectively. For example,  $w(s) = 1/(s - \sqrt{s})$  is well defined on  $D = (1, +\infty]$ . However, overestimation in the interval evaluation of the denominator leads to a division by zero error for  $S = [4, 100] \in \mathbb{I}D$ , so that the interval extension is not defined at this point:  $[w](S) = 1/([4, 100] - [2, 10]) = 1/[-6, 98] = \text{NaN}$ . Assuming well-defined interval and McCormick extensions on all of  $\mathbb{M}D$  is not essential for the following developments, but simplifies the presentation.

This simplest way to infer information about  $\mathbf{h}$  from the factorable representation of  $\ell$  is through a direct rearrangement of (4.1) of the form  $\mathbf{r} = \mathbf{h}(\mathbf{s})$ ,  $\forall \mathbf{s} \in D_s$ . However, such a rearrangement is rarely possible. Instead, we pursue the more modest feat of deriving a factorable function  $\psi$  that satisfies  $\mathbf{r} = \psi(\mathbf{s}, \mathbf{r})$  for all  $(\mathbf{s}, \mathbf{r}) \in D_s \times D_r$  such that  $\ell(\mathbf{s}, \mathbf{r}) = \mathbf{0}$ . In general, even this cannot be accomplished precisely as written due to some technicalities discussed below. However, we will derive a similar expression and then provide a computational test capable of verifying that this expression is well-defined, and that a unique implicit function  $\mathbf{h}$  exists, on some given intervals  $S \times R$ . This then implies that  $\mathbf{h}(\mathbf{s}) = \psi(\mathbf{s}, \mathbf{h}(\mathbf{s}))$ ,  $\forall \mathbf{s} \in S$ , which provides the opportunity to compute inclusion and relaxation functions for  $\mathbf{h}$  through iterative procedures.

We begin by presenting the details of a particular interval Newton method called the interval Hansen-Sengupta method [59]. The existence of a unique implicit function to be bounded is not assumed a priori. Instead, the algorithm is centered around the following more general task: Given intervals  $S \subset D_s$  and  $R \subset D_r$ , compute a refined interval  $R' \subset R$  that contains every  $\mathbf{r} \in R$  for which (4.1) holds for some  $\mathbf{s} \in S$ . If it is possible to derive a factorable function  $\psi$  satisfying the implication

$$(\mathbf{s}, \mathbf{r}) \in S \times R, \quad \ell(\mathbf{s}, \mathbf{r}) = \mathbf{0} \quad \implies \quad \mathbf{r} = \psi(\mathbf{s}, \mathbf{r}), \quad (4.2)$$

then this task is accomplished by the update  $R' = R \cap [\psi](S, R)$ , where  $[\psi]$  is the natural interval extension of  $\psi$ .

The key step in deriving such a function is an application of the mean-value theorem. Choose a fixed reference point  $\tilde{\mathbf{r}} \in R$ . For each fixed  $\mathbf{s} \in S$  and  $i \in \{1, \dots, n\}$ , applying the mean-value theorem to the function  $\ell_i(\mathbf{s}, \cdot)$  ensures that there exists  $\lambda_i(\mathbf{s}) \in [0, 1]$  such that

$$\frac{\partial \ell_i}{\partial \mathbf{r}}(\mathbf{s}, \tilde{\mathbf{r}} + \lambda_i(\mathbf{s})(\mathbf{r} - \tilde{\mathbf{r}})) (\mathbf{r} - \tilde{\mathbf{r}}) = \ell_i(\mathbf{s}, \mathbf{r}) - \ell_i(\mathbf{s}, \tilde{\mathbf{r}}). \quad (4.3)$$

Define  $\lambda : S \rightarrow [0, 1]$  by choosing  $\lambda_i(\mathbf{s})$  in this way for all  $\mathbf{s} \in S$  and every  $i \in \{1, \dots, n\}$ . Additionally, define

$$\mathbf{M}(\mathbf{s}, \mathbf{r}, \lambda') \equiv \begin{bmatrix} \frac{\partial \ell_1}{\partial \mathbf{r}}(\mathbf{s}, \tilde{\mathbf{r}} + \lambda'_1(\mathbf{r} - \tilde{\mathbf{r}})) \\ \vdots \\ \frac{\partial \ell_n}{\partial \mathbf{r}}(\mathbf{s}, \tilde{\mathbf{r}} + \lambda'_n(\mathbf{r} - \tilde{\mathbf{r}})) \end{bmatrix}, \quad \forall (\mathbf{s}, \mathbf{r}, \lambda') \in D_s \times D_r \times \mathbb{R}^n. \quad (4.4)$$

Then, from the definition of  $\lambda$ ,  $\mathbf{M}(\mathbf{s}, \mathbf{r}, \lambda(\mathbf{s})) (\mathbf{r} - \tilde{\mathbf{r}}) = \ell(\mathbf{s}, \mathbf{r}) - \ell(\mathbf{s}, \tilde{\mathbf{r}})$ ,  $\forall (\mathbf{s}, \mathbf{r}) \in S \times R$ . In particular,

$$(\mathbf{s}, \mathbf{r}) \in S \times R, \quad \ell(\mathbf{s}, \mathbf{r}) = \mathbf{0} \quad \implies \quad \mathbf{M}(\mathbf{s}, \mathbf{r}, \lambda(\mathbf{s})) (\mathbf{r} - \tilde{\mathbf{r}}) = -\ell(\mathbf{s}, \tilde{\mathbf{r}}). \quad (4.5)$$

Thus, any  $(\mathbf{s}, \mathbf{r})$  satisfying (4.1) must have  $(\mathbf{r} - \tilde{\mathbf{r}})$  as a solution of an  $n \times n$  linear system. To obtain (4.2) from (4.5), we consider rearrangements of these linear equations that isolate each  $r_i$  on the left-hand side. Assume for the moment that the diagonal elements of  $\mathbf{M}(\mathbf{s}, \mathbf{r}, \lambda(\mathbf{s}))$  are nonzero. Then, the  $i$ th linear relation above can be rearranged as

$$r_i = \tilde{r}_i + \frac{1}{m_{ii}(\mathbf{s}, \mathbf{r}, \lambda(\mathbf{s}))} \left( -\ell_i(\mathbf{s}, \tilde{\mathbf{r}}) - \sum_{j \neq i} m_{ij}(\mathbf{s}, \mathbf{r}, \lambda(\mathbf{s})) (r_j - \tilde{r}_j) \right). \quad (4.6)$$

With this in mind, we make the following definition for every  $(\mathbf{s}, \mathbf{r}, \boldsymbol{\lambda}') \in D_s \times D_r \times \mathbb{R}^n$  such that  $m_{ii}(\mathbf{s}, \mathbf{r}, \boldsymbol{\lambda}') \neq 0$  for all  $i \in \{1, \dots, n\}$ :

$$\boldsymbol{\psi}(\mathbf{s}, \mathbf{r}, \boldsymbol{\lambda}') \equiv \mathbf{r}', \quad (4.7)$$

where, using the abbreviation  $m_{ij} = m_{ij}(\mathbf{s}, \mathbf{r}, \boldsymbol{\lambda}')$ , for  $i = 1, \dots, n$ ,

$$r'_i = \tilde{r}_i + \frac{1}{m_{ii}} \left( -\ell_i(\mathbf{s}, \tilde{\mathbf{r}}) - \sum_{j < i} m_{ij}(r'_j - \tilde{r}_j) - \sum_{j > i} m_{ij}(r_j - \tilde{r}_j) \right). \quad (4.8)$$

The reason for splitting the sum above will become clear shortly. In any case, it is easily shown from (4.6) that this definition satisfies

$$(\mathbf{s}, \mathbf{r}) \in S \times R, \quad \ell(\mathbf{s}, \mathbf{r}) = \mathbf{0} \quad \implies \quad \mathbf{r} = \boldsymbol{\psi}(\mathbf{s}, \mathbf{r}, \boldsymbol{\lambda}(\mathbf{s})). \quad (4.9)$$

In this last implication,  $\boldsymbol{\psi}$  is not quite in the form we set out to derive due to the third argument,  $\boldsymbol{\lambda}(\mathbf{s})$ . However, in this form,  $\boldsymbol{\psi}$  is a factorable function wherever it is defined. Indeed, by Assumption 4.1, both  $\ell$  and  $\frac{\partial \ell}{\partial \mathbf{r}}$  are factorable. Thus,  $\mathbf{M}$  is factorable, and by (4.8), so is  $\boldsymbol{\psi}$ . On the other hand,  $\boldsymbol{\lambda}$  is a theoretical construct that is known to exist on account of the mean-value theorem, but does not have a known factorable representation. Fortunately, we are assured that  $\boldsymbol{\lambda}(\mathbf{s}) \in [\mathbf{0}, \mathbf{1}]$ ,  $\forall \mathbf{s} \in S$ , so that a factorable representation is not necessary in order to bound  $\boldsymbol{\lambda}$ . Thus, our task of computing a refined interval  $R'$  is now accomplished through the update

$$R' = R \cap [\boldsymbol{\psi}](S, R, [\mathbf{0}, \mathbf{1}]). \quad (4.10)$$

Here  $[\boldsymbol{\psi}]$  is the natural interval extension of  $\boldsymbol{\psi}$ , derived by applying interval arithmetic to (4.7) and (4.8). In particular,  $[\boldsymbol{\psi}](S, R, [\mathbf{0}, \mathbf{1}]) \equiv R'$ , where, using the abbreviation  $[m_{ij}] = [m_{ij}](S, R, [\mathbf{0}, \mathbf{1}])$ ,

$$R'_i = \tilde{r}_i + \frac{1}{[m_{ii}]} \left( -[\ell_i](S, \tilde{\mathbf{r}}) - \sum_{j < i} [m_{ij}](R'_j - \tilde{r}_j) - \sum_{j > i} [m_{ij}](R_j - \tilde{r}_j) \right). \quad (4.11)$$

For the moment, it is assumed that  $0 \notin [m_{ii}](S, R, [\mathbf{0}, \mathbf{1}])$  for every  $i$ , so that interval division by  $[m_{ii}](S, R, [\mathbf{0}, \mathbf{1}])$  is defined [56]. Note that, by splitting the sum in the definition of  $\boldsymbol{\psi}$ , we have arranged that  $[\psi_i]$  is computed with the updated intervals  $R'_j$  in place of  $R_j$  for all  $j < i$ , which helps to tighten the refinement  $R'$  in (4.10). In fact, a further improvement can be achieved by carrying out the intersection

in (4.10) component-wise during the computation of  $[\psi]$ . Thus, we further define  $[\psi_\cap](S, R, [\mathbf{0}, \mathbf{1}]) \equiv R'$ , where, using the abbreviation  $[m_{ij}] = [m_{ij}](S, R, [\mathbf{0}, \mathbf{1}])$ ,

$$R'_i = R_i \cap \left( \tilde{r}_i + \frac{1}{[m_{ii}]} \left( -[\ell_i](S, \tilde{\mathbf{r}}) - \sum_{j < i} [m_{ij}](R'_j - \tilde{r}_j) - \sum_{j > i} [m_{ij}](R_j - \tilde{r}_j) \right) \right). \quad (4.12)$$

This gives the modified update  $R' = [\psi_\cap](S, R, [\mathbf{0}, \mathbf{1}])$ .

In order to extend the definition of  $[\psi_\cap]$  to intervals  $S$  and  $R$  such that  $0 \in [m_{ii}](S, R, [\mathbf{0}, \mathbf{1}])$  for some  $i$ , it is necessary to define a special interval operator:

$$\Gamma(A, B, Z) \equiv \text{hull}(\{z \in Z : az = b, a \in A, b \in B\}), \quad (4.13)$$

where  $\text{hull}(Q)$  denotes the smallest interval containing the set  $Q$ . Given an a priori bound  $Z$ ,  $\Gamma(A, B, Z)$  is a bound on the solutions  $z \in Z$  of a single interval linear equation, even when rearranging that equation for  $z$  would involve division by an interval containing zero. It has been shown that  $\Gamma$  can be evaluated computationally through the simple rules

$$\Gamma(A, B, Z) \equiv \begin{cases} (B/A) \cap Z & \text{if } 0 \notin A \\ \text{hull}(Z \setminus \text{int}([b^L/a^L, b^L/a^U])) & \text{if } 0 \in A \text{ and } b^L > 0 \\ \text{hull}(Z \setminus \text{int}([b^U/a^U, b^U/a^L])) & \text{if } 0 \in A \text{ and } b^U < 0 \\ Z & \text{if } 0 \in A \text{ and } 0 \in B \end{cases}, \quad (4.14)$$

where  $\text{int}(Q)$  is the interior of  $Q$  [59]. Using  $\Gamma$ , we now define  $[\psi_\Gamma](S, R, [\mathbf{0}, \mathbf{1}]) \equiv R'$ , where, using the abbreviation  $[m_{ij}] = [m_{ij}](S, R, [\mathbf{0}, \mathbf{1}])$ ,

$$R'_i = \tilde{r}_i + \Gamma \left( [m_{ii}], -[\ell_i](S, \tilde{\mathbf{r}}) - \sum_{j < i} [m_{ij}](R'_j - \tilde{r}_j) - \sum_{j > i} [m_{ij}](R_j - \tilde{r}_j), (R_i - \tilde{r}_i) \right). \quad (4.15)$$

In contrast to  $[\psi_\cap]$ ,  $[\psi_\Gamma]$  is defined for all arguments with  $(S, R) \in \mathbb{I}D_s \times \mathbb{I}D_r$ . Moreover, we have the following variant of Theorem 5.1.8 in [59] proven in [78]:

**Theorem 4.1** *Let  $S \in \mathbb{I}D_s$ ,  $R \in \mathbb{I}D_r$ ,  $\tilde{\mathbf{r}} \in R$ , and define  $R' \equiv [\psi_\Gamma](S, R, [\mathbf{0}, \mathbf{1}])$ . The following conclusions hold:*

1. *If  $(\mathbf{s}, \mathbf{r}) \in S \times R$  satisfies  $\ell(\mathbf{s}, \mathbf{r}) = \mathbf{0}$ , then  $\mathbf{r} \in R'$ .*
2. *If  $R' = \emptyset$ , then  $\exists \mathbf{z}(\mathbf{s}, \mathbf{r}) \in S \times R$  such that  $\ell(\mathbf{s}, \mathbf{r}) = \mathbf{0}$ .*
3. *If  $\tilde{\mathbf{r}} \in \text{int}(R)$  and  $\emptyset \neq R' \subset \text{int}(R)$ , then  $\exists \mathbf{h} \in C^k(S, R')$  such that, for every  $\mathbf{s} \in S$ ,  $\mathbf{r} = \mathbf{h}(\mathbf{s})$  is the unique element of  $R$  satisfying  $\ell(\mathbf{s}, \mathbf{r}) = \mathbf{0}$ . Moreover,  $[\mathbf{M}](S, R, [\mathbf{0}, \mathbf{1}])$  does not contain a singular matrix and does not contain zero in any of its diagonal elements.*

*Remark 4.2* Since  $\tilde{\mathbf{r}} \in R$ , it is straightforward to show that  $\tilde{\mathbf{r}} + [\mathbf{0}, \mathbf{1}](R - \tilde{\mathbf{r}}) = R$ , and hence

$$[\mathbf{M}](S, R, [\mathbf{0}, \mathbf{1}]) \equiv \begin{bmatrix} [\frac{\partial \ell_1}{\partial \mathbf{r}}](S, R) \\ \vdots \\ [\frac{\partial \ell_n}{\partial \mathbf{r}}](S, R) \end{bmatrix} = \left[ \frac{\partial \ell}{\partial \mathbf{r}} \right] (S, R). \quad (4.16)$$

The statement of Theorem 4.1 in [78] uses this substitution. Conclusion 3 of Theorem 4.1 then states that if  $\tilde{\mathbf{r}} \in \text{int}(R)$  and the inclusion  $\emptyset \neq [\psi_r](S, R, [\mathbf{0}, \mathbf{1}]) \subset \text{int}(R)$  holds, then  $\left[ \frac{\partial \ell}{\partial \mathbf{r}} \right] (S, R)$  contains no singular matrices and does not contain zero in any of its diagonal elements. In practice, it is almost always necessary to precondition (4.1) by a non-singular matrix  $\mathbf{C} \in \mathbb{R}^{n \times n}$  for this to be true, and hence for the desired inclusion to hold. A common choice is the midpoint inverse,  $\mathbf{C} \equiv (m(\left[ \frac{\partial \ell}{\partial \mathbf{r}} \right] (S, R)))^{-1}$ , in which case  $[\mathbf{M}](S, R, [\mathbf{0}, \mathbf{1}]) = \mathbf{C} \left[ \frac{\partial \ell}{\partial \mathbf{r}} \right] (S, R)$  is an interval enclosure of the identity matrix. For ease of notation, the preconditioner  $\mathbf{C}$  is omitted from Theorem 4.1 and all subsequent developments.

Without further note, we will always define  $[\psi_r](S, R, [\mathbf{0}, \mathbf{1}])$  with  $\tilde{\mathbf{r}} = m(R)$ , so that  $\tilde{\mathbf{r}} \in \text{int}(R)$  whenever  $R$  has nonempty interior. Now, suppose that intervals  $S^*$  and  $R^*$  have been found such that  $\emptyset \neq [\psi_r](S^*, R^*, [\mathbf{0}, \mathbf{1}]) \subset \text{int}(R^*)$ . Then, Conclusion 3 of Theorem 4.1 ensures the existence of an implicit function satisfying  $\ell(\mathbf{s}, \mathbf{h}(\mathbf{s})) = \mathbf{0}$  on all of  $S^*$ , and further ensures that  $\mathbf{h}$  is the only such function taking values in  $R^*$ . Thus, the inclusion test of Conclusion 3 isolates a single parametric solution branch of (4.1), and provides the bound  $\mathbf{h}(S^*) \subset [\psi_r](S^*, R^*, [\mathbf{0}, \mathbf{1}]) \subset \text{int}(R^*)$ . Applying Conclusion 1 of Theorem 4.1, this bound can be iteratively refined by the scheme

$$R_{k+1} = [\psi_r](S^*, R_k, [\mathbf{0}, \mathbf{1}]), \quad R_0 = R^*. \quad (4.17)$$

This is known as the interval Hansen-Sengupta method. More generally, this scheme defines a sequence of progressively tighter inclusion functions  $H_k : \mathbb{I}S^* \rightarrow \mathbb{I}\mathbb{R}^n$  for  $\mathbf{h}$  via

$$H_{k+1}(S) = [\psi_r](S, H_k(S), [\mathbf{0}, \mathbf{1}]), \quad H_0(S) = R^*, \quad \forall S \in \mathbb{I}S^*. \quad (4.18)$$

The fact that  $[\mathbf{M}](S^*, R^*, [\mathbf{0}, \mathbf{1}])$  does not contain zero in any of its diagonal elements also has important consequences. Most importantly, it follows that the matrix  $\mathbf{M}(\mathbf{s}, \mathbf{r}, \lambda(\mathbf{s}))$  has nonzero diagonals for all  $(\mathbf{s}, \mathbf{r}) \in S^* \times R^*$ , so that  $\psi$  is defined on all of  $(\mathbf{s}, \mathbf{r}) \in S^* \times R^*$  and, by (4.9),

$$\mathbf{h}(\mathbf{s}) = \psi(\mathbf{s}, \mathbf{h}(\mathbf{s})), \quad \forall \mathbf{s} \in S^*. \quad (4.19)$$

Additionally, it can be shown that  $[\psi_\Gamma](R, S, [\mathbf{0}, \mathbf{1}]) = [\psi_\cap](R, S, [\mathbf{0}, \mathbf{1}])$ ,  $\forall (S, R) \in \mathbb{IS}^* \times \mathbb{IR}^*$ , and, using the fact that  $\emptyset \neq [\psi_\Gamma](S^*, R^*, [\mathbf{0}, \mathbf{1}]) \subset \text{int}(R^*)$ ,

$$[\psi_\Gamma](R^*, S^*, [\mathbf{0}, \mathbf{1}]) = [\psi_\cap](R^*, S^*, [\mathbf{0}, \mathbf{1}]) = [\psi](R^*, S^*, [\mathbf{0}, \mathbf{1}]). \quad (4.20)$$

We are now prepared to show how the developments of Sect. 3.2 can be used to compute convex and concave relaxations of the implicit function  $\mathbf{h}$  on  $S \in \mathbb{IS}^*$ . It is assumed that the inclusion test of Theorem 4.1 has been passed with  $(S^*, R^*)$ , so that  $\mathbf{h} : S^* \rightarrow R^*$  exists. Moreover, it is assumed that an inclusion function  $H : \mathbb{IS}^* \rightarrow \mathbb{IR}^n$  is available, e.g., by truncation of (4.18). By analogy to (4.18), we will derive an iterative scheme that generates refined inclusion functions  $\mathcal{H}_k : \mathbb{MS}^* \rightarrow \mathbb{MR}^n$ . Recall that  $\mathbb{MS}^*$  contains all  $\mathcal{S} = (S, [\mathbf{s}^{cv}, \mathbf{s}^{cc}])$  with  $S \subset S^*$  (although we will be primarily concerned with arguments of the form  $\mathcal{S} = (S, [\mathbf{s}, \mathbf{s}])$  with  $\mathbf{s} \in S$ ). To initialize this scheme, we note that one relaxation function is given by

$$\mathcal{H}_0(\mathcal{S}) = (H(S), H(S)), \quad \forall \mathcal{S} = (S, [\mathbf{s}^{cv}, \mathbf{s}^{cc}]) \in \mathbb{MS}^*. \quad (4.21)$$

Using the notation  $[\mathbf{h}^L(S), \mathbf{h}^U(S)] = H(S)$ , this is true because the constant functions  $\mathbf{h}^{cv}(\mathbf{s}) = \mathbf{h}^L(S)$  and  $\mathbf{h}^{cc}(\mathbf{s}) = \mathbf{h}^U(S)$  are trivially convex and concave relaxations of  $\mathbf{h}$  on  $S$ .

In order to refine  $\mathcal{H}_0$ , we again use the semi-explicit characterization of  $\mathbf{h}$  given in (4.19). To begin, note that  $\lambda(\mathbf{s}) \in [\mathbf{0}, \mathbf{1}]$ ,  $\forall \mathbf{s} \in S$ , so that the constant function  $\mathbb{MS}^* \ni \mathcal{S} \mapsto \mathcal{L} \equiv ([\mathbf{0}, \mathbf{1}], [\mathbf{0}, \mathbf{1}]) \in \mathbb{MR}^n$  is a relaxation function for  $\lambda$  on  $\mathbb{MS}^*$ . Now, since  $\psi$  is factorable, the natural relaxation function  $\{\psi\}$  can be computed by simply applying McCormick arithmetic to (4.8). However, it is more useful to define  $\{\psi_\cap\}$  by analogy to  $[\psi_\cap]$  as

$$\{\psi_\cap\}(\mathcal{S}, \mathcal{R}, \mathcal{L}) \equiv \mathcal{R}', \quad (4.22)$$

where, using the abbreviation  $\{m_{ij}\} = \{m_{ij}\}(\mathcal{S}, \mathcal{R}, \mathcal{L})$ ,

$$\mathcal{R}'_i \equiv \mathcal{R}_i \cap \left( \tilde{r}_i + \frac{1}{\{m_{ii}\}} \left( -\{\ell_i\}(\mathcal{S}, \tilde{\mathbf{r}}) - \sum_{j < i} \{m_{ij}\}(\mathcal{R}'_j - \tilde{r}_j) - \sum_{j > i} \{m_{ij}\}(\mathcal{R}_j - \tilde{r}_j) \right) \right). \quad (4.23)$$

The intersection above is defined for arbitrary  $\mathcal{R}$  and  $\overline{\mathcal{R}}$  in  $\mathbb{MR}^n$  by  $\mathcal{R} \cap \overline{\mathcal{R}} = (R^B \cap \overline{R}^B, R^C \cap \overline{R}^C)$ . Division by a McCormick object is defined whenever its interval part does not contain zero. But, for any  $\mathcal{S} \in \mathbb{MS}^*$ ,  $\{m_{ii}\}(\mathcal{S}, \mathcal{R}, \mathcal{L})$  has interval part  $[m_{ii}](S, R, [\mathbf{0}, \mathbf{1}]) \subset [m_{ii}](S^*, R^*, [\mathbf{0}, \mathbf{1}])$ , which is guaranteed not to contain zero. Thus,  $\{\psi_\cap\}$  is defined for all  $\mathcal{S} \in \mathbb{MS}^*$ , and the same is easily seen to be true of  $\{\psi\}$ .

Now, a sequence of progressively refined relaxation functions  $\mathcal{H}_k : \mathbb{MS}^* \rightarrow \mathbb{MR}^n$  can be defined via

$$\mathcal{H}_{k+1}(\mathcal{S}) = \{\psi_\cap\}(\mathcal{S}, \mathcal{H}_k(\mathcal{S}), \mathcal{L}), \quad \mathcal{H}_0(\mathcal{S}) = (H(S), H(S)), \quad (4.24)$$

for all  $\mathcal{S} = (S, [\mathbf{s}^{cv}, \mathbf{s}^{cc}]) \in \mathbb{M}S^*$ . Convex and concave relaxations of  $\mathbf{h}$  on any  $S \subset S^*$  are evaluated computationally at a particular point  $\mathbf{s} \in S$  by simply setting  $\mathcal{S} = (S, [\mathbf{s}, \mathbf{s}])$  and executing the iteration (4.24). For any  $k \geq 0$ , the result is a McCormick object  $\mathcal{H}_k(\mathcal{S}) = (H_k(S), [\mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{h}_k^{cc}(\mathbf{s})])$  with the obvious interpretation.

To see that this works in more detail, denote  $\mathcal{H}_k(\mathcal{S}) = (H_k(S), [\mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{h}_k^{cc}(\mathbf{s})])$  and assume that  $\mathbf{h}_k^{cv}$  and  $\mathbf{h}_k^{cc}$  are convex and concave relaxations of  $\mathbf{h}$  on  $S$ , which is trivially true for  $k = 0$ . Now consider a single iteration of (4.24). Using the definition of a relaxation function in terms of composite relaxations (see Sect. 3.2), the computation of  $\mathbf{h}_{k+1}^{cv}$  and  $\mathbf{h}_{k+1}^{cc}$  in (4.24) can be expressed as

$$\mathbf{h}_{k+1}^{cv}(\mathbf{s}) = \max(\mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{u}_\psi(\mathbf{s}, \mathbf{s}, \mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{h}_k^{cc}(\mathbf{s}))), \quad (4.25)$$

$$\mathbf{h}_{k+1}^{cc}(\mathbf{s}) = \min(\mathbf{h}_k^{cc}(\mathbf{s}), \mathbf{o}_\psi(\mathbf{s}, \mathbf{s}, \mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{h}_k^{cc}(\mathbf{s}))), \quad (4.26)$$

where  $\mathbf{u}_\psi$  and  $\mathbf{o}_\psi$  are convex and concave composite relaxations of  $\psi$  on  $S \times H_k(S)$ , respectively. Since  $\mathbf{h}_k^{cv}$  and  $\mathbf{h}_k^{cc}$  are convex and concave relaxations of  $\mathbf{h}$  on  $S$  by hypothesis, the properties of composite relaxations ensure that the functions  $\mathbf{s} \mapsto \mathbf{u}_\psi(\mathbf{s}, \mathbf{s}, \mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{h}_k^{cc}(\mathbf{s}))$  and  $\mathbf{s} \mapsto \mathbf{o}_\psi(\mathbf{s}, \mathbf{s}, \mathbf{h}_k^{cv}(\mathbf{s}), \mathbf{h}_k^{cc}(\mathbf{s}))$  are, respectively, convex and concave relaxations of  $\mathbf{s} \mapsto \psi(\mathbf{s}, \mathbf{h}(\mathbf{s}))$  (and hence of  $\mathbf{h}$ ) on  $S$ . The same is true of  $\mathbf{h}_{k+1}^{cv}$  and  $\mathbf{h}_{k+1}^{cc}$  because the maximum (resp. minimum) of two convex (resp. concave) functions is convex (resp. concave).

## 4.1 Example

Section 8 presents a parameter estimation problem in semi-explicit DAEs with the following algebraic equation:

$$0 = y_1^3 + 2m_3y_1^2 + K_by_1 - (K_b + y_1^2)x_2, \quad (4.27)$$

where  $m_3$  and  $K_b$  are known constants. As discussed in the following section, the methods of this section are applied to the algebraic equations in semi-explicit DAE systems by interpreting  $\mathbf{r} = \mathbf{y}$  as the dependent variables and  $\mathbf{s} = (t, \mathbf{p}, \mathbf{x})$  as the independent variables. Thus, we may write (4.27) in the notation of this section as

$$0 = r^3 + 2m_3r^2 + K_br - (K_b + r^2)s. \quad (4.28)$$

The derivative  $\frac{\partial \ell}{\partial r}$  is given by

$$\frac{\partial \ell}{\partial r}(s, r) = r(3r + 2(2m_3 - s)) + K_b. \quad (4.29)$$

The equations above are written to represent the specific factorable representations used in the computation of bounds and relaxations of the implicit function  $h(s)$  below. They are arranged so as to reduce the conservatism of the bounds and relaxations of these expressions when they are evaluated with interval- or McCormick-valued arguments. In general, this is done by minimizing the number of appearances of interval- or McCormick-valued variables. As a simple example,  $a(b + c)$  is preferred to  $ab + ac$  because  $a$  appears twice in the latter expression. Since these appearances of  $a$  will be considered independent when evaluated, e.g., in interval arithmetic, the latter expression gives a weaker enclosure. With  $A = [2, 3]$ ,  $B = [-2, -1]$ , and  $C = [1, 2]$ , the first expression gives  $[2, 3]([-2, -1] + [1, 2]) = [2, 3][-1, 1] = [-3, 3]$ , whereas the second gives  $[2, 3][-2, -1] + [2, 3][1, 2] = [-6, -2] + [2, 6] = [-4, 4]$ . In arranging (4.28), we have used the observation that the evaluations of this function needed for computing  $[\psi_r]$  and  $\{\psi_\cap\}$  always use a real-valued reference value for  $r$ . Thus, multiple appearances of  $r$  are not problematic. On the other hand, (4.28) is evaluated with interval- and McCormick-valued arguments for  $s$ . Thus, (4.28) is arranged so that  $s$  appears only once, which would not be the case if, for example, the  $r^2$  terms were collected. In contrast to the situation with (4.28), computing  $[\psi_r]$  and  $\{\psi_\cap\}$  does involve evaluating (4.29) with interval- and McCormick-valued arguments for  $r$ . Moreover, (4.29) cannot be arranged so that  $r$  appears only once. The preferred arrangement in this case is the so-called Horner's form [56].

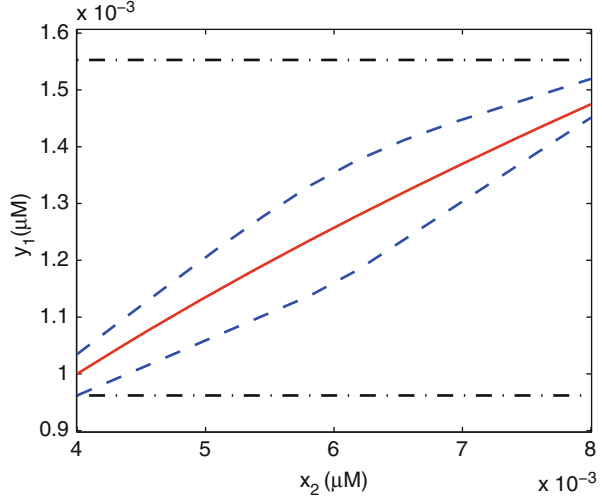
Using (4.28) and (4.29), Eqs. (4.15) and (4.23) become, respectively,

$$R' = \tilde{r} + \left[ (R - \tilde{r}) \cap \left( \frac{-\tilde{r}^3 - 2m_3\tilde{r}^2 - K_b\tilde{r} + (K_b + \tilde{r}^2)S}{R(3R + 2(2m_3 - S)) + K_b} \right) \right], \quad \text{and} \quad (4.30)$$

$$\mathcal{R}' = \tilde{r} + \left[ (\mathcal{R} - \tilde{r}) \cap \left( \frac{-\tilde{r}^3 - 2m_3\tilde{r}^2 - K_b\tilde{r} + (K_b + \tilde{r}^2)\mathcal{S}}{\mathcal{R}(3\mathcal{R} + 2(2m_3 - \mathcal{S})) + K_b} \right) \right], \quad (4.31)$$

provided that no division by an interval or McCormick object containing zero occurs. Choosing the values  $m_3 = 100$ ,  $K_b = \frac{36}{540}$ , and  $S = [4, 8] \times 10^{-3}$ , we find that  $\emptyset \neq R' \subset \text{int}(R)$  with  $R = [0.96164, 1.5531] \times 10^{-3}$ . This  $R$  interval was found using a nonsmooth Newton iteration as described in [79]. It follows from Theorem 4.1 that there exists a unique implicit function  $r = h(s)$  satisfying (4.28) on  $S$  and bounded by  $R$ . This bound changes by less than  $10^{-6}$  after ten iterations of (4.30). Relaxations of  $h$  can now be computed by iteration on (4.31) with  $\mathcal{R}$  initially set to  $(R, R)$  and  $\mathcal{S} = (S, [s, s])$  for values of  $s$  in  $S$ . The resulting relaxations after five iterations are given in Fig. 1.

**Fig. 1** Interval bounds (dot-dashed) and convex and concave relaxations (dashed) for the implicit solution  $y_1 = h(x_2)$  of (4.27) (solid) computed via (4.30) and (4.31)



## 5 State Bounds for Semi-explicit Index-One DAEs

In this section, we summarize the main results in [78, 79] for the computation of interval bounds on the reachable set of (2.1), which we restate here for convenience:

$$\left. \begin{aligned} \dot{\mathbf{x}}(t, \mathbf{p}) &= \mathbf{f}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \\ \mathbf{0} &= \mathbf{g}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \end{aligned} \right\}, \quad (5.1a)$$

$$\mathbf{x}(t_0, \mathbf{p}) = \mathbf{x}_0(\mathbf{p}). \quad (5.1b)$$

In what follows, we consider a regular solution  $(\mathbf{x}, \mathbf{y})$  of (5.1) on  $I \times P$  and compute functions  $\mathbf{x}^L, \mathbf{x}^U : I \rightarrow \mathbb{R}^{n_x}$  and  $\mathbf{y}^L, \mathbf{y}^U : I \rightarrow \mathbb{R}^{n_y}$  such that

$$\mathbf{x}^L(t) \leq \mathbf{x}(t, \mathbf{p}) \leq \mathbf{x}^U(t) \quad \text{and} \quad \mathbf{y}^L(t) \leq \mathbf{y}(t, \mathbf{p}) \leq \mathbf{y}^U(t), \quad \forall (t, \mathbf{p}) \in I \times P.$$

These functions are referred to as *state bounds* for the solution  $(\mathbf{x}, \mathbf{y})$ . As mentioned in Sect. 2.3, the existence and uniqueness of  $(\mathbf{x}, \mathbf{y})$  on  $I \times P$  is verified by the method and need not be assumed. The behavior of the method for DAEs with multiple solutions is discussed below, after some key results are in place.

Exactly as in Sect. 4, the difficulty in computing state bounds for  $(\mathbf{x}, \mathbf{y})$  is that these functions have no known factorable representation, so that the methods of Sect. 3 cannot be applied directly. This complication was overcome in Sect. 4 by instead exploiting the factorable representation of the algebraic system there (specifically,  $\ell$  and  $\frac{\partial \ell}{\partial \mathbf{r}}$ ) to obtain global information about the function of interest (the implicit function  $\mathbf{h}$ ). Our approach in this section and the next is exactly analogous, although the functions of interest here are defined as the solutions of DAEs, so that some additional insights are required. In brief, the factorable

representation of the governing equations in (5.1) will be used to derive an auxiliary system of DAEs describing bounding trajectories  $\mathbf{x}^L$ ,  $\mathbf{x}^U$ ,  $\mathbf{y}^L$ , and  $\mathbf{y}^U$  as its solutions. Accordingly, we make the following assumption on (5.1) throughout.

**Assumption 5.1** *The functions  $\mathbf{f}$ ,  $\mathbf{x}_0$ ,  $\mathbf{g}$  and  $\frac{\partial \mathbf{g}}{\partial \mathbf{y}}$  are factorable with natural interval and McCormick extensions  $[\mathbf{x}_0] : \mathbb{ID}_p \rightarrow \mathbb{IR}^{n_x}$ ,  $\{\mathbf{x}_0\} : \mathbb{MD}_p \rightarrow \mathbb{MR}^{n_x}$ ,  $[\mathbf{f}] : \mathbb{ID}_t \times \mathbb{ID}_p \times \mathbb{ID}_x \times \mathbb{ID}_y \rightarrow \mathbb{IR}^{n_x}$ ,  $\{\mathbf{f}\} : \mathbb{MD}_t \times \mathbb{MD}_p \times \mathbb{MD}_x \times \mathbb{MD}_y \rightarrow \mathbb{MR}^{n_x}$ ,  $[\mathbf{g}] : \mathbb{ID}_t \times \mathbb{ID}_p \times \mathbb{ID}_x \times \mathbb{ID}_y \rightarrow \mathbb{IR}^{n_y}$ ,  $\{\mathbf{g}\} : \mathbb{MD}_t \times \mathbb{MD}_p \times \mathbb{MD}_x \times \mathbb{MD}_y \rightarrow \mathbb{MR}^{n_y}$ ,  $[\frac{\partial \mathbf{g}}{\partial \mathbf{y}}] : \mathbb{ID}_t \times \mathbb{ID}_p \times \mathbb{ID}_x \times \mathbb{ID}_y \rightarrow \mathbb{IR}^{n_y \times n_y}$ , and  $\{\frac{\partial \mathbf{g}}{\partial \mathbf{y}}\} : \mathbb{MD}_t \times \mathbb{MD}_p \times \mathbb{MD}_x \times \mathbb{MD}_y \rightarrow \mathbb{MR}^{n_y \times n_y}$ .*

## 5.1 Theoretical Considerations

The treatment of the algebraic equations in (5.1) follows exactly the developments in Sect. 4. The first step is to apply the main result Theorem 4.1 to the algebraic equations  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$ . In the semi-explicit form (5.1), the role of the algebraic equations  $\mathbf{g}$  is essentially to specify the algebraic variables  $\mathbf{z}_y$  given fixed values of  $t$ ,  $\mathbf{p}$ , and the differential variables  $\mathbf{z}_x$ . Thus, in applying Theorem 4.1 to these equations, we identify the dependent variable  $\mathbf{r}$  with  $\mathbf{z}_y$  and the independent variable  $\mathbf{s}$  with  $(t, \mathbf{p}, \mathbf{z}_x)$ . With these substitutions, the functions  $\mathbf{M}$ ,  $\psi$ ,  $[\psi]$ ,  $[\psi]_\cap$ , and  $[\psi]_\Gamma$  can be defined exactly as in Sect. 4. For example, in this case (4.4) becomes

$$\mathbf{M}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y, \boldsymbol{\lambda}') \equiv \begin{bmatrix} \frac{\partial g_1}{\partial \mathbf{y}}(t, \mathbf{p}, \mathbf{z}_x, \tilde{\mathbf{z}}_y + \lambda'_1(\mathbf{z}_y - \tilde{\mathbf{z}}_y)) \\ \vdots \\ \frac{\partial g_{n_y}}{\partial \mathbf{y}}(t, \mathbf{p}, \mathbf{z}_x, \tilde{\mathbf{z}}_y + \lambda'_{n_y}(\mathbf{z}_y - \tilde{\mathbf{z}}_y)) \end{bmatrix}, \quad (5.2)$$

for all  $(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y, \boldsymbol{\lambda}') \in \mathbb{R}^{1+n_p+n_x+n_y+n_y}$ , where  $\tilde{\mathbf{z}}_y \in \mathbb{R}^{n_y}$  is a fixed reference point. Theorem 4.1 yields the following Corollary:

**Corollary 5.1** *Let  $(J, P, Z_x, Z_y) \in \mathbb{ID}_t \times \mathbb{ID}_p \times \mathbb{ID}_x \times \mathbb{ID}_y$ ,  $\tilde{\mathbf{z}}_y \in Z_y$ , and define*

$$Z'_y \equiv [\psi]_\Gamma(J, P, Z_x, Z_y, [\mathbf{0}, \mathbf{1}]). \quad (5.3)$$

*The following conclusions hold:*

1. *If  $(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) \in J \times P \times Z_x \times Z_y$  satisfies  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$ , then  $\mathbf{z}_y \in Z'_y$ .*
2. *If  $Z'_y = \emptyset$ , then  $\exists \mathbf{h}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) \in J \times P \times Z_x \times Z_y$  such that  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$ .*
3. *If  $\tilde{\mathbf{z}}_y \in \text{int}(Z_y)$  and  $\emptyset \neq Z'_y \subset \text{int}(Z_y)$ , then  $\exists \mathbf{h} \in C^1(J \times P \times Z_x, Z'_y)$  such that, for every  $(t, \mathbf{p}, \mathbf{z}_x) \in J \times P \times Z_x$ ,  $\mathbf{z}_y = \mathbf{h}(t, \mathbf{p}, \mathbf{z}_x)$  is the unique element of  $Z_y$  satisfying  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$ . Moreover,  $[\mathbf{M}](J, P, Z_x, Z_y, [\mathbf{0}, \mathbf{1}])$  does not contain a singular matrix and does not contain zero in any of its diagonal elements.*

*Remark 5.1* As in Remark 4.2, note that  $\tilde{\mathbf{z}}_y \in Z_y$  implies  $\tilde{\mathbf{z}}_y + [\mathbf{0}, \mathbf{1}](Z_y - \tilde{\mathbf{z}}_y) = Z_y$ , and hence

$$[\mathbf{M}](J, P, Z_x, Z_y, [\mathbf{0}, \mathbf{1}]) = \left[ \frac{\partial \mathbf{g}}{\partial \mathbf{y}} \right] (J, P, Z_x, Z_y). \quad (5.4)$$

The statement of Corollary 5.1 in [78] uses this substitution. As mentioned in Sect. 4, it is almost always necessary to precondition the system  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$  in order to pass the inclusion test of Conclusion 3, but we omit the preconditioner for brevity.

Conclusion 3 of Corollary 5.1 is crucial to the state bounding method presented below. First, it provides a computational test for the existence and uniqueness of an implicit function  $\mathbf{h} : J \times P \times Z_x \rightarrow \mathbb{R}^{n_y}$  satisfying  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{h}(t, \mathbf{p}, \mathbf{z}_x)) = \mathbf{0}$ ,  $\forall (t, \mathbf{p}, \mathbf{z}_x) \in J \times P \times Z_x$ . The existence of this implicit function makes (5.1) equivalent to an explicit system of ODEs on  $I \times P$ , which provides an inroad for the application of state bounding methods for ODEs. However, the complication remains that  $\mathbf{h}$  is not factorable. Thus, the second essential piece of information gained from Conclusion 3 is the a priori bound  $\mathbf{h}(J, P, Z_x) \subset Z'_y \subset \text{int}(Z_y)$ .

As discussed in Sect. 4, the inclusion test in Conclusion 3 of Corollary 5.1 has some further useful consequences. Specifically, when it holds for some  $(J, P, Z_x, Z_y)$ , it implies that  $\psi(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y, \lambda')$  is defined for all  $(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y, \lambda') \in J \times P \times Z_x \times Z_y \times [\mathbf{0}, \mathbf{1}]$ , and satisfies

$$\mathbf{h}(t, \mathbf{p}, \mathbf{z}_x) = \psi(t, \mathbf{p}, \mathbf{z}_x, \mathbf{h}(t, \mathbf{p}, \mathbf{z}_x), \lambda(t, \mathbf{p}, \mathbf{z}_x)), \quad \forall (t, \mathbf{p}, \mathbf{z}_x) \in J \times P \times Z_x. \quad (5.5)$$

Moreover, we have  $[\psi_\Gamma](J, P, Z_x, Z_y) = [\psi_\cap](J, P, Z_x, Z_y) = [\psi](J, P, Z_x, Z_y)$ . We will use both of these facts below.

The following theorem is the main result from [78] that enables the computation of state bounds. One of the main hypotheses is that the inclusion test of Conclusion 3 in Corollary 5.1 is satisfied pointwise in time; i.e., with  $J = [t, t]$  and time-varying bounds  $X(t)$  and  $Y(t)$  in place of  $Z_x$  and  $Z_y$ .

**Theorem 5.2** *Let  $(I, P) \in \mathbb{I}D_t \times \mathbb{I}D_p$ . Suppose that  $\mathbf{y}^L, \mathbf{y}^U : I \rightarrow \mathbb{R}^{n_y}$  are continuous,  $\mathbf{x}^L, \mathbf{x}^U : I \rightarrow \mathbb{R}^{n_x}$  are absolutely continuous, and for all  $t \in I$ ,  $\mathbf{x}^L(t) \leq \mathbf{x}^U(t)$ ,  $\mathbf{y}^L(t) \leq \mathbf{y}^U(t)$ ,  $X(t) \equiv [\mathbf{x}^L(t), \mathbf{x}^U(t)] \in \mathbb{I}D_x$ , and  $Y(t) \equiv [\mathbf{y}^L(t), \mathbf{y}^U(t)] \in \mathbb{I}D_y$ . Let the following hypotheses hold:*

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

(ALG): For every  $t \in I$ ,  $\emptyset \neq [\psi_\Gamma]([t, t], P, X(t), Y(t), [\mathbf{0}, \mathbf{1}]) \subset \text{int}(Y(t))$ .

(RHS): For a.e.  $t \in I$  and each index  $i$ ,

1.  $\dot{x}_i^L(t) \leq f_i(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y)$  for all  $(\mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) \in P \times X(t) \times Y(t)$  such that  $z_{x,i} = x_i^L(t)$  and  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$ ,
2.  $\dot{x}_i^U(t) \geq f_i(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y)$  for all  $(\mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) \in P \times X(t) \times Y(t)$  such that  $z_{x,i} = x_i^U(t)$  and  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) = \mathbf{0}$ .

Then every regular solution of (5.1) on  $I \times P$  with  $\mathbf{y}(t_0, \hat{\mathbf{p}}) \in Y(t_0)$  for at least one  $\hat{\mathbf{p}} \in P$  must satisfy  $(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \in X(t) \times Y(t)$  for all  $(t, \mathbf{p}) \in I \times P$ .

The proof of Theorem 5.2 is quite technical and can be found in Theorem 5.5.6 of [78]. However, the main ideas are not difficult to appreciate. Consider a solution  $(\mathbf{x}, \mathbf{y})$  as in the statement of the theorem. First, note that Hypothesis (ALG) and Conclusion 3 of Corollary 5.1 imply that, for every  $t \in I$ , there exists  $\mathbf{h}(t, \cdot, \cdot) : P \times X(t) \rightarrow \text{int}(Y(t))$  such that  $\mathbf{h}(t, \mathbf{p}, \mathbf{z}_x)$  solves  $\mathbf{g}(t, \mathbf{p}, \mathbf{z}_x, \cdot) = \mathbf{0}$  uniquely among elements of  $Y(t)$ , provided that  $(\mathbf{p}, \mathbf{z}_x) \in P \times X(t)$ . With  $\mathbf{z}_x = \mathbf{x}(t, \mathbf{p})$ ,  $\mathbf{y}(t, \mathbf{p})$  satisfies this equation by definition. Thus, we have the following implication for all  $(t, \mathbf{p}) \in I \times P$ :

$$\mathbf{x}(t, \mathbf{p}) \in X(t), \quad \mathbf{y}(t, \mathbf{p}) \in Y(t) \quad \implies \quad \mathbf{y}(t, \mathbf{p}) = \mathbf{h}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p})) \in \text{int}(Y(t)). \quad (5.6)$$

Then, from Hypotheses (IC) and the assumption that  $\mathbf{y}(t_0, \hat{\mathbf{p}}) \in Y(t_0)$ , continuity of  $\mathbf{y}(t_0, \cdot)$  implies that  $\mathbf{y}(t_0, \cdot)$  is confined to the interior of  $Y(t)$  on all of  $P$ . Thus,  $(\mathbf{x}(t_0, \mathbf{p}), \mathbf{y}(t_0, \mathbf{p})) \in X(t_0) \times Y(t_0)$  for all  $\mathbf{p} \in P$ .

Now, in order for  $(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}))$  to leave the bounds  $X(t) \times Y(t)$  at some  $t > t_0$  and for some  $\mathbf{p} \in P$ , it must happen that  $(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}))$  first intersects the boundary of  $X(t) \times Y(t)$ . Here again, (5.6) implies that  $\mathbf{y}(t, \mathbf{p})$  cannot reach the boundary of  $Y(t)$  before  $\mathbf{x}(t, \mathbf{p})$  leaves  $X(t)$ . Thus, if a violation is to occur, we must come to a point such that  $(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \in X(t) \times Y(t)$  and either  $x_i(t, \mathbf{p}) = x_i^L(t)$  or  $x_i(t, \mathbf{p}) = x_i^U(t)$  for at least one  $i$ . Supposing that the first case occurs, we now note that  $(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}))$  satisfies all of the conditions imposed on  $(\mathbf{z}_x, \mathbf{z}_y)$  in Hypothesis (RHS).1 of the theorem, and it follows that

$$\dot{x}_i^L(t) \leq f_i(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) = \dot{x}_i(t, \mathbf{p}). \quad (5.7)$$

From the inequality  $\dot{x}_i^L(t) \leq \dot{x}_i(t, \mathbf{p})$  and some further technical conditions it can then be shown that it is impossible for  $\dot{x}_i(\cdot, \mathbf{p})$  to cross  $\dot{x}_i^L$  at  $t$ , completing the argument.

Note that Hypotheses (IC) and (RHS) in Theorem 5.2 involve global information about the functions  $\mathbf{f}$  and  $\mathbf{x}_0$  in the form of bounds on their images. The content of Theorem 5.2 is then to deduce global information about the solution  $(\mathbf{x}, \mathbf{y})$  from global information about the system equations. In this sense, Theorem 5.2 is well aligned with our strategy so far. Not surprisingly, the implementation of these ideas in the next section uses the techniques of Sect. 3 to obtain the required bounds on the system equations, which have known factorable representations.

## 5.2 Implementation

Using the theoretical developments of the previous two sections, we now derive an auxiliary system of semi-explicit DAEs that describes valid state bounds as its

solutions. Let  $\mathcal{B}_i^L, \mathcal{B}_i^U : \mathbb{R}^{n_x} \rightarrow \mathbb{R}^{n_x}$  be defined by  $\mathcal{B}_i^L([\mathbf{v}, \mathbf{w}]) = \{\mathbf{z} \in [\mathbf{v}, \mathbf{w}] : z_i = v_i\}$  and  $\mathcal{B}_i^U([\mathbf{v}, \mathbf{w}]) = \{\mathbf{z} \in [\mathbf{v}, \mathbf{w}] : z_i = w_i\}$ , for every  $i = 1, \dots, n_x$ . Moreover, let  $[\mathbf{f}]^L$  and  $[\mathbf{f}]^U$  denote the upper and lower bounds of  $[\mathbf{f}]$ , and defined  $[\boldsymbol{\psi}]^L$  and  $[\boldsymbol{\psi}]^U$  analogously. For any continuous and pointwise positive  $\gamma : I \rightarrow \mathbb{R}$ , consider the initial value problem in DAEs

$$\dot{x}_i^L(t) = [f_i]^L([t, t], P, \mathcal{B}_i^L(X(t)), Y(t)), \quad (5.8)$$

$$\dot{x}_i^U(t) = [f_i]^U([t, t], P, \mathcal{B}_i^U(X(t)), Y(t)), \quad (5.9)$$

$$\mathbf{0} = \mathbf{y}^L(t) - [\boldsymbol{\psi}]^L([t, t], P, X(t), Y(t), [\mathbf{0}, \mathbf{1}]) + \mathbf{1}\gamma(t), \quad (5.10)$$

$$\mathbf{0} = -\mathbf{y}^U(t) + [\boldsymbol{\psi}]^U([t, t], P, X(t), Y(t), [\mathbf{0}, \mathbf{1}]) + \mathbf{1}\gamma(t), \quad (5.11)$$

for all  $i = 1, \dots, n_x$ , with initial conditions

$$[\mathbf{x}^L(t_0), \mathbf{x}^U(t_0)] = [\mathbf{x}_0](P). \quad (5.12)$$

A solution of (5.8)–(5.12) is any  $\mathbf{x}^L, \mathbf{x}^U : I \rightarrow \mathbb{R}^{n_x}$  and  $\mathbf{y}^L, \mathbf{y}^U : I \rightarrow \mathbb{R}^{n_y}$  satisfying (5.8)–(5.12) for all  $t \in I$ , with  $\mathbf{x}^L$  and  $\mathbf{x}^U$  continuously differentiable and  $\mathbf{y}^L$  and  $\mathbf{y}^U$  piecewise  $C^1$  (see [22] for a definition of this class of functions).

The main result of [79] is that any solution of (5.8)–(5.12) provides state bounds for a unique regular solution of (2.1) on  $I \times P$ . Of course, this is a direct application of Theorem 5.2. Clearly, (5.12) verifies Hypothesis (IC) since natural interval extensions are inclusion functions (see Sect. 3). Furthermore, (5.10) and (5.11), along with positivity of  $\gamma$ , imply that  $[\boldsymbol{\psi}]([t, t], P, X(t), Y(t), [\mathbf{0}, \mathbf{1}])$  is strictly contained in  $Y(t)$ . Typically,  $\gamma = 10^{-6}$  works well in practice. Observing that  $[\boldsymbol{\psi}] = [\boldsymbol{\psi}_r]$  for all arguments such that (5.10) and (5.11) hold (see the discussion following Corollary 5.1), Hypothesis (ALG) is verified. The function  $[\boldsymbol{\psi}]$  is used in place of  $[\boldsymbol{\psi}_r]$  here for numerical reasons that are beyond the scope of this review [79]. Finally, (5.8) and (5.9) establish Hypothesis (RHS), again through the inclusion properties of the natural interval extension (note that the operators  $\mathcal{B}_i^L$  and  $\mathcal{B}_i^U$  are used to reflect the conditions  $z_{x,i} = x_i^L(t)$  and  $z_{x,i} = x_i^U(t)$  in Hypothesis (RHS), respectively).

With some technical modifications to Eqs. (5.8)–(5.12) that are beyond the scope of this article, it is possible to assert the existence of a regular solution within the computed bounds [79, Theorem 6.6.4]. Note that Theorem 5.2 does not assert existence.

**Theorem 5.3** *Let  $(\mathbf{x}^L, \mathbf{x}^U, \mathbf{y}^L, \mathbf{y}^U)$  be a solution of (5.8)–(5.12) on  $I$ . Then there exists a unique regular solution  $(\mathbf{x}, \mathbf{y})$  of (2.1) on  $I \times P$  satisfying  $\mathbf{x}(t, \mathbf{p}) \in X(t)$  and  $\mathbf{y}(t, \mathbf{p}) \in Y(t)$ ,  $\forall (t, \mathbf{p}) \in I \times P$ .*

In light of the discussion above, state bounds are given by solving the DAEs (5.8)–(5.12) for  $\mathbf{x}^L, \mathbf{x}^U, \mathbf{y}^L$ , and  $\mathbf{y}^U$ . This can be done using any state-of-the-art DAE solver, for example, IDA [27]. Moreover, the required natural interval extensions can be computed automatically using operator overloading as described in Sect. 3.

The only caveat is that the governing equations in (5.8)–(5.12) are nonsmooth and are undefined for certain arguments, e.g.,  $x_i^L(t) > x_i^U(t)$ . Arguments of this type do not occur along solution trajectories, but may occur during numerical integration. For these reasons, some specialized methods are required, as discussed in [79].

Given (5.12), consistent initial conditions for  $\mathbf{y}^L$  and  $\mathbf{y}^U$  can be computed by solving (5.10) and (5.11) at  $t_0$ . If (5.1) has multiple regular solutions, then there may be multiple consistent initial conditions, and which one is chosen determines which solution branch will be bounded by the solution of (5.8)–(5.12). In particular, at  $t_0$ , (5.10) and (5.11) verifies the existence of an implicit function  $\mathbf{h}(t_0, \cdot, \cdot) : P \times X(t_0) \rightarrow Y(t_0)$  providing the unique solution of  $\mathbf{g}(t_0, \mathbf{p}, \mathbf{x}(t_0, \mathbf{p}), \cdot) = \mathbf{0}$  within  $Y(t_0)$ , but says nothing about potential solutions outside of  $Y(t_0)$ . Thus, there may exist other solutions that lie outside  $Y(t_0)$  for all  $\mathbf{p} \in P$ , and these solutions will not be bounded. Example 2.1 provides a simple DAE of this type.

The bounding equations (5.8)–(5.12) only apply to regular (i.e., index-one) solutions. In particular, the algebraic equations (5.10) and (5.11) cannot be satisfied by any  $X(t)$  and  $Y(t)$  containing a solution that is not regular at  $t$ . This is because satisfaction of (5.10) and (5.11) implies by Conclusion 3 of Corollary 5.1 that  $\frac{\partial \mathbf{g}}{\partial \mathbf{y}}(t, \mathbf{p}, \mathbf{z}_x, \mathbf{z}_y)$  is non-singular for every  $(\mathbf{p}, \mathbf{z}_x, \mathbf{z}_y) \in P \times X(t) \times Y(t)$ . Thus, the solution of (5.8)–(5.12) will cease to exist if the bounded solution approaches a bifurcation point. In general, even in the case of a unique regular solution, there is no guarantee that (5.8)–(5.12) has a solution for a given  $I$  and  $P$ . However, (5.8)–(5.12) has been shown to have solutions providing tight state bounds for several examples of practical interest [79]. Moreover, whenever a solution of (5.8)–(5.12) does exist, it is guaranteed that a unique regular solution of (5.1) exists within the computed bounds.

It is possible to tighten the bounds given by the solution of (5.8)–(5.12) quite considerably by refining the argument  $Y(t)$  in (5.8) and (5.9). In fact, this is the standard procedure as presented in [79] and was initially omitted here only for simplicity of exposition. To develop this refinement procedure, we note that from (5.6) we have

$$\mathbf{y}(t, \mathbf{p}) = \mathbf{h}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p})), \quad \forall (t, \mathbf{p}) \in I \times P. \quad (5.13)$$

Then, using (5.5) with  $J = [t, t]$ ,  $Z_x = X(t)$ , and  $Z_y = Y(t)$ , it follows that

$$\mathbf{y}(t, \mathbf{p}) = \boldsymbol{\psi}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}), \boldsymbol{\lambda}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}))), \quad \forall (t, \mathbf{p}) \in I \times P. \quad (5.14)$$

Thus, we may refine  $Y(t)$  at each  $t$  through the iteration

$$Y_{k+1}(t) = [\boldsymbol{\psi}_r]([t, t], P, X(t), Y_k(t), [\mathbf{0}, \mathbf{1}]), \quad Y_0(t) = Y(t). \quad (5.15)$$

Further refinement is possible if we consider refining only the particular interval  $Y(t)$  that will be used to evaluate  $[f_i]^L$  in (5.8). Of course, analogous procedures also apply to the argument of  $[f_i]^U$ , and for  $j \neq i$ . Considering the condition that  $[f_i]^L$  must satisfy according to Hypothesis (RHS) in Theorem 5.2, the argument  $Y(t)$

in (5.8) can be refined by the iteration

$$Y_{k+1}(t) = [\psi_r]([t, t], P, \mathcal{B}_i^L(X(t)), Y_k(t), [\mathbf{0}, \mathbf{1}]), \quad Y_0(t) = Y(t). \quad (5.16)$$

In practice, these refinements have been found to lead to much sharper results than applying (5.8)–(5.12) directly.

### 5.3 Alternative Approaches

As mentioned in the Sect. 1, an alternative state bounding approach for semi-explicit DAEs has been proposed in [65]. The treatment of the algebraic equations in this method is very similar to that presented here, using an interval Newton method as described in Sect. 4. However, in [65] the *interval Krawczyk method* is used instead of the interval Hansen-Sengupta method. The interval Krawczyk method is based on a different rearrangement of (4.5) that leads to a weaker enclosure but avoids the issue of division by an interval containing zero. The reader is referred to [59] for a comprehensive description and comparison of the two methods.

In its treatment of the differential equations, the method proposed in [65] differs significantly from the developments here. Rather than deriving a *bounding system* which can be solved numerically to obtain state bounds [e.g., (5.8)–(5.12)], the method applies the interval Krawczyk method to the differential equations as well. In essence, the authors propose a time-stepping scheme in which, in each interval  $[t_j, t_{j+1}]$ , the interval Krawczyk method is applied to the nonlinear system of equations

$$\mathbf{0} = -\sigma_x + \mathbf{f}(t, \mathbf{p}, \mathbf{x}_j + (t - t_j)\sigma_x, \mathbf{z}_y), \quad (5.17)$$

$$\mathbf{0} = \mathbf{g}(t, \mathbf{p}, \mathbf{x}_j + (t - t_j)\sigma_x, \mathbf{z}_y). \quad (5.18)$$

Above,  $\sigma_x$  and  $\mathbf{z}_y$  are dummy variables for  $\dot{\mathbf{x}}$  and  $\mathbf{y}$ , respectively, and the remaining variables are known to lie within interval bounds at the beginning of the time step (e.g.,  $t \in [t_j, t_{j+1}]$ ,  $\mathbf{p} \in P$ ,  $\mathbf{x}_j \in X_j$ ). The authors of [65] claim that, if an inclusion test similar to that in Conclusion 3 of Theorem 4.1 is passed with some intervals  $Z_y$  and  $\Sigma_x$ , then a unique solution of the DAE system is guaranteed to exist on  $[t_j, t_{j+1}] \times P$ , and  $\mathbf{y}(t, \mathbf{p}) \in Z_y$  and  $\dot{\mathbf{x}}(t, \mathbf{p}) \in \Sigma_x$ ,  $\forall (t, \mathbf{p}) \in [t_j, t_{j+1}] \times P$ . These intervals can then be refined by interval iteration on (5.17)–(5.18) as described in Sect. 4. Finally, the differential state is bounded by

$$\mathbf{x}(t, \mathbf{p}) = \mathbf{x}(t_j, \mathbf{p}) + \int_{t_j}^t \dot{\mathbf{x}}(s, \mathbf{p}) ds \in X_j + [0, (t_{j+1} - t_j)]\Sigma_x, \quad (5.19)$$

for all  $(t, \mathbf{p}) \in [t_j, t_{j+1}] \times P$ , and the next time step can be initialized.

The existence and uniqueness claim is not proven in [65], and does not follow from a careful application of Theorem 4.1 to (5.17) and (5.18). Thus some skepticism is warranted. However, it is likely that the method can be shown to be valid using a fixed-point result for ODE solutions, such as the successive substitution method. A similar proof is used to establish an interval existence and uniqueness test for DAE solutions in Theorem 4.2 of [79].

Compared to the method presented in Sects. 5.1 and 5.2, the method of [65] has the apparent disadvantage that it requires an interval inclusion test to pass on Eqs. (5.17) and (5.18), which notably involve equations related to the differential equations in addition to the original algebraic equations. As discussed in [78], and briefly in Sect. 8, passing the interval inclusion test numerically can be quite difficult when  $P$  is large in width. Thus, applying such a test to an inflated system of equations may cause unnecessary failure of the method for some problems. Ultimately, a thorough computational comparison of these two methods will be required to clarify the possible advantages and disadvantages of each approach.

## 6 State Relaxations for Semi-explicit Index-One DAEs

In this section, we take up the computation of state relaxations for (5.1). It is assumed throughout that state bounds have been computed via the procedure of Sect. 5. Thus,  $X : I \rightarrow \mathbb{IR}^{n_x}$  and  $Y : I \rightarrow \mathbb{IR}^{n_y}$  satisfying (5.8)–(5.12) are available, and hence we are ensured that a unique regular solution  $(\mathbf{x}, \mathbf{y})$  of (5.1) exists satisfying

$$(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \in X(t) \times Y(t), \quad \forall (t, \mathbf{p}) \in I \times P. \quad (6.1)$$

Thus, the objective of this section is to compute functions  $\mathbf{x}^{cv}, \mathbf{x}^{cc} : I \times P \rightarrow \mathbb{R}^{n_x}$  and  $\mathbf{y}^{cv}, \mathbf{y}^{cc} : I \times P \rightarrow \mathbb{R}^{n_y}$  such that

$$\mathbf{x}^{cv}(t, \mathbf{p}) \leq \mathbf{x}(t, \mathbf{p}) \leq \mathbf{x}^{cc}(t, \mathbf{p}) \quad \text{and} \quad \mathbf{y}^{cv}(t, \mathbf{p}) \leq \mathbf{y}(t, \mathbf{p}) \leq \mathbf{y}^{cc}(t, \mathbf{p}), \quad (6.2)$$

for all  $(t, \mathbf{p}) \in I \times P$ , and, for each  $t \in I$ ,  $\mathbf{x}^{cv}(t, \cdot)$  and  $\mathbf{y}^{cv}(t, \cdot)$  are convex on  $P$ , and  $\mathbf{x}^{cc}(t, \cdot)$  and  $\mathbf{y}^{cc}(t, \cdot)$  are concave on  $P$ . These functions are called *state relaxations* for  $(\mathbf{x}, \mathbf{y})$  on  $I \times P$ . More generally, functions  $\mathcal{X} : I \times \mathbb{MP} \rightarrow \mathbb{MR}^{n_x}$  and  $\mathcal{Y} : I \times \mathbb{MP} \rightarrow \mathbb{MR}^{n_y}$  will be computed such that, for each fixed  $t \in I$ ,  $\mathcal{X}(t, \cdot)$  and  $\mathcal{Y}(t, \cdot)$  are relaxation functions for  $\mathbf{x}(t, \cdot)$  and  $\mathbf{y}(t, \cdot)$ , respectively.

As in the previous section, we begin by treating the algebraic equations. However, matters here are much simpler since existence and uniqueness of the solution  $(\mathbf{x}, \mathbf{y})$  has already been established by the state bounding procedure. The remaining task is simply to derive a procedure that computes the inclusion function  $\mathcal{Y}$  supposing that  $\mathcal{X}$  is known. This will then be used during the computation of  $\mathcal{X}$  described below to obtain  $\mathcal{X}$  and  $\mathcal{Y}$  simultaneously. The starting point for this derivation is the relation (5.14), established in the previous section. Mirroring the developments of Sect. 4, this relation immediately suggests that a sequence of

progressively refined relaxation functions for  $\mathbf{y}(t, \cdot)$  can be computed through the iteration

$$\begin{aligned}\mathcal{Y}_{k+1}(t, \mathcal{P}) &= \{\boldsymbol{\psi}_{\cap}\}(t, \mathcal{P}, \mathcal{X}(t, \mathcal{P}), \mathcal{Y}_k(t, \mathcal{P}), \mathcal{L}), \\ \mathcal{Y}_0(t, \mathcal{P}) &= (Y(t), Y(t)),\end{aligned}\tag{6.3}$$

for all  $\mathcal{P} \in \mathbb{MP}$ , where  $\mathcal{L} = ([0, 1], [0, 1])$  is a relaxation function for  $\lambda$  in (5.14). Of course,  $\mathcal{Y}_0(t, \cdot)$  is trivially a relaxation function for  $\mathbf{y}(t, \cdot)$  by (6.1). Assuming this is true of  $\mathcal{Y}_k(t, \cdot)$ , and further considering that  $\mathcal{X}(t, \cdot)$  is assumed to be a relaxation function for  $\mathbf{x}(t, \cdot)$ , the properties of  $\{\boldsymbol{\psi}_{\cap}\}$  imply that  $\mathcal{Y}_{k+1}(t, \cdot)$  is also a relaxation function for  $\mathbf{y}(t, \cdot)$ , as discussed in Sect. 4. Finally, note that all divisions by McCormick objects in the evaluation of  $\{\boldsymbol{\psi}_{\cap}\}$  above are guaranteed to be defined for all  $\mathcal{P} \in \mathbb{MP}$  on account of Conclusion 3 of Corollary 5.1 and (5.10) and (5.11), provided that the interval part of  $\mathcal{X}(t)$  is  $X(t)$ . Then, convex and concave relaxations of  $\mathbf{y}(t, \cdot)$  on  $P' \subset P$  are evaluated computationally at a particular point  $\mathbf{p} \in P'$  by simply setting  $\mathcal{P} = (P', [\mathbf{p}, \mathbf{p}])$  and executing the iteration (6.3). For any  $k \geq 0$ , the result is a McCormick object  $\mathcal{Y}_k(\mathcal{P}) = (Y_k(t, P'), [\mathbf{y}_k^{cv}(t, \mathbf{p}), \mathbf{y}_k^{cc}(t, \mathbf{p})])$  with the obvious interpretation. Moving forward, we use the notation

$$\mathcal{Y}(t, \mathcal{P}) = \mathcal{H}_K(t, \mathcal{P}, \mathcal{X}(t, \mathcal{P})), \quad \forall \mathcal{P} \in \mathbb{MP},\tag{6.4}$$

to denote the relaxation function  $\mathcal{Y}(t, \mathcal{P})$  computed from  $\mathcal{X}(t, \mathcal{P})$  through the iteration (6.3) truncated at  $K > 0$ .

We are now prepared to state the main result from [81] used to compute state bounds for  $(\mathbf{x}, \mathbf{y})$ . By Assumption 5.1,  $\mathbf{f}$  is factorable, so we can consider its natural McCormick extension,  $\{\mathbf{f}\}$ . Below, we use the notation  $\{\mathbf{f}\} = ([\mathbf{f}], \{\mathbf{f}\}^{cv}, \{\mathbf{f}\}^{cc})$ .

**Theorem 6.1** *Let  $K > 0$  and, for every  $\mathcal{P} \in \mathbb{MP}$ , and let  $\mathcal{X}^{cv}(\cdot, \mathcal{P})$ ,  $\mathcal{X}^{cc}(\cdot, \mathcal{P}) : I \rightarrow \mathbb{R}^{n_x}$  be solutions of the following initial value problem in ODEs*

$$\dot{\mathcal{X}}^{cv}(t, \mathcal{P}) = \{\mathbf{f}\}^{cv}(t, \mathcal{P}, \mathcal{X}(t, \mathcal{P}), \mathcal{Y}(t, \mathcal{P})),\tag{6.5}$$

$$\dot{\mathcal{X}}^{cc}(t, \mathcal{P}) = \{\mathbf{f}\}^{cc}(t, \mathcal{P}, \mathcal{X}(t, \mathcal{P}), \mathcal{Y}(t, \mathcal{P})),\tag{6.6}$$

$$\mathcal{X}(t, \mathcal{P}) = (X(t), X(t) \cap [\mathcal{X}^{cv}(t, \mathcal{P}), \mathcal{X}^{cc}(t, \mathcal{P})]),\tag{6.7}$$

$$\mathcal{Y}(t, \mathcal{P}) = \mathcal{H}_K(t, \mathcal{P}, \mathcal{X}(t, \mathcal{P})),\tag{6.8}$$

$$\mathcal{X}(t_0, \mathcal{P}) = \{\mathbf{x}_0\}(\mathcal{P}).\tag{6.9}$$

Then, for each  $t \in I$ ,  $\mathcal{X}(t, \cdot)$  and  $\mathcal{Y}(t, \cdot)$  are relaxation functions for  $\mathbf{x}(t, \cdot)$  and  $\mathbf{y}(t, \cdot)$ , respectively. In particular, state relaxations for  $(\mathbf{x}, \mathbf{y})$  on  $P' \subset P$  are given by the definitions,

$$\mathbf{x}^{cv}(t, \mathbf{p}) = \mathcal{X}^{cv}(t, (P', [\mathbf{p}, \mathbf{p}])), \quad \mathbf{y}^{cv}(t, \mathbf{p}) = \mathcal{Y}^{cv}(t, (P', [\mathbf{p}, \mathbf{p}])),\tag{6.10}$$

$$\mathbf{x}^{cc}(t, \mathbf{p}) = \mathcal{X}^{cc}(t, (P', [\mathbf{p}, \mathbf{p}])), \quad \mathbf{y}^{cc}(t, \mathbf{p}) = \mathcal{Y}^{cc}(t, (P', [\mathbf{p}, \mathbf{p}])),\tag{6.11}$$

for all  $(t, \mathbf{p}) \in I \times P$ .

By Theorem 6.1, convex and concave relaxations of  $\mathbf{x}(t, \cdot)$  and  $\mathbf{y}(t, \cdot)$  on  $P' \subset P$  are evaluated computationally at a particular point  $\mathbf{p} \in P'$  by simply setting  $\mathcal{P} = (P', [\mathbf{p}, \mathbf{p}])$  and solving (6.5)–(6.9). Since  $\mathcal{Y}(t, \mathcal{P})$  can be computed explicitly given  $\mathcal{X}(t, \mathcal{P})$  [i.e., through a fixed number of iterations of (6.3)], this system can be treated as a system of explicit ODEs. Thus, (6.5)–(6.9) can be solved by any numerical integration code (e.g., CVODES [27]), using a McCormick arithmetic library (see Sect. 3).

The proof of Theorem 6.1 is based on the integral form of the differential equations in (5.1),

$$\mathbf{x}(t, \mathbf{p}) = \mathbf{x}_0(\mathbf{p}) + \int_{t_0}^t \mathbf{f}(s, \mathbf{p}, \mathbf{x}(s, \mathbf{p}), \mathbf{y}(s, \mathbf{p}))ds, \quad \forall (t, \mathbf{p}) \in I \times P. \quad (6.12)$$

Based on a similar rearrangement of (6.5) and (6.6) we define a sequence of functions  $\mathcal{X}^{cv, \ell}(\cdot, \mathcal{P})$ ,  $\mathcal{X}^{cc, \ell}(\cdot, \mathcal{P}) : I \rightarrow \mathbb{R}^{n_x}$  with  $\ell \in \mathbb{N}$  by

$$\mathcal{X}^{cv, \ell+1}(t, \mathcal{P}) = \{\mathbf{x}_0\}^{cv}(\mathcal{P}) + \int_{t_0}^t \{\mathbf{f}\}^{cv}(s, \mathcal{P}, \mathcal{X}^{\ell}(s, \mathcal{P}), \mathcal{Y}^{\ell}(s, \mathcal{P}))ds, \quad (6.13)$$

$$\mathcal{X}^{cc, \ell+1}(t, \mathcal{P}) = \{\mathbf{x}_0\}^{cc}(\mathcal{P}) + \int_{t_0}^t \{\mathbf{f}\}^{cc}(s, \mathcal{P}, \mathcal{X}^{\ell}(s, \mathcal{P}), \mathcal{Y}^{\ell}(s, \mathcal{P}))ds, \quad (6.14)$$

$$\mathcal{X}^{\ell}(t, \mathcal{P}) = (X(t), X(t) \cap [\mathcal{X}^{cv, \ell}(t, \mathcal{P}), \mathcal{X}^{cc, \ell}(t, \mathcal{P})]), \quad (6.15)$$

$$\mathcal{Y}^{\ell}(t, \mathcal{P}) = \mathcal{H}_K(t, \mathcal{P}, \mathcal{X}^{\ell}(t, \mathcal{P})). \quad (6.16)$$

Using some regularity properties of  $\{\mathbf{f}\}$ , it can be shown that these sequences converge to the true solutions of (6.5)–(6.9), for every  $\mathcal{P} \in \mathbb{M}P$  and every continuous choice of  $\mathcal{X}^{cv, 0}(\cdot, \mathcal{P})$  and  $\mathcal{X}^{cc, 0}(\cdot, \mathcal{P})$ . Choosing these initial approximations such that  $[\mathcal{X}^{cv, 0}(t, \mathcal{P}), \mathcal{X}^{cc, 0}(t, \mathcal{P})] = X(t)$  for all  $t \in I$  and  $\mathcal{P} \in \mathbb{M}P$ , it follows that  $\mathcal{X}^0(t, \cdot)$  is a relaxation function for  $\mathbf{x}(t, \cdot)$  for each  $t \in I$ . Assuming that this is true of  $\mathcal{X}^{\ell}(t, \cdot)$  it is shown for  $\mathcal{X}^{\ell+1}(t, \cdot)$  as follows. First, from the definition of  $\mathcal{H}_K$  above, it is guaranteed that  $\mathcal{Y}^{\ell}(t, \cdot)$  is a relaxation function for  $\mathbf{y}(t, \cdot)$  for each  $t \in I$ . Then, the definition of  $\{\mathbf{f}\}$  and the composition theorem for relaxation functions (Theorem 3.2) imply that  $\mathcal{P} \mapsto \{\mathbf{f}\}(t, \mathcal{P}, \mathcal{X}^{\ell}(t, \mathcal{P}), \mathcal{Y}^{\ell}(t, \mathcal{P}))$  is a relaxation function for  $\mathbf{p} \mapsto \mathbf{f}(t, \mathbf{p}, \mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p}))$ . Using basic properties of the integral, it can therefore be shown that the right-most terms in (6.13) and (6.14) together form a relaxation function for the right-most term of (6.12). In fact, this same relationship is also true of the first terms on the right-hand sides of (6.13), (6.14), and (6.12). It follows then that  $\mathcal{X}^{\ell+1}(t, \cdot)$  is a relaxation function for  $\mathbf{x}(t, \cdot)$  for each  $t \in I$ . This recovers the inductive hypothesis at  $\ell + 1$ , and hence for all  $\ell$ , and taking limits then guarantees that  $\mathcal{X}(t, \cdot)$  is a relaxation function for  $\mathbf{x}(t, \cdot)$  for each  $t \in I$ , which proves the theorem. For a formal mathematical argument, the reader is referred to [81].

The state relaxations described by (6.5)–(6.9) can be considerably tightened through a slightly more involved formulation [74]:

$$\dot{\mathcal{X}}_i^{cv}(t, \mathcal{P}) = \{f_i\}^{cv}(t, \mathcal{P}, \underline{\mathcal{X}}(i, t, \mathcal{P}), \underline{\mathcal{Y}}(i, t, \mathcal{P})), \quad (6.17)$$

$$\dot{\mathcal{X}}_i^{cc}(t, \mathcal{P}) = \{f_i\}^{cc}(t, \mathcal{P}, \overline{\mathcal{X}}(i, t, \mathcal{P}), \overline{\mathcal{Y}}(i, t, \mathcal{P})), \quad (6.18)$$

$$\underline{\mathcal{X}}(i, t, \mathcal{P}) = (X(t), \mathcal{B}_i^L([\mathcal{X}^{cv}(t, \mathcal{P}), \mathcal{X}^{cc}(t, \mathcal{P})])), \quad (6.19)$$

$$\overline{\mathcal{X}}(i, t, \mathcal{P}) = (X(t), \mathcal{B}_i^U([\mathcal{X}^{cv}(t, \mathcal{P}), \mathcal{X}^{cc}(t, \mathcal{P})])), \quad (6.20)$$

$$\underline{\mathcal{Y}}(i, t, \mathcal{P}) = \mathcal{H}_K(t, \mathcal{P}, \underline{\mathcal{X}}(i, t, \mathcal{P})), \quad (6.21)$$

$$\overline{\mathcal{Y}}(i, t, \mathcal{P}) = \mathcal{H}_K(t, \mathcal{P}, \overline{\mathcal{X}}(i, t, \mathcal{P})), \quad (6.22)$$

$$\mathcal{X}(t_0, \mathcal{P}) = \{\mathbf{x}_0\}(\mathcal{P}). \quad (6.23)$$

The argument that the solutions of this system provide state relaxations is much more involved than that for (6.5)–(6.9). The validity of the bounding relations (6.2) in this case results from an argument similar to that given after Theorem 5.2 with regard to the constraints  $z_{x,i} = x_i^L(t)$  and  $z_{x,i} = x_i^U(t)$  appearing in Hypothesis (RHS) of that theorem. For this reason, the interval operators  $\mathcal{B}_i^{L/U}$  appear in (6.17)–(6.23), as they did in the bounding system (5.8)–(5.12). The convexity and concavity of the state bounds obtained via (6.17)–(6.23) again rely on the composition properties of relaxation functions (Theorem 3.2), but the argument is made difficult by the use of  $\mathcal{B}_i^{L/U}$ . In fact, due to these difficulties, (6.17)–(6.23) can only be guaranteed to provide valid state relaxations when the solutions satisfy  $[\mathcal{X}^{cv}(t, \mathcal{P}), \mathcal{X}^{cc}(t, \mathcal{P})] \subset X(t)$ ,  $\forall t \in I$ . However, this can be ensured by a modification of (6.17)–(6.23) that yields a hybrid system with sliding modes. The reader is referred to [74] for these technical details (an analogous development for explicit ODEs can be found in [82]).

In the next section, it is shown that the convex and concave parameter dependence of state bounds is very useful for solving global dynamic optimization problems. However, for standard problems in reachability analysis, it is typically desirable to have an enclosure that is not parameter dependent. Of course, state bounds provide exactly this kind of enclosure. However, state relaxations can be used to provide a sharper enclosure  $E(t) \subset \mathbb{R}^{n_x+n_y}$  through the definition

$$E(t) \equiv \bigcup_{\mathbf{p} \in P} ([\mathbf{x}^{cv}(t, \mathbf{p}), \mathbf{x}^{cc}(t, \mathbf{p})] \times [\mathbf{y}^{cv}(t, \mathbf{p}), \mathbf{y}^{cc}(t, \mathbf{p})]), \quad \forall t \in I. \quad (6.24)$$

It is straightforward to argue that  $(\mathbf{x}(t, \mathbf{p}), \mathbf{y}(t, \mathbf{p})) \in E(t)$ ,  $\forall (t, \mathbf{p}) \in I \times P$ , and moreover that  $E(t)$  is convex for each fixed  $t \in I$ . From this last observation, it follows that  $E(t)$  can be outer-approximated by a polytope of the form

$$E_{\text{poly}}(t) \equiv \{(\mathbf{z}_x, \mathbf{z}_y) : \boldsymbol{\lambda}_k^T \mathbf{z}_x + \boldsymbol{\sigma}_k^T \mathbf{z}_y \leq c_k(t), \quad k = 1, \dots, K\}, \quad (6.25)$$

by defining

$$c_k(t) \equiv \max_{\mathbf{p} \in P} \left[ \sum_{i=1}^{n_x} \max(\lambda_{k,i} x_i^{cv}(t, \mathbf{p}), \lambda_{k,i} x_i^{cc}(t, \mathbf{p})) + \sum_{i=1}^{n_y} \max(\sigma_{k,i} y_i^{cv}(t, \mathbf{p}), \sigma_{k,i} y_i^{cc}(t, \mathbf{p})) \right], \quad (6.26)$$

for all  $t \in I$ . Using the convexity/concavity properties of the state relaxations, it is straightforward to show that these maximizations are concave for any  $\lambda_k \in \mathbb{R}^{n_x}$  and  $\sigma_k \in \mathbb{R}^{n_y}$ , and can therefore be solved efficiently [74, 75].

## 6.1 Alternative Approaches

At present, no alternative approaches have been proposed for computing state relaxations for DAEs. However, several methods are available for ODEs that can likely be extended to the DAE case without undue complications. Indeed, once state bounds have been computed, the issue of existence and uniqueness is resolved and the relaxation procedure can be seen as a refinement of the bounds. In this context, one promising direction is the extension of the state relaxation methods for ODEs in [70, 71]. These methods are fundamentally different from the state relaxation method proposed here in that they do not define a relaxed dynamic system to be solved numerically [e.g., as in (6.17)–(6.23)]. Rather, these methods first discretize the dynamics, and then compute state relaxations for an approximate solution that is essentially known in factorable form. These relaxations are then made valid by adding a bound on the approximation error. Despite this difference, these methods are quite similar to the method proposed here in their use of factorable representations and McCormick relaxations as fundamental building blocks (in addition to so-called *Taylor-model arithmetic*, a more complex enclosure method for factorable functions that was not discussed in Sect. 3). Thus, the extension of these methods to DAEs following the developments of Sect. 4 is a plausible and potentially fruitful area for future research.

## 7 Global Optimization with Semi-explicit Index-One DAEs Embedded

In this section, we review the method in [74] for deterministic global optimization problems with DAEs embedded. The specific problem under consideration is discussed in detail in Sect. 2.4. As mentioned in Sect. 1, all existing global optimization algorithms for dynamic problems are based on the spatial branch-and-bound (B&B)

framework originally developed for standard NLPs [29, 69]. Accordingly, we begin with a brief overview of this approach, and subsequently extend it to DAE embedded problems using the developments of the last several sections.

## 7.1 The Spatial Branch-and-Bound Global Optimization Algorithm

Consider the standard NLP

$$\begin{aligned} \min_{\mathbf{p} \in P} \quad & J(\mathbf{p}) \\ \text{s.t.} \quad & \mathbf{G}(\mathbf{p}) \leq \mathbf{0}, \end{aligned} \tag{7.1}$$

where  $P \in \mathbb{R}^{n_p}$ , and  $J$  and  $\mathbf{G}$  are continuous on  $P$ . To solve this problem to global optimality, the spatial B&B method considers a sequence of subproblems in which (7.1) is restricted to a subinterval  $P^l \subset P$ :

$$\begin{aligned} \min_{\mathbf{p} \in P^l} \quad & J(\mathbf{p}) \\ \text{s.t.} \quad & \mathbf{G}(\mathbf{p}) \leq \mathbf{0}, \end{aligned} \tag{7.2}$$

The basic requirement for applying spatial B&B is that, for any subinterval  $P^l \subset P$  (which may be  $P$  itself), procedures are available that compute guaranteed upper and lower bounds on the optimal objective value of (7.2). These bounds are denoted by  $UBD^l$  and  $LBD^l$ , respectively. Since the value of the objective function at any feasible point provides an upper bound on the optimal objective value of (7.2),  $UBD^l$  can be computed by solving (7.1) to local optimality. Computing a lower bound is substantially more difficult and is the key step in the spatial B&B algorithm. Methods for accomplishing this are discussed below.

Supposing that upper and lower bounding procedures are available, the spatial B&B algorithm proceeds as follows. First, upper and lower bounds are computed for the optimal objective value of (7.1). Since these bounds apply to the original problem of interest, rather than to the subproblem (7.2), they are denoted by  $UBD$  and  $LBD$ , respectively. If it happens that  $UBD - LBD$  is less than a specified tolerance  $\varepsilon$ , then the B&B algorithm terminates, having bracketed the optimal objective value of (7.1) within the given tolerance. An estimate of the solution value  $\mathbf{p}^*$  is then given by the value which attained the upper bound  $UBD$ . If this termination test fails, then  $P$  is partitioned into two subintervals, termed *branching*, typically by bisection in its dimension of largest width. These subintervals inherit the bounds  $UBD$  and  $LBD$ , which are obviously valid for the corresponding subproblems (7.2) on account of being valid for (7.1). These two subintervals are then added to a stack  $\Sigma$  of subintervals, or *nodes*, to be processed that is maintained throughout the algorithm.

At the beginning of a generic iteration of the algorithm,  $UBD$  and  $LBD$  are the best known upper and lower bounds on the optimal objective value of (7.1), respectively, and the stack  $\Sigma$  contains a number of nodes  $P^l$ , each of which is equipped with upper and lower bounds  $UBD^l$  and  $LBD^l$  that have been inherited from the parent node from which it was generated through bisection. Collectively, the nodes  $P^l$  may not form a partition of  $P$ , but the complement of  $\cup_l P^l$  in  $P$  will have been proven not to contain the optimal solution of (7.1) through the procedures below. The iteration proceeds by selecting from the stack a node  $P^l$  for which  $LBD^l = LBD$ . The upper and lower bounds  $UBD^l$  and  $LBD^l$  are then refined by computing bounds on the optimal objective value of (7.2) using the procedures that we have assumed to be available. If it is found that (7.2) is infeasible, then  $P^l$  is eliminated from further consideration and a new element is selected from the stack. In this case, we say that  $P^l$  is *fathomed by infeasibility*. Otherwise, upper and lower bounds on the optimal objective value of the original problem (7.1) are updated according to

$$UBD := \min_k UBD^k \quad \text{and} \quad LBD := \min_k LBD^k, \quad (7.3)$$

where the min is taken over all elements of  $\Sigma$ . These assignments are valid because the complement of  $\cup_k P^k$  in  $P$  has been shown not to contain a global optimum of (7.1). Moreover, if  $P^l$  was the only element of  $\Sigma$  for which  $LBD^l = LBD$  at the beginning of the iteration, and if  $LBD^l$  was improved by the application of the lower bounding procedure to (7.2), then  $LBD$  is improved by this assignment. If  $UBD$  is improved by this assignment, then there is an opportunity to fathom some nodes in the stack. This is done by checking the inequality  $LBD^k > UBD$  for every  $P^k \in \Sigma$ . If this is true for some  $P^k$ , then the optimal solution cannot lie in  $P^k$  and  $P^k$  is eliminated from further consideration. In this case,  $P^k$  is said to be *fathomed by value dominance*. If  $P^l$  has not been fathomed either by infeasibility or by value dominance, then it is bisected and the two resulting nodes are added to the stack.

The iteration outlined above is repeated until either the stack becomes empty, indicating that (7.1) is infeasible, or it is found in some iteration that  $UBD - LBD < \varepsilon$ , indicating that a point  $\mathbf{p}^*$  has been found which achieves an objective value within  $\varepsilon$  of the globally optimal objective value. Roughly, if the lower bounding procedure has the property that it provides sharper bounds on smaller intervals  $P^l$  and becomes exact in the limit as  $P^l$  tends toward a singleton, then it can be shown that one of these outcomes will occur after finitely many iterations [29]. A notable exception to this result can occur in the presence of inequality constraints if the algorithm is unable to find feasible points at which to compute upper bounds. In the following discussion, we assume that upper bounds can be computed without difficulty.

Due to the repeated partitioning of  $P$ , the spatial B&B algorithm exhibits worst-case exponential run-time with respect to the dimension of  $\mathbf{p}$  and the magnitude of  $1/\varepsilon$ . In practice, the primary determinants of the run-time are the computational cost and the accuracy of the lower bounding procedure. In addition, a number of more

advanced techniques have been developed which can greatly accelerate convergence through the use of domain reduction techniques [67–69]. Thus, while it is true that the basic procedure outlined above can be prohibitively expensive, impressive results have been achieved for many challenging problems using advanced implementations of the method [66, 67, 85, 92].

Several methods are available for computing lower bounds on the optimal objective value of the subproblem (7.2). If  $J$  is factorable, we may use the lower bound of the natural interval extension  $[J](P^l)$ , as described in Sect. 3. Although some early implementations are based on this approach [32], the lower bounds computed in this way are relatively weak. Moreover, these bounds obey a first-order convergence rate property [56], while it has been demonstrated that at least second-order convergence is required to avoid serious convergence problems in spatial B&B algorithms [95]. In most modern implementations, lower bounds are computed by constructing and solving convex underestimating programs [5, 23, 50, 92]. A convex underestimating program for (7.2) is a program that is convex and has an optimal objective value that is guaranteed to underestimate that of (7.2). Although there are many ways to accomplish this, we consider the program

$$\begin{aligned} \min_{\mathbf{p} \in P^l} \quad & J_l^{cv}(\mathbf{p}) \\ \text{s.t.} \quad & \mathbf{G}_l^{cv}(\mathbf{p}) \leq \mathbf{0}, \end{aligned} \tag{7.4}$$

where  $J_l^{cv}$  and  $\mathbf{G}_l^{cv}$  are convex relaxations of  $J$  and  $\mathbf{G}$  on  $P^l$ , respectively. This program is clearly convex, and hence solvable to global optimality using standard local optimization techniques. Moreover, its optimal objective value is easily seen to underestimate that of (7.2). If  $J$  and  $\mathbf{G}$  are factorable functions,  $J_l^{cv}$  and  $\mathbf{G}_l^{cv}$  can be obtained from the natural McCormick extensions  $\{J\}$  and  $\{\mathbf{G}\}$ , as described in Sect. 3. In fact, when  $J$  and  $\mathbf{G}$  are factorable, a number of approaches are available for constructing a convex relaxations [1, 2, 5, 50], and more generally to construct convex underestimating programs for (7.2) [92]. This last method is used in the popular code BARON.

## 7.2 A Lower Bounding Procedure for Optimization with DAEs

We now consider the application of spatial B&B global optimization to the dynamic optimization problem (2.8), which we restate here for convenience:

$$\begin{aligned} \min_{\mathbf{p} \in P} \quad & \phi(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) \\ \text{s.t.} \quad & \eta(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})) \leq \mathbf{0}. \end{aligned} \tag{7.5}$$

Recall that  $(\phi, \eta) : D_p \times D_x \times D_y \rightarrow \mathbb{R} \times \mathbb{R}^{n_c}$  are continuous, and  $(\mathbf{x}, \mathbf{y})$  is a regular solution of (5.1) on  $I \times P$  that uniquely satisfies the additional initial condition

$$\mathbf{y}(t_0, \hat{\mathbf{p}}) = \hat{\mathbf{y}}_0, \quad (7.6)$$

where  $\hat{\mathbf{p}} \in P$  and  $\hat{\mathbf{y}}_0 \in D_y$  satisfy  $\mathbf{g}(t_0, \hat{\mathbf{p}}, \mathbf{x}_0(\hat{\mathbf{p}}), \hat{\mathbf{y}}_0) = \mathbf{0}$  and  $\det \frac{\partial \mathbf{g}}{\partial \mathbf{y}}(t_0, \hat{\mathbf{p}}, \mathbf{x}_0(\hat{\mathbf{p}}), \hat{\mathbf{y}}_0) \neq 0$ . The existence of such a solution can be verified by the state bounding algorithm of Sect. 5, and in fact this will be done during the course of the optimization algorithm described here.

The optimization problem (7.5) can be written in the form of the standard optimization problem (7.1) with the definitions

$$J(\mathbf{p}) \equiv \phi(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})), \quad (7.7)$$

$$\mathbf{G}(\mathbf{p}) \equiv \eta(\mathbf{p}, \mathbf{x}(t_f, \mathbf{p}), \mathbf{y}(t_f, \mathbf{p})). \quad (7.8)$$

Thus, spatial B&B is applicable to (7.5), provided that upper and lower bounds on the subproblems (7.2) can be computed, for any  $P \subset P$ , with the definitions (7.7) and (7.8). Any local dynamic optimization algorithm can be used to compute the required upper bounds. Moreover, a valid lower bound can be computed by solving the convex problem (7.4) locally, provided that convex relaxations  $J^{cv}$  and  $\mathbf{G}^{cv}$  can be derived for (7.7) and (7.8). Although many methods for computing convex relaxations were discussed above, the key assumption in all of these techniques is that the objective and constraint functions are factorable functions of  $\mathbf{p}$ . Of course, this is not the case for (7.7) and (7.8), precisely because the parametric DAE solutions  $\mathbf{x}(t_f, \cdot)$  and  $\mathbf{y}(t_f, \cdot)$  are not factorable. Thus, we find ourselves in the now familiar situation of requiring global information about non-factorable functions.

The difficult work needed to address this problem has already been done in the last several sections, leading to state relaxations for  $(\mathbf{x}, \mathbf{y})$ . What remains is only propagate these relaxations through  $\phi$  and  $\eta$  to obtain  $J^{cv}$  and  $\mathbf{G}^{cv}$ . To do this, we will use the factorable representations of  $\phi$  and  $\eta$  to compute their natural McCormick extensions, and subsequently leverage the composition theorem for relaxation functions. The following assumption is required:

**Assumption 7.1** *The functions  $\phi$  and  $\eta$  are factorable with natural McCormick extensions  $\{\phi\} : \mathbb{M}D_p \times \mathbb{M}D_x \times \mathbb{M}D_y \rightarrow \mathbb{M}\mathbb{R}$  and  $\{\eta\} : \mathbb{M}D_p \times \mathbb{M}D_x \times \mathbb{M}D_y \rightarrow \mathbb{M}\mathbb{R}^{n_c}$ .*

Relaxation functions  $(\mathcal{J}, \mathcal{G}) : \mathbb{M}P \rightarrow \mathbb{M}\mathbb{R} \times \mathbb{M}\mathbb{R}^{n_c}$  for  $(J, \mathbf{G})$  can now be defined through the following procedure for each  $\mathcal{P} = (P^l, [\mathbf{p}^{cv}, \mathbf{p}^{cc}]) \in \mathbb{M}P$ :

1. Compute state bounds  $X(t)$  and  $Y(t)$  on  $P^l$  by solving (5.8)–(5.12) with  $P^l$  in place of  $P$ .
2. Evaluate the relaxation functions  $\mathcal{X}(t_f, \cdot)$  and  $\mathcal{Y}(t_f, \cdot)$  at  $\mathcal{P}$  by solving (6.17)–(6.23).

### 3. Assign

$$\mathcal{J}(\mathcal{P}) \equiv \{\phi\}(\mathcal{P}, \mathcal{X}(t_f, \mathcal{P}), \mathcal{Y}(t_f, \mathcal{P})), \quad (7.9)$$

$$\mathcal{G}(\mathcal{P}) \equiv \{\eta\}(\mathcal{P}, \mathcal{X}(t_f, \mathcal{P}), \mathcal{Y}(t_f, \mathcal{P})). \quad (7.10)$$

Since  $\mathcal{X}(t_f, \cdot)$  and  $\mathcal{Y}(t_f, \cdot)$  have been shown to be relaxation functions for  $\mathbf{x}(t_f, \cdot)$  and  $\mathbf{y}(t_f, \cdot)$ , respectively, and  $\{\phi\}$  is a relaxation function for  $\phi$  by definition, Theorem 3.2 implies that  $\mathcal{J}$  is a relaxation function for  $J$ . An analogous argument holds for  $\mathcal{G}$ . As discussed previously, the required McCormick arithmetic in these computations can be done automatically using the McCormick arithmetic library MC++ (<http://www3.imperial.ac.uk/people/b.chachuat/research>).

Now, using the notation  $\mathcal{J} = ([\mathcal{J}^L, \mathcal{J}^U], [\mathcal{J}^{cv}, \mathcal{J}^{cc}])$ , and similarly for  $\mathcal{G}$ , define

$$J_l^{cv}(\mathbf{p}) \equiv \mathcal{J}^{cv}((P^l, [\mathbf{p}, \mathbf{p}]]), \quad \mathbf{G}_l^{cv}(\mathcal{P}) \equiv \mathcal{G}^{cv}((P^l, [\mathbf{p}, \mathbf{p}])). \quad (7.11)$$

By the properties of relaxation functions with arguments of the form  $(P^l, [\mathbf{p}, \mathbf{p}])$ , it follows that  $J_l^{cv}$  and  $\mathbf{G}_l^{cv}$  are convex relaxations of  $J$  and  $\mathbf{G}$  as defined by (7.7) and (7.8), respectively (see Sect. 3.2). Thus, a valid lower bounding problem for (7.5) on  $P^l$  is given by

$$\begin{aligned} \min_{\mathbf{p} \in P^l} \quad & J_l^{cv}(\mathbf{p}) \\ \text{s.t.} \quad & \mathbf{G}_l^{cv}(\mathbf{p}) \leq \mathbf{0}. \end{aligned} \quad (7.12)$$

Since the procedure for evaluating  $J_l^{cv}(\mathbf{p})$  and  $\mathbf{G}_l^{cv}(\mathbf{p})$  outlined above involves the solution of (6.17)–(6.23) for each  $\mathbf{p}$ , (7.12) is itself a dynamic optimization problem. However, it is guaranteed to be convex, so that it can be solved to global optimality using a local solver, thereby providing the lower bound required by the spatial B&B algorithm. Note that the bounding system (5.8)–(5.12) is solved only once during the solution of (7.12) because (5.8)–(5.12) depend only on  $P^l$ , and not on  $\mathbf{p}$ . Implementing this requires that one stores the interpolating polynomial used by the integrator in each time step of the state bounds integration, so that accurate bound values can be recovered during integration of the state relaxations. This information can be obtained as output from the DAE solver IDA [27].

Due to the use of McCormick's relaxation technique, it is possible that the objective and constraints in (7.12) are non-differentiable. However, subgradients can be computed using the subgradient propagation rules for McCormick relaxations developed in [53], along with the developments for state relaxations in [74]. These subgradients can be computed automatically using the McCormick arithmetic library MC++ (<http://www3.imperial.ac.uk/people/b.chachuat/research>). Because (7.12) is a potentially nonsmooth convex optimization problem, it is best to solve it using a specialized nonsmooth solver, such as a bundle method [44, 48]. However, these methods are not as mature as those for differentiable problems,

and the available solvers of this type remain problematic. Numerical experiments using the sequential quadratic programming code SNOPT are presented in [74], without serious numerical difficulties. An alternative approach is to use the values  $J_l^{cv}(\hat{\mathbf{p}})$  and  $\mathbf{G}_l^{cv}(\hat{\mathbf{p}})$  for some fixed  $\hat{\mathbf{p}} \in P^l$ , along with subgradients at this point, to construct affine underestimators for  $J$  and  $\mathbf{G}$  on  $P^l$ . This reduces the lower bounding problem to a linear program and thereby avoids the issue of nonsmoothness. The lower bounds generated by this scheme are more conservative. However, there is a significant advantage in terms of the computational cost-per-node because (6.17)–(6.23) are integrated only once during the solution of the lower bounding problem. This method is used in the case study in Sect. 8.

### 7.3 Alternative Approaches

As mentioned in Sect. 1, an alternative approach to global dynamic optimization is to combine the simultaneous formulation, in which the dynamics are discretized in time and reduced to a large system of algebraic equations, with state-of-the-art branch-and-bound codes for standard NLPs [15]. A serious drawback of this approach is that it generates a very large number of new decision variables representing the values of the differential and algebraic states at each time point in the discretization scheme. While the special structure of this large-scale NLP makes it tractable for the purposes of local optimization, it has not been demonstrated that this structure can mitigate the characteristic worst-case exponential dependence of branch-and-bound run-time on the number of decision variables. However, even if this could be done, there is a more subtle problem that we are now in a position to appreciate. For a branch-and-bound search to be exhaustive, upper and lower bounds on the permissible values of all decision variables must be provided by the user. Given the fact that the decisions in the simultaneous formulation include the state variables themselves, at numerous time points throughout the horizon of interest, it follows that a reachable set enclosure must be provided as input. Thus, the primary complication of the sequential approach appears again in the simultaneous formulation and apparently eliminates the advantages of discretization.

Looking toward future improvements in relaxation theory for dynamic optimization problems, it is useful to note that state-of-the-art branch-and-bound codes for standard NLPs do not typically apply McCormick's relaxation technique directly, nor do they use the simplistic relaxation of (7.2) by (7.4). In particular, it is widely appreciated that accounting for the interdependence of the objective and constraint functions, as well as the interdependence of individual terms within each of these equations, is essential to obtaining tight relaxations in general. In the context of dynamic optimization problems, these observations have not been applied principally because of the lack of factorable expressions for the states, objective, and constraints. Indeed, the generality of McCormick's relaxation technique, and specifically its composition properties, are central to overcoming this problem. Moreover, moving beyond this approach is not a straightforward application of

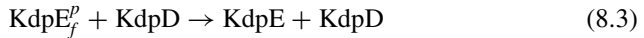
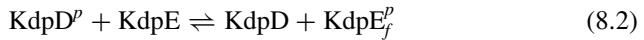
existing relaxation techniques for NLPs. This is because the interdependence of different equations and sub-expressions in dynamic optimization problems are most often a result of correlations between the state variables. Thus, these interdependencies cannot be detected by methods that rely on a factorable representation. Rather, they are properties of the differential-algebraic system that must be inferred from the factorable representations of the governing equations using insights from dynamical systems theory. This is an open challenge that needs to be addressed moving forward. As a specific example, it suggests an alternative approach in which the state variables are not relaxed component-wise, as is done in (6.17)–(6.23). Instead, one might try for a more general convex enclosure of the reachable set as a whole. At present, this idea has not been investigated in the state relaxation literature, either for ODEs or DAEs, and is a promising area for future investigations.

## 8 Numerical Results and Directions for Improvement

In this section, the methods developed in Sects. 5–7 are demonstrated on a parameter estimation problem from [35]. Shortcomings and directions for future research are also highlighted.

### 8.1 Problem Formulation

Consider the following two-component signal transduction network:



This network provides a simplified description of the regulation of  $\text{K}^+$  uptake in *E. coli*, and is mechanistically similar to a variety stimulus response mechanisms found in bacteria, archaea, and plants [35]. A superscript  $p$  above denotes a phosphorylated species. Thus, reaction (8.1) describes the autophosphorylation of KdpD, (8.2) is a phosphoryl transfer reaction, (8.3) describes the dephosphorylation of KdpE<sub>f</sub>, and (8.4) describes the binding of DNA to form the transcription complex KdpE-DNA.

In [35], purified KdpD, KdpE, and DNA fragments were combined in a mixture of ATP/ADP, and the concentrations of the phosphorylated proteins were measured over time. Specifically, the measured concentrations were  $C_{\text{KdpD}^p}$  and  $C_{\text{KdpE}_f^p} + 2C_{\text{KdpE-DNA}}$ , where the latter figure is the total concentration of phosphorylated

KdpE in both its free and complexed forms. The concentration of ATP/ADP were essentially constant during the experiments.

To present the model equations for this system in the notation of previous sections, we write

$$\mathbf{x} \equiv (C_{\text{KdpD}^p}, C_{\text{KdpE}_f^p} + 2C_{\text{KdpE-DNA}}), \quad (8.5)$$

$$\mathbf{y} \equiv (C_{\text{KdpE}_f^p}, C_{\text{DNA}_f}, C_{\text{KdpD}}, C_{\text{KdpE}}). \quad (8.6)$$

We further collect some known initial concentrations in the vector  $\mathbf{m}$  defined as

$$\mathbf{m} \equiv (C_{\text{ATP0}}, C_{\text{ADP0}}, C_{\text{DNA0}}, C_{\text{KdpD0}}, C_{\text{KdpE0}}) = (100, 8, 100, 1, 4) \mu\text{M}. \quad (8.7)$$

Finally, the forward and reverse rate constants of reaction  $i$  are denoted as  $k_i$  and  $k_{-i}$ , the reactions (8.1)–(8.4) are numbered consecutively from 1 to 4, and the equilibrium constant for DNA binding in (8.4) is denoted by  $K_b$ . With these definitions, the model equations are

$$\dot{x}_1 = k_1 m_1 y_3 - k_{-1} m_2 x_1 - k_2 x_1 y_4 + k_{-2} y_3 y_1, \quad (8.8)$$

$$\dot{x}_2 = k_2 x_1 y_4 - k_{-2} y_3 y_1 - k_3 y_3 y_1, \quad (8.9)$$

$$0 = -x_2 + y_1 + 2 \frac{y_1^2 y_2}{K_b}, \quad (8.10)$$

$$0 = -m_3 + y_2 + \frac{y_1^2 y_2}{K_b}, \quad (8.11)$$

$$0 = -m_4 + y_3 + x_1, \quad (8.12)$$

$$0 = -m_5 + y_4 + x_2. \quad (8.13)$$

To arrive at these equations, differential species balances are first derived for  $x_1$ ,  $x_2$ , and  $C_{\text{KdpE-DNA}}$  assuming mass-action kinetics in (8.1)–(8.4). In this form, the system specification is completed by three algebraic equations that specify the concentrations of KdpD, KdpE, and DNA using the observation that the total concentration of each of these species in all its phosphorylated, unphosphorylated, and complexed forms must equal  $C_{\text{KdpD0}}$ ,  $C_{\text{KdpE0}}$ , and  $C_{\text{DNA0}}$ , respectively. This model is then reduced by the observation that DNA binding is fast relative to the other reactions, and can therefore be assumed to be in quasi-equilibrium. This leads to the algebraic relation

$$y_1^2 y_2 = K_b C_{\text{KdpE-DNA}}, \quad (8.14)$$

which is substituted throughout to arrive at the DAEs (8.8)–(8.13).

**Table 1** Data used for the parameter estimation problem (8.15), based on experimental data from Fig. 3(B) in [35]

| $\ell$ | $t_\ell$ (min) | $\hat{x}_1$ ( $\mu\text{M}$ ) | $\hat{x}_2$ ( $\mu\text{M}$ ) |
|--------|----------------|-------------------------------|-------------------------------|
| 0      | 0              | 0                             | 0                             |
| 1      | 1              | 0.0027                        | 0.003                         |
| 2      | 2.5            | 0.0068                        | 0.0075                        |
| 3      | 5              | 0.0068                        | 0.0108                        |
| 4      | 7.5            | 0.0063                        | 0.0148                        |
| 5      | 10             | 0.0069                        | 0.019                         |
| 6      | 12.5           | 0.0067                        | 0.0205                        |
| 7      | 15             | 0.0072                        | 0.0230                        |

In [35], parameter estimation was done using a local dynamic optimization method to find the unknown rate constants  $k_i$  and  $k_{-i}$  for all  $i \in \{1, \dots, 4\}$ . Here, we consider solving this problem globally. To simplify the problem, we fit only  $k_1$  and  $k_3$ , which were found to be the parameters with the highest sensitivities in [35]. Thus, we solve the optimization problem

$$\begin{aligned}
 \min_{k_1, k_3} \quad & \frac{1}{7} \sum_{\ell=1}^7 \left[ \left( \frac{x_1(t_\ell, k_1, k_3) - \hat{x}_1(t_\ell)}{\hat{x}_1(t_\ell)} \right)^2 + \left( \frac{x_2(t_\ell, k_1, k_3) - \hat{x}_2(t_\ell)}{\hat{x}_2(t_\ell)} \right)^2 \right] \\
 \text{s.t.} \quad & k_1 \in [0.001, 0.01] \text{ (h}\mu\text{M)}^{-1}, \quad k_3 \in [10, 300] \text{ (h}\mu\text{M)}^{-1}, \\
 & (8.8)\text{--}(8.13) \quad \text{hold} \quad \forall t \in [0, 0.25] \text{ h}, \\
 & \mathbf{x}(t_0, k_1, k_3) = \mathbf{0} \mu\text{M}, \\
 & \mathbf{y}(t_0, k_1, k_3) = (0, m_3, m_4, m_5) \mu\text{M}.
 \end{aligned} \tag{8.15}$$

Above,  $\hat{x}_1$  and  $\hat{x}_2$  denote measured values, which are given in Table 1. These data are based on experimental values reported in [35]. Unfortunately, the measured values were not reported directly, so the data in Table 1 are estimated from Fig. 3(B) in [35]. Although inaccurate, this data is sufficient for the illustrative purposes of this example. The fixed rate constants in (8.15) are set to the optimal values found in [35]:  $k_{-1} = 0.0029 \text{ (h}\mu\text{M)}^{-1}$ ,  $k_2 = 108 \text{ (h}\mu\text{M)}^{-1}$ ,  $k_{-2} = 1080 \text{ (h}\mu\text{M)}^{-1}$ ,  $K_b = \frac{36}{540} \text{ (}\mu\text{M)}^2$ .

## 8.2 Global Dynamic Optimization

For the purposes of solving (8.15) to global optimality, some further rearrangements of (8.8)–(8.13) are helpful. First, the final three algebraic equations are solved analytically for  $y_2$ ,  $y_3$ , and  $y_4$  respectively, and substituted into the remaining equations to arrive at a system of two differential equations and one nonlinear algebraic equation. Secondly, the resulting equations are rearranged to give factorable

representations that are favorable for interval and McCormick computations. The final form used for the computations below is

$$\dot{x}_1 = k_1 m_1 (m_4 - x_1) - k_{-1} m_2 x_1 - k_2 x_1 (m_5 - x_2) + k_{-2} (m_4 - x_1) y_1, \quad (8.16)$$

$$\dot{x}_2 = -[(k_{-2} + k_3) y_1 + k_2 (m_5 - x_2)] (m_4 - x_1) + k_2 m_4 (m_5 - x_2), \quad (8.17)$$

$$0 = y_1^3 + 2m_3 y_1^2 + K_b y_1 - (K_b + y_1^2) x_2. \quad (8.18)$$

The global dynamic optimization algorithm is also provided with the analytical derivative of the algebraic equation with respect to  $y_1$  with the factorable representation

$$\frac{\partial g}{\partial y_1} = y_1 (3y_1 + 2(2m_3 - x_2)) + K_b. \quad (8.19)$$

The purpose of these rearrangements is to reduce the conservatism of the enclosures obtained through the interval and McCormick methods described in the preceding sections. The analytical solution of (8.11)–(8.13) avoids potentially conservative interval and McCormick iterations on these equations using the methods of Sect. 4. These iterations are applicable to general nonlinear equations, and do not always reduce to the natural interval extension of the analytical solution when the latter is possible. The factorable representations used in (8.18) and (8.19) have already been justified in Sect. 4.1, and are designed to minimize the number of appearances of variables that will be interval- or McCormick-valued in the interval or McCormick evaluations of these expressions. The rearrangements to the right-hand side of (8.17) are similarly motivated. In this case, we note that the state bounding method of Sect. 5 uses exclusively interval evaluations of (8.17) in which  $x_2$  is a degenerate interval, while  $x_1$ ,  $y_1$ , and  $k_3$  are potentially nondegenerate intervals. Thus, the rearranged equation reduces the appearances of  $x_1$  from 2 to 1, without concern for the additional appearances of  $x_2$  so generated. Rearrangements of the type discussed here and in Sect. 4.1 are standard in interval computations and are typically not difficult to perform. However, good arrangements depend on which variables will be interval- or McCormick-valued in a given expression, and can become cumbersome for large systems. Automating these rearrangements is an interesting topic for future research at the intersection of numerical and symbolic computations.

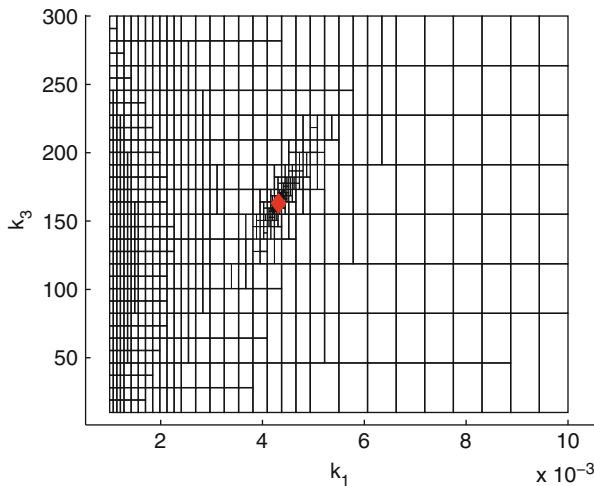
Using Eqs. (8.16)–(8.18), the least-squares estimation problem (8.15) was solved to global optimality using branch-and-bound as described in Sect. 7. In each node, upper bounds were determined by simply evaluating the objective at the midpoint  $\mathbf{p}_{\text{mid}}^l$  of the parameter interval  $P^l$ . To compute a lower bound for the optimal objective value on  $P^l$ , state bounds were first computed on  $P^l$  by solving (5.8)–(5.11). The parameter  $\gamma$  in (5.10) and (5.11) was set to  $10^{-6}$  and (5.8)–(5.11) were solved using IDA [27] with relative and absolute tolerances of  $10^{-6}$ . Subsequently, state relaxations were computed at  $\mathbf{p}_{\text{mid}}^l$  by solving (6.17)–(6.23) with  $K = 5$  using CVODES [27] with relative and absolute tolerances of  $10^{-6}$ . Additionally, sensitivity

equations for (6.17)–(6.23) were integrated in order to provide subgradients for the state relaxations as described in detail in [74]. The state relaxations and subgradients were then propagated through the least-squares objective in (8.15) using McCormick arithmetic along with the subgradient propagation rules in [53]. This gives the relaxation value  $J_i^{cv}(\mathbf{p}_{\text{mid}}^l)$  along with the associated subgradient at  $\mathbf{p}_{\text{mid}}^l$ . Using this information, an affine relaxation of the objective was formulated and minimized on the box  $P^l$  to obtain the final lower bound. The termination condition for the branch-and-bound algorithm was specified as  $|UBD - LBD| \leq 10^{-5}$ . No domain reduction techniques were used. Branching was done using a weighted-width heuristic that bisects the  $i$ th parameter interval if  $i \in \arg\max(W_i w(P_i^l))$ . Here,  $w$  denotes the interval width and  $W_i \in \mathbb{R}_+$  is a weighting factor. For the present problem,  $W_1 = 10^5$  and  $W_2 = 1$  were specified based on the observation that the sensitivity of the objective function to  $k_1$  at  $\mathbf{p}_{\text{mid}}$  of the root node is larger than that with respect to  $k_3$  by a factor of  $10^5$ .

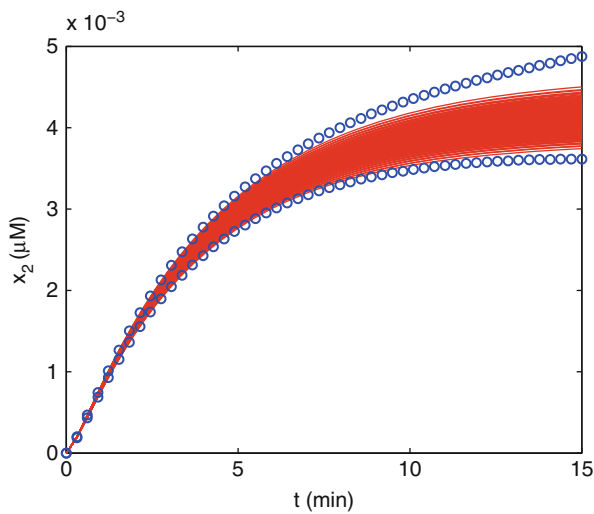
The  $\epsilon$ -global minimum objective value was found to be 0.029069, which is attained at  $\mathbf{p}^* = (k_1^*, k_3^*) = (4.31348 \times 10^{-3}, 162.9297) (\text{h}\mu\text{M})^{-1}$ . This optimal solution is quite different from that found in [35];  $(k_1, k_3) = (2.9 \times 10^{-3}, 90) (\text{h}\mu\text{M})^{-1}$ . However, while it is likely that the value found in [35] is not globally optimal, the discrepancy here is probably primarily due to the error introduced in our estimation of the experimental data from Fig. 3(B) in [35].

The global solution for (8.15) was found in 52s. The optimization was done on a virtual machine with 4 GB of memory running Ubuntu 12.04 on a Macbook Pro OSX 10.9.2 host with a 2.6 GHz Intel Core i7 CPU. The branch-and-bound algorithm visited 1587 nodes in total, generating the partition of the search space shown in Fig. 2. The state bounds and the relaxation of the objective function computed in the 1100th node visited by the algorithm are shown in Figs. 3 and 4 as representative examples.

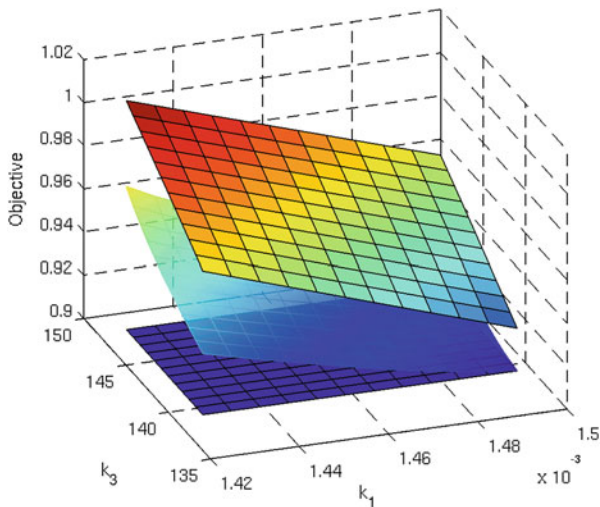
Although the state bounds in Fig. 3 are relatively sharp, the bounding technique can generate rapidly diverging bounds on the larger  $P^l$  intervals that occur early in the branch-and-bound tree. This is a common problem in reachability analysis and also occurs for problems with ODEs embedded. However, a unique feature of the DAE methods here is the need to pass an inclusion test to verify the existence of a solution of the algebraic equations as described in Sect. 4. This manifests itself in the state bounding method through the need to satisfy Eqs. (5.10) and (5.11) everywhere along the bounding trajectories, which can become difficult or impossible when the interval arguments become large in width. For example, the state bounds computed in the 900th node visited by the branch-and-bound algorithm are shown in Fig. 5. These are clearly divergent, and although the bounding procedure succeeds, slightly larger parameter intervals cause it to fail due to an inability to satisfy (5.10) and (5.11) for  $t$  near the end of the horizon. For such nodes, the lower bound is set to  $-\infty$  and the node is branched. These failures make a significant contribution to the overall number of nodes visited by the branch-and-bound algorithm, as well as to the CPU time. Thus, an area for future research is in deriving sharper inclusion tests that can potentially succeed on larger parameter intervals.



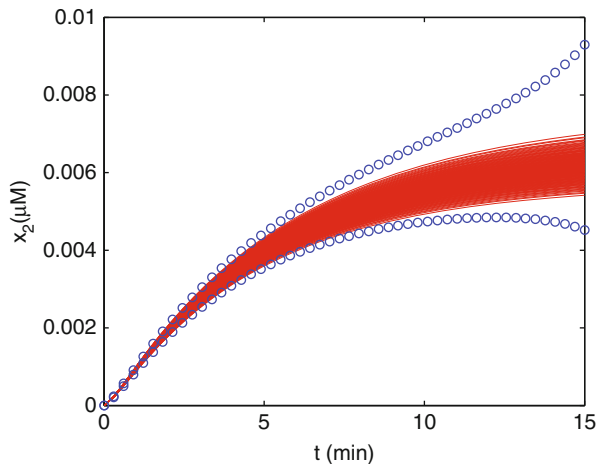
**Fig. 2** Intervals in the search space  $P = [0.001, 0.01] \times [10, 300]$  that were fathomed by value dominance during the branch-and-bound global solution of (8.15). The *diamond* indicates the global solution found



**Fig. 3** Upper and lower state bounds for  $x_2$  (circles) computed on  $P^{1100} = [1.421875 \times 10^{-3}, 1.4921875 \times 10^{-3}] \times [136.875, 145.9375]$ , as well as true solution trajectories of (8.16)–(8.18) (solid) computed on a uniform  $12 \times 12$  grid of  $P^{1100}$ .  $P^{1100}$  is the interval corresponding to the 1100th node processed by the branch-and-bound algorithm while solving (8.15)



**Fig. 4** Objective function of (8.15) on  $P^{1100}$  (upper surface), along with its convex relaxation (middle surface) and interval lower bound (lower surface).  $P^{1100}$  is the interval corresponding to the 1100th node processed by the branch-and-bound algorithm while solving (8.15)



**Fig. 5** Upper and lower state bounds for  $x_2$  (circles) computed on  $P^{900} = [1.5625 \times 10^{-3}, 1.703125 \times 10^{-3}] \times [118.75, 127.8125]$ , as well as true solution trajectories of (8.16)–(8.18) (solid) computed on a uniform  $12 \times 12$  grid of  $P^{900}$ .  $P^{900}$  is the interval corresponding to the 900th node processed by the branch-and-bound algorithm while solving (8.15)

A further area for research in DAE state bounding is to consider the use of a priori information that is known about the DAE reachable set, for example, through physical arguments. This has been studied in the case of ODE state bounding techniques in [76, 80]. It was shown there that many models of interest in chemical engineering applications satisfy affine solution invariants of the form  $\mathbf{M}(\mathbf{x}(t, \mathbf{p}) - \mathbf{x}_0(\mathbf{p})) = \mathbf{0}$ ,  $\forall (t, \mathbf{p}) \in I \times P$ , and that these invariants can be exploited within the state bounding procedure to provide dramatically improved bounds in many cases. The same observation likely holds for DAE models. Indeed, the algebraic equations (8.12) and (8.13) are derived from species balances and are used to replace differential balances on  $y_3$  and  $y_4$ . However, if these differential balances were included in the model, then (8.12) and (8.13) would be redundant and would describe affine solution invariants. Experience with ODEs suggest that these could be used in the bounding procedure to significantly improve the computed bounds on larger  $P^l$  intervals. What remains to be done is to establish methods for exploiting this information within the DAE state bounding theory in a manner analogous to the ODE methods in [76, 80].

Figure 2 clearly shows that there is a large accumulation of small intervals  $P^l$  surrounding the unconstrained global minimum, which adds significantly to the computational cost of global optimization. This phenomenon is known as the cluster problem and has been investigated in a number of articles [18, 95]. The severity of this problem is known to be related to the convergence order of the relaxation method used; i.e., the rate at which the convex relaxations converge to the original objective function as the parameter interval  $P^l$  tends toward degeneracy at a global minimizer,  $P^l \rightarrow [\mathbf{p}^*, \mathbf{p}^*]$ . The cluster problem is severe for first-order convergent methods, non-existent for third-order convergent methods, and second-order methods constitute a boundary case in which the behavior is determined by the size of the pre-factor in the convergence order estimate. In general, computing a third-order convergent relaxation is thought to be NP-hard [60]. At present, there are no published results on the convergence order of relaxation methods for dynamic optimization problems. However, preliminary research suggests that the ODE state relaxation methods in [82, 84] are both second-order convergent, with the pre-factor for the latter being potentially much better than for the former. However, the relaxation methods for implicit functions presented in Sect. 4 are thought to be only first-order convergent, which would then imply first-order convergence of the DAE relaxation methods presented here. Thus, a third area for future research is in developing second-order convergent relaxation methods for implicit functions, and hence for DAE solutions. It is expected that such a development would result in a very substantial reduction in CPU time for DAE embedded global optimization problems.

## 9 Conclusions

In this article, we have given a tutorial overview of the main concepts and tools used for reachability analysis and deterministic global optimization of nonlinear DAE models. These problems are highly interrelated, and are *global* problems in the sense that they concern the parametric solutions of a DAE model over a potentially large range of model parameters, rather than locally about a single value. The fundamental tool used to compute global information in the methods presented here, as well as similar methods in the literature, is the factorable representation of a function, along with its natural interval and McCormick extensions. The primary challenge when dealing with DAE systems is that the solutions are not factorable in general. To circumvent this, the methods detailed in this article repeatedly used the key idea that global information can be computed for a non-factorable function if that function is fully specified as the solution of a system of factorable equations. Using this theme, bounds and relaxations were computed for the solutions of implicit functions, and these methods were then extended to DAEs using some key theorems relating global properties of the DAEs to global properties of the solutions. In particular, these extensions led to computational methods for computing interval bounds, as well as convex and concave relaxations for the parametric solutions of semi-explicit index-one DAEs. Interestingly, these methods were also shown to be capable of computationally verifying the existence and uniqueness of a DAE solution within the computed bounds. Finally, we have highlighted the importance of reachability analysis in the context of global dynamic optimization, and argued that reachable set enclosure methods are a key enabling tool for the application of powerful global optimization algorithms to dynamic problems. In particular, we have illustrated the use of reachable set enclosures described in terms of convex and concave relaxations to extend the spatial branch-and-bound framework to dynamic optimization problems, leading a guaranteed global optimization algorithm for problems with semi-explicit DAE models embedded.

## References

1. Adjiman, C.S., Dallwig, S., Floudas, C.A., Neumaier, A.: A global optimization method,  $\alpha$ BB, for general twice-differentiable constrained NLPs - I. Theoretical advances. *Comput. Chem. Eng.* **22**(9), 1137–1158 (1998)
2. Adjiman, C.S., Androulakis, I.P., Floudas, C.A.: A global optimization method,  $\alpha$ BB, for general twice-differentiable constrained NLPs - II. Implementation and computational results. *Comput. Chem. Eng.* **22**(9), 1159–1179 (1998)
3. Althoff, M., Stursberg, O., Buss, M.: Reachability analysis of linear systems with uncertain parameters and inputs. In: *Proceedings of 46th IEEE Conference on Decision and Control*, pp. 726–732 (2007)
4. Althoff, M., Stursberg, O., Buss, M.: Reachability analysis of nonlinear systems with uncertain parameters using conservative linearization. In: *Proceedings of 47th IEEE Conference on Decision and Control*, pp. 4042–4048 (2008)

5. Androulakis, I.P., Maranas, C.D., Floudas, C.A.:  $\alpha$ BB: a global optimization method for general constrained nonconvex problems. *J. Glob. Optim.* **7**, 337–363 (1995)
6. Ascher, U.M., Petzold, L.R.: *Computer Methods for Ordinary Differential Equations and Differential-Algebraic Equations*. SIAM, Philadelphia (1998)
7. Bellman, R.: *Dynamic Programming*. Princeton University Press, New Jersey (1957)
8. Benyahia, B., Lakerveld, R., Barton, P.I.: A plant-wide dynamic model of a continuous pharmaceutical process. *Ind. Eng. Chem. Res.* **51**(47), 15393–15412 (2012)
9. Berz, M., Makino, K.: Verified integration of ODEs and flows using differential algebraic methods on high-order Taylor models. *Reliab. Comput.* **4**, 361–369 (1998)
10. Bhatia, T., Biegler, L.: Dynamic optimization in the design and scheduling of multiproduct batch plants. *Ind. Eng. Chem. Res.* **35**, 2234–2246 (1996)
11. Biegler, L.T., Zavala, V.M.: Large-scale nonlinear programming using IPOPT: an integrating framework for enterprise-wide dynamic optimization. *Comput. Chem. Eng.* **33**(3), 575–582 (2009)
12. Chachuat, B., Mitsos, A., Barton, P.I.: Optimal start-up of microfabricated power generation processes employing fuel cells. *Optim. Control Appl. Methods* **31**(5), 471–495 (2010)
13. Chernousko, F.L.: Ellipsoidal state estimation for dynamical systems. *Nonlinear Anal.* **63**, 872–879 (2005)
14. Chisci, L., Garulli, A., Zappa, G.: Recursive state bounding by parallelotopes. *Automatica* **32**(7), 1049–1055 (1996)
15. Cizniar, M., Podmajersky, M., Hirmajer, T., Fikar, M., Latifi, A.M.: Global optimization for parameter estimation of differential-algebraic systems. *Chem. Pap.* **63**(3), 274–283 (2009)
16. Cross, E.A., Mitchell, I.M.: Level set methods for computing reachable sets of systems with differential algebraic equation dynamics. In: *Proceedings of 2008 American Control Conference*, pp. 2260–2265 (2008)
17. Cuthrell, J.E., Biegler, L.T.: On the optimization of differential-algebraic process systems. *AIChE J.* **33**(8), 1257–1270 (1987)
18. Du, K.S., Kearfott, R.: The cluster problem in multivariate global optimization. *J. Glob. Optim.* **5**(3), 253–265 (1994)
19. Dunnebie, G., Fricke, J., Klatt, K.U.: Optimal design and operation of simulated moving bed chromatographic reactors. *Ind. Eng. Chem. Res.* **39**(7), 2290–2304 (2000)
20. Esposito, W.R., Floudas, C.A.: Deterministic global optimization in nonlinear optimal control problems. *J. Glob. Optim.* **17**, 97–126 (2000)
21. Esposito, W.R., Floudas, C.A.: Global optimization for the parameter estimation of differential-algebraic systems. *Ind. Eng. Chem. Res.* **39**, 1291–1310 (2000)
22. Facchinei, F., Pang, J.S.: *Finite-Dimensional Variational Inequalities and Complementarity Problems*, vol. 1. Springer, New York (2003)
23. Falk, J.E., Soland, R.M.: An algorithm for separable nonconvex programming problems. *Manag. Sci.* **15**(9), 550–569 (1969)
24. Feehery, W., Tolsma, J., Barton, P.: Efficient sensitivity analysis of large-scale differential-algebraic systems. *Appl. Numer. Math.* **25**(1), 41–54 (1997)
25. Flores-Tlacuahuac, A., Biegler, L.T., Saldívar-Guerra, E.: Optimal grade transitions in the high-impact polystyrene polymerization process. *Ind. Eng. Chem. Res.* **45**(18), 6175–6189 (2006)
26. Harrison, G.W.: Dynamic models with uncertain parameters. In: Avula, X. (ed.) *Proceedings of the First International Conference on Mathematical Modeling*, vol. 1, pp. 295–304 (1977)
27. Hindmarsh, A.C., Brown, P.N., Grant, K.E., Lee, S.L., Serban, R., Shumaker, D.E., Woodward, C.S.: SUNDIALS, suite of nonlinear and differential/algebraic equation solvers. *ACM Trans. Math. Softw.* **31**, 363–396 (2005)
28. Hoefkens, J., Berz, M., Makino, K.: Computing validated solutions of implicit differential equations. *Adv. Comput. Math.* **19**, 231–253 (2003)
29. Horst, R., Tuy, H.: *Global Optimization: Deterministic Approaches*, 3rd edn. Springer, New York (1996)
30. Houska, B., Chachuat, B.: Branch-and-lift algorithm for deterministic global optimization in nonlinear optimal control. *J. Optim. Theor. Appl.* (2014). doi:[10.1007/s10957-013-0426-1](https://doi.org/10.1007/s10957-013-0426-1)

31. Houska, B., Villanueva, M., Chachuat, B.: A validated integration algorithm for non-linear ODEs using Taylor models and ellipsoidal calculus. In: Proceedings of 2013 IEEE 52nd Annual Conference on Decision and Control, pp. 484–489 (2013). doi:[10.1109/CDC.2013.6759928](https://doi.org/10.1109/CDC.2013.6759928)
32. Kearfott, R.: Rigorous Global Search: Continuous Problems. Kluwer Academic Publishers, Dordrecht (1996)
33. Kesavan, P., Lee, J.H.: A set based approach to detection and isolation of faults in multivariable systems. *Comput. Chem. Eng.* **25**, 925–940 (2001)
34. Ko, D., Siriwardane, R., Biegler, L.T.: Optimization of pressure-swing adsorption process using zeolite 13X for CO<sub>2</sub> sequestration. *Ind. Eng. Chem. Res.* **42**(2), 339–348 (2003)
35. Kremling, A., Heermann, R., Centler, F., Jung, K., Gilles, E.: Analysis of two-component signal transduction by mathematical modeling using the KdpD/KdpE system of *Escherichia coli*. *Biosystems* **78**(1–3), 23–37 (2004)
36. Kunkel, P., Mehrmann, V.: Differential-Algebraic Equations: Analysis and Numerical Solution. European Mathematical Society, Zurich (2006)
37. Kurzhanski, A.B., Varaiya, P.: Ellipsoidal techniques for reachability analysis. In: Hybrid Systems: Computation and Control. Lecture Notes in Computer Science, vol. 1790, Springer, Berlin, pp. 202–214 (2000)
38. Le, V.T.H., Stoica, C., Dumur, D., Alamo, T., Camacho, E.F.: Robust tube-based constrained predictive control via zonotopic set-membership estimation. In: Proceedings of 50th IEEE Conference on Decision and Control, pp. 4580–4585 (2011)
39. Limon, D., Bravo, J.M., Alamo, T., Camacho, E.F.: Robust MPC of constrained nonlinear systems based on interval arithmetic. *IEEE Proc. Control Theory Appl.* **152**(3), 325–332 (2005)
40. Lin, Y., Stadtherr, M.A.: Deterministic global optimization for parameter estimation of dynamic systems. *Ind. Eng. Chem. Res.* **45**, 8438–8448 (2006)
41. Lin, Y., Stadtherr, M.A.: Deterministic global optimization of nonlinear dynamic systems. *AIChE J.* **53**(4), 866–875 (2007)
42. Lin, Y., Stadtherr, M.A.: Validated solutions of initial value problems for parametric ODEs. *Appl. Numer. Math.* **57**, 1145–1162 (2007)
43. Lin, Y., Stadtherr, M.A.: Fault detection in nonlinear continuous-time systems with uncertain parameters. *AIChE J.* **54**(9), 2335–2345 (2008)
44. Luksan, L., Vleek, J.: Algorithm 811: NDA: algorithms for nondifferentiable optimization. *ACM Trans. Math. Softw.* **27**(2), 193–213 (2001)
45. Luus, R., Dittrich, J., Keil, F.J.: Multiplicity of solutions in the optimization of a bifunctional catalyst blend in a tubular reactor. *Can. J. Chem. Eng.* **70**, 780–785 (1992)
46. Lygeros, J., Tomlin, C., Sastry, S.: Controllers for reachability specifications for hybrid systems. *Automatica* **35**, 349–370 (1999)
47. Ma, D.L., Chung, S.H., Braatz, R.D.: Worst-case performance analysis of optimal batch control trajectories. *AIChE J.* **45**(7), 1496–1476 (1999)
48. Makela, M.M.: Survey of bundle methods for nonsmooth optimization. *Optim. Methods Softw.* **17**(1), 1–29 (2002)
49. Maly, T., Petzold, L.R.: Numerical methods and software for sensitivity analysis of differential-algebraic systems. *Appl. Numer. Math.* **20**, 57–79 (1996)
50. McCormick, G.P.: Computability of global solutions to factorable nonconvex programs: Part I - convex underestimating problems. *Math. Program.* **10**, 147–175 (1976)
51. Mitchell, I., Bayen, A.M., Tomlin, C.: A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games. *IEEE Trans. Autom. Control* **50**(7), 947–957 (2005)
52. Mitsos, A., Bollas, G.M., Barton, P.I.: Bilevel optimization formulation for parameter estimation in liquid-liquid phase equilibrium problems. *Chem. Eng. Sci.* **64**(3), 548–559 (2009)
53. Mitsos, A., Chachuat, B., Barton, P.I.: McCormick-based relaxations of algorithms. *SIAM J. Optim.* **20**(2), 573–601 (2009)

54. Moisan, M., Bernard, O., Gouze, J.L.: Near optimal interval observers bundle for uncertain bioreactors. *Automatica* **45**(1), 291–295 (2009)
55. Moles, C.G., Mendes, P., Banga, J.R.: Parameter estimation in biochemical pathways: a comparison of global optimization methods. *Genome Res.* **13**(11), 2467–2474 (2003)
56. Moore, R.E.: *Methods and Applications of Interval Analysis*. SIAM, Philadelphia, PA (1979)
57. Nedialkov, N.S., Jackson, K.R., Corliss, G.F.: Validated solutions of initial value problems for ordinary differential equations. *Appl. Math. Comput.* **105**, 21–68 (1999)
58. Neher, M., Jackson, K.R., Nedialkov, N.S.: On Taylor model based integration of ODEs. *SIAM J. Numer. Anal.* **45**(1), 236–262 (2007)
59. Neumaier, A.: *Interval Methods for Systems of Equations*. Cambridge University Press, Cambridge (1990)
60. Neumaier, A.: Complete search in continuous global optimization. In: Iserles, A. (ed.) *Acta Numerica*. Cambridge University Press, Cambridge (2004)
61. Oishi, M., Mitchell, I., Tomlin, C., Saint-Pierre, P.: Computing viable sets and reachable sets to design feedback linearizing control laws under saturation. In: *Proceedings of 45th IEEE Conference on Decision and Control*, San Diego, CA, pp. 3801–3807 (2006)
62. Papamichail, I., Adjiman, C.S.: A rigorous global optimization algorithm for problems with ordinary differential equations. *J. Glob. Optim.* **24**(1), 1–33 (2002)
63. Raissi, T., Ramdani, N., Candau, Y.: Set membership state and parameter estimation for systems described by nonlinear differential equations. *Automatica* **40**, 1771–1777 (2004)
64. Rapaport, A., Dochain, D.: Interval observers for biochemical processes with uncertain kinetics and inputs. *Math. Biosci.* **193**, 235–253 (2005)
65. Rauh, A., Brill, M., Gunther, C.: A novel interval arithmetic approach for solving differential-algebraic equations with Valencia-IVP. *Int. J. Appl. Math. Comput. Sci.* **19**(3), 381–397 (2009)
66. Ryoo, H., Sahinidis, N.: Global optimization of nonconvex NLPs and MINLPs with application in process design. *Comput. Chem. Eng.* **19**(5), 551–566 (1995)
67. Ryoo, H., Sahinidis, N.: A branch-and-reduce approach to global optimization. *J. Glob. Optim.* **2**, 107–139 (1996)
68. Sahinidis, N., Tawarmalani, M.: Accelerating branch-and-bound through a modeling language construct for relaxation-specific constraints. *J. Glob. Optim.* **32**(2), 259–280 (2005)
69. Sahinidis, N., Tawarmalani, M.: A polyhedral branch-and-cut approach to global optimization. *Math. Program.* **130**(2), 225–249 (2005)
70. Sahlodin, A.M., Chachuat, B.: Convex/concave relaxations of parametric ODEs using Taylor models. *Comput. Chem. Eng.* **35**, 844–857 (2011)
71. Sahlodin, A.M., Chachuat, B.: Discretize-then-relax approach for convex/concave relaxations of the solutions of parametric ODEs. *Appl. Numer. Math.* **61**, 803–820 (2011)
72. Schaber, J., Liebermeister, W., Klipp, E.: Nested uncertainties in biochemical models. *IET Syst. Biol.* **3**(1), 1–9 (2009)
73. Schweppe, F.: Recursive state estimation: unknown but bounded errors and system inputs. *IEEE Trans. Autom. Control* **13**(1), 22–28 (1968)
74. Scott, J.K.: *Reachability analysis and deterministic global optimization of differential-algebraic systems*. Ph.D. thesis, Massachusetts Institute of Technology (2012)
75. Scott, J.K., Barton, P.I.: Convex enclosures for the reachable sets of nonlinear parametric ordinary differential equations. In: *Proceedings of 49th IEEE Conference on Decision and Control*, Atlanta, GA, pp. 5695–5700 (2010)
76. Scott, J.K., Barton, P.I.: Tight, efficient bounds on the solutions of chemical kinetics models. *Comput. Chem. Eng.* **34**, 717–731 (2010)
77. Scott, J.K., Barton, P.I.: Convex relaxations for nonconvex optimal control problems. In: *Proceedings of 50th IEEE Conference on Decision and Control*, Orlando, FL, pp. 1042–1047 (2011)
78. Scott, J.K., Barton, P.I.: Interval bounds on the solutions of semi-explicit index-one DAEs. Part 1: Analysis. *Numer. Math.* **125**(1), 1–25 (2011)
79. Scott, J.K., Barton, P.I.: Interval bounds on the solutions of semi-explicit index-one DAEs. Part 2: Computation. *Numer. Math.* **125**(1), 27–60 (2011)

80. Scott, J.K., Barton, P.I.: Bounds on the reachable sets of nonlinear control systems. *Automatica* **49**, 93–100 (2013)
81. Scott, J.K., Barton, P.I.: Convex and concave relaxations for the parametric solutions of semi-explicit index-one DAEs. *J. Optim. Theory Appl.* **156**(3), 617–649 (2013)
82. Scott, J.K., Barton, P.I.: Improved relaxations for the parametric solutions of odes using differential inequalities. *J. Glob. Optim.* **57**(1), 143–176 (2013)
83. Scott, J.K., Stuber, M.D., Barton, P.I.: Generalized McCormick relaxations. *J. Glob. Optim.* **51**, 569–606 (2011)
84. Scott, J.K., Chachuat, B., Barton, P.I.: Nonlinear convex and concave relaxations for the solutions of parametric ODEs. *Optim. Control Appl. Methods* **34**(2), 145–163 (2013)
85. Selot, A., Kuok, L.K., Robinson, M., Mason, T., Barton, P.I.: A short-term operational planning model for natural gas production systems. *AIChE J.* **54**(2), 495–515 (2007)
86. Singer, A.B., Barton, P.I.: Global solution of optimization problems with parameter-embedded linear dynamic systems. *J. Optim. Theory Appl.* **121**, 613–646 (2004)
87. Singer, A.B., Barton, P.I.: Bounding the solutions of parameter dependent nonlinear ordinary differential equations. *SIAM J. Sci. Comput.* **27**, 2167–2182 (2006)
88. Singer, A.B., Barton, P.I.: Global dynamic optimization for parameter estimation in chemical kinetics. *J. Phys. Chem. A* **110**(3), 971–976 (2006)
89. Singer, A.B., Barton, P.I.: Global optimization with nonlinear ordinary differential equations. *J. Glob. Optim.* **34**, 159–190 (2006)
90. Srinivasan, B., Palanki, S., Bonvin, D.: Dynamic optimization of batch processes - I. Characterization of the nominal solution. *Comput. Chem. Eng.* **27**(1), 1–26 (2003)
91. Stuber, M.D., Scott, J.K., Barton, P.I.: Convex and concave relaxations of implicit functions. *Optim. Methods Softw.* **30**(3), 424–460 (2015)
92. Tawarmalani, M., Sahinidis, N.V.: *Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming*. Kluwer Academic Publishers, Dordrecht (2002)
93. Teo, K.L., Goh, G., Wong, K.: *A Unified Computational Approach to Optimal Control Problems*. Wiley, New York (1991)
94. Tsang, T., Himmelblau, D., Edgar, T.: Optimal control via collocation and nonlinear programming. *Int. J. Control* **21**, 763–768 (1975)
95. Wechsung, A., Schaber, S.D., Barton, P.I.: The cluster problem revisited. *J. Glob. Optim.* **58**(3), 429–438 (2014)

Surveys in Differential-Algebraic Equations III

Ilchmann, A.; Reis, T. (Eds.)

2015, IX, 313 p. 42 illus., 21 illus. in color., Softcover

ISBN: 978-3-319-22427-5