Deterministic Global Optimization in Nonlinear Optimal Control Problems

William R. Esposito and Christodoulos A. Floudas*
Department of Chemical Engineering
Princeton University
Princeton, N.J. 08544-5263
Tel: (609) 258-4595
Fax: (609) 258-0211
email: floudas@titan.princeton.edu

Abstract

The accurate solution of optimal control problems is crucial in many areas of engineering and applied science. For systems which are described by a nonlinear set of differential-algebraic equations, these problems have been shown to often contain multiple local minima. Methods exist which attempt to determine the global solution of these formulations. These algorithms are stochastic in nature and can still get trapped in local minima. There is currently no deterministic method which can solve, to global optimality, the nonlinear optimal control problem.

In this paper a deterministic global optimization approach based on a branch and bound framework is introduced to address the nonlinear optimal control problem to global optimality. Only mild conditions on the differentiability of the dynamic system are required. The implementation of the approach is discussed and computational studies are presented for four optimal control problems which exhibit multiple local minima.

Keywords: Optimal Control, Global Optimization, Differential-Algebraic Equations

*Author to whom all correspondence should be addressed

1 Introduction

The rigorous solution of optimal control problems is of primary importance in many areas of engineering and applied science. The applications of these problems range from the determination of rocket trajectories, to the control of batch chemical systems. In every case, the objective is the generation of an optimal control input to reach a minimum or maximum of a performance measure subject to the dynamics of the system under study. In many problems, especially in chemical engineering, the dynamics of the system can be complex. In most cases the systems are nonlinear in nature and described by ODEs, or by a differential-algebraic set of equations. These factors lead to several difficulties, among which the problem of multiple local minima has received very little attention. Luus and Cormack (1972) showed that a rather simple problem, the temperature control of a batch reactor, can exhibit multiple local solutions. Examples also exist in which over 100 local minima have been identified (see for instance Luus et al. (1992)).

The methods most often employed in the literature to address this problem use control parameterization. This approach converts the infinite dimensional problem into a finite one in which the optimization is performed over a set parameters used to describe the time varying nature of the control input. The methods used for the discretization vary from simple piecewise constant functions to complicated polynomials on finite elements. Different approaches have been taken to account for the dynamic nature of the formulation. First, from a deterministic perspective, two types of methods have been employed, a sequential and a simultaneous. A simultaneous approach involves the discretization of not only the control, but also the state profiles. This leads to a completely algebraic formulation which may be addressed using known local NLP methods, or rigorous optimization approaches (e.g., the αBB Adjiman et al. (1998b,a)). Drawbacks to this approach include a large increase in the size of the variable space and the number of both linear and nonlinear constraints. Also, the type of discretization used for the state profiles can have a dramatic effect on the solution due to the error introduced in the approximation.

Villadsen and Michelsen (1978) provided a wide range of polynomial approximation methods which have been used to solve problems of various complexity. Vlassenbroeck (1988) used a Chebyshev series expansion to solve various optimal control problems. Neuman and Sen (1973) presented a suboptimal control algorithm for systems with state constraints using cubic splines. Tsang et al. (1975) discussed a method based on collocation with simple polynomials in time. Biegler (1984) described a method using collocation at the roots of orthogonal polynomials using Lagrange polynomials. A successive quadratic programming (SQP) method was used to solve the resulting NLP. Tieu et al. (1995) added to this approach by including the time endpoint as one of the collocation points. This allowed for the more accurate solution of problems with constraints on the final state values. Renfro et al. (1987) applied a slight departure to the collocation approach. Instead of using the same type of parameterization for both the state and control, they used a global spline approximation for the state, and a piecewise constant profile for the control. Cuthrell and Biegler (1987) extended the orthogonal collocation approach using a series of finite elements. In each element the profiles were approximated once again using Lagrange polynomials. Wang and Chiou (1995) also considered a collocation approach on finite elements, but included the element endpoints as collocation points to better enforce the continuity between elements. Logsdon and Biegler (1989) considered the properties of fully implicit Runge-Kutta algorithm to develop appropriate error and stability constraints. They also showed the applicability of the approach to optimal control problems with state constraints of order greater than one. Smith and Pantelides (1996) proposed a global optimization algorithm using the simultaneous approach. They applied a symbolic reformulation and spatial branch and bound algorithm to the solution of a reactor synthesis problem using orthogonal collocation to describe the dynamic of the PFR.

The second type of method, referred to as a sequential approach, uses an integration routine to solve for the dynamics of the system. At each iteration of the NLP solver, the DAE system is integrated, and some form of gradient evaluation is performed. The size of the optimization space is not increased nor is any approximation error introduced, but the computational effort resulting from the numerous integrations may be large. Sargent and Pollard (1970) discretized the control into two values, u^{max} and u^{min}, and then determined the optimal switching times between these two control states for the control of plate distillation columns. Litt and Delcommune (1984) compared the application of three different types of splines for the control parameterization. Sargent and Sullivan (1978) described a package which uses piecewise constant control profiles and can handle path and final time constraints on the state and the control variables. They also compared the effect various integration routines have on the convergence of the optimization method and the computational effort required. Goh and Teo (1988) presented a unified method also using piecewise linear control to solve problems involving ordinary differential equations with general constraints on the state and control values. Chen and Hwang (1990) extended this approach to differential-algebraic equation systems. Vassiliadis et al. (1994a,b) discussed a similar approach which uses piecewise Lagrange polynomials on variable size elements to parameterize the control for systems with and without path constraints. The approach was applied to optimal control problems which have multiple stages of operation.

All of the aforementioned approaches suffer from the same major drawback, the ability to converge only to a local solution. Attempts to overcome this difficulty have been made. Rosen and Luus (1992) used line search techniques to determine a series of starting points for their NLP solver. Their approach seems to converge to the global solution 75% of the time for simple problems, but fails more often as the number of local minima increase. Strekalovsky and Vasiliev (1997) presented a search method for the global optimization of maximization problems with convex objectives. The global maximum is obtained by enumeration of the possible maxima. Convergence is guaranteed, but only for a very limited class of problems.

A second class of solution methods was proposed that are probabilistic in nature. They are designed to increase the chances of finding the global solution of the problem. Luus and Bojkov (1994) applied dynamic programming methods (Luus, 1990) to solve the bifunctional catalyst problem (known to contain a large number of local minima). The authors show that, on average, 75 % of the time convergence to the global minima is reached. The probability of obtaining the global solution is related to various parameters in the method (Bojokov and Luus, 1993). The approach was also applied to piecewise linear control profiles (Luus, 1993), final state constrained systems (Luus and Rosen, 1991), and inequality path constrained systems (Mekarapiruk and Luus, 1997). Dadebo and Mcauley (1995) applied this technique to solving constrained chemical engineering systems. Luus and Hennessy (1999) applied a direct search technique (Luus and Jaakola, 1973) to the optimization of fed-batch fermentors. Wang and Chiou (1997) used an algorithm based on differential evolution. A piecewise control parameterization with variable switching times was used, while the differential equations were solved using a modified collocation approach. The results of the differential evolution approach were then used as starting points for a standard SQP algorithm to refine the control policy. Carrasco and Banga (1997) presented an approach which used a stochastic optimization method to solve the formulation. Control parameterization was accomplished using piecewise linear functions and the differential-algebraic system was solved using an integration routine. The solution obtained from the stochastic algorithm could then be used as the the starting point for a deterministic method (Carrasco and Banga, 1998). Ali et al. (1997) tested the application of different stochastic algorithms to different types of problems. One of the problems studied is the optimal control of a chemical reactor using a piecewise constant temperature profile. Banga and Seider (1996) applied a random search technique to the solution of control and design problems which arise in the optimization of chemical processes. This algorithm was also applied to state constrained optimal and model-predictive control (Banga et al., 1998), and the optimization of batch and semi-continuous bioprocesses (Banga et al., 1997). For further discussion about recent advances in optimal control theory, algorithms, and applications, the reader is directed to the book of Hager and Pardalos (1998). For a background on the theoretical, algorithmic, applications and test problems of deterministic global optimization, the reader is directed to the recent book of Floudas (2000) and the recent handbook of test problems by Floudas et al. (1999).

All of the attempts to find the global solution in the general case are probabilistic in nature. Even though the likelihood of determining the global minimum is increased, they can still be trapped in a local solution. Hence, a deterministic approach which can guarantee convergence to the global solution of nonlinear optimal control problems is a major challenge. In this study, a method using the sequential approach will be presented. The approach is based on the branch and bound algorithm αBB (Androulakis et al., 1995; Adjiman et al., 1996, 1998b,a) originally developed for algebraic problems. In section 2, the formulation of the optimization problem and methods of control parameterization will be discussed. Section 3 will deal with the proposed approach. The extensions and theorems which allow the application to these dynamic problems will be outlined, and an illustrative example demonstrating the key points will be discussed. Finally, in section 4 a series of example problems will be presented to describe both the theoretical and computational aspects of the proposed approach.

2 Problem Formulation

The general formulation for the problems studied is:

$$
\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{v},\boldsymbol{u}(t)} \quad & f(\boldsymbol{x},\boldsymbol{v}) \\
\text{s.t.} \quad & \\
& \dot{\boldsymbol{z}}_j \;=\; \mathbf{g}(\mathbf{z},\boldsymbol{v},\boldsymbol{u},t) \qquad j \in J \\
& \mathbf{0} \;=\; \mathbf{h}(\boldsymbol{z},\boldsymbol{v},\boldsymbol{u},t) \\
& \boldsymbol{z}(t_o) \;=\; \boldsymbol{z}_o \qquad t \;\in\; [t_o,t_f] \\
& \boldsymbol{u}^L \;\leq\; \boldsymbol{u} \;\leq\; \boldsymbol{u}^U \\
& \\
& c_{m,\mu}(\boldsymbol{x}) \;+\; z_m(t_\mu) \;=\; 0 \qquad m \in M \quad \mu \in P \\
& \mathbf{d}(\boldsymbol{x},\boldsymbol{v}) \;\leq\; \mathbf{0} \\
& \boldsymbol{x}^L \;\leq\; \boldsymbol{x} \;\leq\; \boldsymbol{x}^U \quad , \quad \boldsymbol{v}^L \;\leq\; \boldsymbol{v} \;\leq\; \boldsymbol{v}^U
\end{aligned}
\tag{1}
$$

where $\boldsymbol{z}$ belongs to the set Z of state variables, J is the set of states whose derivatives appear explicitly in the system, M is the set of states which appear in point constraints defined at time points of the set P, $\boldsymbol{u}$ is the time varying set of control variables, $\boldsymbol{v}$ are time invariant parameters which appear in the dynamic system, and $\boldsymbol{x}$ are algebraic variables which do not appear in the dynamic system.

The following conditions on the functions in (1) must also hold:

1. $f(\boldsymbol{x},\boldsymbol{v})$, $\mathbf{c}(\boldsymbol{x})$, $\mathbf{d}(\boldsymbol{x},\boldsymbol{v})$ are twice continuously differentiable functions in the region defined by $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$ and $\boldsymbol{v} \in [\boldsymbol{v}^L, \boldsymbol{v}^U]$.
2. The differential-algebraic system is at most index 1.
3. $\mathbf{g}(\boldsymbol{z},\boldsymbol{v},\boldsymbol{u},t)$ and $\mathbf{h}(\boldsymbol{z},\boldsymbol{v},\boldsymbol{u},t)$ are continuous and twice differentiable with respect to the states, $\boldsymbol{z}$, the controls, $\boldsymbol{u}$, and the parameters, $\boldsymbol{v}$, in the region defined by $\boldsymbol{z} \in [\boldsymbol{z}^L, \boldsymbol{z}^U]$, $\boldsymbol{u} \in [\boldsymbol{u}^L, \boldsymbol{u}^U]$, and $\boldsymbol{v} \in [\boldsymbol{v}^L, \boldsymbol{v}^U]$.

In order to convert the infinite dimensional problem given by (1) into a finite one, control parameterization is used. The control variables are written as a function of the parameters, $\boldsymbol{v}$, and time, t,

$$
\boldsymbol{u}(t) \;=\; \mathcal{U}(\boldsymbol{v},t) \quad . \tag{2}
$$

One common control parameterization is a piecewise constant profile. In the case of a single control variable, this takes the form

$$
u(t) \;=\; v_i \quad \text{for} \quad t_i \leq t < t_{i+1} \quad . \tag{3}
$$

Another control parameterization scheme is a piecewise linear profile:

$$u(t) \;=\; v_i + \frac{v_{i+1} - v_i}{t_{i+1} - t_i}\,(t - t_i) \qquad \text{for} \quad t_i \leq t < t_{i+1} \quad . \tag{4}$$

The control variables, $\boldsymbol{u}$, are then treated as state variables, and the optimization only takes place over the time invariant variables $\boldsymbol{x}$ and $\boldsymbol{v}$. The problem reduces to:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{v}} \quad & f(\boldsymbol{x},\boldsymbol{v}) \\
\text{s.t.} \quad & \\
& \dot{\boldsymbol{z}}_j \;=\; \mathbf{g}(\mathbf{z},\boldsymbol{v},t) \qquad j \in J \\
& \mathbf{0} \;=\; \mathbf{h}(\boldsymbol{z},\boldsymbol{v},t) \\
& \boldsymbol{z}(t_o) \;=\; \boldsymbol{z}_o \qquad t \;\in\; [t_o,t_f] \\
& \\
& c_{m,\mu}(\boldsymbol{x}) \;+\; z_m(t_\mu) \;=\; 0 \qquad m \in M \quad \mu \in P \\
& \mathbf{d}(\boldsymbol{x},\boldsymbol{v}) \;\leq\; \mathbf{0} \\
& \boldsymbol{x}^L \;\leq\; \boldsymbol{x} \;\leq\; \boldsymbol{x}^U \quad , \quad \boldsymbol{v}^L \;\leq\; \boldsymbol{v} \;\leq\; \boldsymbol{v}^U
\end{aligned} \tag{5}$$

A numerical integration routine is used to generate the necessary function and gradient evaluations for the dynamic system in the course of the optimization algorithm (Schweiger and Floudas, 1998b).

3 Global Optimization Approach

Within this section, the basic concepts of the αBB approach will be presented along with the theorems which allow for its extension to dynamic systems of equations. An illustrative example will be provided to show the key points of the approach.

3.1 Overview of the αBB

The αBB global optimization method (Androulakis et al., 1995; Adjiman et al., 1996; Adjiman and Floudas, 1996; Adjiman et al., 1998b,a) guarantees convergence to the global minimum for general twice continuously differentiable constrained and unconstrained NLPs. This is accomplished through the generation of a non-decreasing sequence of lower bounds and a non-increasing sequence of updated upper bounds on the global solution. Finite ϵ-convergence to the global minimum is achieved through the successive subdivision of the region at each level in the branch and bound tree. The sequence of upper bounds on the global solution is obtained by solving, to local optimality, the full nonconvex problem from different starting points. The lower bounds are generated by solving a convex relaxation which underestimates the original problem.

There are three main components in the αBB algorithm which affect the performance of the method. These include: (1) the generation of the underestimating formulation which will allow for the determination of a valid lower bound on the global solution, (2) the method for determining the variable to branch the region on at each level of the tree, and (3) the approaches used to determine appropriate variable bounds at each level. Each component has a distinct effect on the convergence properties of the algorithm and will be discussed.

3.2 Underestimating Formulation

Within the formulation given by (5), there are two different types of terms, an algebraic part, and a dynamic part. The algebraic part consists of the functions, $f(\boldsymbol{x}, \boldsymbol{v})$, $\mathbf{c}(\boldsymbol{x})$, and $\mathbf{d}(\boldsymbol{x}, \boldsymbol{v})$. These are underestimated using the techniques described by Adjiman and Floudas (1996); Adjiman et al. (1998b,a).

A challenging question is how to deal with the dynamic part of the formulation. First, consider the differential-algebraic system of equations given in the formulation as a simple input-output map. The input represents the values of the parameters, $\boldsymbol{v}$, while the output corresponds to the values of the states along time, $\boldsymbol{z}(t)$.

$$\boldsymbol{v} \longrightarrow \boxed{\begin{array}{rcl} \dot{\boldsymbol{z}}_j &=& \mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t) \quad j \in J \\ \mathbf{0} &=& \mathbf{h}(\boldsymbol{z}, \boldsymbol{v}, t) \\ \boldsymbol{z}_j(t_0) &=& \boldsymbol{z}_0 \quad t \in [t_0, t_f] \end{array}} \longrightarrow \boldsymbol{z}(t) \tag{6}$$

Pontryagin (1962) presented two theorems concerning the continuity and differentiability of this input-output map with respect to the parameters, $\boldsymbol{v}$. The theorems are derived and proven for a system of ODEs. It is possible to convert the DAE system in (6) into a set of ODEs either by explicitly solving $\mathbf{0} = \mathbf{h}(\boldsymbol{z}, \boldsymbol{v}, t)$ for the algebraic variables, $z_i \quad i \notin J$, and substituting into $\mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t)$ or through one differentiation of $\mathbf{h}(\boldsymbol{z}, \boldsymbol{v}, t)$ since the system is of most order one (Brenan et al., 1996).

Given the system

$$\dot{\boldsymbol{z}} = \mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t) \quad , \tag{7}$$

the assumption that the right hand side of (7), $\mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t)$, and the partial derivatives,

$$\frac{\partial}{\partial \boldsymbol{z}} \mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t) \tag{8}$$

are defined and are continuous is some domain $\boldsymbol{\Gamma}$ of the space of variables t, $\boldsymbol{z}$, and $\boldsymbol{v}$, we have:

Theorem 1 (Pontryagin (1962), page 170) *If* $(t_0,\ \boldsymbol{z}_0,\ \boldsymbol{v}_0)$ *is an arbitrary point of the domain* $\mathbf{\Gamma}$*, there exist positive numbers* r *and* ρ *such that for:*

$$|\boldsymbol{v} - \boldsymbol{v}_0| < \rho$$

the solution

$$\boldsymbol{z} \ = \ \boldsymbol{\psi}(t, \boldsymbol{\theta})$$

of (7) which satisfies the initial condition

$$\boldsymbol{\psi}(t_0, \boldsymbol{v}) = \boldsymbol{z}_0$$

is defined on the interval $|t - t_0| < r$ *and is a continuous function of all the variables,* t *and* $\boldsymbol{v}$*, on which it depends.*

Theorem 2 (Pontryagin (1962), page 173) *Let the partial derivatives*

$$\mathbf{e}(\boldsymbol{z}, \boldsymbol{v}, t) = \frac{\partial}{\partial \boldsymbol{v}}\ \mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t)$$

exist and be continuous in the domain $\mathbf{\Gamma}$*. Let* $(t_0,\ \boldsymbol{z}_0,\ \boldsymbol{v}_0)$ *be some point of* $\mathbf{\Gamma}$*. Then, there exist positive numbers* r' *and* ρ' *such that for* $|t - t_0| < r'$, $|\boldsymbol{v} - \boldsymbol{v}_0| < \rho'$ *the solution* $\boldsymbol{\psi}(t, \boldsymbol{v})$ *of (7) which satisfies the initial condition:*

$$\boldsymbol{\psi}(t_0, \boldsymbol{v}) = \boldsymbol{z}_0 \ ,$$

has continuous partial derivatives

$$\frac{\partial \boldsymbol{\psi}(t, \boldsymbol{v})}{\partial \boldsymbol{v}} \ .$$

Corollary 1 (Pontryagin (1962), page 177) *If all the partial derivatives of* $\mathbf{g}(\boldsymbol{z}, \boldsymbol{v}, t)$ *with respect to the variables* $\boldsymbol{z}$ *and* $\boldsymbol{v}$ *up to the* m^{th} *order inclusive exist and are continuous, then the functions* $\boldsymbol{\psi}(t, \boldsymbol{v})$ *also have continuous partial derivatives with respect to the parameters,* $\boldsymbol{v}$*, up to the* m^{th} *order inclusive.*

Given the conditions on the differential-algebraic system presented in section 2, and the aforementioned theorems, the values of the states, $\boldsymbol{z}$, at a given time t_μ can be defined as a set of twice continuously differentiable functions of the parameters, $\boldsymbol{v}$:

$$\boldsymbol{z}(t_\mu) \;=\; \boldsymbol{\mathcal{F}}(t_\mu, \boldsymbol{v}) \;\equiv\; \boldsymbol{\mathcal{F}}_\mu(\boldsymbol{v}) \quad . \tag{9}$$

Substituting this function into (5) results in:

$$\begin{aligned}
&\min_{\boldsymbol{x},\boldsymbol{v}} \;\; f(\boldsymbol{x}, \boldsymbol{v}) \\
&\text{s.t.} \\
&\qquad c_{m,\mu}(\boldsymbol{x}) \;+\; \mathcal{F}_{m,\mu}(\boldsymbol{v}) \;=\; 0 \qquad m \in M \quad \mu \in P \\
&\qquad \mathbf{d}(\boldsymbol{x}, \boldsymbol{v}) \;\leq\; \mathbf{0} \\
&\qquad \boldsymbol{x}^L \;\leq\; \boldsymbol{x} \;\leq\; \boldsymbol{x}^U \quad , \quad \boldsymbol{v}^L \;\leq\; \boldsymbol{v} \;\leq\; \boldsymbol{v}^U
\end{aligned} \tag{10}$$

where $\mathcal{F}_{m,\mu}$ is the function which describes the value of state z_m at time point t_μ. The derivatives of this function are the values of the sensitivities of state z_m with respect to the parameters, $\boldsymbol{v}$, at the given time point t_μ. These sensitivities are determined by simultaneously integrating a set of linear equations with the initial system (Brenan et al., 1996; Caracotsios and Stewart, 1985; Vassiliadis et al., 1994a). The set of equations, for this system, take the form of:

$$\begin{bmatrix} \mathbf{I}_{j\times j} & \mathbf{0}_{j\times k} \\ \mathbf{0}_{h\times j} & \mathbf{0}_{h\times k} \end{bmatrix} \dot{\left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{v}}\right)} \;=\; \begin{bmatrix} \frac{\partial \mathbf{g}}{\partial \boldsymbol{z}} \\ \frac{\partial \mathbf{h}}{\partial \boldsymbol{z}} \end{bmatrix} \left(\frac{\partial \boldsymbol{z}}{\partial \boldsymbol{v}}\right) \;+\; \begin{bmatrix} \frac{\partial \mathbf{g}}{\partial \boldsymbol{v}} \\ \frac{\partial \mathbf{h}}{\partial \boldsymbol{v}} \end{bmatrix} \tag{11}$$

where j is the number of states with explicit derivatives, k are the number of states without explicit derivatives, and h is size of the vector of equations $\mathbf{h}$. Therefore the derivatives of the function $\mathcal{F}_{m,\mu}$ with respect to $\boldsymbol{v}$ are defined as:

$$\frac{\partial \mathcal{F}_{m,\mu}}{\partial \boldsymbol{v}} \;=\; \frac{\partial z_m}{\partial \boldsymbol{v}}(t_\mu) \quad . \tag{12}$$

A numerical integration routine is used to generate the necessary function and gradient evaluations for the dynamic system in the course of the optimization algorithm (Schweiger and Floudas, 1998a). The underestimator of this function is generated by adding a quadratic term in $\boldsymbol{v}$:

$$\mathcal{L}_{\mathcal{F}_{m,\mu}}(\boldsymbol{v}) \;=\; \mathcal{F}_{m,\mu}(\boldsymbol{v}) \;+\; \sum_{i\in I} \beta_{m,\mu,i} \left(v_i^U - v_i\right)\left(v_i^L - v_i\right) \tag{13}$$

where I is the set of $\boldsymbol{v}$ variables, and $\mathcal{L}_{\mathcal{F}}$ represents the underestimator of the function $\mathcal{F}$. A simplification of (13) has all the $\beta_{m,\mu,i}$ equal $\forall i \in I$. This reduces (13) to

$$\mathcal{L}_{\mathcal{F}_{m,\mu}}(\boldsymbol{v}) \;=\; \mathcal{F}_{m,\mu}(\boldsymbol{v}) \;+\; \beta_{m,\mu} \sum_{i \in I} \left(v_i^U - v_i\right)\left(v_i^L - v_i\right) \quad . \tag{14}$$

The value of the β parameter needs to be large enough to ensure convexity, but not too large as to overly underestimate the term. The convex relation of formulation (10) including the underestimators for the algebraic functions is written as:

$$\begin{aligned}
\min_{\boldsymbol{x},\boldsymbol{v}} \;\; & \mathcal{L}_f(\boldsymbol{x}, \boldsymbol{v}) \\
\text{s.t.} \;\; & \\
& \mathcal{L}^+_{c_{m,\mu}}(\boldsymbol{x}) \;+\; \mathcal{L}^+_{\boldsymbol{\mathcal{F}}_{m,\mu}}(\boldsymbol{v}) \;\leq\; 0 \qquad m \in M \quad \mu \in P \\
& \mathcal{L}^-_{c_{m,\mu}}(\boldsymbol{x}) \;+\; \mathcal{L}^-_{\boldsymbol{\mathcal{F}}_{m,\mu}}(\boldsymbol{v}) \;\leq\; 0 \qquad m \in M \quad \mu \in P \\
& \mathcal{L}_{\mathbf{d}}(\boldsymbol{x}, \boldsymbol{v}) \;\leq\; 0 \\
& \boldsymbol{x}^L \;\leq\; \boldsymbol{x} \;\leq\; \boldsymbol{x}^U \quad , \quad \boldsymbol{v}^L \;\leq\; \boldsymbol{v} \;\leq\; \boldsymbol{v}^U
\end{aligned} \tag{15}$$

where $\mathcal{L}^+_{c_{m,\mu}}$ is the underestimator of the function $c_{m,\mu}$, $\mathcal{L}^-_{c_{m,\mu}}$ is the underestimator of the function $-c_{m,\mu}$ and $\mathcal{L}^+_{\boldsymbol{\mathcal{F}}_{\mu,m}}$, $\mathcal{L}^-_{\boldsymbol{\mathcal{F}}_{\mu,m}}$, and $\mathcal{L}_{\mathbf{d}}$ are all similarly defined. It is necessary to split the equality point constraints into a positively and a negatively signed inequality, each being underestimated separately. Depending on the nature of the problem and the constraint it may only be necessary to include half of the set.

3.2.1 Determination of β Values

The β values are calculated using the Hessian matrix of the function $\mathcal{F}_{m,\mu}(\boldsymbol{v})$. This matrix is generated using the second order sensitivity of the state $\boldsymbol{z}_m$ with respect to the parameters $\boldsymbol{v}$,

$$\mathcal{H}_{m,\mu} \;\equiv\; \frac{\partial^2 z_m}{\partial \boldsymbol{v}^2}(t_\mu) \quad . \tag{16}$$

Equations (13) and (14) refer to the addition of a diagonal shift matrix to the Hessian matrix of the function,

$$\mathcal{H}_{\mathcal{L}_{m,\mu}} \;=\; \mathcal{H}_{m,\mu} \;+\; 2\Delta_{m,\mu} \quad . \tag{17}$$

where the matrix $\Delta_{m,\mu}$ is composed of diagonal elements $\beta_{m,\mu,i}$. Using a *uniform diagonal shift* method, that is where all elements of the Δ matrix are equal, the value of β needed to make $\mathcal{H}_{\mathcal{L}}$ positive semi-definitive is shown to be (Maranas and Floudas, 1994),

$$\begin{aligned} \beta_{m,\mu} &\geq -\frac{1}{2} \min_{\boldsymbol{v}} \lambda_{m,\mu}^{min}(\boldsymbol{v}) \\ \beta_{m,\mu} &\geq 0 \end{aligned} \tag{18}$$

where $\lambda_{m,\mu}^{min}$ is the minimum eigenvalue of the Hessian matrix $\mathcal{H}_{m,\mu}$. The difficulty arises from the fact that $\mathcal{H}_{m,\mu}$ can not be written as an analytical function of $\boldsymbol{v}$. The elements of the matrix, however, can be determined through an integration of the augmented system at given values of $\boldsymbol{v}$. As a result of this, three different methods for the determination of β values have been developed.

Constant or Semi-Constant β values

In this approach, the values which will be used for the β parameters are preselected. These may either be constant throughout the branch and bound tree, or can be a function of the tree level. A functional form used which varies the value of β by tree level is:

$$\beta = \beta_0 \left\{ \prod_{i \in I} \left(\frac{v_i^U - v_i^L}{v_i^{U,orig} - v_i^{L,orig}} \right) \right\}^{\frac{1}{n}} \tag{19}$$

where I is the set of all branching variables, v_i^U and v_i^L are the bounds of the i variable in the current region, and $v_i^{U,orig}$, and $v_i^{L,orig}$ are the original bounds on the variable at the root node of the branch and bound tree. The product given in (19) varies with the tree level and asytompically approaches 0. The value of n determines the rate at which β will approach 0, starting from β_0. This approach is the easiest, with the least computational effort required. But the drawback is that no second order information, which could be made available, is being used and the validity of the underestimator is not known. This method is only possible when a uniform diagonal shift matrix is being used.

Sampling Approach

The values of the elements of $\mathcal{H}$ are determinable at given values of the parameters $\boldsymbol{v}$. Therefore, in each region, a number of points are selected to evaluate the Hessian matrix and calculate the eigenvalues of these matrices. The minimum of these eigenvalues is then used to calculate the value of β. The number of points sampled and the sampling method are both parameters in this approach. In all cases a uniformly random sample is generated. The number of points used in the sample has both an effect on the validity of the underestimator and the time required to generate it. The more points used, the better the approximation, but the more time required. The size of the sampled set needs to be large enough to generate a valid value for β, but not too large so as to require a substantial computational expense. As with the constant approach, this method is only possible when a uniform diagonal shift matrix is employed.

Sampling with Interval Calculation

In this approach, the values of each element of $\mathcal{H}$ are also determined at given values of the parameters, but the eigenvalues of these Hessian matrices are not directly determined.

Instead, an interval Hessian matrix is generated by determining the minimum and maximum of every element over the sampled points. A valid lower bound on the minimum eigenvalue of this matrix can then by determined using methods presented by Adjiman and Floudas (1996); Adjiman et al. (1998b). Methods have been developed using either a uniform or nonuniform diagonal shift matrix.

3.2.2 Illustrative Example

In order to illustrate the above concepts, consider the system:

$$\begin{aligned} \dot{z} &= u - z^3 \\ z(t_o) &= 9 \quad t \in [0,1] \\ -5 &\le u \le 5 \end{aligned}$$

For the sake of simplicity, a constant control, u, over the whole time horizon will be used. Consider the value of the state, z, at $t = 1$, which is an implicit function of the control, $\mathcal{F}(u)$. This is plotted in Figure 1. Notice the continuity and nonconvex form of this function.

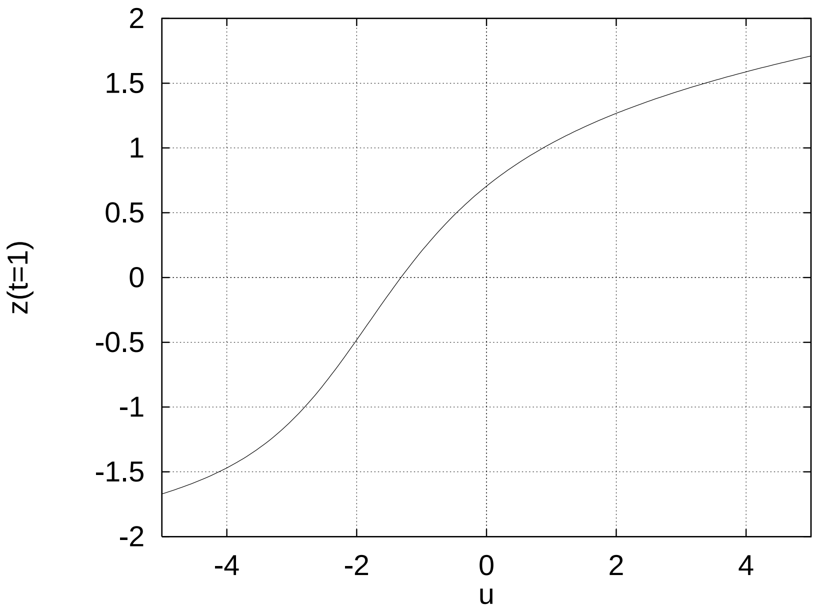

Figure 1: Value of the state at final time versus the control for the illustrative example

The first order sensitivity of the state with respect to the control, $\frac{\partial z}{\partial u}$, is determined by integrating an additional equation with the one given above. Using the expression given by (11), the sensitivity equation is

$$\dot{\frac{\partial z}{\partial u}} = -3\, z^2 \frac{\partial z}{\partial u} + 1 \ . \tag{20}$$

The second order sensitivity, $\frac{\partial^2 z}{\partial u^2}$, for the sake of simplicity, is calculated using a finite difference approximation. These sensitivities are plotted in Figure 2.

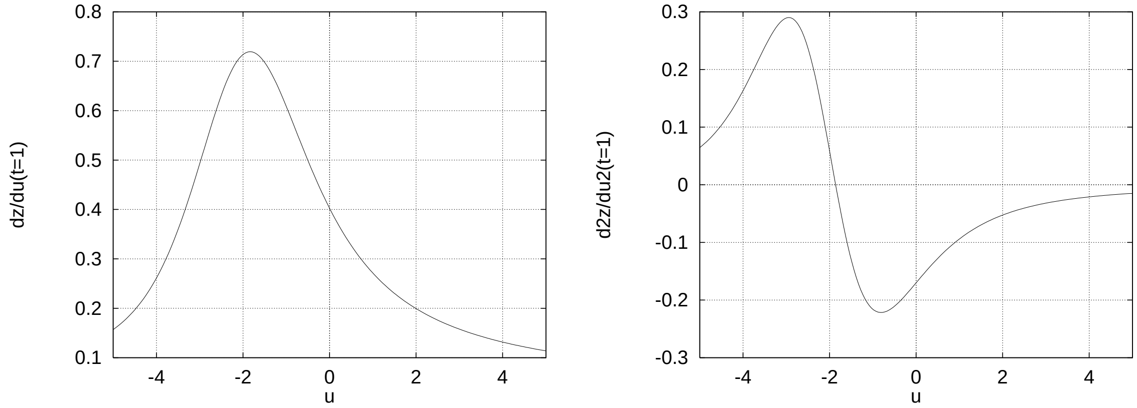

Figure 2: Values of the first and second order sensitivities at final time versus the control for the illustrative example

The minimum of the second order sensitivity is -0.2214. Using the expression given by (18) a β value of 0.1107 is needed to generate a convex relaxation of this function. That relaxation takes the form,

$$\mathcal{L}_{\mathcal{F}}(u) \;=\; \mathcal{F}(u) \;+\; 0.1107 \left(u^U - u\right) \left(u^L - u\right) \tag{21}$$

The original function and the underestimator given by (21) are plotted in Figure 3.

3.3 Determination of a Branching Variable

At each level of the tree, it is necessary to determine which variable will be used to bisect the region. This branching variable is chosen based upon its contribution to the quality of the convex underestimator.

$$i^* \;=\; \arg\max_i \; \delta_i \tag{22}$$

where i^* is the index of the branching variable, and δ_i is the overall contribution calculated for a given variable i from the set $[\boldsymbol{v},\, \boldsymbol{x}]$. The overall contribution is made up of two components,

$$\delta_i \;=\; \delta_i^d \;+\; \delta_i^a \tag{23}$$

where δ_i^d is the contribution calculated from the dynamic part of the formulation, and δ_i^a is from the algebraic part. These components can be determined using two methods. Each

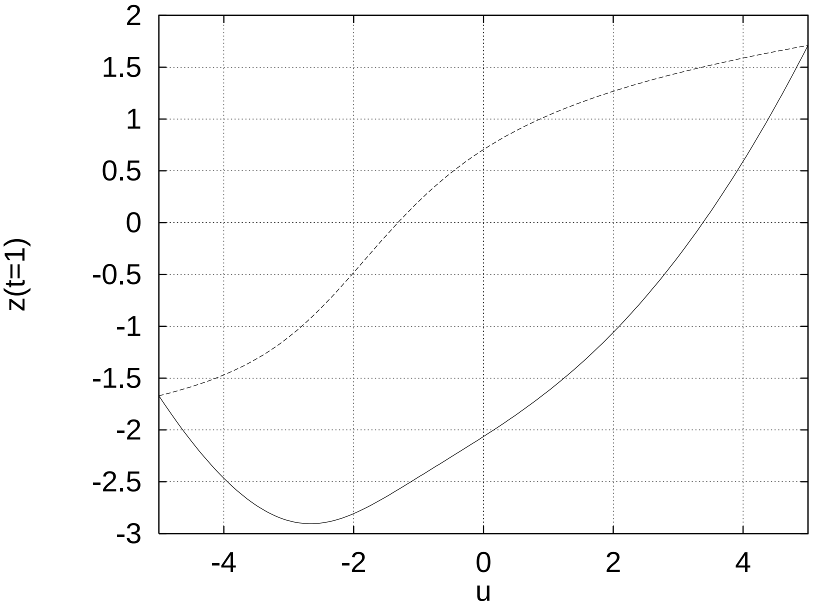

Figure 3: Value of the underestimation function versus the control for the illustrative example

method attempts to determine the overall effect a given variable has on the difference between the original formulation and the convex underestimation.

Method 1: The measure is based on the difference between the original function and the underestimator taken at the solution to the lower bounding problem. For the dynamic part of the formulation, this measure is calculated by

$$\delta_i^d \;=\; \sum_{m \in M} \sum_{\mu \in P} \beta_{m,\mu,i} \, \left(v_i^U - v_i^{sol}\right) \left(v_i^L - v_i^{sol}\right) \tag{24}$$

where $\boldsymbol{v}^{sol}$ is the solution to the lower bounding problem. For the algebraic part, this measure is calculated as:

$$\delta_i^a \;=\; \sum_{k \in K_i} f_k(\boldsymbol{x}^{sol}, \boldsymbol{v}^{sol}) \;-\; \mathcal{L}_{f_k}(\boldsymbol{x}^{sol}, \boldsymbol{v}^{sol}) \tag{25}$$

where K_i is the set of algebraic functions, $f_k(\boldsymbol{x}, \boldsymbol{v})$, in which the given variable i from the set $[\boldsymbol{x}, \boldsymbol{v}]$ participates and $\mathcal{L}_{f_k}(\boldsymbol{x}, \boldsymbol{v})$ represents the underestimator of this function.

Method 2: The measure is based on the maximum separation distance between the original function and the underestimator. For the dynamic part, the measure is calculated by

$$\delta_i^d \;=\; \sum_{m \in M} \sum_{\mu \in P} \beta_{m,\mu,i} \, \frac{1}{4} \, \left(v_i^U - v_i^L\right)^2 \quad . \tag{26}$$

For the algebraic part,

$$\delta_i^a \;=\; \sum_{k \in K_i} \max_{\boldsymbol{x},\boldsymbol{v}} \; [f_k(\boldsymbol{x},\boldsymbol{v}) \;-\; \mathcal{L}_{f_k}(\boldsymbol{x},\boldsymbol{v})] \quad . \tag{27}$$

3.4 Variable Bound Updating

At each iteration it is possible to update the bounds on some or all of the algebraic variables by solving a series of optimization problems,

$$w_j^{L,*}/w_j^{U,*} = \left\{ \begin{array}{ll} \min\limits_{\boldsymbol{x},\boldsymbol{v}} / \max\limits_{\boldsymbol{x},\boldsymbol{v}} & w \\ \quad\quad\quad \text{s.t.} & \mathcal{L}^{+}_{c_{m,\mu}}(\boldsymbol{x}) \;+\; \mathcal{L}^{+}_{\boldsymbol{\mathcal{F}}_{m,\mu}}(\boldsymbol{v}) \;\leq\; 0 \\ & \mathcal{L}^{-}_{c_{m,\mu}}(\boldsymbol{x}) \;+\; \mathcal{L}^{-}_{\boldsymbol{\mathcal{F}}_{m,\mu}}(\boldsymbol{v}) \;\leq\; 0 \\ & \mathcal{L}_{\mathbf{d}}(\boldsymbol{x},\boldsymbol{v}) \;\leq\; \mathbf{0} \\ & \mathcal{L}_f(\boldsymbol{x},\boldsymbol{v}) \;\leq\; UB \\ & \boldsymbol{x}^L \;\leq\; \boldsymbol{x} \;\leq\; \boldsymbol{x}^U \\ & \boldsymbol{v}^L \;\leq\; \boldsymbol{v} \;\leq\; \boldsymbol{v}^U \end{array} \right. \tag{28}$$

where $\boldsymbol{w} \equiv [\boldsymbol{x},\boldsymbol{v}]$, the set of all algebraic variables, and $j \in J$ where J is the set of variables selected for updating. $\mathcal{L}_f(\boldsymbol{x},\boldsymbol{v})$ is the relaxed objective function, and UB is the current best upper bound on the global solution.

3.5 Algorithmic Procedure

The steps of the detailed algorithmic approach are as follows.

Step 1. Initialize the problem:

1. Set the relative, ϵ^{rel}, or the absolute, ϵ^{abs} convergence tolerance.
2. Set the iteration counter, $iter$, to zero.
3. If a sampling method is being used to calculate the β values, set the number of points to use at the first iteration, $p^{initial}$, and the minimum number at each subsequent iteration, p^{every}.

Step 2. Calculate initial β values:

1. Setup the augmented DAE system with the first order sensitivities.
2. Integrate, with sensitivity evaluations, the system at $p^{initial}$ randomly selected values for the $\boldsymbol{v}$ variables.

3. Save the values of the second order sensitivities (the sensitivities of the sensitivity variables) at the necessary time points. These will be used at subsequent iterations.

4. Using the selected method, calculate the values of the β parameters.

Step 3. Determine initial lower and upper bounds on the global solution:

1. Solve the convex relaxation from a randomly chosen starting point.

2. Save the solution and all the variable values:

$$obj^* \rightarrow obj^{1,1} \rightarrow LB \quad \boldsymbol{x}^* \rightarrow \boldsymbol{x}^{1,1} \quad \boldsymbol{v}^* \rightarrow \boldsymbol{v}^{1,1} \quad \boldsymbol{\beta} \rightarrow \boldsymbol{\beta}^{1,1} \quad .$$

3. Using the lower bounding solution as a starting point, solve the original nonconvex problem to local optimality to determine an upper bound (UB) on the global solution.

4. Save the solution and all the variable values:

$$obj^* \rightarrow obj^{UB} \quad \boldsymbol{x}^* \rightarrow \boldsymbol{x}^{UB} \quad \boldsymbol{v}^* \rightarrow \boldsymbol{v}^{UB} \quad .$$

Step 4. Check for convergence
if $UB - LB \leq \epsilon^{abs}$ or $|(UB - LB)/UB| \leq \epsilon^{rel}$ then terminate with the global solution:

$$obj^{glo} = obj^{UB} \quad \boldsymbol{x}^{glo} = \boldsymbol{x}^{UB} \quad \boldsymbol{v}^{glo} = \boldsymbol{v}^{UB}$$

otherwise $iter = iter + 1$.

Step 5. Update Lower Bound:

1. Select the next region to be explored as the one with the lowest solution to the relaxed problem.

2. The region is removed from the list of stored regions, and the solution becomes the new lower bound

$$LB = \min_{iter,r} \quad obj^{iter,r} \quad .$$

Step 6. Update variable bounds:

1. Select a given number of algebraic variables, $\boldsymbol{v}$ or $\boldsymbol{x}$, to have their bounds updated.

2. Solve (28) for each of the selected variables using the current value of the upper bound, UB.

Step 7. Branch the region:

1. Select the variable to branch the region on using the criteria shown in section 3.3.

2. Branch the region into two ($r = 1, 2$) by bisecting on the selected variable.

Step 8. Update β values in each region

1. Search the list of saved integration points to determine the number which belong to the given region, $p^{current}$.
2. If $p^{current} < p^{every}$ select additional points randomly to make up the difference.
3. Save the additional integration results.
4. Calculate the necessary values of the β parameters.
5. The calculated value of every β parameter must be less than or equal to the value used at the parent node of the region.

Step 9. Determine lower and upper bounds in each region:

1. Solve the convex relaxation from a randomly chosen starting point.
2. Save the solution if it is less than the current upper bound (UB):

$$\text{If } obj^{lower} < UB \quad \text{then} \quad obj^* \rightarrow obj^{iter,r} \quad \boldsymbol{x}^* \rightarrow \boldsymbol{x}^{iter,r} \quad \boldsymbol{v}^* \rightarrow \boldsymbol{v}^{iter,r} \quad \boldsymbol{\beta} \rightarrow \boldsymbol{\beta}^{iter,r} \quad .$$

   Otherwise, if the solution is greater than the upper bound, or if there is no feasible solution, remove the region from consideration and go to Step 4.

3. Solve the upper problem in this region using the solution of the lower problem as a starting point.
4. If the solution is less than the current upper bound, then update the bound

$$\text{if } obj^{upper} < UB \quad \text{then} \quad obj^{upper} \rightarrow UB \quad \mathbf{x}^* \rightarrow \mathbf{x}^{UB} \quad \boldsymbol{v}^* \rightarrow \boldsymbol{v}^{UB} \quad .$$

5. Go to Step 4.

This algorithmic procedure has been implemented in an extensive C program with an intuitive front end parser. All the necessary differentiations and generation of additional constraints are performed automatically. A link to the MINOPT optimization program (Schweiger and Floudas, 1998a) is used to perform the local optimizations and integrations. MINOPT itself has links to various local solvers. For these problems, SNOPT (Gill et al., 1997) was used as the local optimization routine, with the integrations being performed by DASOLV (Jarvis and Pantelides, 1992).

4 Computational Studies

In order to illustrate the theoretical and computational aspects of the proposed approach, four example problems will be presented. All example problems were solved on an HP J2240, using one CPU. Explicit formulations for each case study are provided in the handbook of test problems by Floudas et al. (1999).

4.1 Illustrative Example – Continued

This problem involves the same dynamic system used to illustrate the underestimation technique in Section 3.2.2.

$$\begin{aligned} \min_{u} \quad & -z(t_f)^2 \\ \text{s.t.} \quad & \dot{z} = u - z^3 \\ & z(t_o) = 9 \quad t \in [0,1] \\ & -5 \leq u \leq 5 \end{aligned} \tag{29}$$

For the sake of simplicity, a constant control profile will be used over the the entire time range. The problem needs to be reformulated such that the objective becomes an algebraic function. This is accomplished by introducing a point constraint at the final time, t_f, which relates the state, z, to an algebraic variable, x.

$$\begin{aligned} \min_{u,x} \quad & -x^2 \\ \text{s.t.} \quad & \dot{z} = u - z^2 \\ & z(t_o) = 9 \quad t \in [0,1] \\ & x - z(t_f) = 0 \\ & -5 \leq u \leq 5 \end{aligned} \tag{30}$$

In order to underestimate this problem, the equality point constraint needs to be split into two opposite signed inequalities. The final formulation becomes:

$$\begin{aligned} \min_{u\,x} \quad & -x^2 \\ \text{s.t.} \quad & \dot{z} = u - z^3 \\ & z(t_o) = 9 \quad t \in [0,1] \\ & x - z(t_f) \leq 0 \\ & -x + z(t_f) \leq 0 \\ & -5 \leq u \leq 5 \end{aligned} \tag{31}$$

This problem has two local solutions as shown in Table 1, one at each bound of the control variable.

Obj	u	x	Frequency
-2.9246	5	1.7101	59.5 %
-2.7902	-5	-1.6704	40.5 %

Table 1: Local solutions for the illustrative example. Frequency refers to the percentage of starting points resulting in the given solution (determined using 1000 random starting points).

In this formulation, not only will the dynamic system need to be underestimated, but also the objective function. The objective function is concave, and will be underestimated using a line segment from x^L to x^U. The full relaxed problem is:

$$\begin{aligned}
\min_{u\,x} \quad & -\left[\left(x^L\right)^2 + \frac{\left(x^U\right)^2 - \left(x^L\right)^2}{x^U - x^L}\left(x - x^L\right)\right] \qquad (32)\\
\text{s.t.} \quad & \dot{z} = u - z^3 \\
& z(t_o) = 9 \quad t \in [0,1] \\
& x - z(t_f) + \beta^-\left(u^U - u\right)\left(u^L - u\right) \leq 0 \\
& -x + z(t_f) + \beta^+\left(u^U - u\right)\left(u^L - u\right) \leq 0 \\
& -5 \leq u \leq 5 \quad -12 \leq x \leq 9
\end{aligned}$$

The minimum value of β^+ in the full region $u \in [-5, 5]$ was shown to be 0.1107 in Section 3.2.2. Using the same method, the minimum value of β^- required is 0.1450.

This problem was solved using both methods for calculating β values. The results are shown in Table 2. The values used for the constant β method are those given above. In the sampled case 100 points were used initially and at least 20 points at each other iteration. Since there is only one variable which participates in the dynamic system, there is no difference between the direct sampled approach, and the sampled approach with interval analysis. Branching was performed on both x and u using the first method described in Section 3.3 with a relative convergence tolerance of 0.1%. Two different CPU times are reported, the overall solution time, and the time required to perform the integrations to calculate the β values (this is included in the overall time).

The sampled approach has the advantage that at each iteration the value of β is updated. This is evident in the reduction of the required number of iterations. At the same time though, additional effort is required to perform the necessary integrations to update the values. The problem was also solved using bounds updating at each iteration. Table 3 shows results updating and branching on different sets of variables. Also results are shown with and without the use of the convexified objective function as a constraint in the bounds updating

β Calculation Method	Iterations	Total CPU sec.	Integration CPU sec.
Constant	22	8.96	–
Sampled	17	7.59	1.00

Table 2: Global Solution results for the illustrative example using different methods to calculate the β values.

Branching	Bounds Updating	Obj. Constraint	Iter.	CPU sec.	CPU sec/iter.
x, u	x, u	Yes	7	10.81	1.5
x, u	x	Yes	8	9.00	1.2
u	x	Yes	6	7.36	1.2
x, u	x, u	No	7	7.99	1.1
x, u	x	No	8	6.64	0.8
u	x	No	6	4.60	0.8

Table 3: Global Solution results for the illustrative example using different branching and variable bounds updating sets.

formulation. In each case, the first branching method with constant β values and a relative tolerance of 0.1% were used.

From the results it is clear that for this problem, the addition of the convexified objective function constraint does not reduce the number of iterations required to solve the problem. In addition it makes the bounds updating problems more difficult to solve, thus leading to a higher overall computational effort. Also, the best approach was to branch on the control variable, u, and update the bounds on x. Updating the bounds on the control variable, u, does nothing more than increase the computational effort required to solve the problem. If branching is performed on u and bounds updates on x, using a sampled method to calculate the value of β (100 points initially, and at least 20 points at each iteration), the global solution is obtained in 4 iterations and 4.32 CPU sec. Convergence to the solution is achieved with the difference between the upper and lower bounds being less than the optimality tolerance of the local solver, 1×10^{-6}.

4.2 Oil Shale Pyrolysis

This problem involves the determination of the optimal temperature profile in a PFR reactor. The reaction system under study is given by:

i	$\ln a_i$	b_i/R
1	8.86	10215.4
2	24.25	18820.5
3	23.67	17008.9
4	18.75	14190.8
5	20.70	15599.8

Table 4: Data for the oil shale pyrolysis example.

$$
\begin{aligned}
A_1 &\xrightarrow{k_1} A_2 \\
A_2 &\xrightarrow{k_2} A_3 \\
A_1 + A_2 &\xrightarrow{k_3} A_2 + A_2 \\
A_1 + A_2 &\xrightarrow{k_4} A_3 + A_2 \\
A_1 + A_2 &\xrightarrow{k_5} A_4 + A_2
\end{aligned}
$$

Only components A_1 and A_2 are included in the model. This example was studied by Luus (1990), Rosen and Luus (1992) and Carrasco and Banga (1997). The objective is to maximize the production of A_2:

$$
\begin{aligned}
\max_u \quad & z_2(t_f) \\
\text{s.t.} \quad & \dot{z}_1 = -k_1 z_1 - (k_3 + k_4 + k + 5)\, z_1 z_2 \\
& \dot{z}_2 = k_1 z_1 - k_2 z_2 + k_3 z_1 z_2 \\
& k_i = a_i \exp\left[\frac{-b_i/R}{u}\right] \qquad i = 1, \ldots, 5 \\
& \boldsymbol{z}_0 = [1,\, 0] \qquad t \in [0,\, t_f] \\
& 698.15 \le u \le 748.15
\end{aligned} \tag{33}
$$

where R is the ideal gas constant, and the parameters a_i and b_i in the rate expression are defined in Table 4. The control is rescaled between 0 and 1,

$$
u = 698.15 + 50\,\bar{u} \tag{34}
$$

and parameterized using a piecewise constant profile on 10 equally spaced elements. In the original problem, the final time is variable. For this example, we will consider the final time to be fixed at 10. The scaled, parameterized, and reformulated problem becomes:

$$
\begin{aligned}
\min_{\boldsymbol{v},\,x} \quad & -x && (35)\\
\text{s.t.} \quad & \dot{z}_1 \;=\; -k_1\, z_1 \;-\; (k_3 + k_4 + k + 5)\, z_1\, z_2 \\
& \dot{z}_2 \;=\; k_1\, z_1 \;-\; k_2\, z_2 \;+\; k_3\, z_1\, z_2 \\
& k_i \;=\; a_i \,\exp\left[\frac{-b_i/R}{698.15 + 50\,\bar{u}}\right] \qquad i = 1, \ldots, 5 \\
& \bar{u} \;-\; v_i \;=\; 0 \qquad \hat{t}_i \le t < \hat{t}_{i+1} \\
& x \;-\; z_2(t_f) \;\le\; 0 \\
& \boldsymbol{z}_0 \;=\; [1,\, 0] \qquad t \in [0,\, 10] \\
& \hat{\boldsymbol{t}} \;=\; [0,\, 1,\, 2,\, 3,\, 4,\, 5,\, 6,\, 7,\, 8,\, 9,\, 10] \\
& 0 \;\le\; \boldsymbol{v} \;\le\; 1
\end{aligned}
$$

Notice that only one inequality point constraint is required. Since the objective is to minimize the negative of the variable x, only the upper relaxed constraint will be active at the solution. This formulation has 8 known local solutions. The best five are listed in Table 5. The global solution is not the most prevalent solution and is found less than 40% of the time. Additionally, each of the local solutions has a completely different control profile.

Obj	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	Freq.
-0.35343	0	0	0	0.3464	1.00	0.0723	0	0	0	0	39.7%
-0.35327	0	0	0	0	0.1104	1.00	0.2346	0	0	0	41.8%
-0.35261	0.1105	0	0.1455	1.00	0.2649	0	0	0	0	0	6.6%
-0.35175	0	0	0	0	0	0.0538	1.00	0.2331	0	0	9.6%
-0.35134	0.3409	0.2738	1.00	0	0	0	0	0	0	0	0.8%

Table 5: Local solutions for the oil shale pyrolysis example. The frequency was found using 1000 random starting points.

This example was solved with updating different numbers of control parameter variables. In each case, the necessary β values were calculated using the sampling approach with 100 initial points and at least 10 points at each other iteration. The first branching method was used, the objective constraint was included in all bounds updating problems, and an absolute convergence tolerance of 1×10^{-4} was used. The results are presented in Table 6.

These results show how much improvement can be achieved by updating bounds on the control parameters at each iteration. Even though the CPU time per iteration has nearly tripled, the number of iterations required to achieve convergence is greatly decreased. Notice how much of the computational effort, over 50%, is needed to determine valid β values. More efficient second order calculations would lead to a great reduction in the required computational effort. It is important to note, that even though this problem has many local solutions, the upper bound identified at the root node of the branch and bound tree, in every case, was in fact the global solution. The rest of the time was spend proving global optimality.

Variables Updated	Iterations	Total CPU sec.	Integrations CPU sec.	CPU sec/iter.
none	353	13,051	11,988	37
5	157	10,782	6,697	69
all	87	7,408	3,777	85

Table 6: Global Solution results for the oil shale pyrolysis example updating different numbers of variables.

When the computational effort required to determine β values is removed from the problem, the results are much different. Table 7 shows the results using a constant value of β of 8.0×10^{-3}. The algorithmic parameters are the same as the previous results. In this case, the reduction in the number of required iterations to reach convergence is not enough to offset the extra effort needed to solve the bounds updating problem.

Variables Updated	Iterations	CPU sec.	CPU sec/iter.
none	473	1,209	2.6
2	351	4,721	13.4
5	276	5,253	19.0
all	219	6,045	27.6

Table 7: Global Solution results for the oil shale pyrolysis example updating different numbers of variables using a constant β value.

4.3 Singular Control Problem

This example represents a nonlinear singular control problem and appears in Luus (1990) and Rosen and Luus (1992). The problem formulation is:

$$\begin{aligned}
\min_{u(t)} \quad & \int_{t_0}^{t_f} \left[z_1^2 + z_2^2 + 0.0005 \left(z_2 + 16\,t - 8 - 0.1\,z_3\,u^2 \right)^2 \right] \, dt \\
\text{s.t.} \quad & \dot{z}_1 = z_2 \\
& \dot{z}_2 = -z_3\,u + 16\,t - 8 \\
& \dot{z}_3 = u \\
& \boldsymbol{z}_0 = [0,\, -1,\, -\sqrt{5}] \qquad t \in [0,\, 1] \\
& -4 \le u(t) \le 10
\end{aligned} \tag{36}$$

Reformulating the problem into the form given by (5) and parameterizing the control using a piecewise constant profile, results in:

$$
\begin{aligned}
\min_{\boldsymbol{v},\,x} \quad & x \\
\text{s.t.} \quad & \dot{z}_1 \;=\; z_2 \\
& \dot{z}_2 \;= -z_3\, u \;+\; 16\, t \;-\; 8 \\
& \dot{z}_3 \;=\; u \\
& \dot{z}_4 \;=\; z_1^2 + z_2^2 + 0.0005\, \left(z_2 + 16\, t - 8 - 0.1\, z_3\, u^2\right)^2 \\
& u \;-\; v_i \;=\; 0 \qquad \hat{t}_i \le t < \hat{t}_{i+1} \\
& -x \;+\; z_4(t_f) \;\le\; 0 \\
& \boldsymbol{z}_0 \;=\; [0,\, -1,\, -\sqrt{5},\, 0] \qquad t \in [0,\, 1] \\
& -4 \;\le\; \boldsymbol{v} \;\le\; 10
\end{aligned}
\tag{37}
$$

For this example, different numbers of equal sized control intervals were tested. Table 8 shows the local solutions for each number of intervals. In each case, two local solutions exist.

# of intervals	Global Sol.	Frequency	Local Sol.	Frequency
2	0.27711	39%	0.35175	58%
4	0.12374	70%	0.14252	30%
10	0.12012	98%	0.13569	2%

Table 8: Local solutions for the singular control problem using different numbers of control intervals.

Tests were run for each number of control intervals to determine the minimum value of β needed to generate a relaxed problem with only one solution. Also, 1000 points were used to sample the space to determine a value of β needed for convexity. These results are shown in Table 9. Notice how much smaller the β value needed to produce a lower problem with one solution is compared to the value needed for convexity. The convexity condition imposed on the relaxed problem which leads to convergence to the global solution can be relaxed to pseudo-convexity. Therefore, values of β less than that required for convexity can be used. In each case, the value needed for convexity is over 5 times the value needed for one solution.

# of intervals	β (one solution)	β (convexity)	Ratio
2	0.006510	0.03719	5.7
4	0.003545	0.02084	5.9
10	0.001525	0.00812	5.3

Table 9: β values needed for convexity and for a single solution for the singular control problem with different numbers of control intervals.

The two interval case was solved using all three methods for generating β variables. These results are shown in Table 10. The first branching method was used without updating any variable bounds at each iteration. A relative tolerance of 0.1% was used for convergence and for the sampling methods, 100 points were used initially and then at least 20 points at each other iteration. The value of β used for the constant run was that which was found to ensure convexity of the lower problem (0.03719). For the interval calculations, the Gerschgorin method was used when one β was determined, and the scaled Gerschgorin method was used to calculate one β per variable.

β Calculation Method	Iterations	Total CPU sec.	Integration CPU sec.
Constant	21	8.71	–
Sampled	12	15.55	9.41
Sampled/Interval (one β)	60	70.57	39.17
Sampled/Interval (β per variable)[1]	55	66.52	36.63
Sampled/Interval (β per variable)[2]	41	47.04	27.14

Table 10: Global Solution results for the singular control example with 2 control intervals using different methods to calculate the β values. [1] scaled Gerschgorin with scale factor of 1, [2] scaled Gerschgorin with scale factor of $v_i^U - v_i^L$.

For this example, due to the small size of the problem, it is better to use a constant value for β. The extra time required to perform the integrations to determine updated β's results in only a small reduction in the number of iterations. The interval approaches take more than 4 times the standard sampling method (without interval calculations). The interval methods are known to overestimate the required β value. The β value determined using interval calculations with the Gerschgorin method and sampling 100 points is 2.95. This is nearly 2 orders of magnitude larger than that which is necessary for convexity, thus the slower rate of convergence. As another exercise, the same conditions were used with bounds updating on both of the control parameters at each iteration using the objective function constraint. The results are shown in Table 11. The updating does result in a reduction in the number of required iterations to reach convergence, but not enough to offset the additional computational effort.

β Calculation Method	Iterations	Total CPU sec.	Integration CPU sec.
Constant	12	25.14	–
Sampled	8	23.43	9.30
Sampled/Interval (one β)	48	109.81	37.02
Sampled/Interval (β per variable)[1]	44	109.54	34.43
Sampled/Interval (β per variable)[2]	33	71.94	25.66

Table 11: Global Solution results for the singular control example with 2 control intervals using different methods to calculate the β values with bounds updating on both control parameters at each iteration. [1] used scaled Gerschgorin with scale factor of 1, [2] used scaled Gerschgorin with scale factor of $v_i^U - v_i^L$.

Using four control intervals, semi-constant β values given by equation (19) were used.

A relative convergence tolerance of 1% and the first branching option were selected. No bounds were updated at each iteration. Table 12 shows the results obtained by varying the n parameter in the expression. In each case, β_0 was set to the value determined to be needed for convexity of the original lower bounding problem (0.0208). The results, as expected, show for that a greater value of n results in slower convergence. Even with a value of $n = 1$ convergence to the global solution is reached, in fact, in each case the global solution is identified at the root node of the tree. The rest of the iterations are used to prove global optimality. The problem was also solved using the sampling approach with 100 initial points and at least 20 points at each other iteration. This required 187 iterations and 619.67 CPU sec. to achieve convergence. The integrations to calculation the β values required 443.48 CPU seconds.

n	Iterations	CPU sec.
1	21	28.81
3	111	120.05
5	221	210.28
10	460	397.16
15	644	556.25
20	782	627.53
constant	1608	1304.39

Table 12: Results using different n parameters in the calculation of β values for the singular control example with four control intervals.

Using ten control intervals, the problem was solved again with the semi-constant β values with $\beta_0 = 0.00812$ (the value needed for convexity of the original lower bounding problem) and $n = 2$. The solutions were obtained using each of the two branching methods with and without bounds updating. The bounds updating problems included the objective function constraint, and a relative convergence tolerance of 1% was used. These results are shown in Table 13. The second branching option offers slightly faster convergence to the global solution. Bounds updating reduces the number of iterations required (by nearly one half), but more than doubles the computational effort. The global solution was identified at the root node of the tree in each case.

Branch Method	Bounds Updating	Iterations	CPU sec.
1	no	1310	4,509
2	no	1076	3,625
1	yes	585	11,598
2	yes	570	11,391

Table 13: Results using different branching with and without bounds updating for the singular control example with ten control intervals.

4.4 Bifunctional Catalyst Example

This example concerns the optimization of a bifunctional catalyst in converting methylcyclopentane to benzene. The catalyst contains a hydrogenation component and an isomerization component. The objective is to determine the mixture of the two along the length of the reactor which maximizes the concentration of the desired product, A_7, in the reaction scheme given in Figure 4. This problem was studied by Luus et al. (1992), Bojokov and Luus (1993), and Luus and Bojkov (1994). The formulation for this problem using a piecewise constant control profile is:

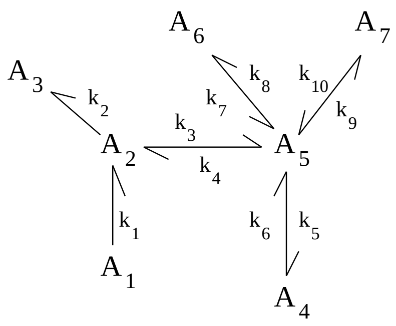

Figure 4: Reaction system of the bifunctional catalyst example.

$$
\begin{aligned}
\min_{\boldsymbol{v},x} \quad & -x \qquad\qquad (38)\\
\text{s.t.} \quad & \dot{z}_1 = -k_1 z_1 \\
& \dot{z}_2 = k_1 z_1 - (k_2 + k_3) z_2 + k_4 z_5 \\
& \dot{z}_3 = k_2 z_2 \\
& \dot{z}_4 = -k_6 z_4 + k_5 z_5 \\
& \dot{z}_5 = k_3 z_2 + k_6 z_4 - (k_4 + k_5 + k_8 + k_9) z_5 + k_7 z_6 + k_{10} z_7 \\
& \dot{z}_6 = k_8 z_5 - k_7 z_6 \\
& \dot{z}_7 = k_9 z_5 - k_{10} z_7 \\
& k_j = \left(c_{j,1} + c_{j,2} u + c_{j,3} u^2 + c_{j,4} u^3\right) v_r \qquad j = 1, \ldots, 10 \\
& u - v_i = 0 \qquad \hat{t}_i \le t < \hat{t}_{i+1} \\
& x - z_7(t_f) \le 0 \\
& \boldsymbol{z}_0 = [1,\ 0,\ 0,\ 0,\ 0,\ 0,\ 0,\ ] \qquad t \in [0,\ 1] \\
& \hat{\boldsymbol{t}} = [0,\ 0.1,\ 0.2,\ 0.3,\ 0.4,\ 0.5,\ 0.6,\ 0.7,\ 0.8,\ 0.9,\ 1] \\
& 0.60 \le \boldsymbol{v} \le 0.90
\end{aligned}
$$

were v_r is the total catalyst volume (2000 g h/mol), and the c matrix of constants for the reaction expressions is defined by:

$$c = \begin{pmatrix} 0.2918487\times10^{-2} & -0.8045787\times10^{-2} & 0.6749947\times10^{-2} & -0.1416647\times10^{-2} \\ 0.9509977\times10^{1} & -0.3500994\times10^{2} & 0.4283329\times10^{2} & -0.1733333\times10^{2} \\ 0.2682093\times10^{2} & -0.9556079\times10^{2} & 0.1130398\times10^{3} & -0.4429997\times10^{2} \\ 0.2087241\times10^{3} & -0.7198052\times10^{3} & 0.8277466\times10^{3} & -0.3166655\times10^{3} \\ 0.1350005\times10^{1} & -0.6850027\times10^{1} & 0.1216671\times10^{2} & -0.6666689\times10^{1} \\ 0.1921995\times10^{-1} & -0.7945320\times10^{-1} & 0.1105666 & -0.5033333\times10^{-1} \\ 0.1323596 & -0.4696255 & 0.5539323 & -0.2166664 \\ 0.7339981\times10^{1} & -0.2527328\times10^{2} & 0.2993329\times10^{2} & -0.1199999\times10^{2} \\ -0.3950534 & 0.1679353\times10^{1} & -0.1777829\times10^{1} & 0.4974987 \\ -0.2504665\times10^{-4} & 0.1005854\times10^{-1} & -0.1986696\times10^{-1} & 0.9833470\times10^{-2} \end{pmatrix}$$

Even though this example is relatively small (7 states and 1 control), it has been shown to exhibit a very large number of local minima. In fact over 300 unique local minima have been identified. Using 1000 random starting points, the global solution was only identified once. Figure 5 gives an idea of just how many local solutions this problem has.

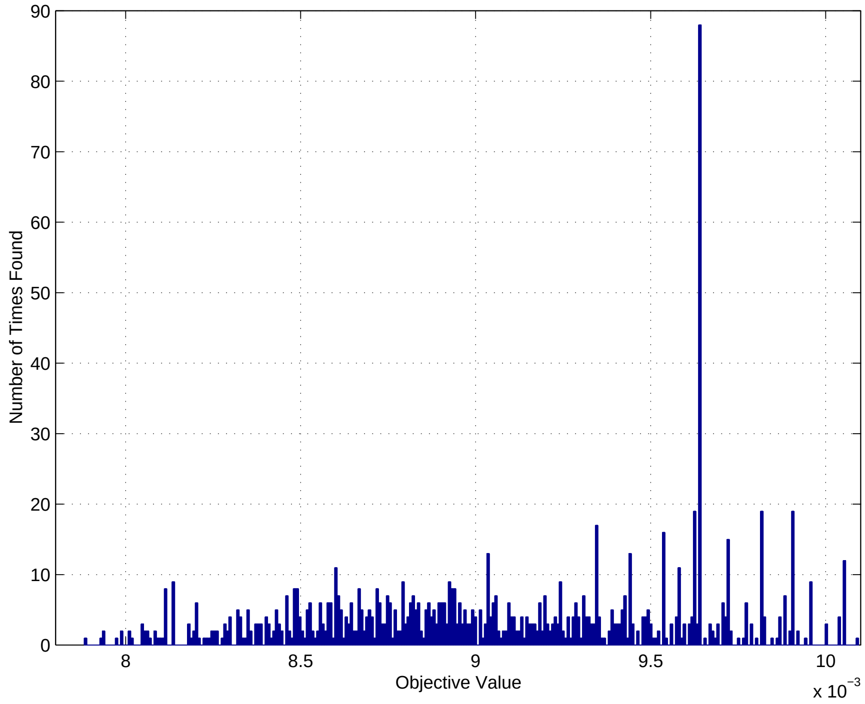

Figure 5: Local solutions and number of times found using 1000 random starting points for the bifunctional catalyst example.

Using 1000 sampled points, the value of β needed to ensure convexity of the initial lower bounding problem was determined to be 0.0698. The problem was solved using this value in expression (19) as β_0 with different values for n and different branching options. A relative convergence tolerance of 0.1% was used and no bounds updating was performed. The results are shown in Table 14. The global solution obtained has an objective value of 10.095×10^{-3}

with a control profile given in Figure 6. The results show some interesting trends. The second branching option allows for faster convergence, but fails more with lower n values and takes more iterations to identify the global solution. This method of branching when one β value is used, results in an ordered selection of branching variables. At each level of the tree the same variable is branched on in each node, starting with v_1 at the first level. Due to the structure of the optimal and local solutions, this results in quicker convergence, but a longer time to identify the global solution. Therefore, when a β value which is too small is used, convergence to a local solution is obtained. With the first branching method, the variable which has the most fractional solution (i.e., closest to the center of the region) is selected. This changes at each node of the tree and results in a more thorough search of the solution space. Therefore, the global solution is identified quicker and convergence is obtained even when invalid β values are used.

n	Branching	Iters.	CPU sec.	Global Solution Identified
2	1	failed (9.4975×10^{-3} in 41 iters)		
2	2	failed (10.0395×10^{-3} in 63 iters)		
3	1	220	2,985	iter. 67
3	2	failed (9.852×10^{-3} in 96 iters)		
4	1	412	4,778	iter. 93
4	2	failed (10.042×10^{-3} in 181 iters)		
5	1	620	6,341	iter. 169
5	2	373	5,675	iter. 268
6	1	820	8,517	iter. 68
6	2	653	7,691	iter. 329

Table 14: Results using different branching and values of the n parameter in the β expression for the bifunctional catalyst example. Failed shows that the algorithm converged to the local solution given in parentheses in the given number of iterations.

This example illustrates an interesting characteristic of the approach. The algorithm acts as a very effective search to determine the global solution. Initial points for the solution of the original problem are determined by solving a convex relaxation. This relaxation acts to smooth the nonconvex nature of the original formulation, thus supplying starting points which are relatively close to the global solution. Consider in this problem that using 1000 randomly chosen starting points only results in the global solution being identified once. Using the first branching method, the global solution was identified in at most 169 iterations and more often less than 100 iterations. Each iteration produces two different starting points for the solution of the upper problem. Therefore, it took no more that 200 points (on average) to find the global solution. This is 5 times better than simply choosing random points. This characteristic is also apparent in the other example problems. In each of those, even though multiple local minima exist, the global solution was always identified at the root node of the branch and bound tree.

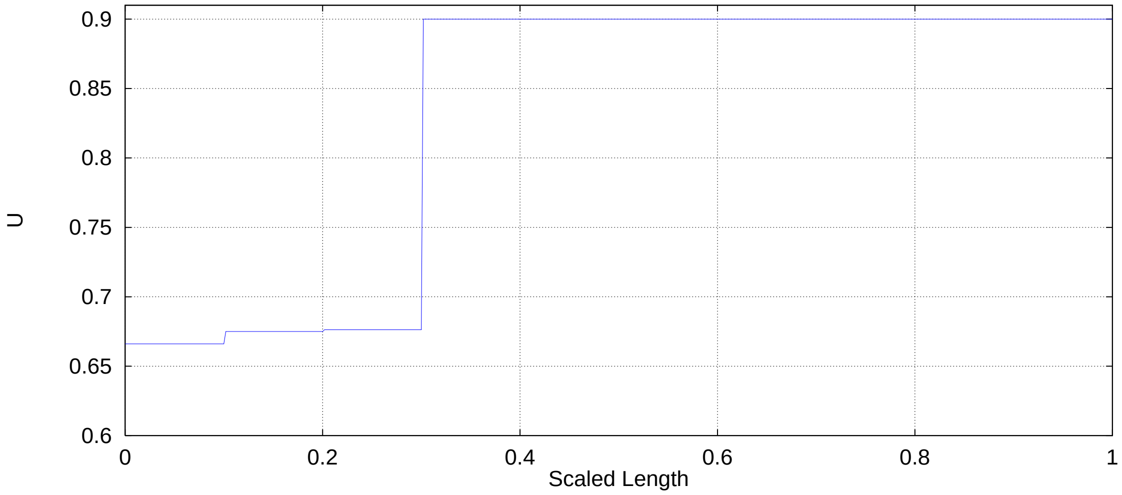

Figure 6: Globally Optimal control profile for the bifunctional catalyst example.

5 Conclusions

In this paper a deterministic global optimization approach has been presented to address nonlinear optimal control problems. The proposed method is constructed on a branch-and-bound framework in which the solution of a convex relaxation is solved to generate a valid lower bound on the global solution. The novelty in this approach is in the generation of a valid convex underestimation of the original nonconvex formulation. The values of the states at given time points are treated as twice continuously differentiable implicit functions of the algebraic variables in the formulation. This allows for the development of an underestimating function using information about the second order sensitivities of these states with respect to the control parameters. The theoretical conditions which make this assumption valid are presented and an example to illustrate the procedure is discussed. Four different optimal control problems which have been shown to exhibit multiple local minima were used to illustrate the theoretical and computational aspects of the proposed approach.

Acknowledgment

The authors gratefully acknowledge financial support from the National Science Foundation.

References

Adjiman C.S., Androulakis I.P., and Floudas C.A., 1998a, A global optimization method, αBB, for general twice–differentiable NLPs – II. Implementation and computational results. *Comp. Chem. Engng.* **22**, 1159–1178.

Adjiman C.S., Androulakis I.P., Floudas C.A., and Neumaier A., 1998b, A global optimization method, αBB, for general twice–differentiable NLPs – I. Theoretical advances. *Comp. Chem. Engng.* **22**, 1137–1158.

Adjiman C.S., Androulakis I.P., Maranas C.D., and Floudas C.A., 1996, A global optimization method, αBB, for process design. *Computers chem. Engng, Suppl.* **20**, S419–S424.

Adjiman C.S. and Floudas C.A., 1996, Rigorous convex underestimators for general twice–differentiable problems. *Journal of Global Optimization* **9**, 23–40.

Ali M.M., Storey C., and Torn A., 1997, Application of stochastic global optimization algorithms to pratical problems. *Journal of Optimization Theory and Applications* **95**, 545–563.

Androulakis I., Maranas C.D., and Floudas C.A., 1995, αBB : A global optimization method for general constrained nonconvex problems. *Journal of Global Optimization* **7**, 337–363.

Banga J.R., Alonso A.A., and Singh R.P., 1997, Stochastic dynamic optimization of batch and semicontinuous bioprocesses. *Biotechnol. Prog.* **13**, 326–335.

Banga J.R., Irizarry-Rivera R., and Seider W.D., 1998, Stochastic optimization for optimal and model-predictive control. *Comp. Chem. Engng.* **22**, 603–612.

Banga J.R. and Seider W.D., 1996, Global optimization of chemical processes using stochastic algorithms. In Floudas C.A. and Pardalos P.M. (eds.), *State of the Art in Global Optimization*, Nonconvex Optimization and its Applications, (pp. 563–583), Kluwer Academic Publishers.

Biegler L., 1984, Solution of dynamic optimization problems by successive quadratic programming and orthogonal collocation. *Comp. Chem. Engng.* **8**, 343–348.

Bojokov B. and Luus R., 1993, Evaluation of the parameters in iterative dynamic programming. *Can. J. Chem. Eng.* **71**, 451–459.

Brenan K.E., L. C.S., and Petzold L.R., 1996, *Numerical Solution of Initial-Value Problems in Differential-Algebraic Equations*. Classics in Applied Mathematics, Society for Industrial and Applied Mathematics, Philadelphia.

Caracotsios M. and Stewart W.E., 1985, Sensitivity analysis of initial value problems with mixed ODEs and algebraic equations. *Comp. Chem. Engng.* **9**, 359–365.

Carrasco E.F. and Banga J.R., 1997, Dynamic optimization of batch reactors using adaptive stochastic algorithms. *I&EC Res.* **36**, 2252–2261.

Carrasco E.F. and Banga J.R., 1998, A hybrid method for the optimal control of chemical processes. In *UKACC International Conference on CONTROL 98*.

Chen C. and Hwang C., 1990, Optimial control computation for differential-algebraic process systems with general constraints. *Chem. Eng. Comm.* **97**, 9–26.

Cuthrell J.E. and Biegler L.T., 1987, On the optimization of differential-algebraic process systems. *AIChE J.* **33**, 1257–1270.

Dadebo S.A. and Mcauley K.B., 1995, Dynamic optimization of constrainted chemical engineering problems using dynamic programming. *Comp. Chem. Engng.* **19**, 513–525.

Floudas C.A., 2000, *Deterministic Global Optimization: Theory, Algorithms and Applications*. Nonconvex Optimization and its Applications, Kluwer Academic Publishers.

Floudas C.A., Pardalos P.M., Adjiman C.S., Esposito W.R., Gümüş Z., Harding S., Klepeis J., Meyer C., and Schweiger C., 1999, *Handbook of Test Problems in Local and Global Optimization*. Kluwer Academic Publishers.

Gill P.E., Murray W., and Saunders M.A., 1997, *User's Guide for SNOPT 5.3: A Fortran Package for Large-Scale Nonlinear Programming*. Systems Optimization Laboratory, Department of Operations Research, Stanford University.

Goh C.J. and Teo K.L., 1988, Control parameterization: A unified approach to optimal control problems with general constraints. *Automatica* **24**, 3–18.

Hager W.W. and Pardalos P.M. (eds.), 1998, *Optimal Control: Theory, Algorithms, and Applications*. Applied Optimization, Kluwer Academic Publishers.

Jarvis R.B. and Pantelides C.C., 1992, *DASOLV: A Differential-Algebraic Equation Solver*. Center for Process Systems Engineering, Imperial College of Science, Technology, and Medicine, London, Version 1.2.

Litt F.X. and Delcommune J., 1984, Implimentation of spline approximations algorithms in numerical optimization. In Rauch H.E. (ed.), *Applications of Nonlinear Programming to Optimization and Control*, (pp. 121–128), Pregamon Press.

Logsdon J.S. and Biegler L.T., 1989, Accurate solution of differential-algebraic optimization problems. *I&EC Res.* **28**, 1628–1639.

Luus R., 1990, Optimal control by dynamic programming using systematic reduction in grid size. *Int. J. Control* **51**, 995–1013.

Luus R., 1993, Piecewise linear continuous optimal control by iterative dynamic programming. *I&EC Res.* **32**, 859–865.

Luus R. and Bojkov B., 1994, Global optimization of the bifunctional catalyst problem. *Can. J. Chem. Eng.* **72**, 160–163.

Luus R. and Cormack D., 1972, Multiplicity of solutions resulting from the use of variational methods in optimal control problems. *Can. J. Chem. Eng.* **50**, 309–311.

Luus R., Dittrich J., and Keil F.J., 1992, Multiplicity of solutions in the optimization of a bifunctional catalyst blend in a tubular reactor. *Can. J. Chem. Eng.* **70**, 780–785.

Luus R. and Hennessy D., 1999, Optimization of fed–batch reactors by the Luus–Jaakola optimization procedure. *I&EC Res.* **38**, 1948–1955.

Luus R. and Jaakola T.H.I., 1973, Optimization by direct search and systemic reduction of the size of search region. *AIChE J.* **19**, 760–766.

Luus R. and Rosen O., 1991, Application of dynamic programming to final state constrained optimal control problems. *I&EC Res.* **30**, 1525–1530.

Maranas C.D. and Floudas C., 1994, Global minimum potential energy conformations of small molecules. *Journal of Global Optimization* **4**, 135.

Mekarapiruk W. and Luus R., 1997, Optimal control of inequality state constrained systems. *I&EC Res.* **36**, 1686–1694.

Neuman C. and Sen A., 1973, A suboptimal control algorithm for constrained problems using cubic splines. *Automatica* **9**, 601–613.

Pontryagin L.S., 1962, *Ordinary Differential Equations*. Addison-Wesley Publishing Co., Translated from Russian by L. Kacinskas and W. B. Counts.

Renfro J.G., Morshedi A.M., and Asbjornsen O.A., 1987, Simultaneous optimization and solution of systems discribed by differential/algebraic equations. *Comp. Chem. Engng.* **11**, 503–507.

Rosen O. and Luus R., 1992, Global optimization approach to nonlinear optimal control. *Journal of Optimization Theory and Applications* **73**, 547–562.

Sargent R. and Pollard G.P., 1970, Off line computation of optimal controls for a plate distillation column. *Automatica* **6**, 59–76.

Sargent R. and Sullivan G., 1978, The development of an efficient optimial control package. In Stoer J. (ed.), *Proceedings of the 8th IFIP Conference on Optimization Techniques*, (pp. 158–168), Springer-Verlag.

Schweiger C.A. and Floudas C.A., 1998a, *MINOPT: A Modeling Language and Algorithmic Framework for Linear, Mixed-Integer, Nonlinear, Dynamic, and Mixed-Integer Nonlinear Optimization*. Princeton University.

Schweiger C.A. and Floudas C.A., 1998b, *MINOPT: A Software package for mixed-integer nonlinear optimization*. 3rd edn.

Smith E.M.B. and Pantelides C.C., 1996, Global optimization of general process models. In Grossmann I.E. (ed.), *Global optimization in engineering design*, (pp. 355–386), Kluwer Academic Publishers.

Strekalovsky A. and Vasiliev I., 1997, On global search for non-convex optimal control. In Bomze I.M. (ed.), *Developments in Global Optimization*, (pp. 121–133), Kluwer Academic Publishers.

Tieu D., Cluett W.R., and Penlidis A., 1995, A comparison of collocation methods for solving dynamic optimization problems. *Comp. Chem. Engng.* **19**, 375–381.

Tsang T.H., Himmelblau D.M., and Edgar T.F., 1975, Optimal control via collocation and non-linear programming. *Int. J. Control* **21**, 763–768.

Vassiliadis V.S., Sargent R.W.H., and Pantelides C.C., 1994a, Solution of a class of multistage dynamic optimization problems 1. Problems without path constraints. *I&EC Res.* **33**, 2111–2122.

Vassiliadis V.S., Sargent R.W.H., and Pantelides C.C., 1994b, Solution of a class of multistage dynamic optimization problems 2. Problems with path constraints. *I&EC Res.* **33**, 2123–2133.

Villadsen J. and Michelsen M.L., 1978, *Solution of Differential Equation Models by Polynomial Approximation*. Prentice-Hall, Inc.

Vlassenbroeck J., 1988, A chebyskev polynomial method for optimal control with state constraints. *Automatica* **24**, 499–506.

Wang F. and Chiou J., 1995, Computation of optimial control for integral and differential-algebraic systems. In *Proceedings of the 1995 American Control Conference*, vol. 5, (pp. 3933–3934).

Wang F. and Chiou J., 1997, Optimal control and optimal time location problems of differential-algebraic systems by differential evolution. *I&EC Res.* **36**, 5348–5357.