An Outer-Inner Approximation for separable MINLPs

Hassan Hijazi
LIX, École Polytechnique, F-91128 Palaiseau, France, hijazi@lix.polytechnique.fr

Pierre Bonami
LIF, Faculté des Sciences de Luminy, Université de Marseille, France, pierre.bonami@lif.univ-mrs.fr.

Adam Ouorou
Orange Labs R&D/CORE/TPN, 38-40 rue du Général Leclerc, 92794 Issy-Les-Moulineaux cedex 9, France, adam.ouorou@orange.com

A common structure in convex mixed-integer nonlinear programs 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. These methods have been implemented in the open source solver Bonmin and are available for download from the COIN-OR 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 average computing time on most tested instances.

Key words: Mixed-Integer Nonlinear Programming, Outer Approximation.

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

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

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 $X := \{\mathbf{x} \in \mathbb{R}^n : \mathbf{A}\mathbf{x} \le \mathbf{b}, \mathbf{l} \le \mathbf{x} \le \mathbf{u}\}$ the region described by the linear constraint of the problem. The functions $g_i : X \to \mathbb{R}$ are assumed to be continuously differentiable and *convex separable* $\forall i \in \{1,\dots,m\}$. Convex separable functions can be written as the sum of the compositions of univariate convex functions and affine functions:

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

(where, for $i = 1,\dots,m$ and $j = 1,\dots,q_i$, $\alpha_{ij} \in \mathbb{R}^n$ and $\beta_{ij} \in \mathbb{R}$).

Note that any problem with a convex separable objective function and constraints can be cast into that form by introducing an extra variable and moving the objective function into the constraints. The separability requirement might seem restrictive in a first look. Let us emphasis 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 Section 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 expect outer approximation to 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 simple extended formulation which can be shown to have better convergence performances. 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 find 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 make this approximation simple to build. Indeed, it suffices to compute chords of each univariate convex function separately.

In the next section, we briefly recall the outer approximation algorithm, and show an example where outer approximation has to perform an exponential number of iterations. In Section 3, we describe our approach. In Section 4, we present computational experiments comparing our approach to state-of-the art solvers. Finally, in Section 5, we present an extension to perspective formulations of problems with indicator variables.

In the remainder, we denote by $\mathbf{g}$, the function $X \to \mathbb{R}^m$ consisting of all the functions g_i and by $I = \{1, 2, .., p\}$ 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 sub-vector with indices in I.

2 Outer Approximation

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

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

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 polyhedral set given by these outer approximations constraints by

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

and the corresponding MILP:

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

The main theoretical justification of outer approximation algorithms is that if (OA($\mathcal{P}$)) contains a suitable set of points then it 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:

$$\boldsymbol{\xi}(\mathbf{y}) := \begin{cases} \arg\min \;\{\mathbf{c}^T\mathbf{x} : \mathbf{g}(\mathbf{x}) \leq 0,\, \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x}_I = \mathbf{y}\} & \text{if it exists,} \\ \arg\min \;\{\sum\limits_{i=1}^{m} \max(g_i(\mathbf{x}), 0) : \mathbf{A}\mathbf{x} \leq \mathbf{b}, \mathbf{x}_I = \mathbf{y}\} & \text{otherwise,} \end{cases}$$

and $\mathcal{P}(\mathcal{Y}) := \{\boldsymbol{\xi}(\mathbf{y}) : \text{for all } \mathbf{y} \in \mathcal{Y}\}$. 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. 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 will mostly consider the first of these two algorithms which is specified in Algorithm 1. Note nevertheless that 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. 2009).

Algorithm 1 The Outer Approximation Decomposition Algorithm

```
0. Initialize.
   $z_U \leftarrow +\infty$. $z_L \leftarrow -\infty$. $\mathbf{x}^* \leftarrow$ NONE. Let $\mathbf{x}^0$ be an optimal solution
   of the continuous relaxation of (sMINLP), $\mathcal{P} \leftarrow \{\mathbf{x}^0\}$.
1. Termination?
   If $z_U - z_L = 0$ or OA($\mathcal{P}$) is infeasible, then $\mathbf{x}^*$ is optimal.
2. Lower Bound
   Let $z_{\mathrm{OA}(\mathcal{P})}$ be the optimal value of OA($\mathcal{P}$) and $\hat{\mathbf{x}}$ its optimal solu-
   tion. $z_L \leftarrow z_{\mathrm{OA}(\mathcal{P})}$
3. Find new linearization point
   $\mathcal{P} \leftarrow \mathcal{P} \cup \{\xi(\hat{\mathbf{x}}_I)\}$.
4. Upper Bound?
   If $\xi(\hat{\mathbf{x}}_I)$ is feasible for (sMINLP) and $\mathbf{c}^T\xi(\hat{\mathbf{x}}_I) < z_U$, then $\mathbf{x}^* \leftarrow$
   $\xi(\hat{\mathbf{x}}_I)$ and $z_U \leftarrow \mathbf{c}^T\xi(\hat{\mathbf{x}}_I)$.
5. Iterate
   $i \leftarrow i + 1$. Go to 1.
```

The convergence of Algorithm 1 follows directly from the equivalence between 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. 2009 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. Below, we exhibit a simple example showing that this is not the case and even if there is one nonlinear constraint with quadratic separable terms the outer approximation algorithm may take 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 $r = \frac{\sqrt{n-1}}{2}$ centered in the point $\rho = \left(\frac{1}{2}, \ldots, \frac{1}{2}\right)$ with the vertices of the unit hypercube $\{0,1\}^n$:

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

Clearly, since the vertices of the hypercube are at distance $\frac{\sqrt{n}}{2}$ from ρ, $B^n = \emptyset$. Note also that the ball is chosen so that it has a non-empty 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 2 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 $B(\rho, \frac{\sqrt{n}}{2})$ 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 non-empty intersection with $[x, y]$, therefore the inequality is not valid for $B(\rho, r)$. $\square$

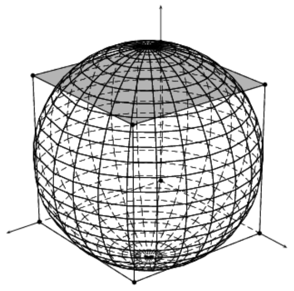

Figure 1: The set $\mathcal{B}^3$ in $\mathbb{R}^3$.

It follows directly from the lemma 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. Since, no linear inequality can cut two vertices of the hypercube simultaneously, if $|\mathcal{P}| < 2^n$, at least one vertex is feasible. □

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 presents 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, we assume that $\alpha_{ij} = e_j$ (with e_j the unit vector with component j equal to 1) and $\beta_{ij} = 0$, for all $i = 1, \ldots, m$, $j = 1, \ldots, q_i$. Note that this is without loss of generality since it can always be achieved by adding sufficiently 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 simple reformulation of (sMINLP) in an extended space. All separable functions in the problem are broken by introducing an auxiliary variable for each univariate function:

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

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 mean 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 (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 relaxation of (sMINLP) of the same strength as a linear 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.

Example1 (continued)
We denote by B^{n*} the extended formulation of the mixed-integer set B^n:

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

Given a set of point $\mathcal{P} := \left\{ \begin{pmatrix} \hat{\mathbf{x}}^{(1)} \\ \hat{\mathbf{z}}^{(1)} \end{pmatrix}, \ldots, \begin{pmatrix} \hat{\mathbf{x}}^{(K)} \\ \hat{\mathbf{z}}^{(K)} \end{pmatrix} \right\}$, the outer approximation of B^{n*} is given by:

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

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

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

This set is empty since $x \in \{0,1\}^n$ implies $z_i \geq \frac{1}{4}$ for $i = 1, \ldots, n$ and $\sum\limits_{i=1}^{n} z_i \geq \frac{n}{4} > \frac{n-1}{4}$. This shows that on this simple problem while the classical version of the outer approximation takes 2^n iterations, the extended formulation can be solved in only 2 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 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 just 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 $S = \{(x_j, y_{ij}) \in \mathbb{R}^2\ s.t.\ g_{ij}(x_j) \leq y_{ij},\ l_j \leq x_j \leq u_j\}$. We enrich the first outer approximation with a set of points of the form

$(\hat{x}_j, g_{ij}(\hat{x}_j))$, for $\hat{x}_j \in [l_j, u_j]$:

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

This set may be obtained in different ways. For example, one could sample 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 function curvature. After testing the three methods, in our experiments, we chose uniform sampling. It typically provided good computational results. Figures 2 and 3 give example of S and S_{out} for the constraint $\frac{1}{6-x} \le y$.

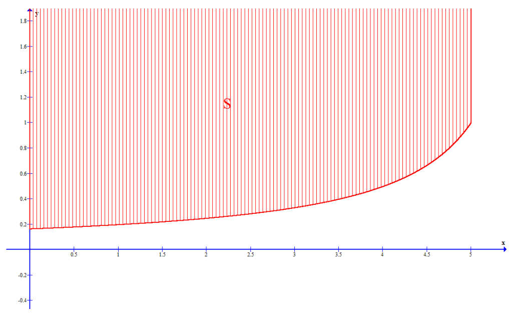

Figure 2: $S = \{(\mathbf{x}, \mathbf{y}) \in \mathbb{R}^2\ s.t.\ \frac{1}{6-x} \le y,\ 0 \le x \le 5\}$.

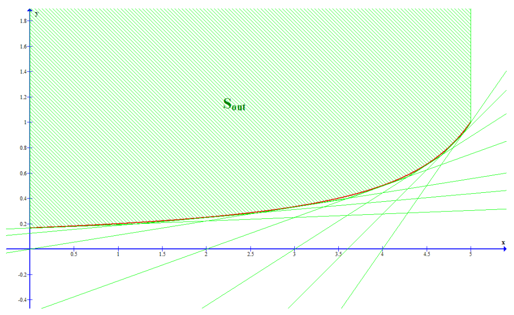

Figure 3: S_{out}.

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 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*) $g_{ij}(x_j) \le y_{ij}$ and the set $S = \{(x_j, y_{ij}) \in \mathbb{R}^2\ s.t.\ g_{ij}(x_j) \le y_{ij},\ l_j \le x_j \le u_j\}$. Instead of taking tangents, we now take chords at the boundary of S. Consider a sequence of K points $(\hat{x}_j^k, g_{ij}(\hat{x}_j^k))$, $k \in \{1, 2, ..., K\}$ such that $l_j = \hat{x}_j^1 \le \hat{x}_j^2 \le \ldots \le \hat{x}_j^K = u_j$. We consider the polyhedral approximation S_{in} of S formed by the constraints:

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

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

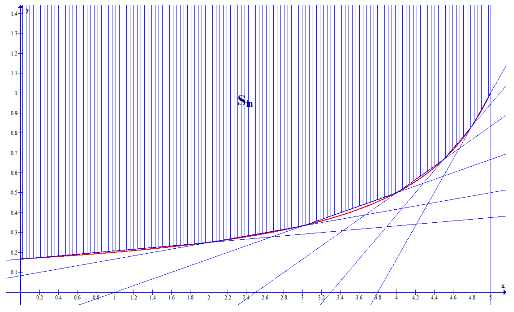

Figure 4: Inner approximation of S with 7 discretization points.

Computing this inner approximation for every nonlinear constraint of the problem, we obtain the Mixed-Integer Linear Program

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

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

Since 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 heurisitic. 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 scheme which hopefully results in a smaller initial gap thanks to good 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 Outer-Inner Approximation in 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 sub-directory `experimental/Separable`. Our program takes as input a convex separable MINLP under the form (sMINLP*) (i.e., already reformulated) and applies to it an outer approximation decomposition scheme augmented with our improvements:

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

The number of discretization points at which the linear approximations are built is a user-set parameter and is initialized to 8 in our experiments.

Below, we present experiments aimed at comparing our algorithm to the standard outer approximation algorithm implemented in Bonmin, called `B-OA`, and when possible to the state of the art solver CPLEX. 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. Second, we report experiments on a larger test set of problems from the IBM CMU MINLP Library (Sawaya et al., 2006).

All our experiments have been performed on a machine equipped with 4 Intel Quad Core Xeon 2.93 GHz CPUs and 120 GiB 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. We use the deterministic multi-threaded version of CPLEX on 10 threads.

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 facilities 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:

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

In this experiment, our main focus is to compare the new scheme implemented in Sepa to the classical outer approximation algorithm `B-OA` found in Bonmin. For completeness, we also compare these two formulations and algorithms with the quadratic solver CPLEX. Let us emphasize that a tighter non-separable formulation for this problem have been proposed by Günlük and Linderoth (2010) using perspective functions, this formulation will be considered in section 5. In Tables 1 and 2, we present statistics on the solution of 6 instances of SQFL of increasing size. We report results using 5 different algorithms. First, on the initial formulation, we consider CPLEX's Mixed Integer Quadratic Programming solver (labeled `CPLEX`) and Bonmin's Outer Approximation algorithm (labeled `B-OA`). Second, on the extended formulation, we compare (`B-OA`) to its variant augmented with a refined initial outer approximation (labeled `B-OA+Ref`) and the inner approximation heuristic. All algorithms where given a time limit of three hours of CPU time except the inner approximation heuristic which was given a time limit of 180 seconds.

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 which has only 8 candidate locations (and therefore 8 binary variables) requires the solution of 110 MILPs which is not so far from the worst case of 2^8 iterations. Using the extended formulation with `B-OA` allows to prove optimality of four instances, but it is still significantly slower than `CPLEX`. `B-OA+Ref` solves the same instances as `CPLEX` and `B-OA` on the extended formulation, but it is significantly faster than `B-OA` and comparable with

			Initial Formulation		Extended formulation			
Inst.	$\|I\|$	$\|J\|$	`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%]	3075.63	464.40	1.21%	71.40
4	30	150	249.31	[48.79%]	2496.28	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 1: CPU times for SQFL instances

	Initial Formulation			Extended formulation			
Inst.	`CPLEX`	`B-OA`		`B-OA`		`B-OA+Ref`	
	# Nodes	#It.	# Nodes	#It.	# Nodes	#It.	# Nodes
1	92	110	13627	11	243	1	33
2	84	230	50502	19	2416	1	54
3	3116	121	31664	20	11396	2	1676
4	2227	116	13538	28	14623	2	980
5	49780	84	740	16	8415	3	1799
6	327	49	159	2	25357	1	199

Table 2: Nodes and iterations for SQFL instances.

`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 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 it is given more time). Finally, from Table 2, we note the very small number of iterations of `B-OA+Ref` (3 at most) and the small number of branch-and-bound nodes (always inferior to `CPLEX`).

4.2 Delay constrained routing problem

The delay constrained routing problem consists in finding an optimal routing in a telecommunication network that satisfies both a given demand and constraints on the 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 demands which are described by the set of paths they can use $P(k) = \{P_k^1, P_k^2, ..., P_k^{n_k}\}$, the quantity of flow v_k that need to be routed and the maximum delay for this demand α_k. A feasible routing is an assignment of a sufficient flow to candidate paths to satisfy demand and 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 $\sum_{e\in P_k^i} \frac{1}{c_e - x_e}$ where x_e is the total quantity of flow going through edge e. Although several formulations of this problem have been proposed (see Ben Ameur and Ouorou, 2006; Hijazi et al., 2011), we only give here the so called big-M formulation. We also report results on a tighter (non-separable) formulation that uses perspective function proposed by Hijazi et al. (2011). Denoting by x_e the total quantity of flow routed through edge e, ϕ_k^i the quantity of flow routed along path P_k^i, and z_k^i an indicator variable indicating if path P_k^i is active, the model reads

$$\begin{aligned}
\min \ & \sum_{e\in E} w_e x_e \\
\text{s.t.} & \sum_{i=1}^{n_k} \phi_k^i \geq 1, && \forall k \in K \\
& \sum_{k\in K} \sum_{P_k^i \ni l} \phi_k^i v_k \leq x_e, && \forall e \in E \\
& x_e \leq c_e, \ \forall e \in E \\
& \sum_{e\in P_k^i} \frac{1}{c_e - x_e} \leq M - z_k^i (M - \alpha_k), && \forall k \in K, \ \forall P_k^i \in P(k). \\
& \sum_{P_k^i \in P(k)} z_k^i \leq N, && \forall k \in K \\
& \phi_k^i \leq z_k^i, && \forall k \in K, \ \forall P_k^i \in P(k) \\
& z_k^i \in \{0,1\}, && \forall k \in K, \ \forall P_k^i \in P(k) \\
& \phi_k^i \in [0,1], && \forall k \in K, \ \forall P_k^i \in P(k) \\
& x_e \in \mathbb{R}, && \forall e \in E.
\end{aligned}$$

The model is clearly separable since the only nonlinear function is $\sum\limits_{e\in P_k^i} \frac{1}{c_e - x_e}$. Also, as shown in (Hijazi, 2010), the delay constraints are second order cone representable. Let $r_e = c_e - x_e$ denote the residual capacity on link e and $t_e = \frac{1}{r_e}$ the corresponding delay value. the delay constraints, denoted $\mathcal{C}_{del}$, are equivalent to the following system of inequalities:

$$\mathcal{C}_{del} \equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, \ \forall e \in P_k^i \\ \frac{1}{r_e} \leq t_e, \ \forall e \in P_k^i \\ \sum\limits_{e\in P_k^i} t_e \leq \alpha_k \end{cases} \equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, \ \forall e \in P_k^i \\ 0 \leq t_e r_e - 1, \ \forall e \in P_k^i \\ \sum\limits_{e\in P_k^i} t_e \leq \alpha_k \end{cases}$$

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

$$\mathcal{C}_{del} \equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, \ \forall e \in P_k^i \\ 0 \leq q_e^2 - r_e^2 - t_e^2 - 2, \ \forall e \in P_k^i \\ q_e \geq r_e + t_e, \ \forall e \in P_k^i \\ \sum\limits_{e\in P_k^i} t_e \leq \alpha_k \end{cases} \equiv \begin{cases} 0 \leq r_e \leq c_e - x_e, \ \forall e \in P_k^i \\ r_e^2 + t_e^2 + 2 \leq q_e^2, \ \forall e \in P_k^i \\ q_e \geq r_e + t_e, \ \forall e \in P_k^i \\ \sum\limits_{e\in P_k^i} t_e \leq \alpha_k \end{cases}$$

Thus, using this representation, the problem can be solved using CPLEX Second Order Cone Programming (SOCP) solver.

In Tables 3 and 4, we report computational experiments of solving the original formulation, the perspective formulation and the extended formulation. Like previously, Table 3 reports computing time to 1% of optimality, or final gap after 3 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 6 with respect to `B-OA` on the initial formulation and by 2 with respect to the perspective formulation. In this experiment, one can note the good performance of the inner approximation method returning a feasible

				Initial Formulation		Persp.	Extended formulation			
Inst.	$\lvert V\rvert$	$\lvert E\rvert$	$\lvert K\rvert$	`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	[∞]	1941.05	628.68	1,359.12	376.90	0.10%	188.38

Table 3: CPU time or final gap for the DCRP instances.

	Initial Formulation			Persp.		Extended formulation			
Inst.	`CPLEX`	`B-OA`		`B-OA`		`B-OA`		`B-OA+Ref`	
	nodes	It.	nodes	It.	nodes	It.	nodes	It.	nodes
1	31681	3	1161	8	37	3	1233	1	90
2	132235	4	1767	10	5218	5	3705	3	5767
3	712896	4	1192	12	3663	3	1067	4	1548
4	636213	3	2020	11	4767	3	2022	2	1026
5	396193	7	17850	17	12953	5	36162	2	6898

Table 4: Number of nodes and number of Master MIPs for the big-M formulations of DCRP instances.

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 4).

4.3 Stochastic Service Systems Design Problems (SSSD)

SSSD consist in configuring optimally the service levels of a network of M/M/1 queues. It was first 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 for operating facility j at rate k, a fixed cost 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}. A convex MINLP model for the problem reads:

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

(where z_{jk} and v_j are auxiliary variable aimed at ensuring convexity of the model). We note that this model also admits a perspective formulation. We will discuss it in Section 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 point here in applying the extended formulation. In this

Inst.	$\lvert I\rvert$	$\lvert J\rvert$	`B-OA`	`B-OA+Ref`	`Inner Gap`	`Inner Time`
1	4	15	3.34	1.08	0.72%	0.2
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	[0.00%]	132.60	0.30%	21.76
5	35	300	[∞]	5300.12	11.78%	182.05

Table 5: CPU times for SSSD instances.

Inst.	`B-OA`		`B-OA+Ref`	
	#it.	# nodes	#it.	# nodes
1	8	4765	3	1145
2	11	71514	4	11190
3	10	24792	3	3982
4	14	2455491	2	288443
5	1	1259175	3	580797

Table 6: Number of OA iterations and nodes for SSSD instances.

case, we only compare (`B-OA`), (`B-OA+Ref`) and the inner approximation method. Let us emphasize that since variables v_j are unbounded in the model, we transform constraints $z_{jk} - \frac{v_j}{1+v_j} \leq 0$ into $\frac{z_{jk}}{1-z_{jk}} - v_j \leq 0$, $\forall j \in J$, $k \in K$, and impose $z_{jk} \leq 1 - \epsilon$, which is a valid constraint for both formulations. Tables 5 and 6 summarize the results for solving 4 instances of SSSD applying the same conditions as before. 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 is 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 while `B-OA` only solves the three easiest. 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 Sawaya et al. (2006) of convex MINLPs. 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; Vecchietti and Grossmann, 1999), different layout problems (Castillo et al., 2005; Sawaya, 2006), Synthesis (Sawaya, 2006) and Retrofit Synthesis (Duran and Grossmann, 1986; Türkay and Grossmann, 1996) problems and trimloss (Harjunkoski et al., 2009) problems. Among these 8 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 extended formulation (sMINLP*) and trivial bounds on variables were added (these bounds do not modify the continuous relaxation of the problems). We composed the instance set presented here by selecting for each class of problem the four most difficult instances.

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 extended formulation, and finally the inner approximation heuristic. `B-OA` and `B-OA+Ref` were given a time limit of three hours of CPU time, while `Inner` was given a time limit of 180 seconds. We report the total CPU time of `B-OA` and `B-OA+Ref`. The last two columns present statistics on the computing time of the inner approximation and the gap of between the feasible solution returned by the inner approximation and the optimal solution.

First, we note that the inner approximation method fails on six problems out of thirty, finds the optimal

	Initial formulation	Reformulation		
Inst.	`B-OA`	`B-OA+Ref`	`Inner Gap`	`Inner Time`
BatchS101006M	4.10	**4.10**	97.99%	1.24
BatchS121208M	8.50	**6.76**	97.96%	1.46
BatchS151208M	18.53	**9.91**	97.97%	5.74
BatchS201210M	**11.74**	12.62	97.96%	2.43
CLay0205M	10.47	**5.18**	0.00%	4.51
CLay0303M	1.61	**1.07**	56.50%	0.14
CLay0304M	13.20	**6.32**	44.09%	0.62
CLay0305M	16.38	**6.00**	0.00%	3.57
FLay04H	22.42	**7.82**	0.00%	4.32
FLay04M	6.55	**2.69**	0.00%	0.91
FLay05H	**1100.98**	1198.21	0.00%	71
FLay05M	261.38	**250.47**	0.00%	13.19
fo7_2	20.86	**19.60**	$[\infty]$	180
fo7	60.86	**26.85**	$[\infty]$	180
fo8	166.58	**62.69**	$[\infty]$	180
fo9	**336.43**	357.78	$[\infty]$	180
o7_2	1288.53	**510.61**	$[\infty]$	180
o7	4948.08	**1642.46**	$[\infty]$	180
RSyn0810M03M	14.48	**7.64**	0.00%	3.7
RSyn0830M02M	11.96	**7.71**	0.06%	2.15
RSyn0830M03M	18.83	**14.79**	0.04%	5.74
RSyn0830M04M	35.19	**26.32**	0.73%	9.61
SLay09H	292.30	**128.22**	3.18%	1.52
SLay09M	35.55	**0.78**	1.69%	0.07
SLay10H	9185.64	**212.55**	0.49%	8.97
SLay10M	1379.23	**14.85**	0.18%	0.3
Syn30M04M	0.53	**0.53**	0.00%	1.22
Syn40M02M	0.32	**0.28**	0.00%	0.42
Syn40M03M	0.77	**0.50**	3.37%	1.58
Syn40M04M	**0.51**	0.57	0.13%	2.02

Table 7: CPU times for IBM CMU MINLP Library instances.

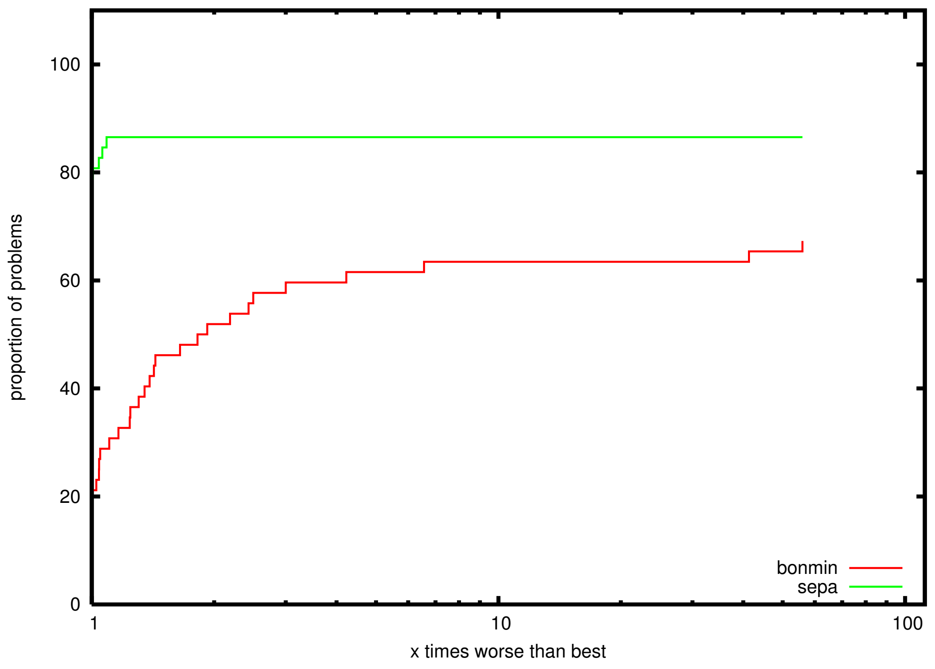

Figure 5: Performance profile of solution time on IBM CMU library.

solution on one third of the problems, and finds a solution more than 3.5% away from the optimum for 6 problems. Note that in the six cases were no solution was found, we could find a solution by increasing the number of discretization points up to 50.

Second, we note that `B-OA+Ref` is faster than `B-OA` on 86% of problems. On average,`B-OA+Ref` is much faster than `B-OA` with an average CPU time of 152 secs. vs. 642 secs. for `B-OA` and a geometrical mean of 14 secs. vs 31 secs. The node reduction is also very significant `B-OA+Ref` needs on average 3.5 times less nodes than `B-OA` (the reduction factor is 3.4 if one considers geometrical means).

To conclude this section, we plot in Figure 5 a performance profile comparing the CPU time for solving all the instances presented in this section ran with Bonmin's `B-OA` algorithm and our improved implementation `B-OA+Ref`. This figure reflects the significant improvement brought by the refined outer approximation algorithm when applied to the extended formulation.

5 Extension to perspective formulations

As shown in the previous section, convex separable MINLPs are very frequent in the literature. A notable exception of non-separable 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 which are driven by a collection of indicator variables where each indicator variable control 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 0 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 $g_i : \mathbb{R}^n \to \mathbb{R}$ 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 closure of the set $W^- = \{(\mathbf{x}, z) \in \mathbb{R}^{n+1} : zg_i(\mathbf{x}/z) \leq 0, i = 1, \dots, m, \mathbf{l}z \leq x \leq \mathbf{u}z, 0 < z \leq 1\}$. The function $zg_i(\mathbf{x}/z)$ is usually called perspective function in convex analysis and for this reason W^- is called perspective formulation of W.

Problems with indicator variables are quite common. For example, in our previous section, both the SQFL and SSSD problems admit a perspective formulation proposed in Günlük and Linderoth (2010).

DCRP includes a different type of on/off constraints but still admits a perspective formulation proposed in Hijazi et al. (2011).

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 $z_{jk} - \frac{v_j}{1+v_j} \leq 0$ can be replaced with $z_{jk} - \frac{v_j}{1+v_j/y_{jk}} \leq 0$

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

$$\nabla g_i(\hat{\mathbf{x}})^T(\mathbf{x} - z\hat{\mathbf{x}}) + z g_i(\hat{\mathbf{x}}) \leq 0, \tag{3}$$

(this constraint is the one proposed in Frangioni and Gentile (2006)). Note that the latter constraint strictly dominates the initial outer approximation constraint, since 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 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 attributed a unique identifier with the suffix `onoff_c`, the corresponding indicator variables is attributed the same identifier through the suffix `onoff_v`. The outer and inner approximations 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 solving directly the perspective formulation.

In Table 8, we present a comparison on the SQFL instances. The three method tested are solving the perspective formulation given in Günlük and Linderoth (2010) with `CPLEX`, running `B-OA+Ref` with the strengthened cuts and running the inner approximation heuristic. 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 B-OA+Ref, the number of OA iterations of B-OA+Ref, and the gap between the solution found by the inner approximation and the optimal solution.

As can been seen from the table, all the 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+Ref`. `B-OA+Ref` is faster than `CPLEX` for the five smallest instances but `CPLEX` is significantly faster on the largest instance. The inner approximation also performs significantly better, it finds a feasible solution to all problem with a maximal gap of 2%.

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

The comparison between the initial (Table 5) 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.

Inst.	CPLEX		B-OA+Ref			Inner	
	CPU	Nodes	CPU	It.	Nodes	Gap	Time
1	3.06	9	1.02	1	8	0.00%	0.23
2	43.20	19	10.31	1	1	0.00%	5.03
3	85.45	1	37.11	2	2	0.00%	26.12
4	261.41	89	73.82	2	47	2.13%	35.14
5	2218.88	1	1712.66	6	153	0.62%	27.90
6	1226.41	1	4218.87	6	22	0.91%	87.78

Table 8: CPU times for perspective formulations of SQFL instances.

Inst.	CPLEX		B-OA+Ref			Inner	
	CPU	Nodes	CPU	It.	Nodes	Gap	Time
1	0.454931	886	0.413937	3	203	0.51%	0.15
2	2.73359	8532	5.728129	3	10304	0.28%	3.07
3	4.78927	3225	5.205208	2	1638	0.48%	0.83
4	739.132	286996	301.9271	3	757568	0.00%	92.19
5	[20.72%]	242092	[0.34%]	2	1177617	12.05%	181.36

Table 9: Time, nodes and iterations for SSSD instances with perspective 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 scheme they could as well be applied in the context of an outer approximation based branch-and-cut. As a conclusion and suggestion for extensions, we outline a few ideas for applying similar approaches to more general classes of MINLP. First, note that the transformation we have applied to obtain (sMINLP*) is similar to the classical Mc-Cormick reformulation applied in global optimization and thus similar inner-approximation schemes could be tried for very broad classes of MINLPs. Second, the scheme could also be applied to problems featuring quadratic constraints by using spectral decomposition to make each constraint separable in a similar fashion as in Saxena et al. (2009). Finally, the inner approximation heuristic may be extended to higher dimensional spaces using the description of the outer approximation polytope which applies in any finite dimension. The main idea is to sufficiently shrink the polytope in order to guarantee the feasibility of its extreme points. These underlined issues represent a line of research for future contributions.

Acknowledgments

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. Second author's research was supported by ANR Grant BLAN06-1-0375 and Google Research Award. We thank Jeff Linderoth and Mustafa Kılınç for sharing the UFL and SSSD instances with us.

References

Abhishek, K., S. Leyffer, J.T. Linderoth. 2010. FilMINT: An Outer Approximation-Based Solver for Convex Mixed-Integer Nonlinear Programs. *INFORMS Journal on Computing* ijoc.1090.0373doi:10.1287/ijoc.1090.0373.

Ben Ameur, W., A. Ouorou. 2006. Mathematical models of the delay constrained routing problem. *Algorithmic Operations Research* **1** 94–103.

Bonami, P., L. T. Biegler, A. R. Conn, G. Cornuéjols, I. E. Grossmann, C. D. Laird, J. Lee, A. Lodi, F. Margot, N. Sawaya, A. Wächter. 2008. An algorithmic framework for convex mixed-integer nonlinear programs. *Discrete Optimization* **5** 186–204.

Bonami, P., M. Kilinç, J. Linderoth. 2009. Algorithms and Software for Convex Mixed Integer Nonlinear Programs. Technical Report #1664, Computer Sciences Department, University of Wisconsin-Madison, 2009.

Castillo, I., J. Westerlund, S. Emet, T. Westerlund. 2005. Optimization of block layout deisgn problems with unequal areas: A comparison of milp and minlp optimization methods. *Computers and Chemical Engineering* **30** 54–69.

Duran, M. A., I. Grossmann. 1986. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming* **36** 307–339.

Elhedhli, S. 2006. Service System Design with Immobile Servers, Stochastic Demand, and Congestion. *Manufacturing & Service Operations Management* **8** 92–97. doi:10.1287/msom.1050.0094.

Fletcher, R., S. Leyffer. 1994. Solving mixed-integer nonlinear programs by outer approximation. *Mathematical Programming* **66** 327–349.

Fourer, R., D. M. Gay, B. W. Kernighan. 1993. *AMPL: A Modeling Language for Mathematical Programming*. The Scientific Press.

Frangioni, A., C. Gentile. 2006. Perspective cuts for a class of convex 0-1 mixed-integer programs. *Mathematical Programming* **106** 225–236.

Grossmann, I., J. Viswanathan, A. Vecchietti. Ramesh Raman, Erwin Kalvelagen. 2001. GAMS/DICOPT: A discrete continuous optimization package. *Math. Methods Appl. Sci* **11** 649–664.

Günlük, O., J. Lee, R. Weismantel. 2007. MINLP strengthening for separable convex quadratic transportation-cost UFL. Tech. Rep. RC24213 (W0703-042), IBM Research Division.

Günlük, O., J. Linderoth. 2010. Perspective relaxation of mixed-integer nonlinear programs with indicator variables. *Mathematical Programming* **124** 183–205.

Harjunkoski, Iiro, Ray Pörn, Tapio Westerlund. 2009. MINLP: Trim-loss problem. Christodoulos A. Floudas, Panos M. Pardalos, eds., *Encyclopedia of Optimization*. Springer, 2190–2198. URL `http://dx.doi.org/10.1007/978-0-387-74759-0_387`.

Hijazi, H. 2010. Mixed-Integer NonLinear Optimization approaches for Network Design in Telecommunications. Ph.D. thesis.

Hijazi, H., P. Bonami, G. Cornuejols, A. Ouorou. 2011. Mixed-integer nonlinear programs featuring on/off constraints. *Computational Optimization and Applications* **http://dx.doi.org/10.1007/s10589-011-9424-0** 1–22.

Lougee-Heimer, R. 2003. The common optimization interface for operations research. *IBM Journal of Research and Development* **47** 57–66. `http://www.coin-or.org`.

Quesada, I., I. E. Grossmann. 1992. An LP/NLP based branch–and–bound algorithm for convex MINLP optimization problems. *Computers and Chemical Engineering* **16** 937–947.

Ravemark, Dag E., David W. T. Rippin. 1998. Optimal design of a multi-product batch plant. *Computers & Chemical Engineering* **22** 177 – 183. doi:DOI:10.1016/S0098-1354(96)00357-2. URL `http://www.sciencedirect.com/science/article/B6TFT-3SH47TV-9/2/806675a2f8c9a3effa55a87be596801b`.

Sawaya, N. 2006. Reformulations, relaxations and cutting planes for generalized disjunctive programming. Ph.D. thesis, Chemical Engineering Department, Carnegie Mellon University.

Sawaya, N., C. D. Laird, L. T. Biegler, P. Bonami, A. R. Conn, G. Cornuéjols, I. E. Grossmann, J. Lee, A. Lodi, F. Margot, A. Wächter. 2006. CMU-IBM open source MINLP project test set. `http://egon.cheme.cmu.edu/ibm/page.htm`.

Saxena, A., P. Bonami, J. Lee. 2009. Convex relaxations of non-convex mixed-integer quadratically constrained programs: Projected formulations. *Mathematical Programming, Series A* To appear.

Tawarmalani, M., N. V. Sahinidis. 2005. A polyhedral branch-and-cut approach to global optimization. *Mathematical Programming* **103** 225–249.

Türkay, Metin, Ignacio E. Grossmann. 1996. Logic-based minlp algorithms for the optimal synthesis of process networks. *Computers & Chemical Engineering* **20** 959 – 978. doi:DOI:10.1016/0098-1354(95)00219-7. URL `http://www.sciencedirect.com/science/article/B6TFT-3TKMDRH-1/2/c97c3f6887f1cbadbf2c6177237b9093`.

Vecchietti, Aldo, I. E. Grossmann. 1999. LOGMIP: a disjunctive 0-1 non-linear optimizer for process system models. *Computers and Chemical Engineering* **23** 555–565. doi:DOI:10.1016/S0098-1354(98)00293-2. URL `http://www.sciencedirect.com/science/article/B6TFT-3XY28Y0-B/2/2709e69a55450cf2263efcc5368850db`.

Wächter, A., L. T. Biegler. 2006. On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming. *Mathematical Programming* **106** 25–57.