Global Optimisation for Dynamic Systems using Interval Analysis

Carlos Pérez-Galván & I. David L. Bogle*

carlos.galvan.12@ucl.ac.uk
d.bogle@ucl.ac.uk
**corresponding author*

Centre for Process Systems Engineering, Department of Chemical Engineering, University College London. London WC1E 7JE, United Kingdom

Abstract

Engineers seek optimal solutions when designing dynamic systems but a crucial element is to ensure bounded performance over time. Finding a globally optimal bounded trajectory requires the solution of the ordinary differential equation (ODE) systems in a verified way. To date these methods are only able to address low dimensional problems and for larger systems are unable to prevent gross overestimation of the bounds. In this paper we show how interval contractors can be used to obtain tightly bounded optima. A verified solver constructs tight upper and lower bounds on the dynamic variables using contractors for initial value problems (IVP) for ODEs within a global optimisation method. The solver provides guaranteed bound on the objective function and on the first order sensitivity equations in a branch and bound framework. The method is compared with three previously published methods on three examples from process engineering.

Keywords: Global optimisation; Dynamic systems; Verified integration; Interval analysis; Interval contractors.

1 Introduction

In chemical engineering dynamic processes arise in many applications and often an optimal trajectory is sought. For example we need the control path for optimal resource consumption in changes of product grade on polymer plants, estimating dynamic states for process control as well as in model building applications (Esposito and Floudas, 2000b; Singer et al., 2006), design of batch systems in chemical engineering, including dynamic rectors and a system with a reactor, heat exchanger, separator and distillation column (Bhatia and Biegler, 1996), simultaneous dynamic optimisation for the calculation of optimal transient trajectories for a polymerisation process (Flores-Tlacuahuac et al., 2005), dynamic optimisation of distillation columns with particular focus on equilibrium constraints (Raghunathan et al., 2004), computation of optimal time trajectories for the control of the different flows in a simulated moving-bed process (Kloppenburg and Gilles, 1999), and a logic-based solution approach applied to a multistage batch distillation process of a ternary mixture (Oldenburg et al., 2003).

There are many problems that require guaranteed bounded performance along the whole trajectory often because of safety critical and environmental limits. In these applications, it is not admissible to allow a certain variable to go beyond some prescribed limits. For example, the content of a certain compound in a stream is not allowed to be present in a higher concentration than the level permitted by the environmental regulator who might otherwise oblige the plant to shut down. The engineer has to make sure that this concentration is within the admissible limits at all times, however, he must do it without compromising too much the cost of the plant operation and the qualities of all products. Safety critical plants require that the variables of interest, for example temperature or pressure of plant equipment, be within limits for the whole time of operation.

Obtaining the optimal performance is not an easy task since dynamic models in chemical engineering often exhibit non-convexities due to the combination of nonlinear terms, and thus, multiple local minima arise in the model. Moreover, to guarantee we are within limits we are required to rigorously make sure we are including all possible solutions. In this sense, we want our computations to rigorously determine the optimal solution by obtaining mathematically verified upper and lower bounds on the global optima.

This can be achieved if we obtain bounds on the dynamic variables that are mathematically verified to be within a safe operating range. In this context, the term "mathematically verified" means that we are able to include all the solutions within upper and lower bounds. For example, interval amounts can represent ranges of values from a lower to an upper endpoint and they include all the values within a range because the lower and the upper endpoint are rounded downwards and upwards, respectively.

Sequential and simultaneous approaches have been used for the solution of the deterministic global optimisation problem for dynamic systems. In the first, the dynamic system is integrated and in turn the objective function and gradients are evaluated. In the second, the dynamic system is converted to a set of algebraic equations by using collocation-based discretisations leaving a fully algebraic nonlinear programming (NLP) problem. Stochastic methods have also been used although they are not considered in this paper because they are not able to provide a guarantee that the global optima have been found.

Several chemical engineering researchers have devoted their efforts to solve the guaranteed global optimisation problem for dynamic systems. They have used complete search methods such as branch and bound frameworks to make sure no solution is left out and to focus on finding bounds for the dynamic variables. Therefore, their algorithms rigorously find all global optima within bounds or are at least able to provide a theoretical guarantee that the global optima have been found. A branch and bound framework was used with the αBB method (Adjiman et al., 1996) in a sequential approach and applied to four different optimal control problems including the optimal control of batch and semi-batch processes (Esposito and Floudas, 2000b), and to parameter estimation problems to determine reaction kinetic constants from time series data (Esposito and Floudas, 2000a). In principle the method of (Esposito and Floudas, 2000a,b) provides a guaranteed global optimum. The rigorous underestimators needed are hard to obtain and here only sampling approaches were proposed. Another sequential approach implementing a branch and bound framework was developed with a convex underestimating procedure (Papamichail and Adjiman, 2002) which they applied to parameter estimation, chemical kinetics and modelling and optimal control problems. Some years later, again using a sequential approach and a branch and bound framework, Papamichail and Adjiman (2004) used McCormick relaxations and constant and affine bounds in the solution of parameter estimation and optimal control problems. The approach used by Papamichail and Adjiman (2002, 2004) is computationally expensive in constructing tight affine underestimators and overestimators. Singer and Barton (2006) presented a sequential approach using another branch and bound framework for problems with an integral objective function. This algorithm implements differential inequalities and McCormick relaxations to construct the convex/concave relaxations and the method was applied to parameter estimation and optimal control problems. Rauh et al. (2006) presented a global optimisation method for discrete-time and continuous-time systems applied to a mechanical positioning system. In the continuous-time system their algorithm uses a prototype implementation of an interval extension of Taylor series as a verified solver. The guaranteed solution has also been considered for mixed-integer dynamic optimisation. Chachuat et al. (2006) presents a review on these methods focusing on systems with embedded ODEs and branch and bound frameworks. They stress out the importance of tight state relaxations and the use of heuristics in the global optimisation algorithm. The dynamic optimisation problem has also been addressed using Taylor Models in a sequential approach and a branch and bound framework with focus on the tightness of the ODEs state bounds. This method uses Taylor Models method with an interval remainder term and was applied to several parameter estimation problems (Lin and Stadtherr, 2006b). Later, they applied the same method but this time with a branch and reduce approach using a domain reduction technique Lin and Stadtherr (2007a) and the applications were an optimal control and a final time formulation of the oil shale pyrolysis problems. The later method was extended to account for inequality path constraints in a rigorous way (Zhao and Stadtherr, 2011) and was applied to three semi-batch reactor problems.

Only a few research groups have worked on this problem and they have mainly made use of a branch and bound framework in a sequential approach. The available methods for solving dynamic systems to global optimality are only able to address low dimensional problems. According to Table 1 of the order of 2-7 state variables and 1 to 8 decision variables.

In these methods one of the main challenges to overcome is on the construction of tight and efficient bounds for the dynamic constraints. This problem arises because there is overestimation present in the construction of the bounds which causes the bounds to be overconservative (yielding to poor objective function and gradient evaluation) or in the worst case they tend to $\pm\infty$ and the algorithm has to be stopped.

Bounds on the dynamic variables overestimate because usually some form of overapproximation (such as intervals or relaxations) is used to guarantee that all the solutions are included. However, the set or relaxation used is merely a representation of the true solution set and when operations with overestimated values are propagated (in an integration algorithm for example) the result can be extremely overconservative.

Table 1: Number of state variables and system parameters in global optimisation for dynamic systems

Reference	**Maximum number of state variables**	**Maximum number of decision variables**
Esposito and Floudas (2000b)	7	1
Esposito and Floudas (2000a)	3	5
Papamichail and Adjiman (2002)	2	3
Papamichail and Adjiman (2004)	2	3
Singer and Barton (2006)	4	3
Lin and Stadtherr (2006b)	2	4
Lin and Stadtherr (2007a)	5	5
Zhao and Stadtherr (2011)	4	8

In this work, we propose to use interval contracting methods in a verified integration method in order to obtain tight and efficient bounds for global optimisation for dynamic systems applications.

The paper is organised as follows, in Section 2 the problem formulation of the global optimisation and the dynamic system are presented. Section 3 describes the verified integration method used in this work. Section 4 presents the sources of overestimation in interval arithmetic and ways to tackle them are presented. Numerical case studies on verified integration are reported on Section 5. The description of the global optimisation solution methodology is given in Section 6 and numerical case studies from global optimisation are presented in Section 7. Finally, conclusive remarks are presented in Section 8.

2 Problem formulation

We assume that the system can be described by an ODE model $\dot{\mathbf{y}}(t) = \mathbf{f}(t, \mathbf{y}(t), \boldsymbol{\theta})$, with $\mathbf{y}$, the vector of state variables and $\boldsymbol{\theta}$, the vector of system parameters. In this article the bold type notation is adopted to indicate vector-valued quantities and square brackets are used for interval-valued quantities unless otherwise specified. The lower and upper endpoints of an interval $[x]$ are specified by $[\underline{x}, \overline{x}]$, the width or diameter of an interval is given by $w([x]) = \overline{x} - \underline{x}$ and the midpoint by $\hat{x} = m([x]) = (\overline{x} - \underline{x})/2$. A dynamic optimisation problem involving a dynamic model can be formulated as

$$\begin{aligned} \min_{\boldsymbol{\theta}} \; & \phi(\mathbf{y}(t_i, \boldsymbol{\theta}),\, \boldsymbol{\theta};\; i = 1, \ldots, ns) \\ s.t. \;\; & \dot{\mathbf{y}} = \mathbf{f}(t, \mathbf{y}(t), \boldsymbol{\theta}) \\ \mathbf{y}(t_0, \boldsymbol{\theta}) & = \mathbf{y}_0(\boldsymbol{\theta}) \\ & t \in [t_0, t_f] \\ & \boldsymbol{\theta} \in [\boldsymbol{\theta}] \end{aligned} \tag{1}$$

where ϕ is the objective function, $[\boldsymbol{\theta}] = [\overline{\boldsymbol{\theta}}, \underline{\boldsymbol{\theta}}]$ are the decision variables, an interval vector where $\overline{\boldsymbol{\theta}}$ and $\underline{\boldsymbol{\theta}}$ are lower and upper endpoints, respectively.

The dynamic simulation problem we are aiming to solve within the optimisation problem which is as follows:

$$\dot{\mathbf{y}}(t) = \mathbf{f}(t, \mathbf{y}(t), \boldsymbol{\theta}),\; \mathbf{y}(t_0, \boldsymbol{\theta}) = \mathbf{y}_0(\boldsymbol{\theta}),\; \mathbf{y}_0(\boldsymbol{\theta}) \in [\mathbf{y}_0],\; \boldsymbol{\theta} \in [\boldsymbol{\theta}] \tag{2}$$

where $t \in [t_0, t_f]$, $\boldsymbol{\theta}$ represent the time-invariant parameters with $[\boldsymbol{\theta}] = [\underline{\boldsymbol{\theta}}, \overline{\boldsymbol{\theta}}]$, $\mathbf{y}$ represent the vector of state variables, $\mathbf{y}_0$ are the initial conditions at time t_0 with $[\mathbf{y_0}] = [\underline{\mathbf{y}_0}, \overline{\mathbf{y}_0}]$. Here, $\mathbf{f}$ is assumed to be $(k-1)$ times continuously differentiable with respect to the state variables. The parameters of the simulation problem correspond to the decision variables of the optimisation problem.

A solution of (2) from the initial condition $\mathbf{y}_j$ at t_j and parameter values $\boldsymbol{\theta}$ is denoted by $\mathbf{y}(t; t_j, \mathbf{y}_j, \boldsymbol{\theta})$. The solution set of (2) with initial conditions $[\mathbf{y}_j]$ at t_j and parameter values $[\boldsymbol{\theta}]$ is given by $\mathbf{y}(t; t_j, [\mathbf{y}_j], [\boldsymbol{\theta}]) = \{\mathbf{y}(t; t_j, \mathbf{y}_j, \boldsymbol{\theta}) \mid \mathbf{y}_j \in [\mathbf{y}_j], \boldsymbol{\theta} \in [\boldsymbol{\theta}]\}$. The objective of this method is to compute an enclosure

$$[\mathbf{y}_j] \supseteq \mathbf{y}(t_j; t_0, [\mathbf{y}_0], [\boldsymbol{\theta}]) = \{\mathbf{y}(t_j; t_0, \mathbf{y}_0, \boldsymbol{\theta}) \mid \mathbf{y}_0 \in [\mathbf{y}_0], \boldsymbol{\theta} \in [\boldsymbol{\theta}]\} \tag{3}$$

at $t_j \in [t_0, t_f]$, $j = 1, \ldots N_s$. In the next section a method based on interval analysis to compute the enclosure above is described. As we will see the main problem with these methods is addressing the overestimation generated in the integration.

This work considers a sequential approach in which the dynamic constraints need to be integrated in order to evaluate the objective function and gradients. When a sequential approach is used a verified ODE method integrates the dynamic part of the optimisation problem leaving a problem only constrained by the system parameters $\boldsymbol{\theta}$. In the global optimisation algorithm (Table 2) two calls to the verified integration method are done per iteration. The tightness of the bounds from the verified integration method directly affect the global optimisation as we evaluate the objective function and gradients with these. Hence, it is desirable to obtain the tightest bounds possible.

In the next section the verified integration method used in the global optimisation algorithm is described.

Table 2: Global optimisation algorithm

```
GOalgorithm(In: $\phi^*_{local}$, $[\boldsymbol{\theta}]_0$, $\varepsilon$; Out: $[\phi*]$, $C$, $L$)
Initialise: $[\boldsymbol{\theta}] = [\boldsymbol{\theta}]_0$, $\phi_{ub} = \phi^*_{local}$ or $\phi_{ub} = \phi(\hat{\boldsymbol{\theta}})$, $L = \varnothing$, $C = \varnothing$
while $L \neq \varnothing$
    Bisect $[\boldsymbol{\theta}]$ such that $w([\boldsymbol{\theta}]_i) = w([\boldsymbol{\theta}]) = max_{1 \leq i \leq n} w([\boldsymbol{\theta}]_i)$ : $[\boldsymbol{\theta}] = [\boldsymbol{\theta}]^{(1)} \cup [\boldsymbol{\theta}]^{(2)}$
    Call ITSContractor($[\mathbf{y}_0]$,$[\boldsymbol{\theta}]^{(1)}$) to evaluate $\phi([\boldsymbol{\theta}]^{(1)})$ and $\frac{\partial \phi}{\partial \theta}([\boldsymbol{\theta}]^{(1)})$
    Call ITSContractor($[\mathbf{y}_0]$,$[\boldsymbol{\theta}]^{(2)}$) to evaluate $\phi([\boldsymbol{\theta}]^{(2)})$ and $\frac{\partial \phi}{\partial \theta}([\boldsymbol{\theta}]^{(2)})$
    $\phi_{ub} = \min\{\phi_{ub}, \overline{\phi}(\hat{\boldsymbol{\theta}}^{(1)}), \overline{\phi}(\hat{\boldsymbol{\theta}}^{(2)})\}$
    if $\max\{\overline{\phi}([\boldsymbol{\theta}]^{(1)}), \overline{\phi}([\boldsymbol{\theta}]^{(2)})\} - \min\{\underline{\phi}([\boldsymbol{\theta}]^{(1)}), \underline{\phi}([\boldsymbol{\theta}]^{(2)})\} < \varepsilon$ then
        if $0 \notin \frac{\partial \phi}{\partial \theta}([\boldsymbol{\theta}]^{(1)})$ then Discard $[\boldsymbol{\theta}]^{(1)}$
        else if $0 \notin \frac{\partial \phi}{\partial \theta}([\boldsymbol{\theta}]^{(2)})$ then Discard $[\boldsymbol{\theta}]^{(2)}$
        end if
        Place $[\boldsymbol{\theta}]^{(1)}$ and $[\boldsymbol{\theta}]^{(2)}$ into $C$ in order
        if $L \neq \varnothing$ then
            Remove the first item from $L$ and place its box into $[\boldsymbol{\theta}]$
            if $\underline{\phi}([\boldsymbol{\theta}]) > \phi_{ub}$ then
                return with $\underline{\phi}$ equal to lower bound on the first box in $C$
                and with the lists $C$ and $L$
            end if
        else
            return with $\underline{\phi}$ equal to lower bound on the first box in $C$
            and with list $C$
        end if
    else
        if $0 \notin \frac{\partial \phi}{\partial \theta}([\boldsymbol{\theta}]^{(1)})$ then Discard $[\boldsymbol{\theta}]^{(1)}$
        else if $0 \notin \frac{\partial \phi}{\partial \theta}([\boldsymbol{\theta}]^{(2)})$ then Discard $[\boldsymbol{\theta}]^{(2)}$
        end if
        Enter the items $([\boldsymbol{\theta}]^{(1)}, \underline{\phi}([\boldsymbol{\theta}]^{(1)})$ and $([\boldsymbol{\theta}]^{(2)}, \underline{\phi}([\boldsymbol{\theta}]^{(2)})$ in proper order in list $L$
        Set $[\boldsymbol{\theta}]$ as the argument of the first item in $L$ (with lowest $\underline{\phi}([\boldsymbol{\theta}])$) and remove
        the item $([\boldsymbol{\theta}], \underline{\phi}([\boldsymbol{\theta}])$ from the list
    end if
end while
```

3 Verified integration of dynamic systems

Dynamic simulation represents a key step in the deterministic solution for dynamic optimisation problems. When the objective is to rigorously obtain the best solution of the dynamic optimisation problem, a global optimisation technique is required as well as the ability to manage the round off and truncation errors in the integration stage which can be done using a verified integration method for the solution of the dynamic system.

The solution of the initial value problems (IVPs) for ODEs plays a key role in the rigorous deterministic solution of dynamic optimisation problems and there are methods for this purpose which are mostly based on interval analysis. A number of researchers have developed several methods for the verified solution of IVPs for ODEs. One of the first attempts to find a verified solution for these systems was proposed by (Moore, 1962). His idea consists of the use of the Picard-Lindelhof iteration in the system of ODEs and the Taylor expansion of the same. In his method he provides a constant enclosure step to determine an a priori enclosure (coarse enclosure) of the solution.

Several other modifications to Moore's method have been made by other researchers including Eijgenraam (1981). The purpose of these modifications were to reduce the wrapping effect (Section 4.1) so Eijgenraam proposed several transformations. The modifications proposed by Lohner (1992) also had an important impact as he devised a method involving a QR factorization of the matrix product responsible for the wrapping effect. This method is still one of the best alternatives to tackle this problem. Nedialkov et al. (1999) provide an excellent review about these methods. Another Lohner-type of method that computes bounds on the partial derivatives with respect to the initial conditions was developed by Wilczak and Zgliczynski (2011). They employed the CAPD library (CAPD, 2013).

Other alternatives to bound the ODE states with upper and lower bounds have been proposed, for example the Interval Hermite-Obreschkoff method has been developed by Nedialkov (2011) and Walawska and Wilczak (2016) have used a similar method to compute bounds on the first order variational equations of ODEs, the Taylor Models approach by Lin and Stadtherr (2006a) and by Makino and Berz (2003), the differential inequalities approach by Scott and Barton (2013) and the validated enclosure of an approximate solution by Rauh et al. (2009). McCormick relaxations (convex and concave relaxations) have been used in the remainder term of the interval Taylor series (ITS) method (Sahlodin and Chachuat, 2011b). These relaxations have also been used in the Taylor Models method in the remainder term (Sahlodin and Chachuat, 2011a). A kind of Taylor model with ellipsoidal remainder term has also been devised (Houska et al., 2013). Fazal and Neumaier (2013) computed state bounds using conditional differential inequalities. The method requires global optimisation subproblems that can make the method verified if a rigorous global optimisation solver is used.

However, there is still need for a method able to obtain tighter and more efficient bounds since solving solving dynamic optimisation problems of practical size remains an issue. Considering all the previous methods, this work makes use of an ITS method because of its implementation simplicity. This method is prone to be modified and take new implementations in an easier manner especially in the development stage and when proposing new methods. The interval Taylor series has already been subject to some changes recently. It is the basis of the method of the popular software package VNODE (Nedialkov, 2011) which also used the interval Hermite-Obreschkoff approach. McCormick relaxations have also been used in the method in Sahlodin and Chachuat (2011b). In this paper we propose to implement interval contractors (Krawczyk and Newton/Gauss-Seidel) at each iteration of ITS method so as to reduce the overestimation generated. The next section describes the traditional ITS method.

3.1 Interval Taylor series

In this section an interval Taylor series (ITS) method has been used for obtaining bounds on the dynamic variables of the ODE models. The bounding method used in this paper consists of two stages, the approach is similar the one of the VNODE software package (Nedialkov, 2011) except that the parametric dependency is being explicitly accounted for. The first stage is the validation of existence and uniqueness of a solution in which also a suitable a priori enclosure and a time step are obtained. The second stage involves the computation of a tighter enclosure which consists on the use of a high order Taylor series to refine the solution obtained in the first stage.

3.1.1 First stage. Validation of existence and uniqueness

In the first stage the validation of existence and uniqueness is carried out and an appropriate time step h_j as well as an a priori enclosure $[\tilde{\mathbf{y}}_j] \supseteq \mathbf{y}(t; t_j, [\mathbf{y}_j], [\boldsymbol{\theta}])$, for all $t \in [t_j, t_{j+1}]$ with $t_{j+1} = t_j + h_j$ are obtained making use of the High Order Enclosure (HOE) approach (Nedialkov et al., 2001). According to this approach h_j and $[\tilde{\mathbf{y}}_j]$ must satisfy the following equation:

$$[\tilde{\mathbf{y}}_j] = [\mathbf{y}_j] + \sum_{i=1}^{k-1} [0, h_j]^i \mathbf{f}^{[i]}([\mathbf{y}_j], [\boldsymbol{\theta}]) + [0, h_j]^k \mathbf{f}^{[k]}([\tilde{\mathbf{y}}_j^0], [\boldsymbol{\theta}]) \subseteq [\tilde{\mathbf{y}}_j^0] \tag{4}$$

where k is the order of the Taylor series expansion, $[\mathbf{y}_j]$ is the vector of tight enclosures of the solutions with ranges in $[\tilde{\mathbf{y}}_j^0]$, $[\boldsymbol{\theta}]$ is the vector of system parameters and $\mathbf{f}^{[i]}$ are the Taylor coefficients defined according to

$$\begin{aligned} \mathbf{f}^{[0]}(\mathbf{y}, \boldsymbol{\theta}) &= \mathbf{y}_j \\ \mathbf{f}^{[i]}(\mathbf{y}, \boldsymbol{\theta}) &= \frac{1}{i} \left(\frac{\partial \mathbf{f}^{[i-1]}}{\partial \mathbf{y}} \mathbf{f} \right) (\mathbf{y}, \boldsymbol{\theta}) \text{ for } i \geq 1 \end{aligned} \tag{5}$$

These Taylor coefficients can be calculated using automatic differentiation.

3.1.2 Second stage. Tightening of the solution

The second stage involves the computation of a tight enclosure $[\mathbf{y}_{j+1}] \supseteq \mathbf{y}(t_{j+1}; t_0, [\mathbf{y}_0], [\boldsymbol{\theta}])$ given interval bounds $[\mathbf{y}_j]$ at t_j. This stage refines the a priori enclosure $[\tilde{\mathbf{y}}_j]$ on $t \in [t_j, t_{j+1}]$

provided in the first stage. It satisfies the next equation

$$\begin{aligned}[\mathbf{y}_{j+1}] = &\overbrace{\hat{\mathbf{y}}_j + \sum_{i=1}^{k-1} h_j^i \mathbf{f}^{[i]}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}})}^{\mathbf{u}_{j+1}} + \overbrace{\left\{ \mathbf{I} + \sum_{i=1}^{k-1} h_j^i \frac{\partial \mathbf{f}^{[i]}}{\partial \mathbf{y}}([\mathbf{y}_j], [\boldsymbol{\theta}]) \right\}}^{[\mathbf{S}^y_{j+1}]}([\mathbf{y}_j] - \hat{\mathbf{y}}_j) \\ &+ \underbrace{\left\{ \sum_{i=0}^{k-1} h_j^i \frac{\partial \mathbf{f}^{[i]}}{\partial \boldsymbol{\theta}}([\mathbf{y}_j], [\boldsymbol{\theta}]) \right\}}_{[\mathbf{S}^\theta_{j+1}]}([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}}) + \underbrace{h_j^k \mathbf{f}^{[k]}([\tilde{\mathbf{y}}_j], [\boldsymbol{\theta}])}_{[\mathbf{z}_{j+1}]}\end{aligned} \tag{6}$$

where $\mathbf{I}$ is the identity matrix and $\hat{\mathbf{y}}_j = (\underline{[\mathbf{y}_j]} + \overline{[\mathbf{y}_j]})/2$. For the sake of simplicity, we rewrite equation (6) as

$$[\mathbf{y}_{j+1}] = \mathbf{u}_{j+1} + [\mathbf{S}^y_{j+1}]([\mathbf{y}_j] - \hat{\mathbf{y}}_j) + [\mathbf{S}^\theta_{j+1}]([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}}) + [\mathbf{z}_{j+1}] \tag{7}$$

In this method the interval matrix-vector product $[\mathbf{S}^y_{j+1}]([\mathbf{y}_j] - \hat{\mathbf{y}}_j)$ in equation (7) is known to be one of the main contributors of the wrapping effect (Moore, 1966). Because of this, a number of methods have been developed to try to avoid direct evaluations of this matrix-vector product (Nedialkov et al., 1999). In this article the QR factorization technique devised by Lohner (1992) is used in the interval Taylor series method. This technique or a similar variation is also used in Nedialkov (2011), Lin and Stadtherr (2006a), Sahlodin and Chachuat (2011b) and Sahlodin and Chachuat (2011a).

The QR factorization technique consists of the substitution of the term $[\mathbf{S}^y_{j+1}]([\mathbf{y}_j] - \hat{\mathbf{y}}_j)$ by another term containing a matrix that induces an orthogonal coordinate system that often provides a better enclosure than the original coordinate system.

$$[\mathbf{y}_{j+1}] = \mathbf{u}_{j+1} + ([\mathbf{S}^y_{j+1}]\mathbf{A}_j)[\boldsymbol{\Gamma}_j] + [\mathbf{S}^\theta_{j+1}]([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}}) + [\mathbf{z}_{j+1}] \tag{8}$$

where,

$$[\boldsymbol{\Gamma}_{j+1}] = \mathbf{A}^{-1}_{j+1}([\mathbf{z}_{j+1}] - \hat{\mathbf{z}}_{j+1}) + \mathbf{A}^{-1}_{j+1}([\mathbf{S}^y_{j+1}]\mathbf{A}_j)[\boldsymbol{\Gamma}_j] + (\mathbf{A}^{-1}_{j+1}[\mathbf{S}^\theta_{j+1}])([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}}) \tag{9}$$

Here, $[\boldsymbol{\Gamma}_0] = [\mathbf{y}_0] - \hat{\mathbf{y}}_0$, $\mathbf{A}_0 = \mathbf{I}$ and $\mathbf{A}_{j+1}$ is chosen as the orthogonal matrix in the QR factorization of $mid([\mathbf{S}^y_{j+1}]\mathbf{A}_j)$, the parallelepiped enclosure of $[\boldsymbol{\Gamma}_{j+1}]$.

As discussed before, overestimation is present in verified integration methods because intervals and relaxations provide only approximations of the true solution set. The ITS method on its own is not able to address significant widths in the system parameters or initial conditions. When intervals with big widths are used in the method overestimation is generated by the dependency, cancellation and wrapping effect problems. The excess of overestimation causes the bounds to be overconservative and in the worst case makes the bounds tend to $\pm\infty$ and consequently the integration stops. The next section describes the overestimation sources, some techniques to reduce it and an implementation of these techniques in a verified integrator.

4 Overestimation in verified simulation

Verified methods exist for the solution of IVPs for ODEs which often rely on interval analysis (Moore, 1962). These methods provide upper and lower bounds in which the true solution of the problem is guaranteed to be contained. A major challenge in these methods is the reduction of the overestimation generated in the integration process which is mainly caused by the dependency and wrapping effect problems.

To tackle this problem some approaches have been proposed such as the ITS with Hermite-Obreschkoff approach by Nedialkov (2011), the ITS with convex/concave remainder term using McCormick relaxations (Sahlodin and Chachuat, 2011b), the differential inequalities with interval analysis approach (Scott and Barton, 2013), the validated enclosure of an approximate solution for ODEs and DAEs by (Rauh et al., 2009) and the Taylor Models approach with interval remainder term (Lin and Stadtherr, 2006a), with convex/concave remainder term using McCormick relaxations (Sahlodin and Chachuat, 2011a) and with ellipsoidal remainder term (Houska et al., 2013).

However, there is still a need for effective ways to reduce the overestimation in order to address problems with more state variables and bigger uncertainties. Being able to address bigger uncertain values means that in the dynamic optimisation problem we are able to provide bounds for the objective function and gradients of larger regions of the decision space which leads to speed-up of the algorithm as less calls to the verified integration algorithm are required. Section 4.1 presents the sources of overestimation in interval analysis which directly affect verified ODE integration. Interval contractors able to reduce the overestimation in nonlinear functions are presented in Section 4.2 as an option for tackling this problem. Finally, Section 4.3 describes a algorithm implementing interval contractors in an ITS method to enhance its overestimation reduction capabilities.

4.1 Reduction of the overestimation via interval contractors

In verified simulation there are three main sources of overestimation namely, dependence, cancellation and wrapping.

4.1.1 Dependence

The dependency problem arises in interval analysis because the method does not account for the dependency of multiple repetitions of the variables in a mathematical model. In other words, a variable repeated twice in a model is treated as two separate variables because in the definition of the interval multiplication rule the left and right factors vary independently. For example, $[x]^2 = [-1,1]^2 = [0,1]$ with $[x] = [-1,1]$ and $[x] \times [x] = [-1,1] \times [-1,1] = [-1,1]$ and hence $[x]^2 \neq [x] \times [x]$ in interval arithmetic. In order to manage this problem ways of reducing to a minimum the number of repetitions of the same variable are sought.

4.1.2 Cancellation

When the mathematical expressions contain at least one addition or subtraction some overestimation is generated. Contrary to floating point arithmetic the widths in interval arithmetic are additive instead of cancelling.

4.1.3 Wrapping

According to Lohner (2001) it is the undesirable overestimation of a solution set of an iteration or recurrence which occurs if this solution set is replaced by a superset of some 'simpler' structure (e.g. an interval) and this superset is then used to compute enclosures for the next step which may eventually lead to an exponential growth of the overestimation. More generally, according to Neumaier (2003) wrapping is the overestimation due to the depth of a computational graph describing the ODE system, caused by long sequences of nested operations depending on a limited number of variables only, which also magnifies bounds on rounding errors and hence can give wide meaningless results even for problems with exact data.

4.2 Fixed-point interval contractors

When a real valued function is evaluated using interval arithmetic usually some overestimation is present in the range of the function. Interval contractors offer the possibility to contract the estimated range in an interval evaluated function. Some of these contractors, such as the Krawczyk contractor and the Newton contractor (Neumaier, 1990; Jaulin et al., 2001), are able to contract nonlinear functions. Consider the case in which we have n_y variables linked by n_f relations of the form

$$f_q(y_1, y_2, ..., y_n) = 0,\ q \in 1, 2, ..., n \tag{10}$$

Each variable y_j belongs to a domain Y_j, an interval. Equation (10) can be written in a vector form and a constraint satisfaction problem (CSP) can be formulated as in equation (11).

$$(\mathbf{f}(\mathbf{y}) = 0,\ \mathbf{y} \in [\mathbf{y}_j]) \tag{11}$$

The solution set B of (11) is defined as

$$B = \{\mathbf{y} \in [\mathbf{y}_j] | \mathbf{f}(\mathbf{y}) = 0\} \tag{12}$$

Contracting the CSP in (11) means replacing $[\mathbf{y}_j]$ by a smaller domain $[\mathbf{y}'_j]$ such that the solution set remains unchanged, i.e. $B \subset [\mathbf{y}'] \subset [\mathbf{y}]$. The contractors considered in this work are interval counterparts of classical point algorithms such as Gauss-Seidel and Newton algorithms. For more details see (Jaulin et al., 2001) and for application examples in deterministic global optimisation refer to (Balendra and Bogle, 2009).

As nonlinear problems are being considered contractors for nonlinear functions were used in this work as opposed to contractors for linear functions (Gauss elimination, Gauss-Seidel, Linear Programming). The Krawczyk and Newton/Gauss-Seidel nonlinear contractors were implemented in the method as described in Section 4.3. This method was then used in case studies to assess the effectiveness in the reduction of the overestimation.

4.2.1 Interval Krawczyk contractor

The Krawczyk contractor (see Table 3) considers a CSP as in (11) where the number of variables is the same as the number of relations and $\mathbf{f}$ is assumed to be differentiable.

Table 3: Krawczyk contractor algorithm

```
KContractor(In: $[\mathbf{y}_j]$, $[\mathbf{S}^y_{j+1}]$, $\mathbf{S}^y_{j+1}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}})$, $\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])$; Out: $[\mathbf{y}_j]$)
begin
    $\hat{\mathbf{y}}_j = m([\mathbf{y}_j])$
    $\mathbf{M} = \mathbf{S}^y_{j+1}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}})$
    $[\mathbf{J}_\psi] = \mathbf{I} - \mathbf{M}[\mathbf{S}^y_{j+1}]$
    $[\mathbf{r}] = \hat{\mathbf{y}}_j - \mathbf{M}\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) + [\mathbf{J}_\psi]([\mathbf{y}_j] - \hat{\mathbf{y}}_j)$
    $[\mathbf{y}_j] \leftarrow [\mathbf{y}_j] \cap [\mathbf{r}]$
end
```

The function

$$\boldsymbol{\psi}(\mathbf{y}) = \mathbf{y} - \mathbf{M}\mathbf{f}(\mathbf{y}) \tag{13}$$

is a fixed-point subsolver for (11) where $\mathbf{M}$ is any invertible matrix. Its centred inclusion function is

$$\boldsymbol{\Psi}([\mathbf{y}_j]) = \boldsymbol{\psi}(\hat{\mathbf{y}}_j) + \{\mathbf{J}_\psi([\mathbf{y}_j])\}([\mathbf{y}_j] - \hat{\mathbf{y}}_j) \tag{14}$$

where $\mathbf{J}_{\boldsymbol{\psi}}$ is an inclusion function for the Jacobian matrix of $\boldsymbol{\psi}$ with $\hat{\mathbf{y}}_j$ as the midpoint of $[\mathbf{y}_j]$. The intersection between the original domain and the one obtained with (14) results in the fixed-point contractor as in equation (15).

$$[\mathbf{y}_j] \leftarrow [\mathbf{y}_j] \cap \boldsymbol{\psi}(\hat{\mathbf{y}}_j) + \{\mathbf{I} - \mathbf{M}\mathbf{J}_f([\mathbf{y}_j])\}([\mathbf{y}_j] - \hat{\mathbf{y}}_j) \tag{15}$$

4.2.2 Interval Newton/Gauss-Seidel contractor

We consider again a CSP as in (11) and apply the mean value theorem to obtain

$$(\mathbf{f}(\hat{\mathbf{y}}_j) + \mathbf{J}_f(\boldsymbol{\xi})(\mathbf{y}_j - \hat{\mathbf{y}}_j) = 0,\ \mathbf{y}_j \in [\mathbf{y}_j],\ \boldsymbol{\xi} \in [\mathbf{y}_j]) \tag{16}$$

The CSP in (16) can be arranged as

$$\begin{pmatrix} \mathbf{A}\mathbf{p} + \mathbf{f}(\hat{\mathbf{y}}_j) = 0 \\ \mathbf{p} = (\mathbf{y}_j - \hat{\mathbf{y}}_j) \\ \mathbf{A} = \mathbf{J}_f(\boldsymbol{\xi}) \\ \mathbf{b} = -\mathbf{f}(\hat{\mathbf{y}}_j) \\ \mathbf{y}_j \in [\mathbf{y}_j],\ \boldsymbol{\xi} \in [\mathbf{y}_j] \end{pmatrix} \tag{17}$$

In this way, a linear contractor can be used such as the Gauss-Seidel contractor (see Table 4). The Gauss-Seidel contractor is able to contract domains of linear systems of the form

$$\mathbf{A}\mathbf{p} - \mathbf{b} = 0 \tag{18}$$

Table 4: Gauss-Seidel Preconditioned contractor algorithm

```
GSContractor(In: $[\mathbf{A}]$, $[\mathbf{p}]$, $[\mathbf{b}]$; Out: $[\mathbf{A}]$, $[\mathbf{p}]$, $[\mathbf{b}]$)
begin
    $\mathbf{A}_0 = m([\mathbf{A}])$
    $[\mathbf{A}'] = \mathbf{A}_0[\mathbf{A}]$
    $[\mathbf{b}'] = \mathbf{b}_0[\mathbf{b}]$
    $diag([\mathbf{A}']) + extdiag([\mathbf{A}']) = [\mathbf{A}']$
    $[\mathbf{p}] \leftarrow [\mathbf{p}] \cap (diag([\mathbf{A}']))^{-1}([\mathbf{b}'] - extdiag([\mathbf{A}'])[\mathbf{p}])$
    $[\mathbf{b}] \leftarrow \mathbf{A}_0[\mathbf{b}'] \cap [\mathbf{b}]$
    $[\mathbf{A}] \leftarrow \mathbf{A}_0[\mathbf{A}'] \cap [\mathbf{A}]$
end
```

If A is square it can be decomposed as the sum of a diagonal matrix and a matrix with zeroes on its diagonal (extdiag):

$$diag(\mathbf{A})\mathbf{p} + extdiag(\mathbf{A})\mathbf{p} = \mathbf{b} \tag{19}$$

Also if $\mathbf{A}$ is invertible then (19) can be rewritten as

$$\mathbf{p} = (diag(\mathbf{A}))^{-1}(\mathbf{b} - extdiag(\mathbf{A})\mathbf{p}) \tag{20}$$

Hence, the solution of the Gauss-Seidel contractor is defined as the intersection of the original domain $\mathbf{p}$ and the new $\mathbf{p}$ calculated with (20). This results in:

$$\mathbf{p} \leftarrow \mathbf{p} \cap (diag(\mathbf{A}))^{-1}(\mathbf{b} - extdiag(\mathbf{A})\mathbf{p}) \tag{21}$$

Finally, the Gauss-Seidel contractor solution in (21) is used to update $[\mathbf{y}_j]$ and the intersection $[\mathbf{y}_j] \leftarrow [\mathbf{y}_j] \cap (\mathbf{p} + \hat{\mathbf{y}}_j)$ is obtained to finish with the Newton procedure (see Table 5).

4.3 Interval Taylor series with contractors

In order to apply the interval contractors in the verified method with the aim of reducing the overestimation this work considers an implicit form of equation (7) so as to formulate a CSP ($\mathbf{f}(\mathbf{y}) = 0$) at each time step. In a similar way if in other verified ODE solvers (e.g. based on Taylor models or differential inequalities) this formulation can be done then interval contractors can be applied as well. If we make this reformulation it is possible to consider the new implicit equation as a CSP problem in the form of (11). The formulation is the following

$$\mathbf{g}(\mathbf{y}) = \mathbf{u}_{j+1} + ([\mathbf{S}^{y}_{j+1}]\mathbf{A}_j)[\mathbf{\Gamma}_j] + [\mathbf{S}^{\theta}_{j+1}]([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}}) + [\mathbf{z}_{j+1}] - [\mathbf{y}_{j+1}] = 0,\ \mathbf{y} \in [\mathbf{y}_j]. \tag{22}$$

Table 5: Newton/Gauss-Seidel contractor algorithm

```
NGSContractor(In: $[\mathbf{y}_{j+1}]$, $[\mathbf{S}^y_{j+1}]$, $\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])$; Out: $[\mathbf{y}_{j+1}]$)
begin
    $\hat{\mathbf{y}}_j = m([\mathbf{y}_j])$
    $[\mathbf{A}] = [\mathbf{S}^y_{j+1}]$
    $[\mathbf{p}] = [\mathbf{y}_j] - \hat{\mathbf{y}}_j$
    $[\mathbf{p}] \supseteq [\mathbf{p}]_{GSP}$ =GSContractor($[\mathbf{A}]$, $[\mathbf{p}]$, $\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])$)
    $[\mathbf{y}_j] \leftarrow [\mathbf{y}_j] \cap [\mathbf{p}]$
end
```

However this reformulation does not yield a form as in (11) as the subtraction of identical vectors in interval arithmetic does not cancel but the widths are summed together. So a midpoint evaluation is performed and we get

$$\begin{aligned}\mathbf{g}(\hat{\mathbf{y}}_j) = \mathbf{u}_{j+1}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}}) + \{\mathbf{S}^y_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])\mathbf{A}_j\}\boldsymbol{\Gamma}_j(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) + \{\mathbf{S}^\theta_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])\}([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}})\\ + \mathbf{z}_{j+1}(\tilde{\mathbf{y}}_j, [\boldsymbol{\theta}]) - m(\mathbf{y}_{j+1}([\mathbf{y}_j], [\boldsymbol{\theta}])) = 0\end{aligned} \tag{23}$$

which has the form of (11). Now since

$$m(\mathbf{y}_{j+1}([\mathbf{y}_j], [\boldsymbol{\theta}])) = \mathbf{u}_{j+1}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}}) + m(\mathbf{z}_{j+1}([\tilde{\mathbf{y}}_j], [\boldsymbol{\theta}]))$$

we have

$$\begin{aligned}\mathbf{g}(\hat{\mathbf{y}}_j) &= \mathbf{u}_{j+1}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}}) + \{\mathbf{S}^y_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])\mathbf{A}_j\}\boldsymbol{\Gamma}_j(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) + \{\mathbf{S}^\theta_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])\}([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}})\\ &\quad + \mathbf{z}_{j+1}(\tilde{\mathbf{y}}_j, [\boldsymbol{\theta}]) - (\mathbf{u}_{j+1}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}}) + m(\mathbf{z}_{j+1}([\tilde{\mathbf{y}}_j], [\boldsymbol{\theta}]))) = 0\\ \\ &= \{\mathbf{S}^y_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])\mathbf{A}_j\}\boldsymbol{\Gamma}_j(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) + \{\mathbf{S}^\theta_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])\}([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}})\\ &\quad + \mathbf{z}_{j+1}(\tilde{\mathbf{y}}_j, [\boldsymbol{\theta}]) - m(\mathbf{z}_{j+1}([\tilde{\mathbf{y}}_j], [\boldsymbol{\theta}])) = 0.\end{aligned}$$

Also recalling equation (9)

$$[\boldsymbol{\Gamma}_{j+1}] = \mathbf{A}^{-1}_{j+1}([\mathbf{z}_{j+1}] - \hat{\mathbf{z}}_{j+1}) + \mathbf{A}^{-1}_{j+1}([\mathbf{S}^y_{j+1}]\mathbf{A}_j)[\boldsymbol{\Gamma}_j] + (\mathbf{A}^{-1}_{j+1}[\mathbf{S}^\theta_{j+1}])([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}})$$

we get that

$$\mathbf{g}(\hat{\mathbf{y}}_j) = \mathbf{A}_{j+1}\boldsymbol{\Gamma}_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) \tag{24}$$

which corresponds to the global error in the verified simulation (Lohner, 1992).

Now when there is no uncertainty then $\mathbf{g}(\hat{\mathbf{y}}_j) = \mathbf{A}_{j+1}\boldsymbol{\Gamma}_{j+1}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) = 0$. In the present work uncertainty has been considered in the initial conditions and parameters of the ODEs and it will be shown in the next section that when a number of iterations of the

Table 6: Interval Taylor series with fixed-point contractors algorithm

```
**Initialise**: $\mathbf{A}_0 = \mathbf{I}$, $[\boldsymbol{\Gamma}_0] = [\mathbf{y}_0] - \hat{\mathbf{y}}_0$

ITSContractor(In: $\mathbf{A}_j$, $[\boldsymbol{\Gamma}_j]$, $h_j$, $[\tilde{\mathbf{y}}_j]$, $[\mathbf{y}_j]$, $[\boldsymbol{\theta}]$; Out: $\mathbf{A}_{j+1}$, $[\boldsymbol{\Gamma}_{j+1}]$, $[\mathbf{y}_{j+1}]$)
begin
    $\mathbf{u}_{j+1} = \hat{\mathbf{y}}_j + \sum_{i=1}^{k-1} h_j^i \mathbf{f}^{[i]}(\hat{\mathbf{y}}_j, \hat{\boldsymbol{\theta}})$
    $[\mathbf{z}_{j+1}] = h_j^k \mathbf{f}^{[k]}([\tilde{\mathbf{y}}_j], [\boldsymbol{\theta}])$
    $[\mathbf{S}_{j+1}^y] = \mathbf{I} + \sum_{i=1}^{k-1} h_j^i \frac{\partial \mathbf{f}^{[i]}}{\partial \mathbf{y}}([\mathbf{y}_j], [\boldsymbol{\theta}])$
    $[\mathbf{S}_{j+1}^\theta] = \sum_{i=0}^{k-1} h_j^i \frac{\partial \mathbf{f}^{[i]}}{\partial \boldsymbol{\theta}}([\mathbf{y}_j], [\boldsymbol{\theta}])$
    $\mathbf{Q}_j \mathbf{R}_j = m([\mathbf{S}_{j+1}^y]\mathbf{A}_j)$
    $\mathbf{A}_{j+1} = \mathbf{Q}_j$
    $[\boldsymbol{\Gamma}_{j+1}] = \mathbf{A}_{j+1}^{-1}([\mathbf{z}_{j+1}] - \hat{\mathbf{z}}_{j+1}) + \mathbf{A}_{j+1}^{-1}([\mathbf{S}_{j+1}^y]\mathbf{A}_j)[\boldsymbol{\Gamma}_j] + (\mathbf{A}_{j+1}^{-1}[\mathbf{S}_{j+1}^\theta])([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}})$
    $[\mathbf{y}_{j+1}] = \mathbf{u}_{j+1} + ([\mathbf{S}_{j+1}^y]\mathbf{A}_j)[\boldsymbol{\Gamma}_j] + [\mathbf{S}_{j+1}^\theta]([\boldsymbol{\theta}] - \hat{\boldsymbol{\theta}}) + [\mathbf{z}_{j+1}]$
    *Newton/Gauss-Seidel contractor*:
    $\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}]) = \mathbf{A}_{j+1}[\boldsymbol{\Gamma}_{j+1}](\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])$
    $[\mathbf{y}_{j+1}] \supseteq [\mathbf{y}_{j+1}]_N$ =NGSContractor($[\mathbf{y}_{j+1}]$, $[\mathbf{S}_{j+1}^y]$, $\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])$)
    or
    $[\mathbf{y}_{j+1}] \supseteq [\mathbf{y}_{j+1}]_N$ =KContractor($[\mathbf{y}_{j+1}]$, $[\mathbf{S}_{j+1}^y]$, $\mathbf{g}(\hat{\mathbf{y}}_j, [\boldsymbol{\theta}])$)
    $[\mathbf{y}_{j+1}] \leftarrow [\mathbf{y}_{j+1}] \cap [\mathbf{y}_{j+1}]_N$
end
```

Krawczyk and Newton steps are used in equation (24) reduction of the overestimation is achieved.

In summary, since equation (24) is defined at each time step it is possible to implement the contractors as in an equation of the form of (11). In this way at each time step the interval Taylor series method obtains an interval ($[\mathbf{y}_{j+1}]$) that encloses the solution of the problem. The midpoint $\hat{\mathbf{y}}_{j+1}$ is then used to define equation (23). After a number of iterations if sufficient reduction is achieved the verified method obtains a new $[\mathbf{y}_{j+1}]$ and the contraction step is repeated otherwise the algorithm returns the best $[\mathbf{y}_{j+1}]$ found so far. The steps to obtain the guaranteed enclosures using contractors are illustrated in Table 6.

5 Testing the dynamic simulation method

Numerical experiments on test problems from chemical and biochemical processes were carried out. Seven ODE models were used in the experiments and in each model the interval contractors were applied alongside an interval Taylor series method to demonstrate the effectiveness of the methods. The test problems have different numbers of state variables and system parameters. Table 7 shows a summary of the test problems used which includes columns with the number of state variables, the number of system parameters and the importance in dynamic optimisation. Appendix 1 extends this information and includes the mathematical models, the values of the system parameters and initial conditions and specifies which ones have been used as uncertain values.

In this section results on the implementation of the Krawczyk and Newton/Gauss-Seidel (K and NGS, respectively) interval contractors in an interval Taylor series method are presented for the test problems in Table 7. For comparison purposes the simulations were also carried out using no contractors (NC method) where when necessary the uncertain amounts were subdivided, solved and then the bounds combined so as to provide a more fair comparison in terms of the time taken by the methods with contractors. The fourth column indicates the number of inital boxes used to simulate the case study. Finally, Taylor model bounds provided by the software package VSPODE version 1.4 (Lin and Stadtherr, 2007b) (VSPODE method) are also used for comparison purposes. This information is displayed in figures in the present section alongside with a Monte Carlo simulation (MC) that provides an approximation of the reachable set.

Tables 8 and 9 report the results obtained after using the four methods on the seven test problems. The results include the widths of the bounds and the CPU time in seconds for each of the problems and the methods. The widths given corresponds to a selected time t_s and the CPU time reported here is using an Intel^{R} CORE^{TM} i5 computer with 8Gb RAM, running Ubuntu. Depending on the problem studied a number of contractor iterations (or Contractors as specified in Tables 8 and 9) has been used to *contract* the width of the bounds.

Figures 1 to 7 show the trajectories in the NC, K, NGS and VSPODE methods with dot-dot-dashed blue, dot-dashed orange, dashed black and solid red lines, respectively. The MC simulation is shown as a shaded grey area.

The algorithm was implemented in C++ and the libraries FADBAD++ (Bendtsen and Stauning, 1996) and PROFIL/BIAS (Knuppel, 1994) were used for the automatic differentiation and interval operations, respectively. The Profil/Bias library was used since it is faster when only interval arithmetic operations and the square function are needed (Žilinskas, 2005) as in the examples considered.

Table 7: Test problems used, number of state variables and parameters and relevance in dynamic optimisation

Problem	**States**	**Parameters**	**Relevance**
1. First order irreversible series reaction	2	2	Parameter estimation for model development
2. First order reversible series reaction	2	4	Parameter estimation for model development
3. Exothermic batch reactor	2	8	Guaranteed safe operation
4. Two-state bioreactor	2	6	Parameter estimation for model development
5. Three-state bioreactor	3	8	Parameter estimation for model development
6. Reactor separator model	6	9	Optimal control
7. Glucagon receptor model	5	22	Guaranteed safe operation

In order to provide a fair comparison a number of subdivisions of the input uncertain intervals (box) was obtained and the solutions from each box were combined. In the first five case studies the most expensive method using the original input intervals (no subdivisions, 1 box) was determined and the simulations with the rest of the methods were carried out by subdividing the box into smaller boxes. The number of boxes was chosen so as to roughly take the same amount of time of the most expensive method using a single box. In the last two case studies, bounds were obtained up to the final time horizon regardless of the time taken. Therefore, in the first five case studies we compare the widths at final time or at selected time since the computational times are roughly the same. In the last two examples we compare the simulation times and the bound widths at final time.

Using a prespecified uncertainty in the initial conditions and system parameters the K, NGS and VSPODE methods managed to bound all of the case studies (Figures 1 to 7). The NC method provided bounds for all the case studies except the fifth one (Figure 5).

In the first five case studies it was observed that in the methods with contractors (K and NGS) provided tighter bounds than the NC method in the Exothermic batch reactor, the Two-state bioreactor and the Three-state bioreactor problems (See Figures 3 to 5 for problems 3, 4 and 5). According to Table 8 in the case studies 3, 4 and 5 the methods using contractors represent on average the 78, 36 and 17 % of the widths of the NC method. In the case studies 1 and 2 (Figures 1 and 2) the NC method bounds have widths that represent on average the 83 and 74 %, respectively, of the widths of the K and NGS methods. In these first five case studies VSPODE method constructs tighter bounds in four of them. In problems 1, 2, 4 and 5 the widths are on average the 73, 49, 16 and 26 % of the widths of the best method. Only case study three exhibits widths at final time that are on average 98 % of the size of the VSPODE method widths.

Table 8: Results on the implementation of interval contractors in the first five case studies

Method	**Contractors**	**CPU(s)**	**Boxes**	**Width at t_s**
1. First order irreversible series reaction ($t_s = 1$ day)				
NC	0	0.0058	4	$w(C_A(t_s))$=0.009225 $w(C_B(t_s))$=0.9190
K	1	0.0055	2	$w(C_A(t_s))$=0.01242 $w(C_B(t_s))$=1.0002108
NGS	1	0.0056	2	$w(C_A(t_s))$=0.01221 $w(C_B(t_s))$=0.9962
VSPODE	N/A	0.0053	1	$w(C_A(t_s))$=0.007070 $w(C_B(t_s))$=0.6317
2. First order reversible series reaction ($t_s = 0.5$ day)				
NC	0	0.027	2	$w(C_A(t_s))$=0.8611 $w(C_B(t_s))$=0.3028
K	1	0.029	1	$w(C_A(t_s))$=0.9458 $w(C_B(t_s))$=0.5412
NGS	1	0.028	1	$w(C_A(t_s))$=0.9342 $w(C_B(t_s))$=0.5391
VSPODE	N/A	0.029	1	$w(C_A(t_s))$=0.4317 $w(C_B(t_s))$=0.1435
3. Exothermic batch reactor ($t_s = 60$ s)				
NC	0	0.067	2	$w(x(t_s))$=0.07566 $w(T(t_s))$=75.2560
K	2	0.069	1	$w(x(t_s))$=0.06179 $w(T(t_s))$=57.3130
NGS	2	0.068	1	$w(x(t_s))$=0.06176 $w(T(t_s))$=57.1690
VSPODE	N/A	0.075	1	$w(x(t_s))$=0.06351 $w(T(t_s))$=57.6620
4. Two-state bioreactor ($t_s = 10$ day)				
NC	0	0.103	4	$w(X(t_s))$=0.9387 $w(S(t_s))$=2.1959
K	2	0.105	2	$w(X(t_s))$=0.3498 $w(S(t_s))$=0.7629
NGS	2	0.106	2	$w(X(t_s))$=0.3491 $w(S(t_s))$=0.7412
VSPODE	N/A	0.118	1	$w(X(t_s))$=0.03555 $w(S(t_s))$=0.1542
5. Three-state bioreactor ($t_s = 7.7$ day)				
NC	0	0.669	16	$w(x_1(t_s))$=4.4063 $w(x_2(t_s))$=4.1199
K	2	0.701	2	$w(x_1(t_s))$=0.7211 $w(x_2(t_s))$=0.7876
NGS	2	0.702	2	$w(x_1(t_s))$=0.7209 $w(x_2(t_s))$=0.7868
VSPODE	N/A	0.601	1	$w(x_1(t_s))$=0.1278 $w(x_2(t_s))$=0.2666

Table 9: Results on the implementation of interval contractors in the last two case studies

Method	**Contractors**	**CPU(s)**	**Boxes**	**Width at t_s**
6. Reactor separator model ($t_s = 100$ s)				
NC	0	6381.810	128	$w(x_1(t_s))$=0.000238 $w(x_6(t_s))$=0.00007
K	3	79.597	1	$w(x_1(t_s))$=0.000231 $w(x_6(t_s))$=0.000069
NGS	3	82.627	1	$w(x_1(t_s))$=0.000231 $w(x_6(t_s))$=0.000069
VSPODE	0	25.875	1	$w(x_1(t_s))$=0.000229 $w(x_6(t_s))$=0.000067
7. Glucagon receptor model ($t_s = 100$ s)				
NC	0	40.610	4	$w(R_s(t_s))$=24.5 $w(PLC_*(t_s))$=0.0001216
K	3	42.115	1	$w(R_s(t_s))$=52.5 $w(PLC_*(t_s))$=0.0002621
NGS	3	34.266	1	$w(R_s(t_s))$=51.1 $w(PLC_*(t_s))$=0.0002517
VSPODE	N/A	8.874	1	$w(R_s(t_s))$=3.4539 $w(PLC_*(t_s))$=0.000006

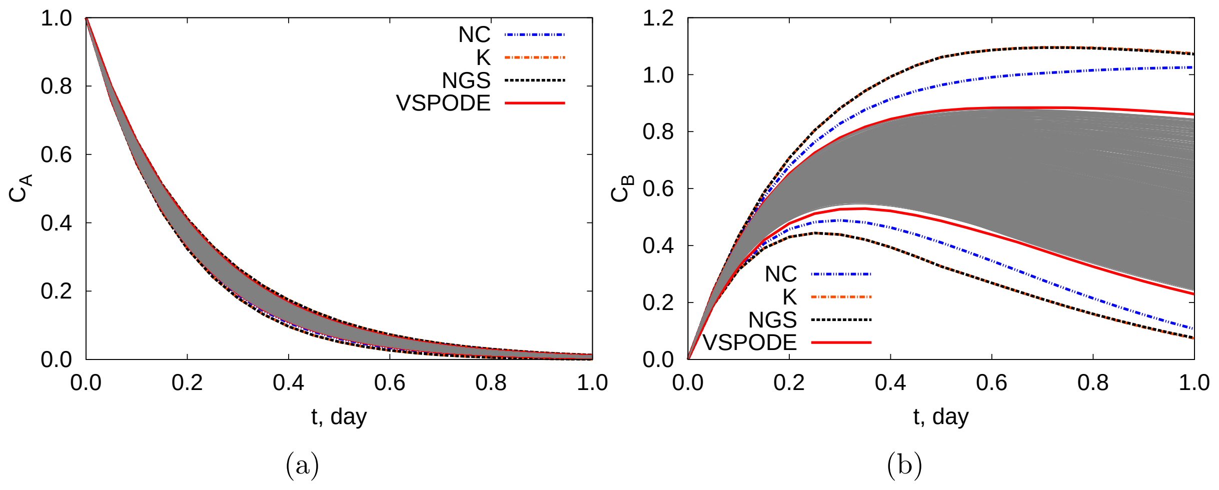
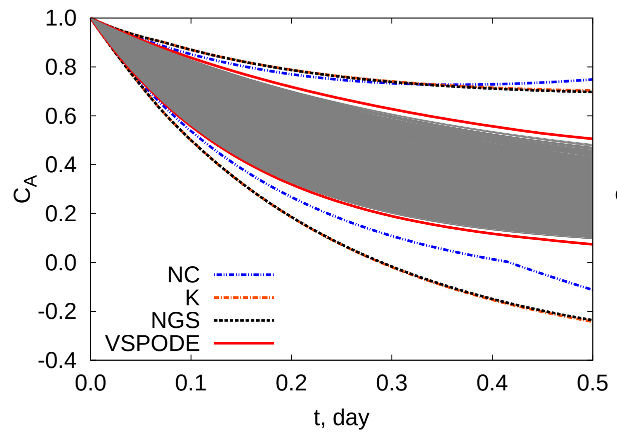
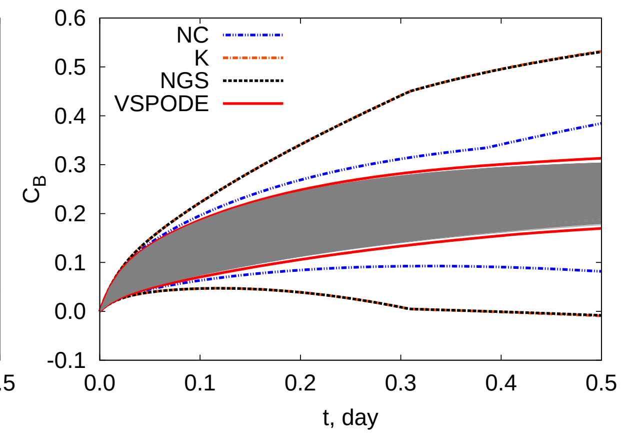


Figure 1: First order irreversible series reaction

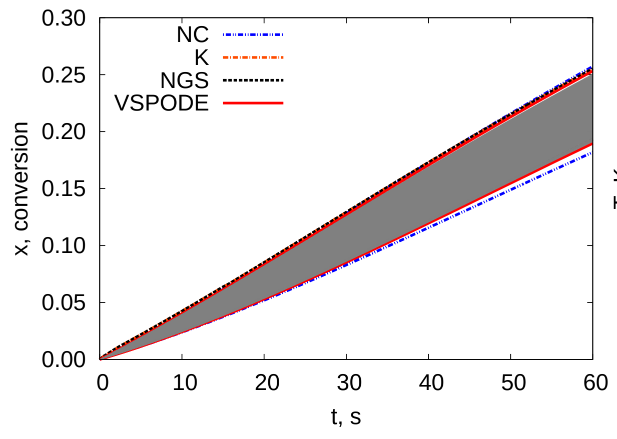


(a)

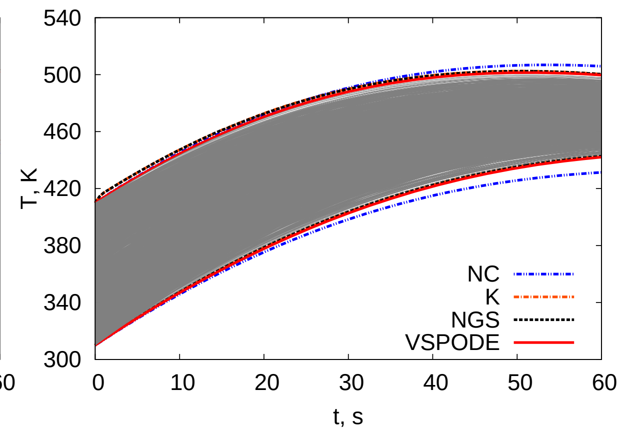


(b)

Figure 2: First order reversible series reaction

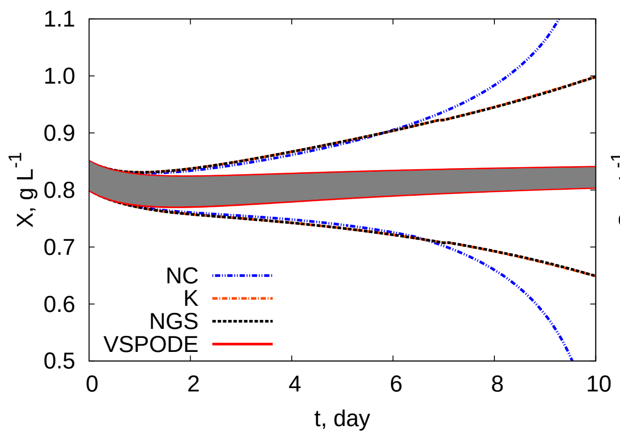


(a)

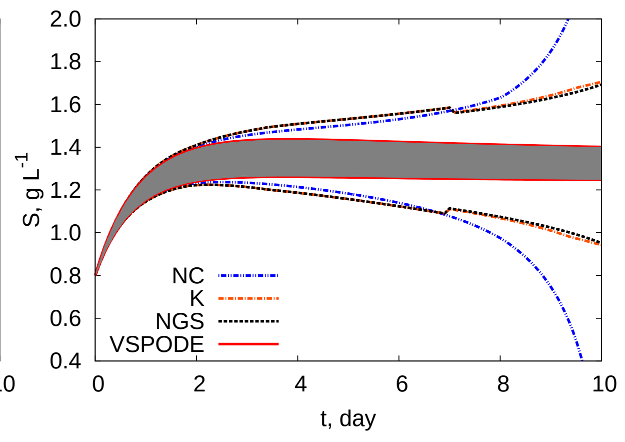


(b)

Figure 3: Exothermic batch reactor

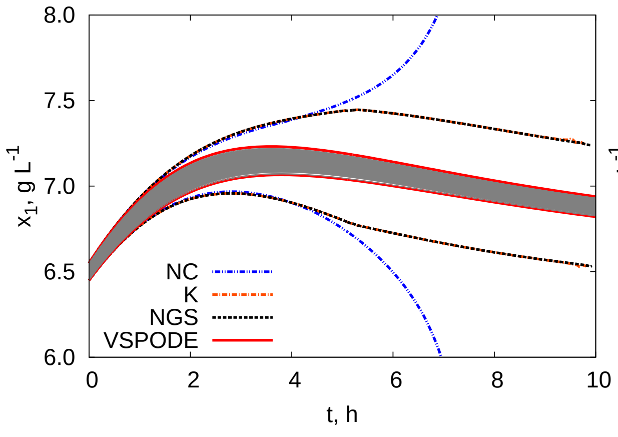


(a)

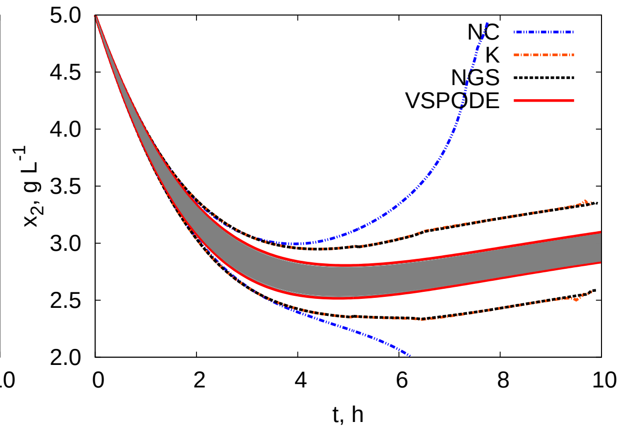


(b)

Figure 4: Two-state bioreactor

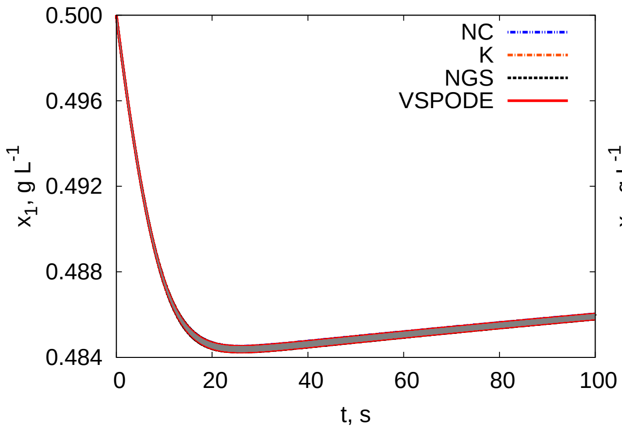


(a)

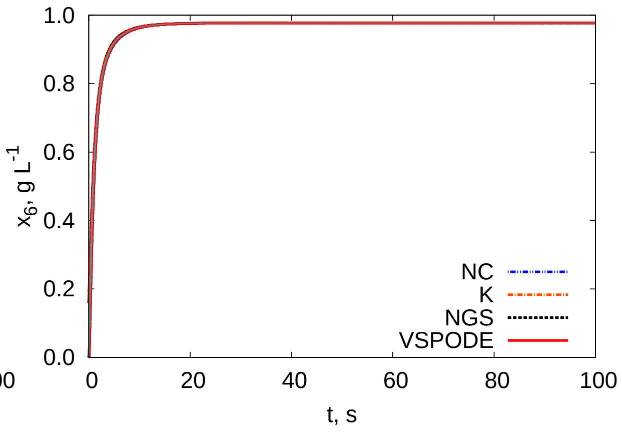


(b)

Figure 5: Three-state bioreactor

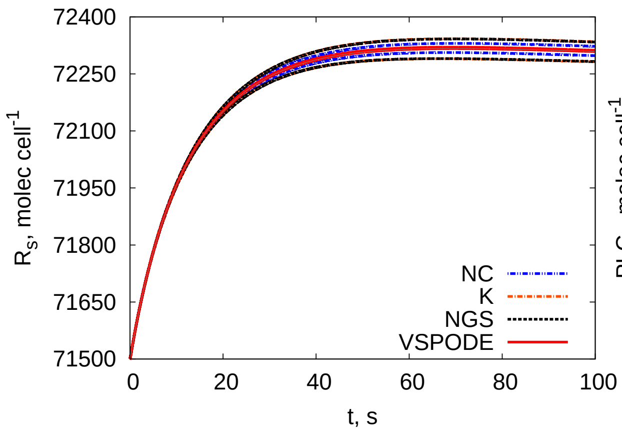


(a)

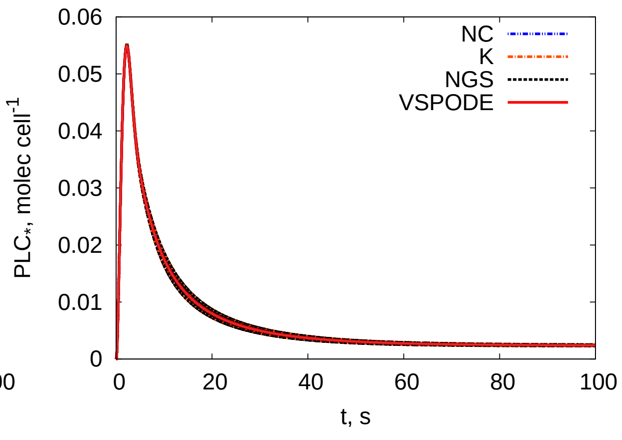


(b)

Figure 6: Reactor separator model



(a)



(b)

Figure 7: Glucagon receptor model

The last two case studies (Table 9), the Reactor separator and the Glucagon receptor models as shown in Figures 6 and 7 were also compared. In Table 9 all the methods in the Reactor separator model (Figure 6) have a similar width so the main comparison in this example comes in terms of time. The slowest method was the NC method requiring more than 6000 seconds and 128 interval boxes to complete the simulation with this final width. The K and NGS method took roughly 80 seconds each and the VSPODE method took 25.875 seconds representing 32.5 and 31.3 % of the methods with contractors, respectively.

The NC method provided bounds that were roughly half as wide as the bounds of the K and NGS methods and the tightest bounds were constructed by VSPODE with bounds that are the 15 % the width of the NC method. In terms of computational time, the slowest methods, the NC and K methods performed the simulation in about 40 seconds and the NGS and VSPODE methods in 34 and 9 seconds, respectively.

So far the paper has demonstrated that the addition of interval contractors to verified integration methods can be advantageous to reduce the overestimation generated in the integrated. Only in one case study (Exothermic batch reactor) the method obtained tighter bounds bounds and in less computational time than the VSPODE software package.

Now that we have the verified integration method we will use it as the bounding technique in a global optimisation algorithm. The next section explores the use of a verified integration method (ITS) with interval contractors in the context of dynamic optimisation. The contractor that was used was the one that has the best performance in terms of tightness and computational time: the Newton/Gauss-Seidel contractor.

6 Global optimisation for dynamic systems

The main objective of the present section is to describe the algorithm to find the global optima using the verified integration method with contractors in a sequential approach. As far as the authors know this is the first time an interval Taylor series method with a Newton/Gauss-Seidel contractor is used in a sequential approach to solve dynamic optimisation problems to global optimality. When a sequential approach is used a verified ODE method is applied to the dynamic part of the optimisation problem leaving a problem only constrained by the system parameters $\boldsymbol{\theta}$. The spatial search procedure used was similar to a standard branch and bound algorithm by Moore et al. (2009). The global optimisation method considers a problem of the form:

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \phi(\boldsymbol{\theta}) \\ &s.t.\, \boldsymbol{\theta} \in [\boldsymbol{\theta}] \end{aligned} \tag{25}$$

The global optimisation method that has been used in the numerical experiments is given in Table 2. In this algorithm each time a box is branched new bounds are needed and so the bounding routine (ITS method with contractors) is called for each box. This call represents the most expensive part in the branch and bound framework. Therefore it is needed to reduce the number of calls by discarding as many boxes as possible prior to the bounding step. A condition that tests whether or not zero is contained in the gradient of the objective function is being used in this algorithm in which we need to compute the first order sensitivity equations.

$$\frac{\partial}{\partial t}\left(\frac{\partial \mathbf{y}}{\partial \boldsymbol{\theta}}\right) = \frac{\partial \mathbf{f}}{\partial \mathbf{y}}\frac{\partial \mathbf{y}}{\partial \boldsymbol{\theta}} + \frac{\partial \mathbf{f}}{\partial \boldsymbol{\theta}} \tag{26}$$

The first order sensitivity equations are integrated in a verified way together with the ODE system.

7 Global optimisation results

The following examples were solved in a sequential approach and using the interval Taylor series with the Newton/Gauss-Seidel contractor to bound the dynamic variables. The CPU times are for the same computer as in Section 5. The programs were also written in C++ and the third party libraries FADBAD++ (Bendtsen and Stauning, 1996) and Profil/Bias (Knuppel, 1994) were used for the automatic differentiation and the interval arithmetic operations, respectively.

The main comparison criteria in this section are the CPU times. Two other works have been used to compare the present work. In order to provide a fair comparison the times reported by Lin and Stadtherr (2006b) and Lin and Stadtherr (2007a) have been adjusted by multiplying the time it takes VSPODE to perform a simulation of the ODE system times the number of iterations reported by each of the works. We believe this provides a better comparison since the simulations are carried out in the same machine. In order to adjust the times of Singer and Barton (2006) and Papamichail and Adjiman (2004) we use the relation used by Lin and Stadtherr (2006a) in which according to the SPEC benchmark the times of the computers used by Lin and Stadtherr (2006a), Lin and Stadtherr (2006b) and Lin and Stadtherr (2007a) correspond to roughly half of the time reported in Singer and Barton (2006) and to roughly 0.128 times the time reported in Papamichail and Adjiman (2004).

7.1 First order irreversible series reaction

A first-order irreversible chain reaction taken from Tjoa and Biegler (1991) considers the following reaction

$$A \xrightarrow{k_1} B \xrightarrow{k_2} C$$

The algorithm presented in Table 2 is applied to the parameter estimation problem with two parameters and two dynamic variables. The experimental data has been taken from Esposito and Floudas (2000a). The problem has been solved to global optimality using an absolute tolerance $\varepsilon_{abs} = 10^{-4}$ in 0.40 seconds. Lin and Stadtherr (2006b) solved the problem with a relative tolerance $\varepsilon_{rel} = 10^{-3}$ and exactly ($\varepsilon = 0$) in 0.023 and 0.059 seconds, respectively. The machine used was an Intel Pentium 4 3.2 GHz. Singer and Barton (2006) solved the problem with an absolute tolerance $\varepsilon_{abs} = 10^{-4}$ in 0.036 seconds in an AMD Athlon XP2000+ 1667 MHz. However, the differential inequalities approach is used in this work in which the auxiliary system is solved by a conventional solver and hence the solution is not computationally verified.

Table 10 shows these results and a column with the adjusted CPU time. In the case of the work by Singer and Barton (2006) the table shows two numbers in the columns CPU (s) and Iterations because they correspond to the cases without and with heuristics. Papamichail and Adjiman (2004) solved the problem in 9 and 11 seconds using constant, and constant and affine underestimation schemes, respectively. The tolerance they used was $\varepsilon_{rel} = 10^{-7}$ in an UltraSPARC-II 2×360 MHz.

$$\begin{aligned} \min_{\mathbf{k}} \phi &= \sum_{j=1}^{10} \sum_{i=1}^{2} (x_i(t_j) - x_i^{exp}(t_j)) \\ s.t.\ \dot{x}_1 &= -k_1 x_1 \\ \dot{x}_2 &= k_1 x_1 - k_2 x_2 \\ x_1(0) &= 1,\ x_2(0) = 0 \\ t &\in [0,1] \\ \mathbf{k} &\in [0,10] \times [0,10] \end{aligned} \tag{27}$$

In the first order irreversible series reaction case study the adjusted CPU time provided by the algorithm was much smaller than the Papamichail and Adjiman (2004) method, it was higher compared to the other two works and the number of iterations was the highest of all.

7.2 Singular control problem

The procedure from Section 6 is applied to a nonlinear singular control problem taken from Luus (1990)

$$\begin{aligned} \min_{u} \phi &= \int_{t_0}^{t_f} [x_1^2 + x_2^2 + 0.0005(x_2 + 16t - 8 - 0.1 x_3 u^2)^2] dt \\ s.t.\ \dot{x}_1 &= x_2 \\ \dot{x}_2 &= -x_3 u + 16t - 8 \\ \dot{x}_3 &= u \\ x_1(0) &= 0,\ \ x_2(0) = -1,\ \ x_3(0) = -\sqrt{5} \\ t &\in [0,1] \\ u &\in [-4,10] \end{aligned} \tag{28}$$

Table 11 shows the results of the singular control problem with a column containing the adjusted CPU time. As in the revious case study in the work by Singer and Barton (2006) the table shows two numbers in the columns CPU (s) and Iterations because they correspond to the cases without and with heuristics. The sequential approach was used to provide a solution for this problem.

The computational time taken for solving the problem was 1.42 seconds with an absolute tolerance of $\varepsilon_{abs} = 10^{-3}$. The problem was also solved by Lin and Stadtherr (2007a) in 0.02 seconds with the same absolute tolerance. It is worth mentioning that their global optimisation algorithm implements a branch and reduce approach and is able to reduce the search space. The machine used was an Intel Pentium 4 3.2 GHz. Also Singer and Barton (2006) provided a (non-verified) solution for the problem with the same tolerance in 2 and 1.8 seconds without and with heuristics. They used an AMD Athlon XP2000+ 1667 MHz to solve the problem. In the singular control problem the adjusted CPU time of the present algorithm resulted to be competitive as it was similar to the quadrature variable formulation of Singer and Barton (2006) but is was bigger than Lin and Stadtherr (2007a) and the original formulation in Singer and Barton (2006).

7.3 Oil shale pyrolysis

The optimal temperature profile in a plug flow reactor is considered. The reactions involved and the model of the problem are shown in (29). In the model only components A_1 and A_2 are included and the objective is to maximise the production of A_2. Here u is the adjustable parameter and is taken as a piecewise constant profile.

$$
\begin{aligned}
& & & \min_u \ \phi = -x_2(t_f) \\
& A_1 \xrightarrow{k_1} A_2 & & s.t.\ \dot{x}_1 = -k_1 x_1 - (k_3 + k_4 + k_5) x_1 x_2 \\
& A_2 \xrightarrow{k_2} A_3 & & \dot{x}_2 = k_1 x_1 - k_2 x_2 + k_3 x_1 x_2 \\
& A_1 + A_2 \xrightarrow{k_3} 2A_2 & & k_i = a_i e^{\left(\frac{-b_i/R}{698.15+50u}\right)},\ i = 1, \ldots, 5 \\
& A_1 + A_2 \xrightarrow{k_4} A_3 + A_2 & & x_1(0) = 1,\ \ x_2(0) = 0 \\
& A_1 + A_2 \xrightarrow{k_5} A_4 + A_2 & & t \in [0, 10] \\
& & & u \in [0, 1]
\end{aligned}
\tag{29}
$$

Table 12 reports the results of the oil shale pyrolysis example and it includes a column with the adjusted CPU time. Again the columns CPU (s) and Iteration in the work by Singer and Barton (2006) correspond to the cases without and with heuristics. The dynamic optimisation problem has been solved to global optimality using an absolute tolerance of $\varepsilon_{abs} = 10^{-3}$ in 5.22 seconds. The same problem was solved by Lin and Stadtherr (2006a) in 3.2 seconds using $\varepsilon_{abs} = 10^{-3}$ and an Intel Pentium 4 3.2 GHz. Singer and Barton (2006) solved the problem in a non-verified manner in 27.30 and 26.20 seconds without and with heuristics, respectively. The machine used was an AMD Athlon XP2000+ 1667 MHz. In the oil shale pyrolysis problem the adjusted CPU time of the present method was smaller than those of Singer and Barton (2006) with and without heuristics and very similar to Lin and Stadtherr (2007a).

8 Conclusions

The paper has presented a global optimisation algorithm for dynamic systems in which for the first time an interval Taylor series with Newton/Gauss-Seidel contractor is used to integrate the dynamic system and thus evaluate the objective function and constraints of the optimisation problem.

The global optimisation method performed reasonably well in two out of three case studies. In the first case study the number of iterations turned out to be the worst of the

Table 10: Global optimisation results. First order irreversible series reaction ($\varepsilon_{abs} = 1 \times 10^{-4}$)

Method	**Objective function**	**Optimiser**	**CPU (s)**	**CPU (s) adjusted**	**Iterations**
This work	1.185×10^{-6}	(5.0035, 1.0000)	0.40	0.40	75
Singer and Barton (2006)	1.22×10^{-6}	(5.0, 1.0)	0.036	0.017	-
Lin and Stadtherr (2006b)	1.185×10^{-6}	(5.0035, 1.0000)	0.023 & 0.059	0.022	4 & 2
Papamichail and Adjiman (2004)	1.185×10^{-6}	(5.0035, 1.0000)	9 & 11	1.10 & 1.35	1

Table 11: Global optimisation results. Singular control problem ($\varepsilon_{abs} = 1 \times 10^{-3}$)

Method	**Objective function**	**Optimiser**	**CPU (s)**	**CPU (s) adjusted**	**Iterations**
This work	0.4965	(4.071)	1.42	1.42	34
Singer and Barton (2006) (quadrature variable)	0.497	(4.07)	5.2 & 3.4	1.82 & 1.19	33 & 15
Singer and Barton (2006) (original)	0.497	(4.07)	2.0 & 1.8	0.7 & 0.63	21 & 15
Lin and Stadtherr (2007a)	0.4965	(4.071)	0.02	0.014	9

Table 12: Global optimisation results. Oil Shale Pyrolysis ($\varepsilon_{abs} = 1 \times 10^{-3}$)

Method	**Objective function**	**Optimiser**	**CPU (s)**	**CPU (s) adjusted**	**Iterations**
This work	-0.3479	(0.231)	5.22	5.22	39
Singer and Barton (2006)	-0.3480	(0.231)	27.3 & 26.2	19.67 & 18.87	115
Lin and Stadtherr (2007a)	-0.3479	(0.984)	3.2	4.61	21

compared works and the CPU time was only better than the method by Papamichail and Adjiman (2004). However, in the oil shale pyrolysis case study the number of iterations was similar and the CPU time was better than the differential inequalities (Singer and Barton, 2006) approach and slightly lower than the Taylor models approach (Lin and Stadtherr, 2007a). In the singular control problem the CPU time resulted better than the quadrature variable formulation without heuristics of Singer and Barton (2006) and slightly higher time when using heuristics. The CPU time of the present method was not better than the rest of the compared works and the number of iterations was similar to Singer and Barton (2006) but not better than Lin and Stadtherr (2007a).

The verified integration technique at the heart of the global optimisation algorithm consists of an interval Taylor series method with Krawczyk and Newton/Gauss-Seidel interval contractor. The method was compared with and without contractors in seven case studies leading to the conclusion that contractors help reduce the overestimation and are faster than the standalone method in most of the cases. In three of the first five case studies the widths of the method using contractors represented from 18 to 78 % of the widths of the method with no contractors.

In terms of the Krawczyk and Newton/Gauss-Seidel contractors. The Newton/Gauss-Seidel contractor performed better than the Krawczyk contractor as it consistently produced tighter widths at almost the same computational time. Furthermore in the glucagon receptor model it provided a much better time than the Krawczyk contractor.

The methods with contractors were also compared with the software package VSPODE. Such comparison resulted in that the method performs better in one case study (Exothermic batch reactor). However, in the rest of the case studies VSPODE showed that the widths of its bounds represent from 15 to 72 % of the width of the bounds constructed by the method with contractors.

While the method presented is not faster than the method that features Taylor Models (VSPODE) it provides the basis of an interval contracting technique that can be extended to other verified simulation methods (such as Taylor Models and convex/concave differential inequalities) and in turn other dynamic simulation methods.

The algorithm developed in this paper can be better than the differential inequalities approach and very similar to the Taylor models approach. It also offers the guarantee that the global optima have been bounded in a verified way. In future work an interesting idea would be to implement a constraint propagation contractor (Jaulin et al., 2001; Benhamou et al., 1999) instead of a fixed-point contractor and to apply the different contracting techniques to other kind of verified integrators. The global optimisation algorithm presented in this paper introduced an verified integration method with an interval Newton/Gauss-Seidel contractor which could potentially enhance the overestimation reduction capabilities of other methods, for example, global optimisation methods using the Taylor models or the convex/concave differential inequalities with interval contractors.

Acknowledgements

Financial support from CONACYT (Mexico), SEP DGRI (Mexico) and UCL is gratefully acknowledged. The authors thank Mark Stadtherr and Youdong Lin for kindly providing the software package VSPODE.

Appendix I

Table 13 extends the information about the case studies presented in the numerical experiments section.

Table 13: Mathematical models of the seven dynamic simulation case studies

Mathematical model	**Initial conditions and parameters**
First order irreversible series reaction $\dot{C}_A = -k_1 C_A$ $\dot{C}_B = k_1 C_A - k_2 C_B$	$C_A(0) = 1$ $C_B(0) = 0$ **Uncertain values** $k_1 = [4.5, 5.5]$ $k_2 = [0.2, 1.8]$
First order reversible series reaction $\dot{C}_A = -k_1 C_A + k_{-1} C_B$ $\dot{C}_B = k_1 C_A - (k_{-1} + k_2) C_B + k_{-2}(1 - C_A - C_B)$	$C_A(0) = 1$ $C_B(0) = 0$ $k_{-1} = 2$ $k_{-2} = 20$ **Uncertain values** $k_1 = [2, 6]$ $k_{-1} = [1, 3]$
Exothermic batch reactor $\dot{x} = k_0(1 - x)\mathrm{e}^{-\frac{E_a}{RT}}$ $\dot{T} = \frac{UA}{C_{A0} V C_p}(T_a - T) - \frac{\Delta H_R}{C_p} k_0 (1 - x)\mathrm{e}^{-\frac{E_a}{RT}}$	$x(0) = 0$ $k_0 = 0.022$ $V = 0.1$ $C_p = 60$ $E_a = 6000$ $R = 8.314$ $\Delta H_R = -140000$ $UA = 3$ $C_{A0} = 10$ **Uncertain values** $T(0) = [310, 410]$ $T_a = [290, 310]$
Two-state bioreactor $\dot{X} = (\mu - \alpha D) X$ $\dot{S} = D(S_f - S) - k \mu X$ $\mu = \frac{\mu_{max} S}{K_s + S}$	$S(0) = 0.8$ $\alpha = 0.5$ $k = 10.53$ $D = 0.36$ $S_f = 5.7$ $K_s = 7.0$ $k_0 = 0.022$ **Uncertain values** $X(0) = [0.80, 0.85]$ $\mu_{max} = [1.1, 1.2]$

Table 13: Mathematical models of the seven dynamic simulation case studies

Mathematical model	**Initial conditions and parameters**
Three-state bioreactor $\dot{x}_1 = (\mu - D)x_1$ $\dot{x}_2 = D(x_{2f} - x_2) - \frac{\mu x_1}{Y}$ $\dot{x}_3 = Dx_3 + (\alpha\mu + \beta)x_1$ $\mu = \frac{\mu_{max}[1 - (\frac{x_3}{x_{3m}})]x_2}{k_s + x_2}$	$x_2(0) = 5.0$ $x_3(0) = 15.0$ $D = 0.202$ $x_{2f} = 20$ $Y = 0.4$ $\beta = 0.2$ $x_{3m} = 50$ $\alpha = 0.5$ **Uncertain values** $x_1(0) = [6.45, 6.55]$ $\mu_{max} = [0.46, 0.47]$ $k_s = [1.05, 1.1]$
Reactor separator model $\dot{x}_1 = \frac{F+B}{H}(x_F - x_1) + kx_1(1 - x_1)$ $\dot{x}_2 = (L + F + B)x_3 - Bx_2 - Vy_2$ $\dot{x}_3 = (L + F + B)(x_4 - x_3) + V(y_2 - y_3)$ $\dot{x}_4 = (F + B)x_1 + Lx_5 - (L + F + B)x_4 +$ $V(y_3 - y_4)$ $\dot{x}_5 = L(x_6 - x_5) + V(y_4 - y_5)$ $\dot{x}_6 = -(L + D)x_6 + Vy_5$ $x_F = \frac{Fx_{F0} + Bx_2}{F + B}$ $y_i = \frac{\alpha x_i}{1 + (\alpha - 1)x_i} \quad i = 2 \dots 5$	$x_1(0) = 0.5$ $x_2(0) = 0.0$ $x_3(0) = 0.0$ $x_4(0) = 0.0$ $k = 0.06$ $x_{F0} = 0$ $D = 1$ $H = 63.33$ $\alpha = 7.5$ $B = 1.2$ $L = 1.704$ $F = 1.0$ **Uncertain values** $x_5(0) = [0.0, 0.08]$ $x_6(0) = [0.0, 0.16]$

Mathematical model
Glucagon receptor model $\dot{R}_r = k_{-1}LR_u - Lk_1R_r - k_sR_r + k_rR_s$ $\dot{R}_s = k_{sp}LR_p + G_iK_{2s}LR_u + k_s(LR_u + R_r) - k_rR_s$ $\dot{G}_i = -G_iK_{23}LR_u + G_*\left(k_h + \frac{Ca\,k_{Gdeg,Cal}}{K_{Gdeg,Cal} + G_*} + \frac{PLC_*\,k_{Gdeg,PLC}}{K_{Gdeg,PLC} + G_*}\right)$ $\dot{LR}_p = -k_{sp}LR_p + k_p\left(1 + \frac{A_0}{1 + B_1G_*^{-n_1}}\right)\left(\frac{LR_u}{LR_u + B_2}\right)$ $\dot{PLC}_* = k_{PC}G_* - \frac{PLC_*k_{PC,deg}}{K_{PC,deg} + PLC_*}$ $G_* = G_0 - G_i$ $R_0 = R_r + R_s + LR_u + LR_p$
Initial conditions and parameters

Table 13: Mathematical models of the seven dynamic simulation case studies

Mathematical model	Initial conditions and parameters
	$k_1 = 100$
$k_s = 5.2 \times 10^{-3}$	$k_{sp} = k_s$
$K_{2s} = 2 \times 10^{-8}$	$k_r = 4 \times 10^{-3}$
$K_{23} = 1 \times 10^{-7}$	$k_h = 2 \times 10^{-1}$
$k_{Gdeg,Cal} = 1.47 \times 10^3$	$K_{Gdeg,Cal} = 3.54 \times 10^1$
$k_{Gdeg,PLC} = 2.19 \times 10^3$	$K_{Gdeg,PLC} = 5.7$
$k_p = 6.5 \times 10^4$	$A_0 = 3$
$k_{-1} = 10$	$n_1 = 1$
$B_2 = 1 \times 10^6$	$R_0 = 5.5 \times 10^4$
$G_0 = 1 \times 10^5$	$k_{pc} = 6.06 \times 10^{-4}$
$k_{PC,deg} = 2.82 \times 10^{-1}$	$K_{PC,deg} = 2.55 \times 10^{-1}$
Uncertain values	
	$B_1 = [98.8, 102.2]$

References

C. S. Adjiman, I. Androulakis, C. Maranas, and C. A. Floudas. A global optimization method, αBB, for process design. *European Symposium of Computer Aided Process Engineering*, 20(96):419–424, 1996.

S. Balendra and I. D. L. Bogle. Modular global optimisation in chemical engineering. *Journal of Global Optimization*, 45(1):169–185, 2009. ISSN 0925-5001. doi: 10.1007/s10898-009-9401-7.

C. Bendtsen and O. Stauning. FADBAD, a flexible C++ package for automatic differentiation. Technical report, Technical University of Denmark, Lyngby, 1996.

F. Benhamou, F. Goualard, L. Granvilliers, and J.-F. Puget. Revising hull and box consistency. In *Proceedings of the 1999 International Conference on Logic Programming*, pages 230–244, Cambridge, MA, USA, 1999. Massachusetts Institute of Technology. ISBN 0-262-54104-1.

T. Bhatia and L. T. Biegler. Dynamic optimization in the design and scheduling of multiproduct batch Plants. *Industrial & Engineering Chemistry Research*, 35(7):2234–2246, 1996.

CAPD. Computer assisted proofs in dynamics, a package for rigorous numerics, 2013. URL `http://capd.ii.uj.edu.pl`. accessed 09.02.17.

B. Chachuat, A. B. Singer, and P. I. Barton. Global methods for dynamic optimization and mixed-integer dynamic optimization. *Industrial & engineering chemistry research*, 45(25):8373–8392, 2006.

P. Eijgenraam. *The solution of initial value problems using interval arithmetic: formulation and analysis of an algorithm.* Mathematisch Centrum, Amsterdam, 1981. ISBN 9061962307 9789061962304.

W. R. Esposito and C. A. Floudas. Global optimization for the parameter estimation of differential-algebraic systems. *Industrial & Engineering Chemistry Research*, 39(5): 1291–1310, 2000a. ISSN 0888-5885. doi: 10.1021/ie990486w.

W. R. Esposito and C. A. Floudas. Deterministic Global Optimization in Nonlinear Optimal Control Problems. *Journal of Global Optimization*, 17(1-4):97–126, 2000b. ISSN 1573-2916. doi: 10.1023/A:1026578104213.

Q. Fazal and A. Neumaier. Error bounds for initial value problems by optimization. *Soft Computing*, 17(8):1345–1356, 2013. ISSN 1433-7479. doi: 10.1007/s00500-013-1007-9.

A. Flores-Tlacuahuac, L. T. Biegler, and E. Saldívar-Guerra. Dynamic optimization of HIPS open-loop unstable polymerization reactors. *Industrial & Engineering Chemistry Research*, 44(8):2659–2674, 2005. ISSN 0888-5885. doi: 10.1021/ie049534p.

B. Houska, M. E. Villanueva, and B. Chachuat. A Validated Integration Algorithm for Nonlinear ODEs using Taylor Models and Ellipsoidal Calculus. In *52nd IEEE Conference on Decision and Control*, pages 484–489, Florence, Italy, 2013.

L. Jaulin, M. Kieffer, O. Didrit, and E. Walter. *Applied Interval Analysis.* Springer, London, 2001.

E. Kloppenburg and E. D. Gilles. A new concept for operating simulated moving-bed processes. *Chemical Engineering & Technology*, 22(10):813–817, 1999. ISSN 0930-7516. doi: 10.1002/(SICI)1521-4125(199910)22:10⟨813::AID-CEAT813⟩3.0.CO;2-G.

O. Knuppel. PROFIL/BIAS-A Fast Interval Library. *Computing*, 53:277–287, 1994.

Y. Lin and M. A. Stadtherr. Deterministic Global Optimization for Dynamic Systems Using Interval Analysis. *12th GAMM - IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2006)*, pages 38–38, 2006a. doi: 10.1109/SCAN.2006.14.

Y. Lin and M. A. Stadtherr. Deterministic Global Optimization for Parameter Estimation of Dynamic Systems. *Industrial & Engineering Chemistry Research*, 45(25):8438–8448, 2006b. ISSN 0888-5885. doi: 10.1021/ie0513907.

Y. Lin and M. A. Stadtherr. Deterministic Global Optimization of Nonlinear Dynamic Systems. *AIChE Journal*, 53(4):866–875, 2007a. doi: 10.1002/aic.

Y. Lin and M. A. Stadtherr. Validated solutions of initial value problems for parametric ODEs. *Applied Numerical Mathematics*, 57(10):1145–1162, 2007b. ISSN 01689274. doi: 10.1016/j.apnum.2006.10.006.

R. J. Lohner. Computation of Guaranteed Enclosures for the Solutions of Ordinary Initial and Boundary Value Problems. In J. R. Cash and I. Gladwell, editors, *Computational Ordinary Differential Equations*, pages 425–435. Clarendon Press, Oxford, 1992.

R. J. Lohner. On the ubiquity of the wrapping effect in the computation of error bounds. In *Perspectives on enclosure methods*, pages 201–216. Springer, 2001.

R. Luus. Optimal control by dynamic programming using systematic reduction in grid size. *International Journal of Control*, 51(5):995–1013, 1990. doi: 10.1080/00207179008934113.

K. Makino and M. Berz. Taylor models and other validated functional inclusion methods. *International Journal of Pure and Applied Mathematics*, 4(4):379–456, 2003.

R. E. Moore. *Interval Arithmetic and Automatic Error Analysis in Digital Computing.* PhD thesis, Stanford University, 1962.

R. E. Moore. *Interval analysis*, volume 4. Prentice-Hall Englewood Cliffs, 1966.

R. E. Moore, R. B. Kearfort, and M. J. Cloud. *Introduction to Interval Analysis*, volume 22. SIAM, Philadelphia, 2009. ISBN 9780898716696. doi: 10.2307/2004792.

N. S. Nedialkov. Implementing a Rigorous ODE Solver Through Literate Programming. In *Modeling, Design, and Simulation of Systems with Uncertainties*, volume 3, pages 3–19. Springer, 2011. ISBN 9783642159558. doi: 10.1007/978-3-642-15956-5{_}1.

N. S. Nedialkov, K. R. Jackson, and G. F. Corliss. Validated solutions of initial value problems for ordinary differential equations. *Applied Mathematics and Computation*, 105:21–68, 1999.

N. S. Nedialkov, K. R. Jackson, and J. D. Pryce. An Effective High-Order Interval Method for Validating the Existence and Uniqueness of the Solution of an IVP for an ODE. *Reliable Computing*, 7(6):449–465, 2001.

A. Neumaier. *Interval Methods for Systems of Equations.* Cambridge University Press, Cambridge, 1990. ISBN 052133196X.

A. Neumaier. Taylor FormsUse and Limits. *Reliable Computing*, 9(1):43–79, 2003. ISSN 1385-3139. doi: 10.1023/A:1023061927787.

J. Oldenburg, W. Marquardt, D. Heinz, and D. B. Leineweber. Mixed-logic dynamic optimization applied to batch distillation process design. *AIChE Journal*, 49(11):2900–2917, 2003. ISSN 00011541. doi: 10.1002/aic.690491120.

I. Papamichail and C. S. Adjiman. A Rigorous Global Optimization Algorithm for Problems with Ordinary Differential Equations. *Journal of Global Optimization*, (24):1–33, 2002.

I. Papamichail and C. S. Adjiman. Global optimization of dynamic systems. *Computers & Chemical Engineering*, 28(3):403–415, 2004. ISSN 00981354. doi: 10.1016/S0098-1354(03)00195-9.

A. U. Raghunathan, M. Soledad Diaz, and L. T. Biegler. An MPEC formulation for dynamic optimization of distillation operations. *Computers & Chemical Engineering*, 28(10):2037–2052, 2004. ISSN 00981354. doi: 10.1016/j.compchemeng.2004.03.015.

A. Rauh, J. Minisini, and E. P. Hofer. Interval techniques for design of optimal and robust control strategies. In *Scientific Computing, Computer Arithmetic and Validated Numerics, 2006. SCAN 2006. 12th GAMM-IMACS International Symposium on*, pages 37–37. IEEE, 2006.

A. Rauh, M. Brill, and C. Günther. A Novel Interval Arithmetic Approach for Solving Differential-Algebraic Equations with ValEncIA-IVP. *International Journal of Applied Mathematics and Computer Science - Verified Methods: Applications in Medicine and Engineering*, 19(3):381–397, 2009.

A. M. Sahlodin and B. Chachuat. Convex/concave relaxations of parametric ODEs using Taylor models. *Computers & Chemical Engineering*, 35:844–857, 2011a.

A. M. Sahlodin and B. Chachuat. Discretize-then-relax approach for convex/concave relaxations of the solutions of parametric ODEs. *Applied Numerical Mathematics*, 61 (7):803–820, 2011b. ISSN 01689274. doi: 10.1016/j.apnum.2011.01.009.

J. K. Scott and P. I. Barton. Bounds on the reachable sets of nonlinear control systems. *Automatica*, 49(1):93–100, 2013.

A. B. Singer and P. I. Barton. Global Optimization with Nonlinear Ordinary Differential Equations. *Journal of Global Optimization*, 34(2):159–190, 2006. ISSN 0925-5001. doi: 10.1007/s10898-005-7074-4.

A. B. Singer, J. W. Taylor, P. I. Barton, and W. H. Green. Global dynamic optimization for parameter estimation in chemical kinetics. *Journal of Physical Chemistry*, 110: 971–976, 2006.

I. B. Tjoa and L. T. Biegler. Simultaneous solution and optimization strategies for parameter estimation of differential-algebraic equation systems. *Industrial & Engineering Chemistry Research*, 30(2):376–385, 1991. ISSN 0888-5885. doi: 10.1021/ie00050a015.

I. Walawska and D. Wilczak. An implicit algorithm for validated enclosures of the solutions to variational equations for ODEs. *Applied Mathematics and Computation*, 291 (2014):303–322, 2016.

D. Wilczak and P. Zgliczynski. Cr-Lohner algorithm. *Schedae Informaticae*, 20:9–46, 2011.

Y. Zhao and M. A. Stadtherr. Rigorous Global Optimization for Dynamic Systems Subject to Inequality Path Constraints. *Industrial & Engineering Chemistry Research*, 50(22):12678–12693, 2011. ISSN 0888-5885. doi: 10.1021/ie200996f.

J. Žilinskas. Comparison of packages for interval arithmetic. *Informatica*, 2005.