INFORMS Journal on Computing

Publication details, including instructions for authors and subscription information:
<http://pubsonline.informs.org>

An Outer-Inner Approximation for Separable Mixed-Integer Nonlinear Programs

Hassan Hijazi, Pierre Bonami, Adam Ouorou

To cite this article:

Hassan Hijazi, Pierre Bonami, Adam Ouorou (2013) An Outer-Inner Approximation for Separable Mixed-Integer Nonlinear Programs. INFORMS Journal on Computing

Published online in Articles in Advance 04 Oct 2013

. <http://dx.doi.org/10.1287/ijoc.1120.0545>

Full terms and conditions of use: <http://pubsonline.informs.org/page/terms-and-conditions>

This article may be used only for the purposes of research, teaching, and/or private study. Commercial use or systematic downloading (by robots or other automatic processes) is prohibited without explicit Publisher approval. For more information, contact permissions@informs.org.

The Publisher does not warrant or guarantee the article's accuracy, completeness, merchantability, fitness for a particular purpose, or non-infringement. Descriptions of, or references to, products or publications, or inclusion of an advertisement in this article, neither constitutes nor implies a guarantee, endorsement, or support of claims made of that product, publication, or service.

Copyright © 2013, INFORMS

Please scroll down for article—it is on subsequent pages

INFORMS is the largest professional society in the world for professionals in the fields of operations research, management science, and analytics.

For more information on INFORMS, its publications, membership, or meetings visit <http://www.informs.org>

An Outer-Inner Approximation for Separable Mixed-Integer Nonlinear Programs

Hassan Hijazi

National ICT Australia, Canberra ACT 2601, Australia,
hassan.hijazi@nicta.com.au

Pierre Bonami

Laboratoire d'Informatique Fondamentale, Faculté des Sciences de Luminy, Université de Marseille,
13009 Marseille, France, pierre.bonami@lif.univ-mrs.fr

Adam Ouorou

Orange Labs R&D/CORE/TPN, 92794 Issy-Les-Moulineaux Cedex 9, France,
adam.ouorou@orange.com

A common structure in convex mixed-integer nonlinear programs (MINLPs) is separable nonlinear functions. In the presence of such structures, we propose three improvements to the outer approximation algorithms. The first improvement is a simple extended formulation, the second is a refined outer approximation, and the third is a heuristic inner approximation of the feasible region. As a side result, we exhibit a simple example where a classical implementation of the outer approximation would take an exponential number of iterations, whereas it is easily solved with our modifications. These methods have been implemented in the open source solver BONMIN and are available for download from the Computational Infrastructure for Operations Research project website. We test the effectiveness of the approach on three real-world applications and on a larger set of models from an MINLP benchmark library. Finally, we show how the techniques can be extended to perspective formulations of several problems. The proposed tools lead to an important reduction in computing time on most tested instances.

Key words: mixed-integer nonlinear programming; outer approximation

History: Accepted by Karen Aardal, Area Editor for Design and Analysis of Algorithms; received February 2011; revised March 2012, October 2012; accepted November 2012. Published online in *Articles in Advance*.

1. Introduction

A well-known approach for solving convex mixed-integer nonlinear programs (MINLPs where one minimizes a convex objective over the intersection of a convex region and integrality requirements) is the *outer approximation* of Duran and Grossmann (1986). Outer approximation consists in building a mixed-integer linear equivalent to the feasible region of the problem by taking tangents at well-specified points. Outer approximation is a very successful approach to solve convex MINLPs. Different algorithms have been devised to build the mixed-integer linear equivalent and they are implemented in several state-of-the-art solvers such as DICOPT (Grossmann et al. 2001), BONMIN (Bonami et al. 2008), and filMINT (Abhishek et al. 2010).

In this paper, we propose several improvements to outer approximation for a subclass of MINLPs featuring particular structures. Specifically, we study

separable convex MINLPs of the form

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & g_i(\mathbf{x}) \leq 0, \quad \forall i \in \{1, \dots, m\}, \\ & \mathbf{Ax} \leq \mathbf{b}, \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \\ & x_j \in \mathbb{Z}, \quad \forall j \in \{1, \dots, p\}, \\ & x_j \in \mathbb{R}, \quad \forall j \in \{p+1, \dots, n\}, \end{aligned} \quad (\text{sMINLP})$$

where $\mathbf{A} \in \mathbb{Q}^{l \times n}$ represents the coefficient matrix of the linear constraints, $\mathbf{b} \in \mathbb{Q}^l$ the right-hand side, $\mathbf{l}$ and $\mathbf{u} \in \mathbb{R}^n$ are finite lower and upper bounds on the decision variables $\mathbf{x}$, and, for $i = 1, \dots, m$, g_i are the nonlinear functions describing the feasible region. We denote by $X := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{Ax} \leq \mathbf{b}, \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}\}$ the region described by the linear constraints of the problem. Furthermore, the functions $g_i : X \rightarrow \mathbb{R}$ are assumed to be continuously differentiable and *convex*

separable. By convex separable function we mean that each g_i can be written as a sum of compositions of a univariate convex function and an affine function:

$$g_i(\mathbf{x}) = \sum_{j=1}^{q_i} g_{ij}(\boldsymbol{\alpha}_{ij}^T \mathbf{x} + \beta_{ij}),$$

where, for $i = 1, \dots, m$ and $j = 1, \dots, q_i$, $\boldsymbol{\alpha}_{ij} \in \mathbb{R}^n$ and $\beta_{ij} \in \mathbb{R}$.

Note that any problem with a convex separable objective function and constraints can be cast into this form by introducing an extra variable and moving the objective function into the constraints. The separability requirement might seem restrictive. Let us emphasize that it is a very common feature of convex MINLPs in the literature. In particular, note that any convex quadratic function can be put into that form by using a spectral decomposition. As we will see in §4.4, almost all convex MINLPs available in benchmark libraries are convex separable.

The problem (sMINLP) is a simple form of MINLP and one could hope that outer approximation-based methods perform well on it. We show that this is not the case and even a problem with one quadratic constraint can require an exponential number of iterations to be solved by outer approximation. We then propose three improvements allowing us to overcome this issue. First, we exploit the separability to build a univariate extended formulation that can be shown to have better convergence. Second, we build a good initial linear approximation of the problem. Third, we propose a heuristic method for finding near-optimal feasible solutions.

Our approach to finding these solutions consists in building a mixed-integer linear approximation of the feasible region of (sMINLP) from the inside. The separable structure of the problem we consider makes this approximation simple to build. Indeed, it suffices to compute chords of each univariate convex function separately.

We present extensive computational results aimed at assessing the effectiveness of the improvements we propose.

Finally, we consider an important class of MINLPs from the literature that are not separable, namely, the perspective formulations of separable MINLPs with indicator variables developed by Frangioni and Gentile (2006) and Günlük and Linderoth (2010). We show that our approach can be extended to these problems.

In the next section, we briefly recall the outer approximation algorithm and show an example of separable MINLP where outer approximation has to perform an exponential number of iterations. In §3 we describe our approach. In §4 we present computational experiments comparing our approach to state-of-the-art solvers. Finally, in §5, we present

the extension to perspective formulations of problems with indicator variables.

In the remainder of the paper, we denote by $\mathbf{g}$, the function $X \rightarrow \mathbb{R}^m$ consisting of all the functions g_i and by $I = \{1, 2, \dots, p\}$ the set of the indices of the integer constrained variables. Given a vector $\mathbf{x} \in \mathbb{R}^n$, we denote by $\mathbf{x}_I \in \mathbb{R}^p$ the subvector with indices in I .

2. Outer Approximation

Consider any point $\hat{\mathbf{x}} \in X$, by convexity, all $\mathbf{x} \in X$ such that $g_i(\mathbf{x}) \leq 0$ verify the following *outer approximation* constraint:

$$\nabla g_i(\hat{\mathbf{x}})^T (\mathbf{x} - \hat{\mathbf{x}}) + g_i(\hat{\mathbf{x}}) \leq 0. \quad (1)$$

More generally, we call outer approximation constraints the set of constraints of the form (1) taken for $i \in \{1, \dots, m\}$.

Outer approximation algorithms are based on the mixed-integer linear relaxation of (sMINLP) obtained by replacing the nonlinear constraints by their linear outer approximations taken in a set of points $\mathcal{P} = \{\hat{\mathbf{x}}^{(1)}, \dots, \hat{\mathbf{x}}^{(K)}\}$. We denote the mixed-integer set given by these outer approximation constraints by

$$\mathcal{X}(\mathcal{P}) := \begin{cases} \nabla g_i(\hat{\mathbf{x}}^{(k)})^T (\mathbf{x} - \hat{\mathbf{x}}^{(k)}) + g_i(\hat{\mathbf{x}}^{(k)}) \leq 0, \\ \quad \forall i \in \{1, \dots, m\}, \forall \hat{\mathbf{x}}^{(k)} \in \mathcal{P}, \\ \mathbf{Ax} \leq \mathbf{b}, \\ \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \\ x_j \in \mathbb{Z}, \quad \forall j \in \{1, \dots, p\}, \\ x_j \in \mathbb{R}, \quad \forall j \in \{p+1, \dots, n\}, \end{cases}$$

and the corresponding mixed-integer linear program (MILP)

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \mathbf{x} \in \mathcal{X}(\mathcal{P}). \end{aligned} \quad (\text{OA}(\mathcal{P}))$$

The main theoretical justification of outer approximation algorithms is that if $\mathcal{P}$ contains a suitable set of points, then $(\text{OA}(\mathcal{P}))$ has the same optimal value as (sMINLP). More precisely, let $\mathcal{Y} := \{\mathbf{y} \in \mathbb{Z}^p: l_j \leq y_j \leq u_j, j \in \{1, \dots, p\}\}$ be the set of all possible assignments for the integer constrained variables. For each $\mathbf{y} \in \mathcal{Y}$ (note that by our assumptions $\mathcal{Y}$ is finite), we define

$$\xi(\mathbf{y}) := \begin{cases} \arg \min \{ \mathbf{c}^T \mathbf{x}: \mathbf{g}(\mathbf{x}) \leq 0, \mathbf{Ax} \leq \mathbf{b}, \mathbf{x}_I = \mathbf{y} \} \\ \quad \text{if it exists,} \\ \arg \min \left\{ \sum_{i=1}^m \max(g_i(\mathbf{x}), 0): \mathbf{Ax} \leq \mathbf{b}, \mathbf{x}_I = \mathbf{y} \right\} \\ \quad \text{otherwise,} \end{cases}$$

and $\mathcal{P}(\mathcal{Y}) := \{\xi(\mathbf{y}): \text{for all } \mathbf{y} \in \mathcal{Y}\}$. It can be shown that $\text{OA}(\mathcal{P}(\mathcal{Y}))$ has the same optimal value as (sMINLP) (Duran and Grossmann 1986, Fletcher and Leyffer 1994, Bonami et al. 2008).

The aim of outer approximation algorithms is to build this equivalent mixed-integer linear program.

Figure 1 The Set $\mathcal{B}^3$ in $\mathbb{R}^3$

To do so, two algorithmic ideas have been developed: the outer approximation decomposition algorithm (Duran and Grossmann 1986) and the outer approximation branch and cut (Quesada and Grossmann 1992). In this paper we only consider the first of these two algorithms that is specified in Algorithm 1. Nevertheless, our improvements could also be applied in the context of the second one (for refinements of these two algorithms and implementation details the reader should report to Fletcher and Leyffer 1994, Bonami et al. 2008, Abhishek et al. 2010, Bonami et al. 2012).

Algorithm 1 (The Outer Approximation Decomposition Algorithm)

0. **Initialize.**

$z_U \leftarrow +\infty$, $z_L \leftarrow -\infty$, $\mathbf{x}^* \leftarrow \text{NONE}$. Let $\mathbf{x}^0$ be an optimal solution of the continuous relaxation of (sMINLP), $\mathcal{P} \leftarrow \{\mathbf{x}^0\}$.

1. **Termination?**

If $z_U - z_L = 0$ (and $\mathbf{x}^*$ is optimal) or $\text{OA}(\mathcal{P})$ is infeasible (and so is (sMINLP)), Stop.

2. **Lower Bound**

Let $z_{\text{OA}(\mathcal{P})}$ be the optimal value of $\text{OA}(\mathcal{P})$ and $\hat{\mathbf{x}}$ its optimal solution. $z_L \leftarrow z_{\text{OA}(\mathcal{P})}$.

3. **Find new linearization point**

$\mathcal{P} \leftarrow \mathcal{P} \cup \{\xi(\hat{\mathbf{x}}_l)\}$.

4. **Upper Bound?**

If $\xi(\hat{\mathbf{x}}_l)$ is feasible for (sMINLP) and $\mathbf{c}^T \xi(\hat{\mathbf{x}}_l) < z_U$, then $\mathbf{x}^* \leftarrow \xi(\hat{\mathbf{x}}_l)$ and $z_U \leftarrow \mathbf{c}^T \xi(\hat{\mathbf{x}}_l)$.

5. **Iterate**

Go to 1.

The convergence of Algorithm 1 follows directly from the equivalence between $\text{OA}(\mathcal{P}(\mathcal{Y}))$ and (sMINLP). Note that, in the worst case, the algorithm may have to solve a MILP for all points $\mathbf{y} \in \mathcal{Y}$. Nevertheless, in practice the algorithm is one of the most

competitive algorithms available (see, for example, Bonami et al. 2012 for a recent experimental comparison). In that light, one could expect it to perform well on simple MINLPs where all the nonlinear functions are separable. Next, we exhibit a simple example showing that this is not the case and even for a problem with one quadratic constraint the outer approximation algorithm may require an exponential number of iterations to converge.

EXAMPLE 1 (OUTER APPROXIMATION TAKING EXPONENTIALLY MANY ITERATIONS). We consider the subset of $\mathbb{Z}^n$ obtained by intersecting the ball $B(\rho, r)$ of radius $r = (\sqrt{n-1})/2$ centered in the point $\rho = (1/2, \dots, 1/2)$ with the vertices of the unit hypercube $\{0, 1\}^n$ (see Figure 1):

$$B^n = \left\{ \mathbf{x} \in \{0, 1\}^n \mid \sum_{j=1}^n \left(x_j - \frac{1}{2} \right)^2 \leq \frac{n-1}{4} \right\}.$$

Clearly, because the vertices of the hypercube are at distance $\sqrt{n}/2$ from ρ , $B^n = \emptyset$. Note also that the ball is chosen so that it has a nonempty intersection with all the edges of the cube.

In our first lemma, we show that an outer approximation constraint obtained from the ball defining $\mathcal{B}^n$ cannot cut simultaneously two vertices of the hypercube. Remember that, by definition, outer approximation constraints are valid linear inequalities for $B(\rho, r)$ (i.e., are satisfied by all points in $B(\rho, r)$).

LEMMA 2.1. A valid linear inequality $\alpha^T x \leq \beta$ for the ball $B(\rho, \sqrt{n}/2)$ cannot cut two vertices of the unit hypercube simultaneously.

PROOF. Suppose that an inequality cuts two vertices of the hypercube x and y (i.e., $\alpha^T x > \beta$ and $\alpha^T y > \beta$), then it cuts the whole segment $[x, y]$. By construction, $B(\rho, r)$ has a nonempty intersection with $[x, y]$, therefore the inequality is not valid for $B(\rho, r)$. $\square$

It follows directly from Lemma 2.1 that the set $\mathcal{P}$ of linearization points would need to contain at least 2^n points for the outer approximation set $\mathcal{X}(\mathcal{P})$ to be empty.

THEOREM 2.2. The mixed-integer linear program ($\text{OA}(\mathcal{P})$) built from the set B^n is feasible if $|\mathcal{P}| < 2^n$.

PROOF. Because no linear outer approximation can cut two vertices of the hypercube simultaneously, if $|\mathcal{P}| < 2^n$, at least one vertex is feasible. $\square$

It follows directly from this theorem that Algorithm 1 would need 2^n iterations to converge (remember that each iteration involves the solution of a MILP). We note that in the context of an outer approximation branch and cut also, at least 2^n nodes would need to be enumerated. This indicates that even on very simple examples of separable MINLPs, the outer approximation might present extremely bad behavior.

In the next section we will show that, by a simple modification, this problem can be solved with an outer approximation procedure in much less iterations.

3. The New Scheme

In the remainder of the paper, we assume that $\alpha_{ij} = \mathbf{e}_j$ (with $\mathbf{e}_j$ the unit vector with component j equal to 1) and $\beta_{ij} = 0$, for all $i = 1, \dots, m$, $j = 1, \dots, q_i$. Note that this is without loss of generality because it can always be achieved by sufficiently adding many artificial variables to be equal to $\alpha_{ij}\mathbf{x} + \beta_{ij}$ and reordering the univariate functions g_{ij} .

3.1. Univariate Extended Formulation

Our first ingredient is a reformulation of (sMINLP) in an extended space. All separable functions in the problem are broken by introducing an auxiliary variable for each univariate function:

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & g_{ij}(x_j) \leq y_{ij}, \quad \forall i \in \{1, \dots, m\}, \forall j \in \{1, \dots, n\}, \\ & \sum_{j=1}^n y_{ij} \leq 0 \quad \forall i \in \{1, \dots, m\}, \\ & \mathbf{Ax} \leq \mathbf{b}, \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \\ & x_j \in \mathbb{Z} \quad \forall j \in \{1, \dots, p\}, \\ & x_j \in \mathbb{R} \quad \forall j \in \{p+1, \dots, n\}. \end{aligned} \quad (\text{sMINLP}^*)$$

Clearly, if $(\mathbf{x}, \mathbf{y})$ is an optimal solution to (sMINLP*), then $\mathbf{x}$ is optimal for (sMINLP).

Extended formulations are widely used in integer programming and combinatorial optimization as a means to obtain stronger relaxations. The difference between (sMINLP) and (sMINLP*) may seem innocuous. It is quite clear that the continuous relaxations of both problems are identical. Nevertheless, as noted previously by Tawarmalani and Sahinidis (2005), linear outer approximations of (sMINLP*) yield tighter approximations than those of (sMINLP). More precisely, Proposition 5 in Tawarmalani and Sahinidis (2005) shows that to obtain a linear outer approximation of (sMINLP) of the same strength as a linear outer approximation of (sMINLP*) one needs exponentially many more linearization points. The univariate extended formulation obtained from Example 1 gives a particularly striking illustration of this in the context of the outer approximation algorithm.

EXAMPLE 1 (CONTINUED). We denote by B^{n*} the univariate extended formulation of the mixed-integer set B^n :

$$\begin{aligned} B^{n*} = \left\{ (\mathbf{x}, \mathbf{z}) \in \{0, 1\}^n \times \mathbb{R}^n \mid \left(x_j - \frac{1}{2}\right)^2 \leq z_j, \right. \\ \left. \forall j \in \{1, 2, \dots, n\}, \sum_{j=1}^n z_j \leq \frac{n-1}{4} \right\}. \end{aligned}$$

Given a set of points

$$\mathcal{P} := \left\{ \begin{pmatrix} \hat{\mathbf{x}}^{(1)} \\ \hat{\mathbf{z}}^{(1)} \end{pmatrix}, \dots, \begin{pmatrix} \hat{\mathbf{x}}^{(K)} \\ \hat{\mathbf{z}}^{(K)} \end{pmatrix} \right\},$$

the outer approximation of B^{n*} is given by

$$\mathcal{B}^{n*}(\mathcal{P}) = \left\{ (\mathbf{x}, \mathbf{z}) \in \{0, 1\}^n \times \mathbb{R}^n \mid \begin{aligned} & (2\hat{x}_j^{(k)} - 1)x_j \leq (\hat{x}_j^{(k)})^2 \\ & -\frac{1}{4} + z_j, \forall j \in \{1, 2, \dots, n\}, \forall \hat{\mathbf{x}}^{(k)} \in \mathcal{P}^K, \\ & \sum_{j=1}^n z_j \leq \frac{n-1}{4} \end{aligned} \right\}.$$

Note that $\hat{\mathbf{z}}^{(k)}$ is not present in the outer approximation, because variables z_i only appear in linear expressions, therefore it can be neglected. It can easily be seen that $\mathcal{P}$ only needs two points to make $\mathcal{B}^{n*}$ empty. Take $\mathbf{x}^1 \in \{0, 1\}^n$ and $\mathbf{x}^2$ the complement of $\mathbf{x}^1$ ($x_i^2 = 1 - x_i^1$, $i = 1, \dots, n$). The set $\mathcal{B}^{n*}(\{\mathbf{x}^1, \mathbf{x}^2\})$ is

$$\left\{ (\mathbf{x}, \mathbf{z}) \in \{0, 1\}^n \times \mathbb{R}^n \mid \begin{aligned} & \frac{1}{4} - x_j \leq z_j, \forall j \in \{1, 2, \dots, n\}, \\ & x_j - \frac{3}{4} \leq z_j, \forall j \in \{1, 2, \dots, n\}, \\ & \sum_{j=1}^n z_j \leq \frac{n-1}{4} \end{aligned} \right\}.$$

This set is empty because $x \in \{0, 1\}^n$ implies $z_j \geq 1/4$ for $j = 1, \dots, n$ and $\sum_{j=1}^n z_j \geq n/4 > (n-1)/4$. This shows that on this simple problem although the classical version of the outer approximation takes 2^n iterations, the univariate extended formulation can be solved in only two iterations (note that if we count the number of constraints instead of linearization points, $2n$ constraints are generated versus 2^n). Thus, in general, we can expect the univariate extended formulation to perform much better than the initial one in an outer approximation algorithm.

3.2. Refined Initial Outer Approximation

The second ingredient of our approach consists in building a better initial outer approximation of the feasible region of (sMINLP*). Note that in Algorithm 1, the first outer approximation is built by taking only one linearization point at an optimal solution of the continuous relaxation of (sMINLP). In general, it is not obvious how to choose other linearization points that will improve the bound obtained by solving the associated MILP. Here, we exploit the simplicity of univariate functions to choose a better initial set of points.

Consider one of the nonlinear constraints of (sMINLP*) $g_{ij}(x_j) \leq y_{ij}$ and the set $S = \{(x_j, y_{ij}) \in \mathbb{R}^2 \mid g_{ij}(x_j) \leq y_{ij}, l_j \leq x_j \leq u_j\}$. We enrich the first

outer approximation with a set of points of the form $(\hat{x}_j, g_{ij}(\hat{x}_j))$, for $\hat{x}_j \in [l_j, u_j]$:

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \nabla g_{ij}(\hat{x}_j^k)(x_j - \hat{x}_j^k) + g_{ij}(\hat{x}_j^k) \leq y_{ij} \\ & \forall i \in \{1, \dots, m\}, \forall j \in \{1, \dots, q_i\}, \forall k = \{1, \dots, K\}, \\ & \sum_{j=1}^{q_i} y_{ij} \leq 0 \quad \forall i \in \{1, \dots, m\}, \\ & \mathbf{Ax} \leq \mathbf{b}, \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \\ & x_j \in \mathbb{Z} \quad \forall j \in \{1, \dots, p\}, \\ & x_j \in \mathbb{R} \quad \forall j \in \{p+1, \dots, n\}. \end{aligned} \quad (\text{OA}^*)$$

This set may be obtained in different ways. For example, one could choose points uniformly in the interval $[l_j, u_j]$ or perform a random sampling, or try to make a finer approximation by taking into account the curvature of g_i . After testing the three methods, in our experiments, we choose a uniform sampling. It typically provides good computational results. Figures 2 and 3 give examples of S and S_{out} for the constraint $1/(6-x) \leq y$.

This initial approximation can be used as a starting point by any classical outer approximation-based algorithm.

3.3. Inner Approximation

The third and last ingredient is a heuristic procedure for finding a first feasible solution to the problem. To do so, we build a second mixed-integer linear program. The difference is that instead of building an outer approximation, we now approximate the feasible region from the inside.

Consider again one of the nonlinear constraints of (sMINLP), $g_{ij}(x_j) \leq y_{ij}$, and the set $S = \{(x_j, y_{ij}) \in \mathbb{R}^2 \text{ s.t. } g_{ij}(x_j) \leq y_{ij}, l_j \leq x_j \leq u_j\}$. Instead of taking tangents, we now take chords at the boundary of S . Consider a sequence of K points $(\hat{x}_j^k, g_{ij}(\hat{x}_j^k))$, $k \in \{1, 2, \dots, K\}$

Figure 2 $S = \{(x, y) \in \mathbb{R}^2 \text{ s.t. } 1/(6-x) \leq y, 0 \leq x \leq 5\}$

Figure 3 S_{out}

such that $l_j = \hat{x}_j^1 \leq \hat{x}_j^2 \leq \dots \leq \hat{x}_j^K = u_j$. We consider the polyhedral approximation S_{in} of S formed by the constraints

$$\frac{g_{ij}(\hat{x}_j^{k+1}) - g_{ij}(\hat{x}_j^k)}{\hat{x}_j^{k+1} - \hat{x}_j^k}(x_j) + \frac{g_{ij}(\hat{x}_j^k)\hat{x}_j^{k+1} - g_{ij}(\hat{x}_j^{k+1})\hat{x}_j^k}{\hat{x}_j^{k+1} - \hat{x}_j^k} \leq y_{ij}, \quad (2)$$

for $k = 1, \dots, K-1$. It is trivial to check that $S_{\text{in}} \subset S$. Figure 4 illustrates this inner approximation on the constraint of Figure 2.

Computing this inner approximation for every nonlinear constraint of the problem, we obtain the mixed-integer linear program

$$\begin{aligned} \min \quad & \mathbf{c}^T \mathbf{x} \\ \text{s.t.} \quad & \sum_{j=1}^n y_{ij} \leq 0 \quad \forall i \in \{1, \dots, m\}, \\ & \hat{\sigma}_{ij}^k x_j + \hat{\gamma}_{ij}^k \leq y_{ij} \quad \forall i \in \{1, \dots, m\}, \\ & \quad \forall j \in \{1, \dots, n\}, \forall k \in \{1, \dots, K\}, \\ & \mathbf{Ax} \leq \mathbf{b}, \\ & \mathbf{l} \leq \mathbf{x} \leq \mathbf{u}, \\ & x_i \in \mathbb{Z} \quad \forall i \in \{1, \dots, p\}, \\ & x_i \in \mathbb{R} \quad \forall i \in \{p+1, \dots, n\}, \end{aligned} \quad (\text{IA})$$

where $\hat{\sigma}_{ij}^k = (g_{ij}(\hat{x}_j^{k+1}) - g_{ij}(\hat{x}_j^k))/(\hat{x}_j^{k+1} - \hat{x}_j^k)$ and $\hat{\gamma}_{ij}^k = (g_{ij}(\hat{x}_j^k)\hat{x}_j^{k+1} - g_{ij}(\hat{x}_j^{k+1})\hat{x}_j^k)/(\hat{x}_j^{k+1} - \hat{x}_j^k)$.

Because the nonlinear constraints are inner approximated, every solution to (IA) is feasible for (sMINLP). Let us emphasize that (IA) may be infeasible regardless of the feasibility of (sMINLP). Whenever it is the case that (IA) is infeasible, one could build and solve a better approximation by enriching the set of discretization points, but here we just use the initial approximation as a quick heuristic. Thus, an initial primal point is retained for (sMINLP) only if (IA) returns a feasible solution.

The combination of (OA($\mathcal{P}$)) and (IA) gives rise to an outer-inner approximation algorithm that hopefully results in a smaller initial gap thanks to good

Figure 4 Inner Approximation of S with Seven Discretization Points

lower and upper bounds. In the next section we present computational experiments aimed at assessing the effectiveness of this scheme.

4. Computational Results

We have implemented the enhancements we propose to the outer approximation decomposition algorithm into the open source solver BONMIN (Bonami et al. 2008) from COIN-OR (Lougee-Heimer 2003). Our implementation, called SEPA can be found in the stable distribution of BONMIN in the subdirectory experimental/Separable. Our program takes as input a convex separable MINLP under the form (sMINLP*) (i.e., already in the form of a univariate formulation) and applies to it an outer approximation decomposition scheme augmented with our improvements:

- the refined initial outer approximation described in §3.2, and
- the inner approximation scheme described in §3.3.

The number of discretization points at which the initial linear approximations are built is a user-set parameter and is initialized to eight in all our experiments.

Next, we present experiments aimed at assessing the effectiveness of our modified algorithm. We report several series of experiments. First, we report detailed experiments on three classes of problems from the literature: separable quadratic facility location, delay constrained routing, and stochastic service design problems. The goal of these detailed experiments is to assess the effect of each of the modifications we propose to the outer approximation algorithm.

To this purpose, we compare different versions of the algorithm:

- the standard implementation of BONMIN called B – OA applied either to the initial formulation of the problem or the univariate extended formulation;
 - the outer approximation with an initial outer approximation called B – OA + Ref applied only to the univariate extended formulation; and
 - the inner approximation heuristic called Inner applied only to the univariate extended formulation.
- We do not report on the combination of B – OA + Ref and Inner. As will be seen from the results, in most cases B – OA + Ref takes a very limited number of iterations and a good solution is already found in its first iteration. The combination only helps in very few hard problems. When possible, we also present comparisons with the state-of-the-art solver CPLEX 12.3. CPLEX can only solve mixed-integer quadratic (MIQP) and mixed-integer second order cone (MISOCP) programs.

In the second experiment, we test a larger test set of problems from the IBM CMU MINLP Library (Sawaya et al. 2006).

All of our experiments are performed on a machine equipped with 4 Intel Quad Core Xeon 2.93 GHz CPUs and 120 GB of RAM. We use the stable version 1.6 of BONMIN with Ipopt 3.8 (Wächter and Biegler 2006) as the nonlinear programming solver and IBM-CPLEX version 12.3 as the MILP solver. In all experiments, we use the deterministic multi-threaded version of CPLEX on 10 threads with all other settings set to their default values. It is worth stressing that the multithreaded version of CPLEX is

used with the same settings both as a stand-alone solver and within BONMIN's different algorithms.

4.1. Separable Quadratic Facility Location Problems (SQFL)

SQFL is a variant of the classical uncapacitated facility location problem introduced by Günlük et al. (2007). We are given a set of customers J and a set of candidate locations I . Each customer has a unit demand, and each facility has an unlimited capacity. There is a fixed cost for opening a facility plus a shipping cost that is proportional to the square of the quantity delivered. The problem can be modeled as follows:

$$\begin{aligned} \min \quad & \left\{ \sum_{i \in I} f_i z_i + \sum_{i \in I, j \in J} q_{ij} x_{ij}^2 \right\} \\ \text{s.t.} \quad & x_{ij} \leq z_i \quad \forall i \in I, j \in J, \\ & \sum_{i \in I} x_{ij} = 1 \quad \forall j \in J, \\ & z_i \in \{0, 1\}, x_{ij} \in [0, 1] \quad \forall i \in I, j \in J. \end{aligned}$$

Let us emphasize that a tighter nonseparable formulation for this problem has been proposed by Günlük and Linderoth (2010). This formulation will be considered in §5.

In Tables 1 and 2, we present statistics on the solution of six instances of SQFL of increasing size. We report results using five different settings. First, on the initial formulation, we consider CPLEX's MIQP solver (labeled CPLEX) and BONMIN's B – OA. Second, on the univariate extended formulation, we consider B – OA, its variant augmented with a refined initial outer approximation (labeled B – OA + Ref) and the inner approximation heuristic. All algorithms were given a time limit of three hours of CPU time except the inner approximation heuristic, which was given a time limit of 180 seconds.

Note that we do not report results on solving the univariate formulation with CPLEX. This is for two reasons. First, CPLEX's branch and bound does not take any advantage of the structure of the univariate formulation. Second, putting the quadratic terms of the objective into the constraints forces CPLEX to use its

MISOCP solver rather than its MIQP solver. The former is substantially slower than the latter.

In Table 1, we present the main characteristics of each instance (number of customers and location) and for each method the total CPU time to solve the instance, or, if the instance could not be solved within the time limit, the final relative gap between brackets. The last two columns present statistics on the relative gap to the optimum of the feasible solution returned by the inner approximation (Inner sol) and the time needed to solve the corresponding MILP to optimality (Inner time).

In Table 2, we give the number of outer approximation iterations (i.e., number of MILPs solved) for the two outer approximation-based algorithms, and the total number of branch-and-bound nodes for all the exact algorithms (note that for the outer approximation-based algorithms, this is a cumulative number over all iterations).

The results show that for this problem the classical outer approximation method applied on the original formulation is inefficient, solving only one instance out of six. One can note that this instance that has only eight candidate locations (and therefore eight binary variables) requires the solution of 110 MILPs that is not so far from the worst case of 2^8 iterations. Using the univariate extended formulation with B – OA allows us to prove optimality of four instances, but it is still significantly slower than CPLEX. The method B – OA + Ref solves the same instances as CPLEX and B – OA on the univariate extended formulation, but it is significantly faster than B – OA and comparable with CPLEX in terms of CPU time. We note also a significant reduction of the final gap on the hard instances that remained unsolved using all approaches when using B – OA + Ref. The inner approximation method allows us to find a feasible solution on four instances out of five with an average computing time of 82 seconds. On the instance for which no solution is found by the inner approximation, no solution was found within the time limit of 180 seconds, but the inner approximation is feasible (a solution can be found if more time is given). Finally, from Table 2, we note the very small number of iterations of B – OA + Ref (three at most) and

Table 1 CPU Times for SQFL Instances

Inst.	I	J	Initial formulation		Univariate formulation			
			CPLEX	B – OA	B – OA	B – OA + Ref	Inner sol (%)	Inner time
1	8	30	0.91	70.01	7.36	2.20	0.00	0.35
2	20	100	25.36	[16.59%]	153.66	25.06	0.00	6.47
3	20	150	141.95	[64.96%]	3,075.63	464.40	1.21	71.40
4	30	150	249.31	[48.79%]	2,496.28	338.32	2.13	49.26
5	35	300	[12.42%]	[72.58%]	[54.62%]	[4.57%]	9.73	180.00
6	50	500	[77.91%]	[91.72%]	[64.51%]	[51.78%]	[∞]	180.00

Table 2 Nodes and Iterations for SQFL Instances

Inst.	Initial formulation			Univariate formulation			
	CPLEX		B – OA	B – OA		B – OA + Ref	
	No. of nodes	No. of it.		No. of it.	No. of nodes	No. of it.	No. of nodes
1	92	110	13,627	11	243	1	33
2	84	230	50,502	19	2,416	1	54
3	3,116	121	31,664	20	11,396	2	1,676
4	2,227	116	13,538	28	14,623	2	980
5	49,780	84	740	16	8,415	3	1,799
6	327	49	159	2	25,357	1	199

the small number of branch-and-bound nodes (always inferior to CPLEX).

4.2. Delay Constrained Routing Problem (DCRP)

The delay constrained routing problem consists of finding an optimal routing in a telecommunication network that satisfies both a given demand and constraints on the end-to-end delay of communications. The problem was first introduced by Ben Ameur and Ouorou (2006). We are given a graph $G = (V, E)$ with arc costs w_e and capacities c_e and a set of K source-destination demands. Each demand is described by the set of paths it can use $P(k) = \{P_k^1, P_k^2, \dots, P_k^{n_k}\}$, the quantity of flow v_k that needs to be routed, and a maximum authorized end-to-end delay α_k . A feasible routing is an assignment of a sufficient flow to candidate paths so as to satisfy demand such that the total delay on each active path P_k^i is smaller than α_k . The delay on an active path P_k^i is given by the nonlinear function $\sum_{e \in P_k^i} (1/(c_e - x_e))$, where x_e is the total quantity of flow going through edge e . Several formulations for this problem have been proposed (see Ben Ameur and Ouorou 2006, Hijazi et al. 2012). Here, we only present the so-called big- M formulation. We also report results on a tighter (nonseparable) formulation that uses perspective function proposed by Hijazi et al. (2012). Denoting by x_e the total quantity of flow routed through edge e , by ϕ_k^i the proportion of demand k routed along path P_k^i , and by z_k^i an indicator variable indicating if path P_k^i is active; the model reads

$$\min \sum_{e \in E} w_e x_e \quad (3a)$$

$$\text{s.t.} \quad \sum_{i=1}^{n_k} \phi_k^i \geq 1, \quad \forall k \in K, \quad (3b)$$

$$\sum_{k \in K} \sum_{P_k^i \ni e} \phi_k^i v_k \leq x_e \leq c_e, \quad \forall e \in E, \quad (3c)$$

$$\sum_{e \in P_k^i} \frac{1}{c_e - x_e} \leq M - z_k^i (M - \alpha_k), \quad \forall k \in K, \forall P_k^i \in P(k), \quad (3d)$$

$$\sum_{P_k^i \in P(k)} z_k^i \leq N, \quad \forall k \in K, \quad (3e)$$

$$\phi_k^i \leq z_k^i, \quad \forall k \in K, \forall P_k^i \in P(k), \quad (3f)$$

$$\phi_k^i \in [0, 1], \quad z_k^i \in \{0, 1\}, \quad \forall k \in K, \forall P_k^i \in P(k), \quad (3g)$$

$$x_e \in \mathbb{R}, \quad \forall e \in E. \quad (3h)$$

The objective (3a) is to minimize the use of expensive links in the network. Constraints (3b) assure that each demand is entirely routed. Constraints (3c) define the flow variable on an edge as the sum of the flows carried by it. The end-to-end delay constraints are introduced in (3d) using a big enough constant M to handle their on and off aspect. A limit on the number of active paths allowing to route each demand is set in (3e). Finally, the fractional flow variables are linked to the binary activation variables in constraints (3f).

The model is clearly separable because the only nonlinear function is $\sum_{e \in P_k^i} (1/(c_e - x_e))$. Also, as shown in Hijazi (2010), the delay constraints (3d) are second-order cone representable. Let $r_e = c_e - x_e$ denote the residual capacity on link e and $t_e = 1/r_e$ the corresponding delay value. Constraints (3d) are equivalent to the following system of inequalities:

$$(3d) \equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, & \forall e \in P_k^i, \\ 1/r_e \leq t_e, & \forall e \in P_k^i, \\ \sum_{e \in P_k^i} t_e \leq \alpha_k. \end{cases}$$

$$\equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, & \forall e \in P_k^i, \\ 0 \leq t_e r_e - 1, & \forall e \in P_k^i, \\ \sum_{e \in P_k^i} t_e \leq \alpha_k. \end{cases}$$

Moreover, since $2t_e r_e = (r_e + t_e)^2 - r_e^2 - t_e^2$, we can write

$$(3d) \equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, & \forall e \in P_k^i, \\ 0 \leq q_e^2 - r_e^2 - t_e^2 - 2, & \forall e \in P_k^i, \\ q_e \geq r_e + t_e, & \forall e \in P_k^i, \\ \sum_{e \in P_k^i} t_e \leq \alpha_k. \end{cases}$$

$$\equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, & \forall e \in P_k^i, \\ r_e^2 + t_e^2 + 2 \leq q_e^2, & \forall e \in P_k^i, \\ q_e \geq r_e + t_e, & \forall e \in P_k^i, \\ \sum_{e \in P_k^i} t_e \leq \alpha_k. \end{cases}$$

Using this last representation the problem can be solved using CPLEX MISOCP solver.

In Tables 3 and 4, we report computational experiments for solving the original formulation (3a)–(3h) using the classical outer approximation algorithm

Table 3 CPU Time or Final Gap for the DCRP Instances

Inst.	V	E	K	Initial formulation		Persp.	Univariate formulation			
				CPLEX	B – OA	B – OA	B – OA	B – OA + Ref	Inner sol (%)	Inner time
1	60	280	100	126.11	8.56	6.41	9.88	2.15	0.34	2.64
2	100	600	200	[2.19%]	81.62	108.63	222.05	298.28	0.25	73.71
3	100	800	500	7,905.79	53.00	119.54	27.83	60.36	0.13	25.54
4	34	160	946	[2.06%]	246.83	324.41	216.16	235.97	0.00	126.07
5	67	170	761	[∞]	1,941.05	628.68	1,359.12	376.90	0.10	188.38

implemented in BONMIN (B – OA), the SOCP formulation obtained by applying the previous transformation on delay constraints (3d) (CPLEX), the perspective formulation obtained by replacing (3d) by their perspective counterpart as proposed in Hijazi et al. (2012) (Persp.), and the univariate extended formulation (B – OA and B – OA + Ref). Table 3 reports computing time to 1% of optimality, or final gap after three hours of computations; Table 4 reports the total number of branch-and-bound nodes and of outer approximation iterations. The inner approximation heuristic was also tested separately with a time limit of 180 seconds.

First, we note that for this problem, the SOCP formulation solved by CPLEX is not efficient in practice. Meanwhile, B – OA appears to be already efficient on the original and perspective formulations. Nevertheless, B – OA + Ref is the fastest on average. On instance 5 (the hardest one), the CPU time is divided by six with respect to B – OA on the initial formulation and by two with respect to the perspective formulation. In this experiment, one can note the good performance of the inner approximation method returning a feasible solution on all instances with an optimality gap of at most 0.4% and with an average computing time of 84 seconds. It is also worth noting again the very small number of iteration of B – OA + Ref in Table 4 (at most four).

4.3. Stochastic Service Systems Design Problems (SSSD)

SSSD consists in optimally configuring the service levels of a network of $M/M/1$ queues. It was first

Table 4 Number of Nodes and Number of Master MIPs for the Big- M Formulations of DCRP Instances

Inst.	Initial formulation			Persp.			Univariate formulation			
	CPLEX		B – OA		B – OA		B – OA		B – OA + Ref	
	Nodes	It.	Nodes	It.	Nodes	It.	Nodes	It.	Nodes	It.
1	31,681	3	1,161	8	37	3	1,233	1	90	
2	132,235	4	1,767	10	5,218	5	3,705	3	5,767	
3	712,896	4	1,192	12	3,663	3	1,067	4	1,548	
4	636,213	3	2,020	11	4,767	3	2,022	2	1,026	
5	396,193	7	17,850	17	12,953	5	36,162	2	6,898	

proposed by Elhedhli (2006). We are given a set of customers J , a set of facilities I , and a set of service levels K . Each customer has a mean demand rate λ_j . The goal is to determine service rates for the facilities so that the customer demand is met. There is a fixed cost f_{jk} for operating facility j at rate k and a fixed cost c_{ij} for assigning customer i to facility j . A binary variable x_{ij} indicates if customer i is served by facility j , and a binary variable y_{jk} indicates if facility j is operated at service level k . Service level k in facility j has a predetermined mean service rate μ_{jk} and the average waiting cost per unit time is given by t . A convex MINLP model for the problem reads as follows:

$$\begin{aligned}
 \min \quad & \left\{ \sum_{i \in I, j \in J} c_{ij} x_{ij} + t \sum_{j \in J} v_j + \sum_{j \in J, k \in K} f_{jk} y_{jk} \right\} \\
 \text{s.t.} \quad & \sum_{i \in I} \lambda_i x_{ij} = \sum_{k \in K} \mu_{jk} y_{jk} \quad \forall j \in J, \\
 & \sum_{j \in J} x_{ij} = 1 \quad \forall i \in I, \\
 & \sum_{k \in K} y_{jk} \leq 1 \quad \forall j \in J, \\
 & z_{jk} - y_{jk} \leq 0 \quad \forall j \in J, k \in K, \\
 & z_{jk} - \frac{v_j}{1 + v_j} \leq 0 \quad \forall j \in J, k \in K, \\
 & z_{jk} \geq 0, v_j \geq 0, \quad \forall j \in J, k \in K \\
 & x_{ij}, y_{jk} \in \{0, 1\} \quad \forall i \in I, j \in J, k \in K,
 \end{aligned}$$

where z_{jk} and v_j are auxiliary variables aimed at ensuring convexity of the model. We note that this model also admits a perspective formulation. We will discuss this in §5.

The initial model for this problem is already in a form where not more than one univariate function appears in each constraint. Therefore, there is no reformulation to perform. In this case, we only compare B – OA, B – OA + Ref, and the inner approximation method. Let us emphasize that because variables v_j are unbounded in the model, we transform constraints $z_{jk} - v_j/(1 + v_j) \leq 0$ into $z_{jk}/(1 - z_{jk}) - v_j \leq 0$, $\forall j \in J, k \in K$, and impose $z_{jk} \leq 1 - \epsilon$, which is a valid constraint for both formulations. Tables 5 and 6 summarize the results for solving four instances of SSSD.

Table 5 CPU Times for SSSD Instances

Inst.	I	J	B – OA	B – OA + Ref	Inner gap (%)	Inner time
1	4	15	3.34	1.08	0.72	0.20
2	8	15	23.96	6.48	0.00	2.32
3	10	50	31.13	9.19	0.66	2.79
4	20	50	2,902.16	132.60	0.30	21.76
5	35	300	[∞]	5,300.12	11.78	182.05

In Table 5, we report the number of facilities in the instance $|I|$, the number of facilities $|J|$, and for each algorithm, the total CPU time or the final relative gap after reaching the time limit. The number of outer approximation iterations and the total number of nodes are reported in Table 6.

The results show a significant improvement induced by the enhancements made in B – OA + Ref. In particular, B – OA + Ref solves all instances whereas B – OA only solves four. The inner approximation method also scores a good performance on this problem. It finds a solution within 1% of optimality except in the last instance where the MIP search was interrupted by the time limit.

4.4. IBM CMU MINLP Library Instances

For completeness, we have conducted further experiments on the IBM CMU MINLP Library of convex MINLPs (Sawaya et al. 2006). A first important fact to note is that separability is a very common feature in this library of convex MINLPs coming from different applications. Eight different classes of problems are included in the library: batch synthesis problems (Ravemark and Rippin 1998, Vecchiotti and Grossmann 1999), different layout problems (Castillo et al. 2005, Sawaya 2006), synthesis (Duran and Grossmann 1986, Türkay and Grossmann 1996) and retrofit synthesis (Sawaya 2006) problems, and trimloss (Harjunkski et al. 2009) problems. Among these eight classes only one is not separable according to our definition: the trimloss instances (these instances contain functions of the form $-\sqrt{xy}$ in their formulation).

To run our modified algorithm, the models of the library were put in the form of the univariate extended formulation (sMINLP*) and trivial bounds on variables were added (these bounds are directly implied by the linear constraints of the problems and do not

Table 6 Number of OA Iterations and Nodes for SSSD Instances

Inst.	B – OA		B – OA + Ref	
	No. of it.	No. of nodes	No. of it.	No. of nodes
1	8	4,765	3	1,145
2	11	71,514	4	11,190
3	10	24,792	3	3,982
4	14	2,455,491	2	288,443
5	1	1,259,175	3	580,797

modify the continuous relaxations or the linear outer approximations). We composed the instance set presented here by selecting for each class of problems the four most difficult instances. All instances had to be reformulated except the FLayer instances that were already univariate formulations.

Table 7 presents the numerical results obtained for this set of instances. We run three algorithms: BONMIN's classical outer approximation algorithm (B – OA) on the original formulation, the refined outer approximation algorithm B – OA + Ref on the univariate extended formulation, and finally the inner approximation heuristic Inner. Both B – OA and B – OA + Ref are given a time limit of three hours of CPU time, while Inner is given a time limit of 180 seconds. We report here the total CPU time in seconds, the number of outer approximation iterations, and the cumulative number of branch-and-bound nodes of B – OA and B – OA + Ref. For Inner, we present statistics on the gap between the feasible solution returned by the inner approximation and the optimal solution and the CPU time.

First, we note that the inner approximation method fails on six problems out of 30, finds the optimal solution on about one third of the problems, and finds a solution more than 3.5% away from the optimum for six problems. Note that in the six cases where no solution was found, we could find a solution by increasing the number of discretization points to 50.

Second, we note that B – OA + Ref is faster than B – OA on 86% of the problems. On average, B – OA + Ref is much faster than B – OA with an average CPU time of 152 seconds versus 642 seconds for B – OA and a geometrical mean of 14 seconds versus 31 seconds. There is also a very significant reduction in the average number of outer approximation iterations and cumulative number of nodes. We note nevertheless that many instances are already solved in very few iterations by the classical B – OA and only limited improvement can be expected on these instances.

Figure 5 Performance Profile of Solution Time on IBM CMU Library

Table 7 CPU Times for IBM CMU MINLP Library Instances

Inst.	Initial formulation			Univariate formulation				
	B – OA			B – OA + Ref			Inner	
	CPU	It.	Nodes	CPU	It.	Nodes	Gap (%)	CPU
BatchS101006M	4.10	2	1,129	4.10	1	1,112	97.99	1.24
BatchS121208M	8.50	3	1,605	6.76	1	556	97.96	1.46
BatchS151208M	18.53	4	2,535	9.91	1	1,308	97.97	5.74
BatchS201210M	11.74	2	1,403	12.62	1	1,215	97.96	2.43
CLay0205M	10.47	6	23,604	5.18	2	7,323	0.00	4.51
CLay0303M	1.61	10	580	1.07	5	1,349	56.50	0.14
CLay0304M	13.20	16	9,958	6.32	8	6,723	44.09	0.62
CLay0305M	16.38	7	28,902	6.00	2	10,023	0.00	3.57
FLay04H	22.42	9	16,280	7.82	3	6,288	0.00	4.32
FLay04M	6.55	9	15,856	2.69	3	5,987	0.00	0.91
FLay05H	1,100.98	17	988,444	1,198.21	15	920,981	0.00	71
FLay05M	261.38	18	787,980	250.47	15	758,512	0.00	13.19
fo7_2	20.86	2	48,190	19.60	1	39,277	∞	180
fo7	60.86	4	131,937	26.85	1	50,719	∞	180
fo8	166.58	3	353,515	62.69	1	112,976	∞	180
fo9	336.43	3	638,817	357.78	1	498,581	∞	180
o7_2	1,288.53	5	3,377,463	510.61	1	1,178,907	∞	180
o7	4,948.08	9	13,657,744	1,642.46	2	3,631,904	∞	180
RSyn0810M03M	14.48	3	4,916	7.64	2	2,398	0.00	3.7
RSyn0830M02M	11.96	4	4,078	7.71	2	2,267	0.06	2.15
RSyn0830M03M	18.83	4	4,810	14.79	2	2,591	0.04	5.74
RSyn0830M04M	35.19	3	6,421	26.32	3	3,066	0.73	9.61
SLay09H	292.30	31	80,065	128.22	38	21,800	3.18	1.52
SLay09M	35.55	25	21,631	0.78	5	12	1.69	0.07
SLay10H	9,185.64	66	4,403,165	212.55	20	16,830	0.49	8.97
SLay10M	1,379.23	69	963,304	14.85	23	5,160	0.18	0.3
Syn30M04M	0.53	4	4	0.53	2	5	0.00	1.22
Syn40M02M	0.32	3	7	0.28	1	1	0.00	0.42
Syn40M03M	0.77	4	77	0.50	2	18	3.37	1.58
Syn40M04M	0.51	3	3	0.57	2	2	0.13	2.02
Arith. mean	642.42	11.60	852,480.77	151.53	5.53	242,929.70	—	40.88
Geo. mean	30.97	6.45	13,591.20	13.74	2.72	4,031.56	—	5.00

To conclude this section, we plot in Figure 5 a performance profile comparing the CPU time for solving all 46 instances considered in this section run with BONMIN's B – OA algorithm on their initial formulation and our improved implementation B – OA + Ref on the univariate formulation. The figure reflects the significant improvement brought by the refined outer approximation algorithm when applied to the univariate extended formulation. The method B – OA + Ref clearly dominates B – OA.

5. Extension to Perspective Formulations

As shown in the previous section, convex separable MINLPs are very frequent in the literature. A notable class of nonseparable formulations are the so-called perspective formulations of MINLPs with indicator variables introduced by Frangioni and Gentile (2006) and further developed by Günlük and Linderoth (2010).

MINLPs with indicator variables are problems that are driven by a collection of indicator variables; each one of which controls the activation of a subset of decision variables or constraints. The most common examples are problems where a Boolean variable z commands the value of a number of other variables by fixing them to zero whenever $z = 0$ and forcing them to belong to a bounded convex set whenever $z = 1$, i.e., sets of the forms $W = \{(\mathbf{x}, z) \in \mathbb{R}^n \times \{0, 1\} : g_i(\mathbf{x}) \leq 0, i = 1, \dots, m, \mathbf{l}z \leq \mathbf{x} \leq \mathbf{u}z\}$ with $g_i: \mathbb{R}^n \rightarrow \mathbb{R}$ such that $g_i(\mathbf{0}) \leq 0$ for $i = 1, \dots, m$. Frangioni and Gentile (2006) and Günlük and Linderoth (2010) show that the convex hull of W is the topological closure of the set $W^- = \{(\mathbf{x}, z) \in \mathbb{R}^{n+1} : zg_i(\mathbf{x}/z) \leq 0, i = 1, \dots, m, \mathbf{l}z \leq \mathbf{x} \leq \mathbf{u}z, 0 < z \leq 1\}$. The function $zg_i(\mathbf{x}/z)$ is usually called perspective function in convex analysis and for this reason W^- is referred to as the perspective formulation of W .

Problems with indicator variables are quite common. In the previous section, SQFL, DCRP, and SSSD are three examples. Both the SQFL and SSSD problems admit a perspective formulation proposed in

Günlük and Linderoth (2010). DCRP includes a different type of on and off constraint but still admits a perspective formulation proposed in Hijazi et al. (2012). We explain next how to extend our results to perspective formulations of the type of SQFL and DCRP. Extending the result to the perspective formulations of DCRP is possible by using the same type of approach but is more involved in practice. For this reason, we do not detail this case.

To obtain the perspective formulation of SQFL instances, we first need, for every $i \in I$ and $j \in J$, an additional variable ξ_{ij} that represents x_{ij}^2 ; x_{ij}^2 is then replaced by ξ_{ij} in the objective function and the constraint $x_{ij}^2 \leq \xi_{ij}$ is added. Now by applying the convexification result of Günlük and Linderoth (2010) using the indicator variables z , this latter constraint can be strengthened to $x_{ij}^2 \leq \xi_{ij}z_i$.

Likewise, in the SSSD problem, $y_{jk} = 0$ implies $z_{jk} = 0$ and therefore the constraint $z_{jk} - v_j/(1 + v_j) \leq 0$ can be replaced with $z_{jk} - v_j/(1 + v_j/y_{jk}) \leq 0$.

For our purpose, it is important to note that even if the functions $g_i(x)$ used to describe W^1 are separable, the perspective functions $zg_i(x/z)$ used to describe W^- are not. Nevertheless, our approach can be extended to perspective formulations in a simple manner. Instead of modifying the initial formulation of the model, we use the perspective formulation to generate stronger outer and inner approximation constraints. This way, the separability property of the initial model is not lost.

First, we consider the outer approximation taken in a point $(\hat{x}, \hat{z})$. The outer approximation constraints coming from the naive formulation of W , $\nabla g_i(\hat{x})^T \cdot (x - \hat{x}) + g_i(\hat{x}) \leq 0$, can be strengthened by considering the outer approximation to the perspective formulation W^- in the point $(\hat{x}, 1)$:

$$\nabla g_i(\hat{x})^T (x - z\hat{x}) + zg_i(\hat{x}) \leq 0 \quad (4)$$

(this constraint is proposed in Frangioni and Gentile 2006). Note that the latter constraint strictly dominates the initial outer approximation constraint, because the constant term is multiplied by $z \in [0, 1]$.

A stronger initial outer approximation can also be generated in a similar manner. By considering the constraints defining W , one can obtain the constraint (2).

The constraints can be lifted by introducing variable z in the following manner:

$$\frac{g_{ij}(\hat{x}_j^{k+1}) - g_{ij}(\hat{x}_j^k)}{\hat{x}_j^{k+1} - \hat{x}_j^k}(x_j) + \frac{g_{ij}(\hat{x}_j^k)\hat{x}_j^{k+1} - g_{ij}(\hat{x}_j^{k+1})\hat{x}_j^k}{\hat{x}_j^{k+1} - \hat{x}_j^k}z \leq y_{ij}.$$

By construction, these constraints are redundant when $z = 0$. They are identical to the inner approximations (2) when $z = 1$.

The strengthened cuts have been included in the experimental solver presented in the previous section. To implement it, we use the suffixes features of the AMPL modeling language (Fourer et al. 1993). Constraints to which the perspective strengthening should be applied are assigned a unique identifier with the suffix `onoff_c`; the corresponding indicator variables are attributed the same identifier through the suffix `onoff_v`. The outer and inner approximation constraints for all the corresponding constraints are then strengthened using the indicator variable.

We now present two experiments aimed at assessing the efficiency of the perspective strengthening on the two problems SQFL and SSSD and comparing it with directly solving the perspective formulation.

In Table 8, we present a comparison on the SQFL instances. The three methods tested are solving the perspective formulation given in Günlük and Linderoth (2010) with CPLEX, running $B - OA + \text{Ref}$ with the strengthened cuts ($B - OA + \text{Ref} + \text{Persp}$) and running the inner approximation heuristic. The first three columns repeat results reported in Tables 1 and 2, corresponding to the refined outer approximation without perspective cut generation. As before, we report for each instance the total CPU time of each method, the number of nodes in the branch-and-bound tree for CPLEX and for $B - OA + \text{Ref} + \text{Persp}$, the number of outer approximation iterations for $B - OA + \text{Ref} + \text{Persp}$, and the gap between the solution found by the inner approximation and the optimal solution.

As can be seen from Table 8, all algorithms using the perspective formulation are significantly better than their counterparts using the initial formulation. All instances are solved by both CPLEX and $B - OA + \text{Ref} + \text{Persp}$. The latter is faster than the former for the five

Table 8 CPU Times for Perspective Formulations of SQFL Instances

Inst.	B - OA + Ref			CPLEX		B - OA + Ref + Persp			Inner	
	CPU	It.	Nodes	CPU	Nodes	CPU	It.	Nodes	Gap (%)	CPU
1	2.20	1	33	3.06	9	1.02	1	8	0.00	0.23
2	25.06	1	541	43.20	19	10.31	1	1	0.00	5.03
3	464.40	2	1,676	85.45	1	37.11	2	2	0.00	26.12
4	338.32	2	980	261.41	89	73.82	2	47	2.13	35.14
5	[4.57%]	3	1,799	2,218.88	1	1,712.66	6	153	0.62	27.90
6	[51.78%]	1	199	1,226.41	1	4,218.87	6	22	0.91	87.78

Table 9 Time, Nodes, and Iterations for SSSD Instances with Perspective Instances

Inst.	B – OA + Ref			CPLEX		B – OA + Ref + Persp			Inner	
	CPU	It.	Nodes	CPU	Nodes	CPU	It.	Nodes	Gap (%)	CPU
1	1.08	3	1,145	0.45	886	0.41	3	203	0.51	0.15
2	6.48	4	11,190	2.73	8,532	5.72	3	10,304	0.28	3.07
3	9.19	3	3,982	4.78	3,225	5.20	2	1,638	0.48	0.83
4	132.60	2	288,443	739.13	286,996	301.92	3	757,568	0.00	92.19
5	5,300.12	3	580,797	[20.72%]	242,092	[0.34%]	2	1,177,617	12.05	181.36

smallest instances but CPLEX is significantly faster on the largest instance. The inner approximation also performs significantly better with strengthened cuts; it finds a feasible solution for all problems and the maximal gap is 2%.

In Table 9, we present a comparison on the SSSD instances. The three methods tested and the information reported are the same as in Table 8.

The comparison between the initial and perspective formulations of the SSSD instances is less favorable than for the SQFL problem, only the first three instances benefit from the strengthened cuts, nevertheless, the enhanced OA outperforms CPLEX on these instances.

6. Conclusion

In this paper, we have shown how separability in MINLPs can be used to improve outer approximation schemes, both theoretically and in practice. Computational results have shown that our methods are successful on a broad family of convex MINLPs including perspective formulations. Although our improvements have been applied to the outer approximation decomposition algorithm, they could as well be applied in the context of an outer approximation-based branch and cut. We believe that this would lead to similar improvements.

An interesting line of research to extend our results would be to consider problems with nonseparable nonlinear functions. Only few convex instances from the literature do not fit our separability assumptions. We have shown an extension to the case of perspective formulations. Other instances that we know of are the trimloss instances of Harjunkoski et al. (2009). These instances are known to be very difficult to solve (as far as we know, only the three smallest have been solved to optimality). As noted before, these instances use functions of the form $-\sqrt{xy}$ that are not separable. Of course, it is still possible to apply ideas similar to the univariate formulation and the refined outer approximation to these instances. A preliminary experiment indicates that doing so improves the solution time. Generalizing, the inner approximation is more involved, as it requires the inner approximation of a function from $\mathbb{R}^2$ to $\mathbb{R}$. We believe it is an interesting line for future research and the good results we

obtained with separable functions lead us to believe there is potential for further improvements.

Acknowledgments

The authors thank Jeff Linderoth and Mustafa Kılınç for sharing the UFL and SSSD instances with us. The first author's research is currently supported by [Grants Digiteo Emergence PASO, Digiteo Chair 2009-14D RMNCCO, and Digiteo Emergence 2009-55D ARM]. A major part of this work was accomplished at Orange Labs R&D. The second author's research was supported by ANR [Grant BLAN06-1-0375] and the Google Research Award.

References

- Abhishek K, Leyffer S, Linderoth J (2010) FilMINT: An outer approximation-based solver for convex mixed-integer nonlinear programs. *INFORMS J. Comput.* 22(4):555–567.
- Ben Ameer W, Ouorou A (2006) Mathematical models of the delay constrained routing problem. *Algorithmic Oper. Res.* 1(2):94–103.
- Bonami P, Kılınç M, Linderoth J (2012) Algorithms and software for convex mixed integer nonlinear programs. Lee J, Leyffer S, eds. *Mixed Integer Nonlinear Programming, The IMA Volumes in Mathematics and Its Applications*, Vol. 154 (Springer, New York), 1–39.
- Bonami P, Biegler LT, Conn AR, Cornuéjols G, Grossmann IE, Laird CD, Lee J, et al. (2008) An algorithmic framework for convex mixed-integer nonlinear programs. *Discrete Optim.* 5(2):186–204.
- Castillo I, Westerlund J, Emet S, Westerlund T (2005) Optimization of block layout design problems with unequal areas: A comparison of MILP and MINLP optimization methods. *Comput. Chemical Engrg.* 30(1):54–69.
- Duran MA, Grossmann I (1986) An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Math. Programming* 36(3):307–339.
- Elhedhli S (2006) Service system design with immobile servers, stochastic demand, and congestion. *Manufacturing Service Oper. Management* 8:92–97.
- Fletcher R, Leyffer S (1994) Solving mixed-integer nonlinear programs by outer approximation. *Math. Programming* 66:327–349.
- Fourer R, Gay DM, Kernighan BW (1993) *AMPL: A Modeling Language for Mathematical Programming* (Duxbury Press, Pacific Grove, CA).
- Frangioni A, Gentile C (2006) Perspective cuts for a class of convex 0-1 mixed-integer programs. *Math. Programming* 106:225–236.
- Grossmann I, Viswanathan J, Vecchietti A, Raman R, Kalvelagen E (2001) GAMS/DICOPT: A discrete continuous optimization package. *Math. Methods Appl. Sci.* 11:649–664.
- Günlük O, Linderoth J (2010) Perspective relaxation of mixed-integer nonlinear programs with indicator variables. *Math. Programming* 124:183–205.
- Günlük O, Lee J, Weismantel R (2007) MINLP strengthening for separable convex quadratic transportation-cost UFL. Technical Report RC24213 (W0703-042), IBM Research Division, T.J. Watson Research Center, Cambridge, MA.

- Harjunkski I, Pörn R, Westerlund T (2009) MINLP: Trim-loss problem. Floudas CA, Pardalos PM, eds. *Encyclopedia of Optimization* (Springer, New York), 2190–2198.
- Hijazi H (2010) Mixed-integer nonlinear optimization approaches for network design in telecommunications. Ph.D. thesis, Aix-Marseille Université, Marseille, France.
- Hijazi H, Bonami P, Cornuéjols G, Ouorou A (2012) Mixed-integer nonlinear programs featuring on/off constraints. *Comput. Optim. Appl.* 52:537–558.
- Lougee-Heimer R (2003) The common optimization interface for operations research. *IBM J. Res. Development* 47:57–66.
- Quesada I, Grossmann IE (1992) An LP/NLP based branch-and-bound algorithm for convex MINLP optimization problems. *Comput. Chemical Engrg.* 16(10–11):937–947.
- Ravemark DE, Rippin DWT (1998) Optimal design of a multiproduct batch plant. *Comput. Chemical Engrg.* 22(1–2):177–183.
- Sawaya N (2006) Reformulations, relaxations and cutting planes for generalized disjunctive programming. Ph.D. thesis, Chemical Engineering Department, Carnegie Mellon University, Pittsburgh, PA.
- Sawaya N, Laird CD, Biegler LT, Bonami P, Conn AR, Cornuéjols G, Grossmann IE, et al. (2006) CMU-IBM open source MINLP project test set. Accessed December 2011, <http://egon.cheme.cmu.edu/ibm/page.htm>.
- Tawarmalani M, Sahinidis NV (2005) A polyhedral branch-and-cut approach to global optimization. *Math. Programming* 103(2):225–249.
- Türkay M, Grossmann IE (1996) Logic-based MINLP algorithms for the optimal synthesis of process networks. *Comput. Chemical Engrg.* 20(8):959–978.
- Vecchiotti A, Grossmann IE (1999) LOGMIP: A disjunctive 0-1 nonlinear optimizer for process system models. *Comput. Chemical Engrg.* 23(4–5):555–565.
- Wächter A, Biegler LT (2006) On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Math. Programming* 106(1):25–57.