Global Optimization with Non-Analytical Constraints

Clifford A. Meyer and Christodoulos A. Floudas

Department of Chemical Engineering
Princeton University, Princeton, NJ 08544, USA
email: floudas@cronus.princeton.edu

and

Arnold Neumaier

Institut für Mathematematik, Universität Wien
Strudlhofgasse 4, A-1090 Wien, Austria
email: neum@cma.univie.ac.at

Abstract

This paper presents an approach for the global optimization of constrained nonlinear programming problems in which some of the constraints are *non-analytical* (non-factorable), defined by a computational model for which no explicit analytical representation is available. A two phase approach to the global optimization is considered. In the first phase the non-analytical functions are sampled and an interpolation function is constructed. In the second, the interpolants are used as surrogates in a deterministic global optimization algorithm. The interpolants are designed in such a way as to allow valid over- and under-estimation functions to be constructed so as to provide the global optimization algorithm with a guarantee of ϵ-global optimality.

1 Introduction

Chemical engineering models often cannot be defined by analytical expressions and need to be computed by numerical means. Systems of ordinary or partial differential equations are two frequently encountered models of this type. When such models participate in the constraints of an optimization problem a number of difficulties arise. Firstly, the functions these models describe are likely to be nonconvex and the globally optimal solution to the problem in which they participate can be missed by local optimization methods. Secondly, properties such as Lipschitz constants, on which deterministic global optimization algorithms may be founded, are usually not available. In addition, these functions may be expensive to evaluate.

Deterministic global optimization approaches without global function properties have been studied by several authors. Munack[1] applied a branch and bound approach using interval arithmetic to the optimization of a function with bound constraints. This branch and bound approach, proceeds by fitting a multiquadratic function to a transformation of the objective function in the current box and minimizing the multiquadratic surrogate. Esposito and Floudas[2] considered the application of the αBB algorithm[3] to problems in optimal control. In these studies the α parameter was estimated from a set of sampled Hessians in the function domain. Lipschitzian approaches to global optimization without a known Lipschitz constant have been studied by Wood and Zhang[4], Jones et al.[5], and Hansen et al.[6].

Several authors, including Sacks et al.[7], Jones et al.[8], and Gutmann[9] have considered the global optimization of unconstrained black-box functions, where each function evaluation is very expensive and no information besides these evaluations is available. These methods use a response surface approach in which the objective function is interpolated from a set of sample points, a utility function based on this interpolation is optimized, and the black-box function is sampled at the new optimum point before the process is repeated. Jones[10] provides an overview of global optimization methods based on this idea.

This paper introduces techniques to address global optimization problems which are constrained by differentiable "*non-analytical*" functions. The non-analytical functions we consider here are those which can be computed, along with their gradients, by a deterministic numerical algorithm. No further details of their analytic structure need be known.

The optimization problem may be written as follows:

$$\begin{array}{rcll} & & \min_{x \in [\underline{x}, \overline{x}] \subseteq \mathbb{R}^n} f_0(x) & \\ & & \text{subject to :} & \\ f_i(x) & \leq & 0 & \text{for all} \quad i = 1, \ldots, m \\ f_i(x) & = & 0 & \text{for all} \quad i = m+1, \ldots, p \\ \mathcal{F}_i(x_{A_i}) & = & x_{B_i} & \text{for all} \quad i = p+1, \ldots, q \end{array} \tag{1}$$

Where $A_i \subseteq \{1, \ldots, n\}$, $B_i \subseteq \{1, \ldots, n\}$, $A_i \bigcap B_i = \emptyset$ for all $i = p+1, \ldots, q$, and the notation x_{A_i} refers to the subvector of the x with indices in the set A_i. Each non-analytical function $\mathcal{F}_i : \mathbb{R}^{r_i} \to \mathbb{R}^{s_i}$ maps a vector $x_{A_i} \in \mathbb{R}^{s_i}$ to a vector $x_{B_i} \in \mathbb{R}^{s_i}$. The functions $f_i, i = 0, \ldots, p$ are algebraic functions with structure which can be used in a deterministic global optimization algorithm such as the αBB[11;12;13;14].

The solution of the optimization problem proceeds by three phases. In the *sampling phase* the functions $\mathcal{F}$ and their gradients are evaluated at a set of sample points. It is assumed that the function evaluations are cheap enough to enable a large enough number of points to be sampled for the construction of a sufficiently accurate interpolation. The interpolating functions are used as surrogates for the non-analytical functions in the *global optimization phase*. The replacement of the original non-analytical functions by surrogates serves two purposes: the surrogates are cheaper to evaluate and contain information on which a global optimization algorithm may be founded. In the final *local optimization phase*, the global optimum of the interpolation problem is used as a starting point for a local optimization of the original problem formulation. This is needed to correct for the approximations made in the interpolation phase.

Alfeld[15] provides an overview of the numerous approaches available for the interpolation of multivariate functions. Of these approaches, numerical studies indicate the radial basis functions to have superior accuracy[16]. Such interpolation functions, however, have poor interval extensions and are therefore not suitable for deterministic global optimization. In section 2 the interpolating function which we will refer to as the "blending" function is defined. These functions are designed in such a way as to have properties which can be used to generate rigorous over- and under-estimators. They are computationally efficient versions of blending functions discussed by Shepard[17], Franke and Nielsen[18], Franke[16], Barnhill et al.[19], and Farwig[20].

Section 3 discusses an interval arithmetic approach for generating linear under- and over-estimators for the blending function. An alternative "smooth blending" interpolation function is introduced in section 4 and the approximation properties of the respective functions are compared. In section 5, a cutting plane approach to generating linear under- and overestimators for the blending functions is described. Section 6 describes the algorithm used to sample the interpolation points. The global optimization algorithm is discussed in section 7, and numerical results are presented in section 8.

2 Blending functions

Let $\mathbf{x} := [\underline{x}, \overline{x}]$ and $\mathcal{F} : \mathbb{R}^r \to \mathbb{R}$ be a function for which function values and gradients are available from some subroutine call. We assume that we have a list of **design points** $x_l \in \mathbf{x}$ $(l \in L)$ with components x_{li} $(i = 1, \ldots, r)$ and associated function values and gradients

$$f_l = \mathcal{F}(x_l), \quad g_l = \nabla\mathcal{F}(x_l).$$

Using the local linear approximations

$$f_l(x) := f_l + g_l^T(x - x_l), \tag{2}$$

we define the **blending function**

$$\hat{f}(x) = \begin{cases} f_l & \text{if } x = x_l, \\ \sum\limits_{l\in L} w_l(x) f_l(x) / \sum\limits_{l\in L} w_l(x) & \text{if } x \neq x_l \text{ for all } l \in L, \end{cases} \tag{3}$$

where the $w_l(x)$ are twice continuously differentiable weight functions such that

$$0 \leq w_l(x) < \infty = w_l(x_l) \quad \text{for } x \in \mathbf{x}\backslash\{x_l\},$$

$$\sum_{l\in L} w_l(x) > 0 \quad \text{for } x \in \mathbf{x}.$$

These conditions guarantee that $\hat{f}(x)$ is twice continuously differentiable. A suitable set of weight functions is defined by

$$w_l(x) = \Big(\prod_{i=1}^{r} w_{li}(x_i)_+\Big)^3 \Big/ \sum_{i=1}^{r} \Big(\varepsilon + \Big(\frac{x_i - x_{li}}{\overline{x}_{li} - \underline{x}_{li}}\Big)^2\Big), \tag{4}$$

where $\varepsilon = 0$ (but in practice, ε is 10^{-12} to avoid division by zero),

$$w_{li}(\xi) = \frac{(\xi - \underline{x}_{li})(\overline{x}_{li} - \xi)/(x_{li} - \underline{x}_{li})(\overline{x}_{li} - x_{li})}{1 + (1 + \gamma_{li}(\xi - x_{li}))^2} \tag{5}$$

for suitable constants γ_{li}. The $\mathbf{x}_l = [\underline{x}_l, \overline{x}_l]$ form a family of boxes with $x_l \in \operatorname{int} \mathbf{x}_l$ whose interiors cover $\operatorname{int} \mathbf{x}$. $u_+ = \max(u, 0)$ denotes the positive part of a number u; the third power in (4) ensures that $w_l(x)$ is twice continuously differentiable. A good choice of the γ_{li} is

$$\gamma_{li} = \frac{\underline{x}_{li} + \overline{x}_{li} - 2x_{li}}{\varepsilon + (x_{li} - \underline{x}_{li})(\overline{x}_{li} - x_{li})} \tag{6}$$

for some tiny $\varepsilon > 0$, since we have the following result.

2.1 Proposition. *For the choice* (6) *with $\varepsilon = 0$, $w_{li}(\xi)$ is increasing for $\xi \in [\underline{x}_{li}, x_{li}]$, decreasing for $\xi \in [x_{li}, \overline{x}_{li}]$, and vanishes at the end points $\xi = \underline{x}_{li}, \overline{x}_{li}$.*

Proof. The vanishing of $w_{li}(\xi)$ at $\underline{x}_{li}$ and $\overline{x}_{li}$ is clear. Here we show the increasing, decreasing property. Differentiation and simplification gives

$$\frac{\partial}{\partial \xi} w_{li}(\xi) = \alpha_{li} \frac{(x_{li} - \xi)(2 + \gamma_{li}(\xi - x_{li}))}{(1 + (1 + \gamma_{li}(\xi - x_{li}))^2)^2},$$

where

$$\alpha_{li} = (x_{li} - \underline{x}_{li})^{-2} + (\overline{x}_{li} - x_{li})^{-2},$$

and we have $x_{li} - \underline{x}_{li} > 0$ and $\overline{x}_{li} - x_{li} > 0$ for $\xi \in [\underline{x}_{li}, \overline{x}_{li}]$.

The denominator and α_{li} are both positive. We now show that $2+\gamma_{li}(\xi - x_{li})$ is also positive:

$$\begin{array}{rcl} 2 + \gamma_{li}(\xi - x_{li}) & = & 2 + \frac{(\xi - x_{li})}{(x_{li} - \underline{x}_{li})} - \frac{(\xi - x_{li})}{(\overline{x}_{li} - x_{li})} \\ \frac{(\xi - x_{li})}{(x_{li} - \underline{x}_{li})} & > & -1 \quad \text{for all } \xi \in [\underline{x}_{li}, \overline{x}_{li}] \\ \frac{(\xi - x_{li})}{(\overline{x}_{li} - x_{li})} & > & -1 \quad \text{for all } \xi \in [\underline{x}_{li}, \overline{x}_{li}] \end{array}$$

The sign of the derivative it then the same as the sign of $(x_{li} - \xi)$, which gives us the required result. $\square$

As a consequence, $w_l(x)$ gets infinite weight at $x = x_l$, finite positive weight at $x \in \operatorname{int} \mathbf{x}_l \setminus \{x_l\}$, and zero weight otherwise.

For a given box $\mathbf{x}' \subseteq \mathbf{x}$, the monotonicity properties of $w_{li}(\xi)$ imply that it is straightforward to compute

$$\mathbf{w}_{li} = \{[\underline{w}_{li}(\xi), \overline{w}_{li}(\xi)] \mid \xi \in \mathbf{x}'_{li}\},$$

and hence an enclosure

$$w_l(x) \in \mathbf{w}_l = \Big(\prod_{i=1}^{d} (\mathbf{w}_{li})_+\Big)^3 \Big/ \sum_{i=1}^{d} \Big(\varepsilon + \Big(\frac{\mathbf{x}'_i - x_i}{\overline{x}_{li} - \underline{x}_{li}}\Big)^2\Big) \quad \text{for } x \in \mathbf{x}'. \tag{7}$$

Note that (for $\varepsilon = 0$) the denominators in (7) may contain 0, whence

$$0 \le \underline{w}_l \le \overline{w}_l \le \infty.$$

Example 1 As an illustration, consider the function defined by a single ordinary differential equation:

$$\begin{array}{rcl} \mathcal{F}(x_1, x_2) & = & z(t_f) \\ \frac{dz}{dt} & = & -x_1 x_2 z \\ z(t_0) = 1,\ t_0 & = & 0,\ t_f = 10 \end{array}$$

This will serve as a non-analytical function for illustrative purposes, although an analytical expression can be written, $\mathcal{F}(x_1, x_2) = e^{-10 x_1 x_2}$.

Here the evaluation of the blending function at an arbitrary point $x = (x_1, x_2) = (0.30, 0.37)$ is demonstated. Figure 1 shows the set of boxes which contain this point, and the position of the design points associated with these boxes. No other design point contributes to the evaluation of $\hat{f}(0.30, 0.37)$.

The design points associated with these boxes are labelled 1, 2, 3 and 4 and the data associated is summarized in Table 1.

The blending function evaluation data is summarized in Table 2. Note that, although all 4 points contribute to the function evaluation, the weights are such that the contributions of point 3 and 4 are smaller.

From this data the blending function can be evaluated:

$$\hat{f} = \sum_{l=1}^{4} \frac{w_l(x)}{\sum_{l=1}^{4} w_l(x)} f_l(x) = 0.3213 \tag{8}$$

The function $\mathcal{F}(x)$ equals 0.3296 at this point. Figure 2 illustrates, $w_l(x)/\sum_{l \in L} w_l(x)$, the fractional contributions of design points 1 and 2 to the blending function over the supports of their weight functions.

As the blending function may involve a large number of design points the implementation of the function needs to be efficient. As only a small fraction of the total set of design points contribute a nonzero quantity to $\hat{f}(x)$ for any given $x \in \mathbf{x}$, the key to the efficient evaluation of $\hat{f}(x)$ lies in the determination of the set of containing subregions $\{\mathbf{x}_l : x \in \mathbf{x}_l, l \in [1, L]\}$. This can be facilitated by storing the sample data in a tree structure. Each node of the tree corresponds to a subregion of the sample space, the root node is associated with $\mathbf{x}$, the leaf nodes with the subregions $\mathbf{x}_l, l \in [1, L]$. The nodes which contribute to the blending function at any point can be found by starting at the root node and descending via the nodes associated with subregions which contain $\mathbf{x}$.

3 Linear Under- and Overestimation by Interval Analysis

To determine linear under- and over-estimation functions for the blending function on a box $\mathbf{x}' \subseteq \mathbf{x}$, we observe that for a given $\tilde{x} \in \mathbf{x}$ Equations (2), (3) imply the representation:

$$\hat{f}(x) = \hat{f}(x, \tilde{x}) + \hat{g}(x)^T(x - \tilde{x}) \quad \text{for } x \in \mathbf{x}', \tag{9}$$

where

$$\hat{f}(x, \tilde{x}) = \begin{cases} f_l & \text{if } x = x_l, \\ \sum\limits_{l \in L} w_l(x) f_l(\tilde{x}) / \sum\limits_{l \in L} w_l(x), & \text{if } x \neq x_l \text{for all } l \in L, \end{cases} \tag{10}$$

$$\hat{g}(x) = \begin{cases} g_l & \text{if } x = x_l, \\ \sum\limits_{l \in L} w_l(x) g_l / \sum\limits_{l \in L} w_l(x), & \text{if } x \neq x_l \text{ for all } l \in L. \end{cases} \tag{11}$$

Both (10) and the components of (11) are weighted averages of known constants, hence we can determine interval enclosures using the following result.

3.1 Proposition. *Let the q_l be sorted such that $q_1 \leq q_2 \leq \ldots$, and*

$$q(w) = \sum_{l \in L} w_l q_l / \sum_{l \in L} w_l, \quad w \in \mathbf{w}, \quad \sum_{l \in L} w_l > 0, \tag{12}$$

where $0 \leq \underline{w}_l \leq \overline{w}_l \leq \infty$. Then the minimum of $q(w)$ is attained at $w = w^{\min}(k)$ for some k, where

$$w_l^{\min}(k) = \begin{cases} \overline{w}_l & \textit{if } l \leq k, \\ \underline{w}_l & \textit{if } l > k, \end{cases}$$

and the maximum of $q(w)$ is attained at $w = w^{\max}(k)$ for some k, where

$$w_l^{\max}(k) = \begin{cases} \underline{w}_l & \textit{if } l \leq k, \\ \overline{w}_l & \textit{if } l > k. \end{cases}$$

Proof. This follows from

$$\frac{\partial q(w)}{\partial w_j} = \frac{q_j(\sum\limits_{l\in L} w_l) - \sum\limits_{l\in L} w_l q_l}{(\sum\limits_{l\in L} w_l)^2} = \frac{q_j - q(w)}{\sum\limits_{l\in L} w_l}.$$

Suppose the minimum is attained at $w^{\min}$, then:

$$\frac{\partial q(w^{\min})}{\partial w_j} = \frac{q_j - q(w^{\min})}{\sum\limits_{l\in L} w_l^{\min}}.$$

If $q_j \leq q(w^{\min})$ then $\frac{\partial q(w^{\min})}{\partial w_j} \leq 0$ and the minimum is attained at $w_j = \overline{w}_j$.
If $q_j > q(w^{\min})$ then $\frac{\partial q(w^{\min})}{\partial w_j} > 0$ and the minimum is attained at $w_j = \underline{w}_j$.

Suppose the maximum is attained at $w^{\max}$, then:

$$\frac{\partial q(w^{\max})}{\partial w_j} = \frac{q_j - q(w^{\max})}{\sum\limits_{l\in L} w_l^{\max}}.$$

If $q_j \leq q(w^{\max})$ then $\frac{\partial q(w^{\max})}{\partial w_j} \leq 0$ and the maximum is attained at $w_j = \underline{w}_j$.
If $q_j > q(w^{\max})$ then $\frac{\partial q(w^{\max})}{\partial w_j} > 0$ and the maximum is attained at $w_j = \overline{w}_j$.

$\square$

Note that updating the quantities

$$Z_k^{\min} = \sum_{l\in L} w_l^{\min}(k) q_l, \quad N_k^{\min} = \sum_{l\in L} w_l^{\min}(k),$$

$$Z_k^{\max} = \sum_{l\in L} w_l^{\max}(k) q_l, \quad N_k^{\max} = \sum_{l\in L} w_l^{\max}(k)$$

when k changes to $k+1$ costs only $O(1)$ operations: If we introduce the numbers

$$d_k = \overline{w}_k - \underline{w}_k, \quad c_k = q_k d_k,$$

we have

$$Z_k^{\min} = Z_{k-1}^{\min} + c_k, \quad N_k^{\min} = N_{k-1}^{\min} + d_k,$$

$$Z_k^{\max} = Z_{k-1}^{\max} - c_k, \quad N_k^{\max} = N_{k-1}^{\max} - d_k.$$

Thus the range of $q(w)$ can be found with work proportional to the number of terms in the sum with $\overline{w}_l > 0$. If one or more of the weights are infinite, only a subset of the terms have to be calculated.

Applying this to (10) and (11) using (7), we find enclosures $\hat{\mathbf{f}}(\tilde{x})$ of $\hat{f}(x, \tilde{x})$ and $\hat{\mathbf{g}}$ of $\hat{g}(x)$, valid for all $x \in \mathbf{x}'$. Therefore, (9) implies the linear interval enclosure

$$\hat{f}(x) \in \hat{\mathbf{f}}(\tilde{x}) + \hat{\mathbf{g}}^T(x - \tilde{x}) \quad \text{for } x \in \mathbf{x}'. \tag{13}$$

If $\tilde{x} = \underline{x}'$, we find the linear enclosure

$$\underline{\hat{f}}(\underline{x}') + \underline{\hat{g}}^T(x - \underline{x}') \le \hat{f}(x) \le \overline{\hat{f}}(\underline{x}') + \overline{\hat{g}}^T(x - \underline{x}') \quad \text{for } x \in \mathbf{x}'. \tag{14}$$

Similar enclosures, but with different endpoints of $\hat{\mathbf{g}}$ in the linear parts, can be derived by choosing for $\tilde{x}$ any other corner of $\mathbf{x}'$; for instance, if $\tilde{x} = \overline{x}'$, we get

$$\underline{\hat{f}}(\overline{x}') + \overline{\hat{g}}^T(x - \overline{x}') \le \hat{f}(x) \le \overline{\hat{f}}(\underline{x}') + \underline{\hat{g}}^T(x - \overline{x}') \quad \text{for } x \in \mathbf{x}'.$$

Thus we get up to 2^r lower and 2^r upper bounds. Note that $\hat{\mathbf{g}}$ is independent of $\tilde{x}$, so that only the $\hat{\mathbf{f}}(\tilde{x})$ need to be computed when changing $\tilde{x}$. Note also that if only lower bounds are needed one can save the computation of the $\overline{\hat{f}}(\tilde{x})$; and similarly for upper bounds.

Remark. For improved approximation properties, it may be preferable to apply the blending to a residual function instead of directly to f. In particular, one may use the given data to construct a quadratic or rational approximation function and base the blending on the computed differences.

4 Blending Functions with Blending Derivatives

A smoother function can be derived by blending the derivatives prior to calculating the blending function. We base a local approximation around a design point x_l on the following relation:

$$f(x) \simeq f_l + \int_0^{\|x-x_l\|} \left(\sum_{l\in L} \frac{w_l g_l}{\sum_{l\in L} w_l} \right)^T \frac{(x-x_l)}{\|x-x_l\|} dt \tag{15}$$

Using the simplest approximation of this integral we define a local approximation as follows:

$$\breve{f}_l(x) := f_l + \tfrac{1}{2}(\hat{g}(x) + g_l)^T (x - x_l), \tag{16}$$

where the blending gradient $\hat{g}(x)$ is defined as follows:

$$\hat{g}(x) = \begin{cases} g_l & \text{if } x = x_l, \\ \sum_{l\in L} w_l(x) g_l / \sum\limits_{l\in L} w_l(x) & \text{if } x \neq x_l \text{ for all } l \in L, \end{cases} \tag{17}$$

The blending function based on these local approximations is defined as before:

$$\breve{f}(x) = \begin{cases} \breve{f}_l & \text{if } x = x_l, \\ \sum_{l\in L} w_l(x) \breve{f}_l(x) / \sum\limits_{l\in L} w_l(x) & \text{if } x \neq x_l \text{ for all } l \in L, \end{cases} \tag{18}$$

Example 1 - Continued. Returning to the illustrative function evaluation, we can use the same weight values determined there, to calculate the values of $\hat{g}(x)$ at $x = (0.30, 0.37)$, $\hat{g}(x) = (-1.2453, -1.0143)$. Using these values of $\hat{g}(x)$ and equation (16) we calculate $f_1(x), f_2(x), f_3(x), f_4(x)$ as 0.3282, 0.3305, 0.3325 ,and 0.3306 respectively. Using equation (18) with these values we calculate $\breve{f}(x) = 0.3295$, a very good approximation of $\mathcal{F}(x) = 0.3296$.

Example 2. Here we compare the interpolation properties of the respective blending functions on the six hump camelback function[21] $4x_1^2 - 2.1x_1^4 + \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4$, over the domain $([-2.0, 2.0], [-1.5, 1.5])$. The effects of the amount of overlap and sample density were examined by sampling the function over a uniform $m \times m$ grid, where m varied from 16 to 64. The space was partitioned into $m \times m$ regions, and the design points placed at the center of each of these box regions $\mathbf{x}_l'$. The accuracy of the blending functions was tested at points on a shifted grid, at point $\frac{1}{4}\underline{x}_l' + \frac{3}{4}\overline{x}_l'$ in each region $\mathbf{x}_l'$. The absolute error in the blending function $\hat{f}$ at a test point x is $|\hat{f}(x) - f(x)|$. The mean of the absolute errors, over the set of test points, is denoted $\overline{\delta}(\hat{f})$, and the maximum is denoted $\delta^{\max}(\hat{f})$. In these tests the overlap between boxes was determined by inflating each box $\mathbf{x}_l'$, by a parameter ϕ such that, $\underline{x} = \underline{x}' - \phi(\overline{x}' - \underline{x}')$, and $\overline{x} = \overline{x}' + \phi(\overline{x}' - \underline{x}')$. $\overline{\delta}$ values for the test points are tabulated in Table 3, and the $\delta^{\max}$ values are presented in Table 4.

In Table 3 we see that the approximation by $\hat{f}$ deteriorates with increasing ϕ, while $\breve{f}$ tends to improve with greater overlap, while the error of $\breve{f}$ is an order of magnitude less than that of $\hat{f}$.

Similar trends are evident in Table 4. The maximum error is an order of magnitude greater than the mean error owing to regions of high curvature towards the boundary of the domain.

From Proposition 4.1 we see that linear under- and over-estimation functions for the modified blending function can be derived using a similar approach as before.

4.1 Proposition. $\breve{f}(x) = \breve{f}(x, \tilde{x}) + \hat{g}^T(x - \tilde{x})$ *where*

$$\begin{aligned} \breve{f}(x, \tilde{x}) &= \sum_{l \in L} w_l(x) f_l(\tilde{x}) / \sum_{l \in L} w_l(x) \\ \hat{g}(x) &= \sum_{l \in L} w_l(x) g_l / \sum_{l \in L} w_l(x) \end{aligned}$$

Proof. This follows from

$$\begin{aligned} \breve{f}(x) &= \sum_{l \in L} w_l(x) \breve{f}_l(x) / \sum_{l \in L} w_l(x) \\ &= \sum_{l \in L} w_l(x) \left(f_l + \left(\tfrac{1}{2} g_l^T + \tfrac{1}{2} \frac{\sum_{l \in L} w_l(x) g_l^T}{\sum_{l \in L} w_l(x)} \right) ((x - \tilde{x}) + (\tilde{x} - x_l)) \right) / \sum_{l \in L} w_l(x) \\ &= \breve{f}(x, \tilde{x}) + \sum_{l \in L} w_l(x) \left(\left(\tfrac{1}{2} g_l^T + \tfrac{1}{2} \frac{\sum_{l \in L} w_l(x) g_l^T}{\sum_{l \in L} w_l(x)} \right) (x - \tilde{x}) \right) / \sum_{l \in L} w_l(x) \\ &= \breve{f}(x, \tilde{x}) + \left(\sum_{l \in L} w_l(x) \left(\tfrac{1}{2} g_l^T + \tfrac{1}{2} \frac{\sum_{l \in L} w_l(x) g_l^T}{\sum_{l \in L} w_l(x)} \right) / \sum_{l \in L} w_l(x) \right) (x - \tilde{x}) \end{aligned}$$

$$= \breve{f}(x,\tilde{x}) + \hat{g}^T(x-\tilde{x})$$

□

5 Cutting Plane Approaches

The linear under- and over-estimating functions derived through interval analysis can be relatively weak in regions containing a large number of sample points. An alternative to the interval analysis approach is to use a separation algorithm to construct valid linear inequalities. This is based on the observation that the graph of the blending function $\hat{f}(\cdot)$ is contained in the convex hull of the local linear approximations. This property does not apply to the smooth blending function $\breve{f}(\cdot)$.

5.1 Proposition. *Let $\mathbf{y}_A \subseteq \mathbf{x}_A$ be a hyper-rectangle. For any $x_A \in \mathbf{y}_A$ the following holds:*

$$\begin{aligned} (x_A, \hat{f}(x_A)) &\in \mathcal{C} \\ \mathcal{C} &:= \operatorname{conv}\left((x'_A, f_l(x'_A)) : \quad \textit{for all } x'_A \in \mathcal{E}(L')\right) \\ \mathcal{E}(L') &:= \operatorname{ext}(\mathbf{x}_{Al} \cap \mathbf{y}_A), l \in L' \\ L' &:= \{l \in L : \mathbf{x}_{Al} \cap \mathbf{y}_A \neq \emptyset\} \end{aligned} \tag{19}$$

where $\operatorname{conv}(\cdot)$ *denotes the convex hull, and* $\operatorname{ext}(\cdot)$ *denotes the set of extreme points.*

Proof. From the blending function definition, we see that the function is a convex combination of local linear approximations:

$$\hat{f}(x_A) = \sum_{l \in L'} \lambda_l f_l(x_A)$$

where $\sum_{l \in L'} \lambda_l = 1, \lambda_l \geq 0$. These linear approximations are in turn convex combinations of extreme points. □

Let $(\tilde{x}_A, \tilde{x}_B)$, $\tilde{x}_A \in \mathbb{R}^r, \tilde{x}_B \in \mathbb{R}^s$ be a point which does not satisfy the constraint $\hat{f}(x_A) = x_B$. A linear constraint, ${\pi_A}^T x_A + {\pi_B}^T x_B \leq \pi_0$, separating $(\tilde{x}_A, \tilde{x}_B)$ from the feasible set can be found provided $(\tilde{x}_A, \tilde{x}_B) \notin \mathcal{C}$. $\pi_A \in \mathbb{R}^r, \pi_B \in \mathbb{R}^s$ and $\pi_0 \in \mathbb{R}$ are found through the solution of a linear program of the form:

$$
\begin{gathered}
\max_{\pi_A, \pi_B, \pi_0} {\pi_A}^T \tilde{x}_A + {\pi_B}^T \tilde{x}_B - \pi_0 \\
\text{subject to} \\
{\pi_A}^T x_A + {\pi_B}^T f_l(x_A) - \pi_0 \leq 0 \qquad \text{for all } x_A \in \text{ext}(\mathbf{x}_{Al} \cap \mathbf{y}_A), l \in L \\
\|[\pi_A, \pi_B]\|_1 \leq 1
\end{gathered}
$$

The objective in this problem is to generate a linear constraint which maximizes the infeasibility of the point $(\tilde{x}_A, \tilde{x}_B)$ while ensuring that all points in $\mathcal{C}$ satisfy this constraint. The norm constraint is required to truncate the feasible cone, ensuring that the maximum objective function value is bounded.

Example 3.

In Figure 3 the solid line depicts a blending function representing the constraint $\hat{f}(x_1) = x_2$. This blending function is constructed from the design points (+) given in table 5.

The overlap between adjacent regions is 0.1 times the width of the design box regions, resulting in a function which is almost piecewise linear. The construction of a cut separating the point $(\tilde{x}_A, \tilde{x}_B) = (-0.25, -0.5)$, marked as $\Box$ in Figure 3, from the feasible region in a given interval is illustrated here. The interval under consideration, $x_1 \in [-0.55, 0.25]$, is demarcated by dotted lines. The points in the set $\{(x'_A, f_l(x'_A)) : x'_A \in \mathcal{E}(L')\}$ are marked as $\diamond$ and are given in Table 6. The separation problem is formulated explicitly as follows:

$$
\begin{array}{rcl}
\max_{\pi_1,\pi_2,\pi_0,\pi_1^+,\pi_2^+} & -0.25\pi_1 + -0.5\pi_2 - \pi_0 & \\
& \text{subject to} & \\
-0.55\pi_1 - 0.1800\pi_2 - \pi_0 & \leq & 0 \\
-0.49\pi_1 - 0.1368\pi_2 - \pi_0 & \leq & 0 \\
-0.51\pi_1 - 0.0992\pi_2 - \pi_0 & \leq & 0 \\
-0.29\pi_1 - 0.0288\pi_2 - \pi_0 & \leq & 0 \\
-0.31\pi_1 - 0.0168\pi_2 - \pi_0 & \leq & 0 \\
-0.09\pi_1 + 0.0008\pi_2 - \pi_0 & \leq & 0 \\
-0.11\pi_1 + 0.0000\pi_2 - \pi_0 & \leq & 0 \\
-0.11\pi_1 + 0.0000\pi_2 - \pi_0 & \leq & 0 \\
0.09\pi_1 - 0.0008\pi_2 - \pi_0 & \leq & 0 \\
0.25\pi_1 + 0.0120\pi_2 - \pi_0 & \leq & 0 \\
\pi_1^+ & \geq & \pi_1 \\
\pi_1^+ & \geq & -\pi_1 \\
\pi_2^+ & \geq & \pi_1 \\
\pi_2^+ & \geq & -\pi_1 \\
\pi_1^+ + \pi_2^+ & \leq & 1
\end{array}
$$

The variables π_i^+ have been introduced so as to represent the norm constraint as a set of linear inequalities. From the solution of this linear programming problem we find $\pi_1 = 0.1935$, $\pi_2 = -0.8065$ and $\pi_0 = 0.0387$. This gives us the linear inequality $0.1935x_1 - 0.8065x_2 + 0.0387 \leq 0$, which is represented as the dashed line in Figure 3.

While r is small, the size of the set $\{x_A \in \mathbb{R}^r : x_A \in \mathcal{E}(L')\}$ is $|L'|2^r$, where $|L'|$ may be of the order of 100 000. In addition to the large number of constraints, the problem faces numerical difficulties due to similarities between constraints and redundancy in the constraint set. For these reasons, the separation problem is solved via a sequence of smaller separation problems. The procedure can be summarized as follows:

Determine L'.
Let $\mathcal{E}'$ be a randomly chosen set of n points from $\mathcal{E}(L')$.
Solve the separation problem for π_A, π_B, π_0.
while ($\{x \in \mathcal{E}(L') : {\pi_A}^T x_A + {\pi_B}^T x_B - \pi_0 > 0\} \neq \emptyset$) {
Augment $\mathcal{E}'$ with points in $\{x_A \in \mathcal{E}(L') : {\pi_A}^T x_A + {\pi_B}^T f_l(x_A) - \pi_0 > 0\}$.

Solve the separation problem defined by $\mathcal{E}'$ for π, π_0.

}

Stronger cuts may be derived by reducing the size of the sets $\mathbf{x}_{Al}$. Cuts approximating the constraint $\hat{f}_j(x_{A_j}) = x_{B_j}$ can be used in this reduction during the generation of cuts representing $\hat{f}_i(x_{A_i}) = x_{B_i}$ whenever $A_i \bigcap (A_j \bigcup B_j) \neq \emptyset$. Let such a cut be expressed as:

$$\sum_{k \in A_j \bigcap P} \pi_k x_k + \sum_{k \in A_j \bigcap N} \pi_k x_k + \sum_{k \in B_j \bigcap P} \pi_k x_k + \sum_{k \in B_j \bigcap N} \pi_k x_k - \pi_0 \leq 0$$

where the indices are classified in sets $P = \{i : \pi_i > 0\}$ and $N = \{i : \pi_i < 0\}$. Choose a $k' \in A_i \bigcap (A_j \bigcup B_j)$. Then we can attempt to tighten the bounds on component k' of the box $\mathbf{x}_{Al}$. We denote the original bounds as $\underline{x}_{k'}, \overline{x}_{k'}$, and the updated bounds as as $\underline{x}'_{k'}, \overline{x}'_{k'}$. These bounds may be updated using the formula:

$$\begin{aligned}
\theta &= \tfrac{1}{\pi_{k'}}(\pi_0 - \sum_{k \in (A_j \bigcap P) \backslash k'} \pi_k \underline{x}_k - \sum_{k \in (A_j \bigcap N) \backslash k'} \pi_k \overline{x}_k - \\
&\quad \sum_{k \in (B_j \bigcap P) \backslash k'} \pi_k \underline{x}_k - \sum_{k \in (B_j \bigcap N) \backslash k'} \pi_k \overline{x}_k) \\
\overline{x}'_{k'} &= \min\{\overline{x}_{k'}, \theta\} \quad \text{if} \quad \pi_{k'} > 0 \\
\underline{x}'_{k'} &= \max\{\underline{x}_{k'}, \theta\} \quad \text{if} \quad \pi_{k'} < 0
\end{aligned}$$

For example let $B_j = \{1, 2, 3\}$ and $A_i = \{2, 3, 4\}$. Suppose we have the following valid cut: $\pi_1 x_1 + \pi_2 x_2 + \pi_3 x_3 \leq \pi_0$ and let $\pi_1, \pi_2, \pi_3 > 0$. A relaxation of this cut is $\pi_1 \underline{x}_1 + \pi_2 x_2 + \pi_3 \underline{x}_3 \leq \pi_0$. New upper and lower bounds on x_2 in the box $\mathbf{x}_{Al}$ may be derived using the relation:

$$\overline{x}'_2 = \min\{\overline{x}_2, \frac{\pi_0 - \pi_1 \underline{x}_1 - \pi_3 \underline{x}_3}{\pi_2}\}$$

6 Sampling Algorithm

Here we consider the procedure of choosing the design points to represent the non-analytical function $\mathcal{F} : \mathbb{R}^r \to \mathbb{R}^s$. An initial design is sampled over a grid, the resolution of which is determined by a parameter M. Resampling over a shifted grid is then carried out to assess the error in the blending function approximation. The algorithm for this procedure is summarized below. If the accuracy determined on the shifted grid is too low, points are resampled on a finer initial grid.

Step 1: Partition $\mathbf{x}$ into $L^0 = 2^{M \cdot s}$ subregions, and store the structure of this partitioning scheme. Set $L = L^0$.

Step 2: For every subregion $\mathbf{x}_l, l \in [1, L^0]$ evaluate f_l and g_l at the design point $x_l = \frac{1}{4}\underline{x_l} + \frac{3}{4}\overline{x_l}$.

Step 3: For every subregion $\mathbf{x}_l, l \in [1, L^0]$ evaluate $\hat{f}(x_{L+1})$, f_{L+1} and g_{L+1} at the design point $x_{L+1} = \frac{3}{4}\underline{x_l} + \frac{1}{4}\overline{x_l}$. Set the error estimate $\delta_l = \max\limits_{j=1,\ldots,r} |\hat{f}_j(x_{L+1}) - f_{j,L+1}|$. Set $J = \arg\max\limits_{j=1,\ldots,r} |\hat{f}_j(x_{L+1}) - f_{j,L+1}|$. Set $L = L+1$, and partition $\mathbf{x}_l$ into two equal size regions, $\mathbf{x}_l$ and $\mathbf{x}_L$, bisecting the axis k, where

$$k = \max_{i \in \{1,\ldots,s\}} \left(\left. \frac{\partial f_J}{\partial x_i} \right|_{x_{L+1}} - \left. \frac{\partial f_J}{\partial x_i} \right|_{x_l} \right) (x_{l,i} - x_{L+1,i})$$

such that $x_l \in \mathbf{x}_l$, $x_L \in \mathbf{x}_L$.

7 Branch and Cut Algorithm

The algorithm outlined in this section can determine an ϵ-global optimal solution of the following optimization problem where the non-analytical functions in problem 1 have been replaced by blending function surrogates:

$$\begin{array}{rcll} & & \min_{x \in [\underline{x}, \overline{x}] \subseteq \mathbb{R}^n} f_0(x) & \\ & & \text{subject to :} & \\ f_i(x) & \leq & 0 \quad \text{for all} & i = 1, \ldots, m \\ f_i(x) & = & 0 \quad \text{for all} & i = m+1, \ldots, p \\ \hat{f}_i(x_{A_i}) & = & x_{B_i} \quad \text{for all} & i = p+1, \ldots, q \end{array}$$

Step 0: Initialize
Choose an optimality tolerance ϵ.
Set upper bound on the global minimum objective function, $\overline{f}_0 \leftarrow +\infty$.
Set lower bound on the global minimum objective function, $\underline{f}_0 \leftarrow -\infty$.
Initialize iteration counter: $iter \leftarrow 0$.
Initialize list of subregions: $\mathcal{L} \leftarrow \emptyset$.
Read in blending function data.

Step 1: Compute Upper Bound
Solve NLP problem with bounds $\mathbf{x}$.
If a feasible solution $\tilde{x}$ with objective function $\tilde{f}_0$ was obtained set: $\tilde{f}_0 \rightarrow \overline{f}_0$, $\tilde{x} \rightarrow x^*$.

Step 2: Compute Lower Bound
Generate a relaxed lower bounding problem with convex feasible region using interval linear over- and underestimators to represent the blending functions.
Solve the NLP or LP lower bounding problem.
If the problem is feasible let the solution be $\tilde{x}$ and the optimal objective function value be $\tilde{f}_0$.
If the problem is infeasible, terminate, otherwise generate cuts by solving a series of separation problems, separating $\tilde{x}$ from the relaxation of the feasible region.
If new cuts have been determined, add these cuts to the lower bounding problem and repeat step 2, otherwise purge cuts which are not active at $\tilde{x}$, set $\tilde{x} \rightarrow x^{\mathcal{R}}$, $\tilde{f}_0 \rightarrow f_0^{\mathcal{R}}$, $\mathcal{L} \rightarrow \mathcal{L} \bigcup \mathcal{R}$ and continue to step 3.

Step 3: Check for Convergence
If $\overline{f}_0 - \underline{f}_0 \leq \epsilon$ terminate; otherwise set $iter \rightarrow iter + 1$.

Step 4: Select Subregion
If $\mathcal{L} = \emptyset$ the problem is infeasible; otherwise select the region with the best lower bound: $\mathcal{R} \leftarrow \arg\min_{i \in \mathcal{L}} f_0^i$, and delete this region from the list $\mathcal{L} \rightarrow \mathcal{L} \backslash \mathcal{R}$.

Step 5: Branch
Choose a variable on which to branch and partition $\mathcal{R}$ into subregions $\mathcal{R}_1$ and $\mathcal{R}_2$.
Solve the upper bounding problem in each subregion.

Step 6: Update Bounds on Variables

In each subregion $\mathcal{R}_i$, set $\mathbf{x} \leftarrow \mathcal{R}_i$ and solve a variable bound update problem for a given variable.
Let the solution to this problem be $\tilde{x}$.
If the problem is infeasible, terminate; otherwise update $\mathcal{R}_i$ and $\mathbf{x}$ then generate cuts by solving a series of separation problems, separating $\tilde{x}$ from the relaxation of the feasible region.
If new cuts have been determined, add these cuts to the bound update problem and repeat step 6, otherwise purge cuts which are not active at $\tilde{x}$. Continue to step 7 once the bound updating for all selected bound update variables has been completed in $\mathcal{R}_1$ and $\mathcal{R}_2$.

Step 7: Compute Lower Bounds for Subregions
Set $\mathbf{x} \leftarrow \mathcal{R}_i$
Generate a relaxed lower bounding problem with convex feasible region using interval linear over- and underestimators to represent the blending functions.
Solve the NLP or LP lower bounding problem.
If the problem is feasible let the solution be $\tilde{x}$ and the optimal objective function value be $\tilde{f}_0$.
If the problem is infeasible, terminate; otherwise generate cuts by solving a series of separation problems, separating $\tilde{x}$ from the relaxation of the feasible region.
If new cuts have been determined, add these cuts to the lower bounding problem and repeat step 7; otherwise purge cuts which are not active at $\tilde{x}$, set $\tilde{x} \to x^{\mathcal{R}}$, $\tilde{f}_0 \to f_0^{\mathcal{R}}$, $\mathcal{L} \to \mathcal{L} \bigcup \mathcal{R}$ and continue to step 3.

The algorithm was implemented in C as an extension of the αBB code of Adjiman et al.[13]. CPLEX 7.0[22] was used to solve the LPs, and MINOS 5.4 [23] was used for the NLPs.

8 Computational Studies

8.1 Example 4

$$
\begin{array}{rcl}
\min\limits_{x_1 \in [0.1,1.0], x_2 \in [0.1,1.0]} & x_1 + x_2 & \\
\text{subject to :} & & \\
x_3 & = & 0.5 \\
\mathcal{F}(x) - x_3 & = & 0
\end{array}
$$

where $\mathcal{F}(x)$ is defined as the system of equations:

$$
\begin{array}{rcl}
\mathcal{F}(x_1, x_2) & = & z(t_f) \\
\dfrac{dz}{dt} & = & -x_1 x_2 z \\
z(t_0) = 1,\ t_0 & = & 0,\ t_f = 10
\end{array}
$$

We first assess the influence of the inclusion of cutting planes on the convergence rate.

In Figure 4 the convergence rates of the global optimization algorithm with and without cutting planes are compared. In these runs both lower bounding and bound tightening problems were solved on each iteration, and for the run with cutting planes, cuts where generated for every problem.

The cutting planes produce much stronger lower bounds than the algorithm using only those constraints derived through interval analysis. The effect of the density of sampling points on the convergence rate is illustrated in Figure 5.

The plots represent the gap between lower and upper bounds, $\overline{f} - \underline{f}$, on the global minimum objective function value for runs with 512, 2048 and 8192 evenly distributed design points. We see that the greater the number of the design points the smaller the gap, due to tighter lower bounds, and the fewer the number of iterations required for convergence.

Figure 6 shows the feasible region and optimal point in the x_1, x_2 domain. Cutting planes generated on the first iteration are also represented on this figure. Note the small gap between the optimal point and the cutting planes. The cutting planes are valid for the convex hull, $\mathcal{C}$, defined in Equation 19 which is an extension of the convex hull of the feasible region. Tighter cuts are generated on subsequent iterations once points supporting the earlier cuts have been eliminated from $\mathcal{E}(L')$. The cuts are stronger when there are more design points as $\mathcal{C}$ is then a more accurate representation of the convex hull of the feasible region. Useful as the cuts are, it may not always be possible to close the gap between upper and lower bounds using only cutting planes. For this reason the constraints derived using interval methods are essential for convergence.

The solutions and CPU times obtained for the runs without cutting planes are given in Table 7, the convergence criterion for this example was set as $\overline{f} - \underline{f} < 5 \times 10^{-4}$, CPU times are on an HP C-160. Table 8 shows results when cutting planes are used. The global solution to the problem is $x_1 = x_2 = 0.2633$, $f = 0.5266$.

From Tables 7 and 8 we observe that although the number of iterations required for convergence is substantially less for the runs where cuts were generated, the time needed to generate these cuts, in this case, undermines these savings. The number of different solutions is due to the occurrence of kinks in the blending function representation which do not exist in the original problem. Note that the optimal solution points and objective function values for the interpolated problems are very similar to each other indicating that they are likely to provide good starting points for the local optimization phase, which indeed leads to the global optimum.

8.2 Oilshale Pyrolysis Problem

This problem involves the optimization of the temperature profile of a plug flow reactor in order to maximize of the production rate of a selected species. This example has been studied by Luus[24], Luus and Rosen[25], Carrasco and Banga[26] and Esposito and Floudas[2]. The system of chemical reactions occuring are assumed to be as follows:

$$A_1 \xrightarrow{k_1} A_2$$

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

The following optimal control problem formulation is that of maximizing the valuable product A_2, subject to bounds on the temperature and the system dynamics.

$$\max_{u(t)} z_2(t_f)$$

$$\begin{array}{rcl} & & \text{subject to} \\ \dot{z}_1 & = & -k_1 z_1 - (k_3 + k_4 + k_5) z_1 z_2 \\ \dot{z}_2 & = & k_1 z_1 - k_2 z_2 + k_3 z_1 z_2 \\ & & \\ & & \text{where} \\ k_i & = & a_i \exp\left(\frac{-b_i/R}{u}\right) \quad i = 1, \ldots, 5 \\ z(0) & = & (1, 0) \end{array}$$

$$698.15 \leq u(t) \leq 748.15 \qquad \text{for all } t \in [0, t_f]$$

In this formulation $z_i(t)$ is the concentration of component i at time t; the final time t_f is set to 10 and the temperature is denoted as $u(t)$. Data for the parameters a and b is provided in Table 9. In this study we examine the cases where the control variable $u(t)$ is parametrized using a piecewise constant profile on 2 or 3 equally spaced time intervals. The non-analytical function in this system is defined as follows:

$$\begin{array}{rcl} & & \mathcal{F} : (\, z_1^{i-1},\, z_2^{i-1},\, u^i \,) \rightarrow (\, z_1^i,\, z_2^i \,) \\ (\, z_1^{i-1},\, z_2^{i-1} \,) & = & (\, z_1(0),\, z_2(0) \,) \\ (\, z_1^i,\, z_2^i \,) & = & (\, z_1(t'_f),\, z_2(t'_f) \,) \\ \dot{z}_1 & = & -k_1(u^i) z_1 - (k_3(u^i) + k_4(u^i) + k_5(u^i)) z_1 z_2 \\ \dot{z}_2 & = & k_1(u^i) z_1 - k_2(u^i) z_2 + k_3(u^i) z_1 z_2 \end{array}$$

where $t'_f = t_f/N$, and N is the number of time intervals in the parametrized problem.

This function was sampled on 8192 evenly distributed points over the domain $(z_1^{i-1}, z_2^{i-1}), u^i) \in ([0,1],[0,1],[698.15, 748.15])$ using the numerical integration package DASPK[27]. Using $\mathcal{F}$ the parametrized problem can be expressed as follows:

$$\max_{u^i, z^i} z_2^N$$

$$\begin{array}{rcll} & & \text{subject to} & \\ \mathcal{F}(z_1^{i-1}, z_2^{i-1}, u^i) & = & (z_1^i, z_2^i) & \text{for all } i = 1, \ldots, N \\ z_1^0 & = & 1 & \\ z_2^0 & = & 0. & \end{array}$$

Table 10 shows the results for convergence to an absolution tolerance of 10^{-3}.

Using the results in Table 10 as starting points, the solutions to the original problems were determined using the package MINOPT[28] with the local NLP solver MINOS 5.4[23]. These solutions are presented in Table 11.

In Figure 7 the progress of the upper and lower bounds for the algorithms with cutting planes generation is compared to the progress of the algorithm using constraints generated by interval arithmetic only. In this figure the cutting planes can be seen to greatly improve the rate of convergence. With the cutting planes the initial lower bound is much tighter and the initial rate of convergence is rapid.

8.3 Nonlinear CSTR Problem

The continuous stirred tank reactor system with nonlinear dynamics is an example of optimal control problem with multiple solutions, first studied by Aris and Amundson[29] and subsequently by Lapidus and Luus[30] and Mekarapiruk and Luus[31].

$$\min_{u(t)} z_3(t_f)$$

$$\begin{aligned}
& \text{subject to} \\
\dot{z}_1 &= g(z) - (2+u)(z_1+0.25) \\
\dot{z}_2 &= 0.5 - z_2 - g(z) \\
\dot{z}_3 &= z_1^2 + z_2^2 + 0.1u^2 \\
z_1(t_f) &= 0 \\
z_2(t_f) &= 0 \\
& \text{where} \\
g(z) &= (z_2+0.5)\exp(\tfrac{25z_1}{z_1+2}) \\
z(0) &= (0.09, 0.09, 0.00)
\end{aligned}$$

$$-1 \le u(t) \le 5 \quad \text{for all } t \in [0, t_f = 0.78]$$

The non-analytical function in this system is defined as follows:

$$\begin{aligned}
\mathcal{F} : (\, z_1^{i-1},\, z_2^{i-1},\, u^i\,) &\rightarrow (\, z_1^i,\, z_2^i,\, z_3^i\,) \\
(\, z_1^{i-1},\, z_2^{i-1},\, 0\,) &= (\, z_1(0),\, z_2(0),\, z_3(0)\,) \\
(\, z_1^i,\, z_2^i,\, z_3^i\,) &= (\, z_1(t_f'),\, z_2(t_f'),\, z_3(t_f')\,) \\
\dot{z}_1 &= g(z) - (2+u^i)(z_1+0.25) \\
\dot{z}_2 &= 0.5 - z_2 - g(z) \\
\dot{z}_3 &= (z_1)^2 + (z_2)^2 + 0.1(u^i)^2
\end{aligned}$$

where $t_f' = t_f/N$, and N is the number of time intervals in the parametrized problem.

This function was sampled on 8192 evenly distributed points over the domain $(z_1(t_i), z_2(t_i), u_i) \in ([-0.2, 0.2], [-0.2, 0.2], [-1, 5])$. The parametrized problem, with a penalty term in the objective function, can be expressed as follows:

$$\min_{u^i, z^i} 10((z_1^N)^2 + (z_2^N)^2) + \sum_{i=1}^{N} z_3^i$$

subject to

$$
\begin{aligned}
\mathcal{F}(z_1^{i-1}, z_2^{i-1}, u^i) &= (z_1^i, z_2^i, z_3^i) \qquad \text{for all } i = 1, \ldots, N \\
z_1^0 &= 0.09 \\
z_2^0 &= 0.09
\end{aligned}
$$

Results for convergence to an absolute tolerance of 5×10^{-3} are presented in Table 12. Using these values as a starting point, the local solver obtained the solutions in Table 13. Using the solution from the problem with three time intervals as a starting position, a local solution of the original problem with 21 time intervals was obtained using MINOPT[28]. The control policy shown in Figure 8 was obtained. An optimal objective function value of $z_3(t_f) = 0.14433$ was found and the constraints on the final states were within an acceptable tolerance $z_1(t_f) = 5.57 \times 10^{-8}$ and $z_2(t_f) = -6.74 \times 10^{-8}$.

9 Conclusions

Numerical results indicate that the interpolation by blending functions when used to represent the non-analytical functions in a global optimization algorithm can yield solutions which approximate the global optima. Through the use of rigorous linear over- and under-estimators the proposed global optimization algorithm can guarantee the optimality of the solution for the interpolated problem. Discrepancies between the global optimal solution for the original problem and that using the blending interpolation functions depend on the sample density of the blending function design points. For non-analytical functions involving a small number of variables, the use of blending function interpolation can be a useful strategy. The number of design points required to interpolate grows rapidly as the number of variables increases, limiting the method to functions of low dimensionality. The cutting plane approach was found to greatly enhance the convergence rate of the proposed optimization algorithm. There is scope for the use of similar, cutting from design points, techniques for a broad class of global optimization problems. Such methods may be useful for generating a convex representation of analytical functions if the computational cost of cutting plane generation can be offset by increased convergence rates.

Acknowledgment

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

References

(1) Munack, H. On Global Optimization Using Interval Arithmetic. *Computing* **1992**, *48*, 319.

(2) Esposito, W. R.; Floudas, C. A. Deterministic Global Optimization in Nonlinear Optimal Control Problems. *Journal of Global Optimization* **2000**, *17*, 97.

(3) Adjiman, C. S.; Floudas, C. A. Rigorous Convex Underestimators for General Twice–Differentiable Problems. *J. of Glob. Opt.* **1996**, *9*, 23.

(4) Wood, G. R.; Zhang, B. P. Estimation of the Lipschitz constant of a function. *Journal of Global Optimization* **1996**, *13*, 91.

(5) Jones, D. R.; Perttunen, C. D.; Stuckman, B. E. Lipschitzian Optimization without the Lipschitz Constant. *J. of Optimization Theory and Applications* **1993**, *79*, 157.

(6) Hansen, P.; Jaumard, B.; Lu, S. On Using Estimates of Lipschitz-constants in global optimization. *J. of Optimization Theory and Applications* **1992**, *75*, 195.

(7) Sacks, J.; Welch, W. J.; Mitchell, T. J.; Wynn, H. P. Design and Analysis of Computer Experiments. *Statistical Science* **1989**, *4*, 409.

(8) Jones, D. R.; Schonlau, M.; Welch, W. J. Efficient Global Optimization of Expensive Black-Box Functions. *J. of Glob. Opt.* **1998**, *13*, 455.

(9) Gutmann, H.-M. A Radial Basis Function Method for Global Optimization. *J. of Glob. Opt.* **2001**, *19*, 201.

(10) Jones, D. R. A Taxonomy of Global Optimization Methods Based on Response Surfaces. *J. of Glob. Opt.* **2001**, *21*, 345.

(11) Adjiman, C. S.; Androulakis, I. P.; Maranas, C. D.; Floudas, C. A. A Global Optimisation Method, αBB, for Process Design. *Computers Chem. Engng, Suppl.* **1996**, *20*, S419.

(12) Adjiman, C. S.; Dallwig, S.; Floudas, C. A.; Neumaier, A. A Global Optimization Method, αBB, for General Twice–Differentiable NLPs – I. Theoretical Advances. *Computers Chem. Engng.* **1998a**, *22* (9), 1137.

(13) Adjiman, C. S.; Androulakis, I. P.; Floudas, C. A. A Global Optimization Method, αBB, for General Twice–Differentiable NLPs – II. Implementation and Computational Results. *Computers chem. Engng.* **1998b**, *22* (9), 1159.

(14) Floudas, C. *Deterministic Global Optimization: Theory, Algorithms and Applications*; Kluwer Academic Publishers, 2000.

(15) Alfeld, P. Scattered data interpolation in three or more variables. In *Mathematical Methods in Computer Aided Geometric Design*; Lyche, T.; Schumaker, L., Eds.; Academic Press, 1989, p 1–34.

(16) Franke, R. Scattered Data Interpolation: Tests of Some Methods. *Mathematics of Computation* **1982**, *38*, 181.

(17) Shepard, D. A Two Dimensional Interpolation Function for Irregular Spaced Data. *Proc. ACM Nat. Conf.* **1965**, , 517.

(18) Franke, R.; Nielsen, G. Smooth Interpolation for Large Sets of Scattered Data. *Int. Journal for Numerical Methods in Engineering* **1980**, *15*, 1691.

(19) Barnhill, R.; Dube, R.; Little, F. Properties of Shepard's Surfaces. *Rocky Mountain J. of Math.* **1983**, *13*, 365.

(20) Farwig, R. Rate of Convergence of Shepard's Global Interpolation Formula. *Math. Computation* **1986**, *46*, 577.

(21) Dixon, L.; Szegö, G. Towards Global Optimization. In *Proceedings of a Workshop at the University of Cagliari, Italy* North Holland, 1975 .

(22) *ILOG CPLEX User's Manual*; ILOG Inc Mountain View, CA 2000.

(23) Murtagh, B. A.; Saunders, M. A. *MINOS 5.4 User's Guide*; Systems Optimization Laboratory Dept. of Operations Research, Stanford University, CA. 1983.

(24) Luus, R. Optimal Control by Dynamic Programming using Systematic Reduction in Grid Size. *Int J. Control* **1990**, *51*, 995.

(25) Luus, R.; Rosen, O. Application of Dynamic Programming to Final State Constrained Optimal Control Problems. *I&EC Res.* **1991**, *30*, 1525.

(26) Carrasco, E.; Banga, J. Dynamic Optimization of Batch Reactors Using Adaptive Stochastic Algorithms. *I&EC Res.* **1997**, *36*, 2252.

(27) Maly, T.; Petzold, L. Numerical methods and software for sensitivity analysis of differential-algebraic systems. *Applied Numerical Mathematics* **1996**, *20*, 57.

(28) Schweiger, C. A.; Rojnuckarin, A.; Floudas, C. A. *MINOPT : A Software Package for Mixed–Integer Nonlinear Optimization, User's Guide*; Computer–Aided Systems Laboratory Dept. of Chemical Engineering, Princeton University, NJ 1997.

(29) Aris, R.; Amundson, N. R. An Analysis of Chemical Reactor Stability and Control. *Chemical Engineering Science* **1958**, *7*, 123.

(30) Lapidus, L.; Luus, R. *Optimal Control of Engineering Processes*; Blaisdell, 1967.

(31) Mekarapiruk, W.; Luus, R. Optimal Control of Final State Constrained Systems. *The Canadian Journal of Chemical Engineering* **1997**, *75*, 806.

Figure Captions

Figure 1: Illustration Design Point and Box Position

Figure 2: Fractional Contributions, $w_l(x)/\sum_{l \in L} w_l(x)$, of Design Points 1 and 2

Figure 3: Illustration of Cut Generation

Figure 4: Convergence Rate with (+) and without ($\diamond$) Cutting Planes

Figure 5: Convergence Rate for Cutting Plane Approach with $\diamond$: 512; + 2048 and $\Box$ 8192 Design Points

Figure 6: Illustrative Problem: Feasible Set, Cutting Planes and the Global Solution

Figure 7: Progress of $\overline{f}$ and $\underline{f}$ with and without Cutting Planes

Tables

l	x_{l1}	$\underline{x}_{l1}$	$\overline{x}_{l1}$	x_{l2}	$\underline{x}_{l2}$	$\overline{x}_{l2}$	$\mathcal{F}(x)$	$\frac{\partial \mathcal{F}}{\partial x_1}$	$\frac{\partial \mathcal{F}}{\partial x_2}$
1	0.2875	0.2125	0.3625	0.2875	0.1750	0.4750	0.4376	-1.2580	-1.2580
2	0.2875	0.2125	0.3625	0.4375	0.3250	0.6250	0.2843	-1.2437	-0.8173
3	0.3625	0.2875	0.4375	0.5125	0.3250	0.6250	0.1560	-0.7996	-0.5656
4	0.3625	0.2875	0.4375	0.3625	0.1750	0.4750	0.2687	-0.9741	-0.9741

Table 1: Data Required for Blending Function Evaluation at $x = (0.3, 0.37)$

l	$f_l(x)$	γ_{l1}	γ_{l2}	$w_{l1}(x)$	$w_{l2}(x)$	$w_l(x)$	$\frac{w_l(x)}{\sum_{l=1}^{4} w_l(x)}$
1	0.3180	0	3.5556	0.4861	0.3632	0.06664	0.4412
2	0.3239	0	3.5556	0.4861	0.3448	0.08181	0.5417
3	0.3223	0	-3.5556	0.1528	0.4984	0.00254	0.0168
4	0.2866	0	-3.5556	0.1528	0.1664	0.00004	0.0003

Table 2: Illustration of Blending Function Evaluation

m	16		32		64	
ϕ	$\overline{\delta}(\hat{f})$	$\overline{\delta}(\breve{f})$	$\overline{\delta}(\hat{f})$	$\overline{\delta}(\breve{f})$	$\overline{\delta}(\hat{f})$	$\overline{\delta}(\breve{f})$
0.01	0.338	0.338	0.044	0.044	0.007	0.007
0.5	0.413	0.135	0.053	0.021	0.009	0.004
1.0	0.611	0.115	0.079	0.015	0.013	0.003
2.0	1.568	0.126	0.189	0.013	0.031	0.002

Table 3: Mean error estimates for blending function approximations

m	16		32		64	
ϕ	$\delta^{\max}(\hat{f})$	$\delta^{\max}(\breve{f})$	$\delta^{\max}(\hat{f})$	$\delta^{\max}(\breve{f})$	$\delta^{\max}(\hat{f})$	$\delta^{\max}(\breve{f})$
0.01	4.119	4.119	0.846	0.846	0.189	0.189
0.5	4.684	0.702	0.948	0.124	0.209	0.025
1.0	6.171	0.943	1.157	0.150	0.242	0.029
2.0	11.735	1.006	1.906	0.136	0.364	0.026

Table 4: Maximum error estimates for blending function approximations

x_1	-1	-0.8	-0.6	-0.4	-0.2	0	0.2	0.4	0.6	0.8	1
x_2	-1	-0.512	-0.216	-0.064	-0.008	0	0.008	0.064	0.216	0.512	1

Table 5: Example 3 List of Design Points (+)

x_1	-0.55	-0.51	-0.49	-0.31	-0.29	-0.11	-0.09	0.09	0.11	0.25
x_2	-0.18	-0.0992	-0.1368	-0.0168	-0.0288	0.00	0.0008	-0.0008	0.00	0.012

Table 6: Example 3 List of Cut Generation Points ($\diamond$)

Design Points	x_1	x_2	x_3	f	Iterations	CPU (s)
512	0.2898	0.2361	0.5000	0.5260	55	0.82
2048	0.2503	0.2761	0.5000	0.5264	47	1.49
8192	0.2588	0.2677	0.5000	0.5265	47	6.50

Table 7: Solutions without Cutting Planes to Blending Function Representation

Design Points	x_1	x_2	x_3	f	Iterations	CPU (s)
512	0.2652	0.2607	0.5000	0.5260	14	1.59
2048	0.2611	0.2603	0.5000	0.5264	10	5.15
8192	0.2588	0.2677	0.5000	0.5265	6	12.14

Table 8: Solutions Using Cutting Planes to Blending Function Representation

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 9: Data for the oil shale pyrolysis example.

N	u^1	u^2	u^3	z_2^N
2	713.78	698.15	-	0.3592
3	698.93	722.96	698.15	0.3614

Table 10: Results for Interpolated Oilshale Pyrolysis Problem

N	u^1	u^2	u^3	$z_2(t_f)$
2	719.71	698.15	-	0.3510
3	698.15	724.80	698.15	0.3517

Table 11: Solutions for Oilshale Pyrolysis Problem

N	u^1	u^2	u^3	$z_1^N \times 10^2$	$z_2^N \times 10^2$	$\sum_{i=1}^{N} z_3^i$	obj
2	2.385	-0.625	-	-2.385	-2.980	0.242	0.256
3	2.375	0.329	-0.250	-1.593	-2.327	0.148	0.156

Table 12: Results for Interpolated Nonlinear CSTR Problem

N	u^1	u^2	u^3	$z_1(t_f) \times 10^2$	$z_2(t_f) \times 10^2$	$z_3(t_f)$	obj
2	2.404	-0.482	-	-3.042	-3.053	0.246	0.264
3	2.615	0.197	-0.243	-1.664	-1.993	0.185	0.192

Table 13: Solutions for Nonlinear CSTR Problem

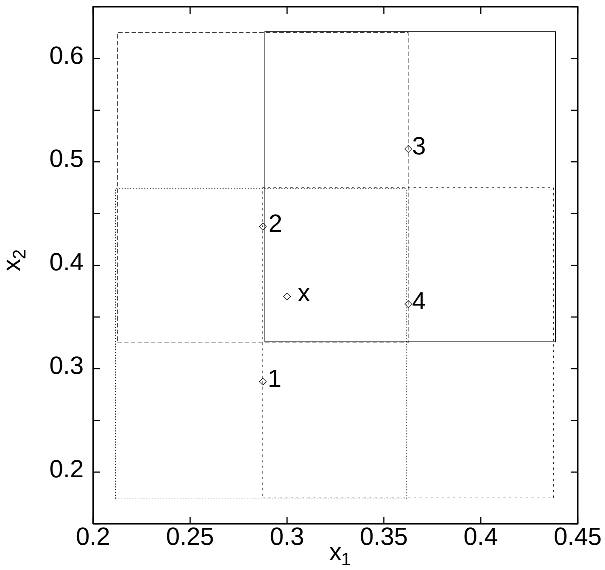

Figure 1: Illustration Design Point and Box Positions

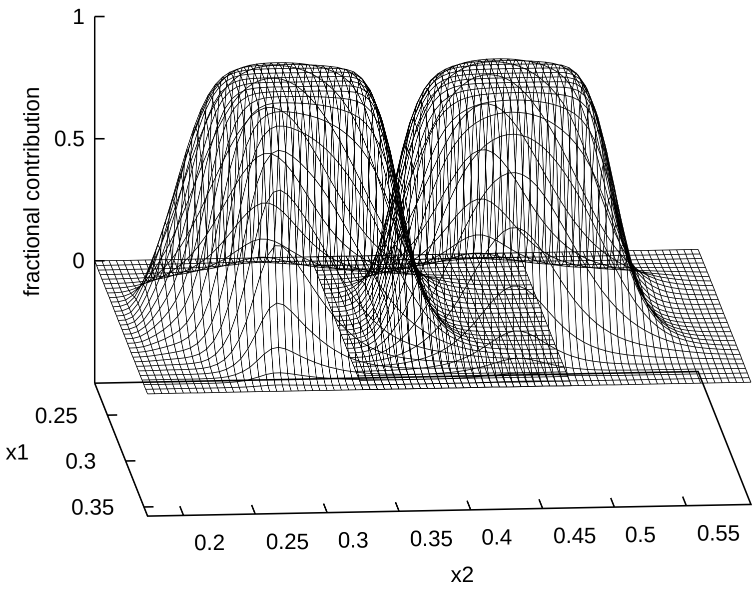

Figure 2: Fractional Contributions, $w_l(x)/\sum_{l\in L} w_l(x)$, of Design Points 1 and 2

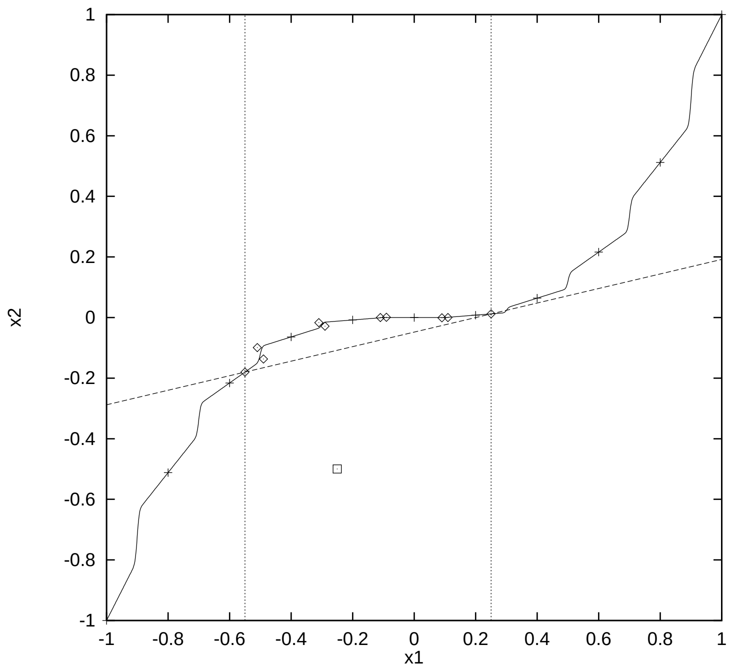

Figure 3: Illustration of Cut Generation

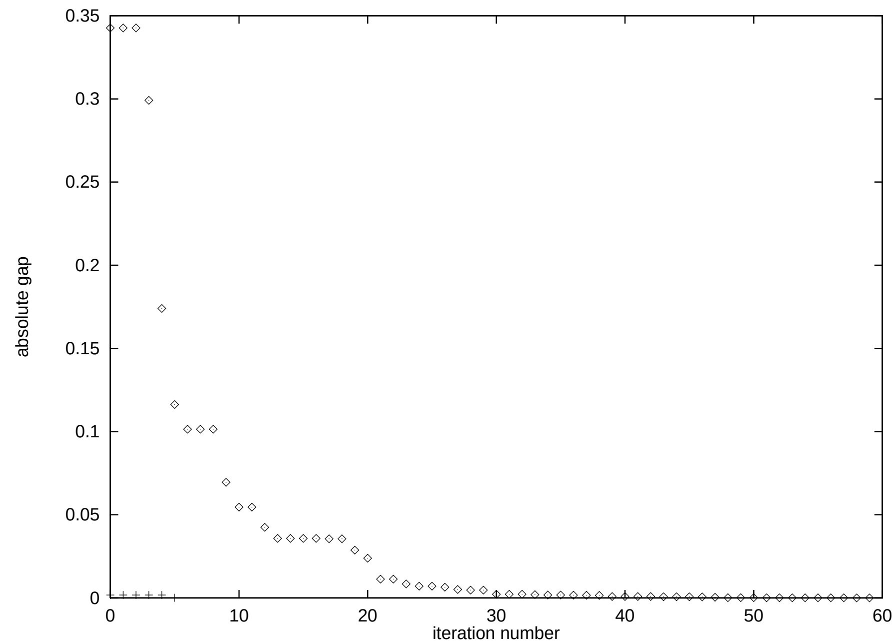

Figure 4: Convergence Rate with (+) and without (⋄) Cutting Planes

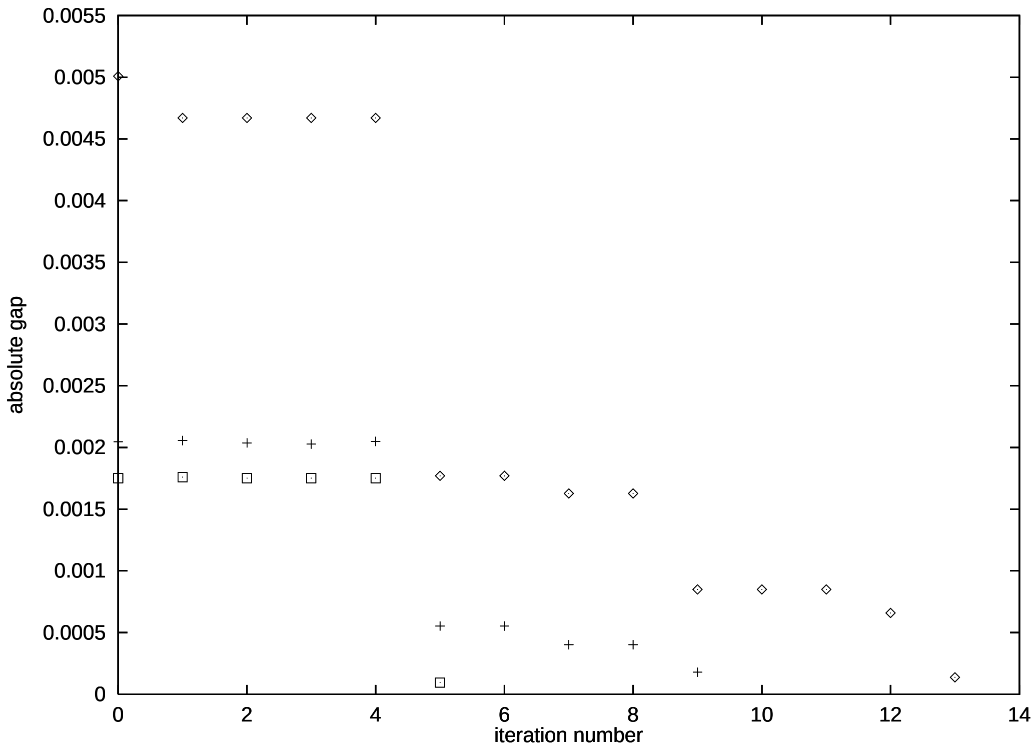

Figure 5: Convergence Rate for Cutting Plane Approach with $\diamond$: 512; + 2048 and $\square$ 8192 Design Points

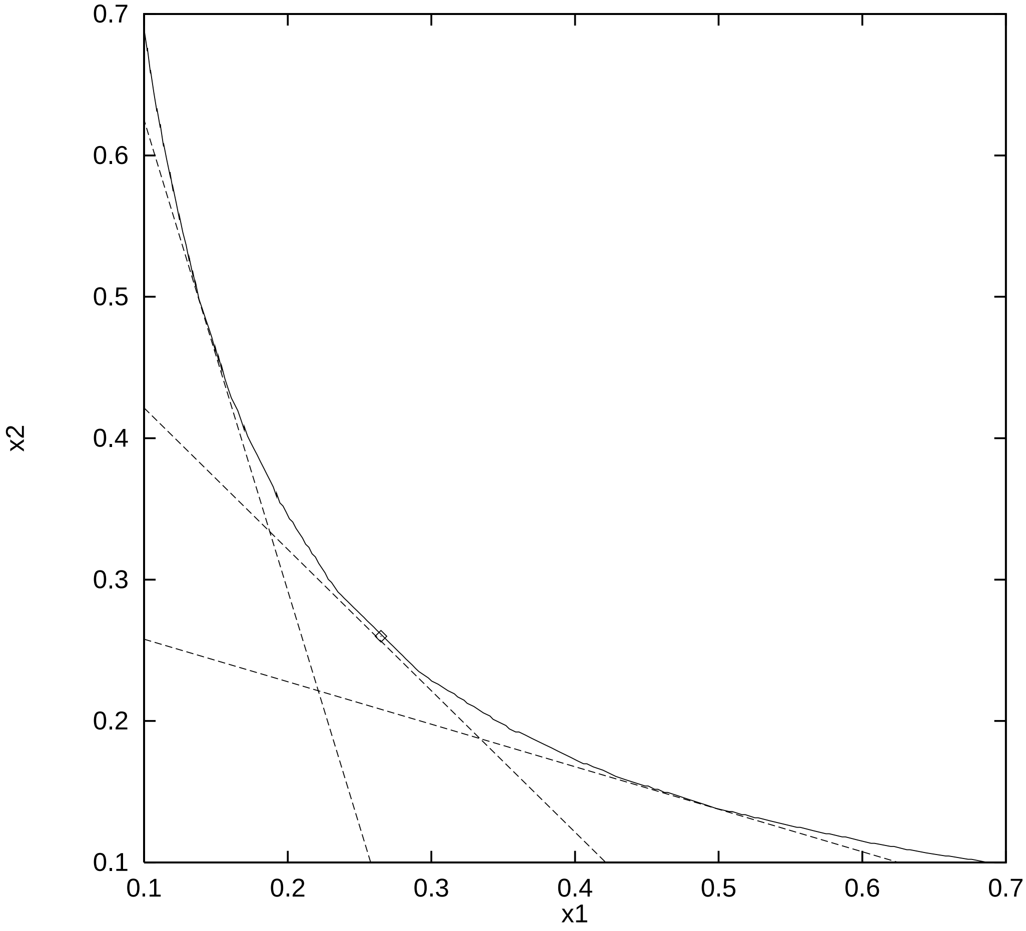

Figure 6: Illustrative Problem: Feasible Set, Cutting Planes and the Global Solution

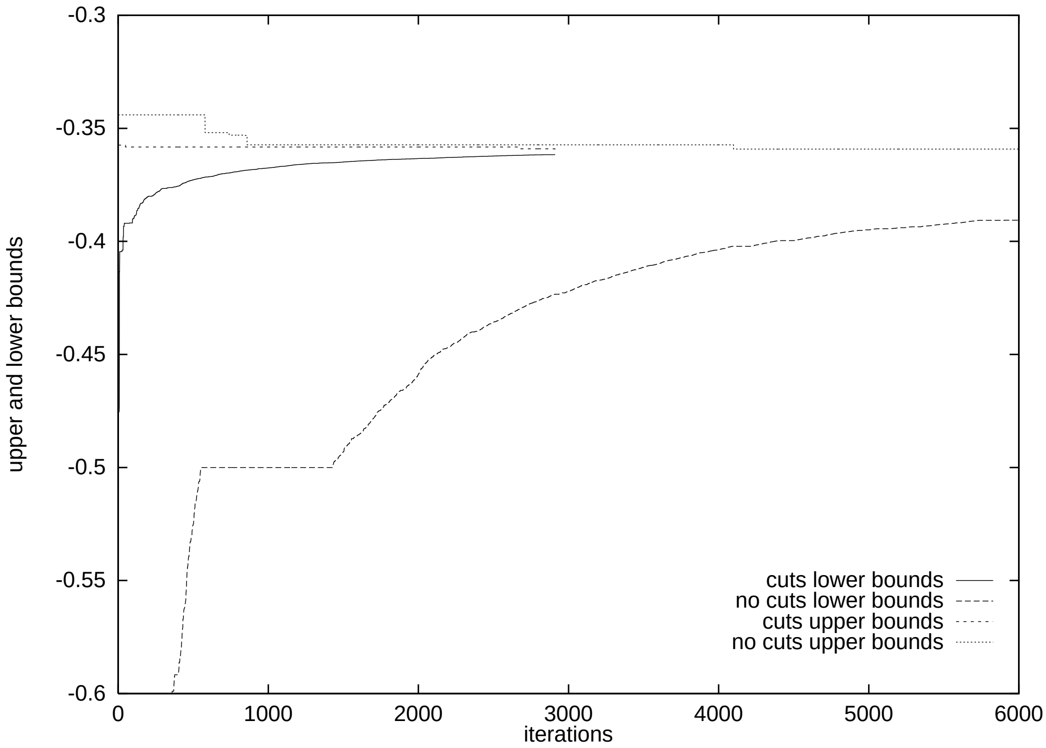

Figure 7: Progress of $\overline{f}$ and $\underline{f}$ with and without Cutting Planes

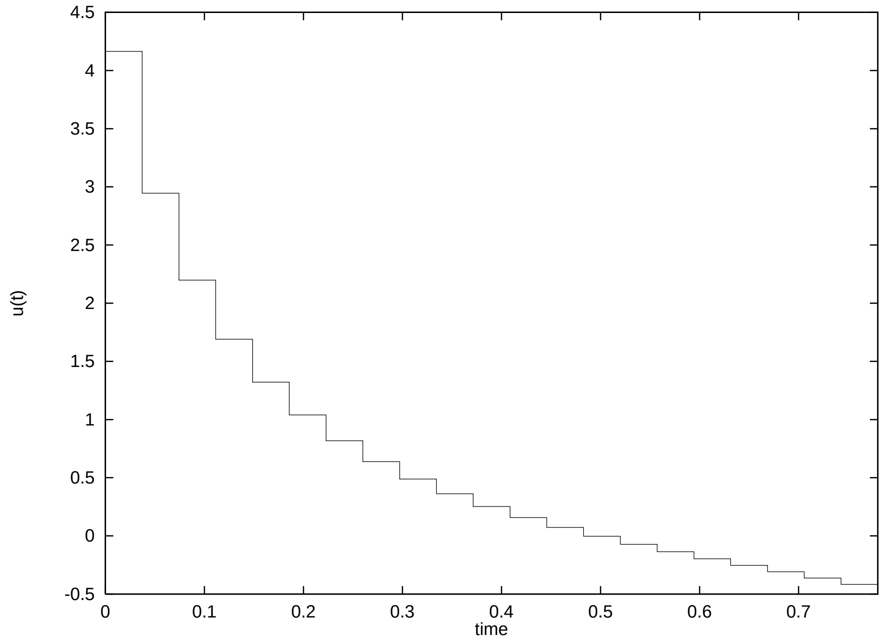

Figure 8: Optimal Control Policy for CSTR Problem