Global Optimization of Bilinear Programs with a Multiparametric Disaggregation Technique

Scott Kolodziej [a], *Pedro M. Castro* [b] *and Ignacio E. Grossmann* [a*]

[a] Department of Chemical Engineering, Carnegie Mellon University, Pittsburgh, PA 15213-3890, USA

[b] Unidade de Modelação Optimização Sistemas Energéticos, Laboratório Nacional de Energia e Geologia, 1649-038 Lisboa, Portugal

Abstract

In this paper, we present the derivation of the multiparametric disaggregation technique by Teles et. al (2001) for solving nonconvex bilinear programs. Both upper and lower bounding formulations corresponding to mixed-integer linear programs are derived using disjunctive programming and exact linearizations, and incorporated into two global optimization algorithms that are used to solve bilinear programming problems. The relaxation derived using the multiparametric disaggregation technique (MDT) is shown to scale much more favorably than the relaxation that relies on piecewise McCormick envelopes, yielding smaller mixed-integer problems and faster solution times for similar optimality gaps. The proposed relaxation also compares well with general global optimization solvers on large problems.

[*] Corresponding author. Tel.:+1 412 268 3642; fax:+1 412 268 7139.

E-mail address: grossmann@cmu.edu (I.E. Grossmann).

Keywords: *Global Optimization; mixed integer linear programming; mixed integer nonlinear programming; quadratic optimization; disjunctive programming.*

1. Introduction

Bilinear programs, for the purpose of this paper, can be written as the following nonconvex nonlinear programming problem:

$$\begin{aligned} &\text{Min } z = f_0 = \sum_{(i,j)\in BL_0} a_{ij0}x_i x_j + h_0(x) \\ &\text{Subject to} \\ &\quad f_q = \sum_{(i,j)\in BL_q} a_{ijq}x_i x_j + h_q(x) \le 0 \quad q \in Q\backslash\{0\} \\ &\quad x \in S \cap \Omega \subset \mathbb{R}^n \end{aligned} \tag{P}$$

where $h_q(x)$ is convex and twice differentiable, a_{ijq} is a scalar with $i \in I$, $j \in J$, and $q \in Q$ represents the set of all functions f_q, including the objective function f_0 and all constraints. BL_q is an (i,j)-index set which defines the bilinear terms present in the problem. Although $i \neq j$ for strictly bilinear problems, $i = j$ can be allowed to accommodate quadratic problems. The set $S \subset \mathbb{R}^n$ is convex, and $\Omega \subset \mathbb{R}^n$ is an n-dimensional hyperrectangle defined in terms of the initial variable bounds x^L and x^U:

$$\Omega = \{x \in \mathbb{R}^n : 0 \le x^L \le x \le x^U\}$$

The global optimization of bilinear programs of the form of (P) is important in such areas as water networks and petroleum blending operations [1, 2, 3, 4]. Nonconvex, bilinear constraints are required to model the mixing of various streams in these systems, and are in some cases the only nonlinearities in the models. The pooling problem, stemming from the original Haverly paper [5], contains these bilinear constraints and has received much attention in the literature [6, 1, 7, 8, 2, 9]. Recently, Misener & Floudas have demonstrated a novel logarithmic relaxation for modeling bilinear terms with piecewise McCormick envelopes while addressing various classes of pooling problems [1].

Water network optimization problems containing bilinear terms have also received much attention in the literature [10, 11, 3, 4, 12]. The same blending constraints present in the pooling

problem are present in water network problems, and thus numerous advances in solving bilinear programs have been made addressing these problems.

The global optimization of general nonconvex bilinear programs has received significant attention in the literature [13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27]. The convex McCormick envelopes [1] coupled with spatial branch and bound search frameworks have been the basis for many of these global optimization techniques, with piecewise McCormick envelopes being a more recent development. Variations of this approach have been suggested, generalizing the convex envelopes to piecewise over- and under-estimators [1, 17]. Novel ways of representing bilinear terms through reformulation have been another approach reported in the literature [16]. Misener and Floudas [1], building on the work of Vielma & Nemhauser [28, 29], have shown that a relaxation of the bilinear terms can be achieved with a logarithmic number of binary variables. Teles & Castro [30] have introduced a novel approximation of polynomial constraints that exhibit a similar property.

In this paper, we introduce the multiparametric disaggregation technique described in [11] and derive the corresponding mixed-integer constraints using generalized disjunctive programming and exact linearization. We also prove that this approximation technique can be used under some conditions to obtain an upper bound. Furthermore, we introduce a rigorous lower bound derived from the upper bounding constraints, and present two algorithms based on these bounds. Finally, we conclude with a comparison of this approach and the McCormick convex envelopes, and report computational results on small and large problems.

2. Discretization with Multiparametric Disaggregation

Given a nonconvex bilinear term $w_{ij} = x_i \cdot x_j$, the multiparametric disaggregation technique described by Teles et al. [11] can be used to obtain an upper bound on problem (**P**). This reformulation can be derived in terms of generalized disjunctive programming (GDP) [31] and exact linearization [32]. For simplicity in the notation, we first rewrite the bilinear product $w_{ij} =$

$x_i \cdot x_j$ as a single bilinear term $w = u \cdot v$. This product can be represented exactly with the following constraints and disjunction:

$$w = u \cdot v \tag{1}$$

$$v = \sum_{\ell \in \mathbb{Z}} \lambda_\ell \tag{2}$$

$$\bigvee_{k=0}^{9} \left[\lambda_\ell = 10^\ell \cdot k\right] \quad \forall\ \ell \in \mathbb{Z} \tag{3}$$

where v is discretized through the disjunction in (3) that selects one digit k for each power $\ell \in \mathbb{Z}$. Here we assume a basis of 10, although in principle other bases can be selected. Note that since (3) is defined over the domain of all the integer numbers, this implies an infinite number of disjunctions. Furthermore, v can represent any positive real number.

First, we consider the convex hull reformulation of the disjunction in (3) [33] after which we introduce the disaggregated variables, $\hat{\lambda}_{k,\ell}$ and the binary variables $z_{k,\ell}$, which then leads to the following equations where $K = \{k \mid k = 0,1,\ldots 9\}$:

$$\lambda_\ell = \sum_{k=0}^{9} \hat{\lambda}_{k,\ell} \quad \forall\ \ell \in \mathbb{Z} \tag{4}$$

$$\hat{\lambda}_{k,\ell} = 10^\ell \cdot k \cdot z_{k,\ell} \quad \forall\ \ell \in \mathbb{Z}, k \in K \tag{5}$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall\ \ell \in \mathbb{Z} \tag{6}$$

$$z_{k,\ell} \in \{0,1\} \quad \forall\ \ell \in \mathbb{Z}, k \in K$$

Substituting equation (5) into equation (4) yields

$$\lambda_\ell = \sum_{k=0}^{9} 10^\ell \cdot k \cdot z_{k,\ell} \tag{7}$$

Furthermore, substituting equation (7) into equation (2) leads to the fully discretized (but still exact representation of) v:

$$v = \sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} \tag{8}$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall \ \ell \in \mathbb{Z} \tag{6}$$

$$z_{k,\ell} \in \{0,1\} \quad \forall \ \ell \in \mathbb{Z}, k \in K$$

Considering the product $w = u \cdot v$ by substituting equation (8) into equation (1) leads to

$$w = u \cdot \left[\sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} \right]$$

$$= \sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot u \cdot z_{k,\ell}$$

which involves the nonlinear term $u \cdot z_{k,\ell}$. Performing an exact linearization [32], we introduce a new continuous variable, $\hat{u}_{k,\ell} = u \cdot z_{k,\ell}$ so that

$$w = \sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{u}_{k,\ell} \tag{9}$$

Since $u \cdot z_{k,\ell} = \begin{cases} 0 & \text{if } z_{k,\ell} = 0 \\ u & \text{if } z_{k,\ell} = 1 \end{cases}$

and $\hat{u}_{k,\ell}$ is non-negative, we introduce the following lower and upper bound constraints:

$$\hat{u}_{k,\ell} \leq u^{U} \cdot z_{k,\ell} \quad \forall \ \ell \in \mathbb{Z}, k \in K \tag{10}$$

$$\hat{u}_{k,\ell} \geq u^{L} \cdot z_{k,\ell} \quad \forall \ \ell \in \mathbb{Z}, k \in K \tag{11}$$

where u^{U} and u^{L} are the non-negative upper and lower bounds on u. Furthermore, to relate u to $\hat{u}_{k,\ell}$, we derive one additional constraint from equation (6):

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall \ \ell \in \mathbb{Z} \tag{6}$$

$$\sum_{k=0}^{9} u \cdot z_{k,\ell} = u \quad \forall \ \ell \in \mathbb{Z}$$

$$\sum_{k=0}^{9} \hat{u}_{k,\ell} = u \quad \forall \ \ell \in \mathbb{Z} \tag{12}$$

In this way, we arrive at the final set of mixed-integer linear constraints for representing the bilinear product $w = u \cdot v$, namely,

$$w = \sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{u}_{k,\ell} \tag{9}$$

$$v = \sum_{\ell \in \mathbb{Z}} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} \tag{8}$$

$$\hat{u}_{k,\ell} \leq u^{U} \cdot z_{k,\ell} \quad \forall \ \ell \in \mathbb{Z}, k \in K \tag{10}$$

$$\hat{u}_{k,\ell} \geq u^{L} \cdot z_{k,\ell} \quad \forall \ \ell \in \mathbb{Z}, k \in K \tag{11}$$

$$\sum_{k=0}^{9} \hat{u}_{k,\ell} = u \quad \forall \ \ell \in \mathbb{Z} \tag{12}$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall \ \ell \in \mathbb{Z} \tag{6}$$

$$z_{k,\ell} \in \{0,1\} \quad \forall \ \ell \in \mathbb{Z}, k \in K$$

Since in practice it is infeasible to compute the infinite sums over all integers, we represent v to a finite level of precision. The constraints in (9)-(12) and (6) are modified below to allow for a maximum power of 10 (P) and a minimum power of 10 (p), and correspond to the equations proposed by Teles et. al. [11]:

$$w = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{u}_{k,\ell}$$

$$v = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell}$$

$$\hat{u}_{k,\ell} \le u^U \cdot z_{k,\ell} \quad \forall\ \ell \in L, k \in K$$

$$\hat{u}_{k,\ell} \ge u^L \cdot z_{k,\ell} \quad \forall\ \ell \in \mathbb{Z}, k \in K \tag{13}$$

$$\sum_{k=0}^{9} \hat{u}_{k,\ell} = u \quad \forall\ \ell \in L$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall\ \ell \in L$$

$$z_{k,\ell} \in \{0,1\} \quad \forall\ \ell \in L, k \in K$$

where $L = \{\ell = p, p+1, \ldots P\}$.

Because of this finite level of precision, these constraints are no longer an exact representation of the product $w = u \cdot v$. When we incorporate the constraints (13) into problem (**P**) by redefining $w_{ij} = x_i \cdot x_j$, and selecting x_j as the variable on which discretization is performed, the resulting problem (**P′**) shown below will represent a mixed-integer approximation to the original problem:

$$\text{Min } z' = f_0 = \sum_{(i,j) \in BL_0} a_{ij0} w_{ij} + h_0(x)$$

Subject to

$$f_q(x) = \sum_{(i,j) \in BL_q} a_{ijq} w_{ij} + h_q(x) \le 0 \quad q \in Q \backslash \{0\}$$

$$w_{ij} = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{x}_{ijk\ell} \quad \forall\ (i,j) \in BL_q, q \in Q \tag{**P′**}$$

$$x_j = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{jk\ell} \quad \forall\ j \in \{j | (i,j) \in BL_q, q \in Q\}$$

$$\hat{x}_{ijk\ell} \le x_i^U \cdot z_{jk\ell} \quad \forall\ (i,j) \in BL_q, q \in Q, k \in K, \ell \in L$$

$$\hat{x}_{ijk\ell} \ge x_i^L \cdot z_{jk\ell} \quad \forall\ (i,j) \in BL_q, q \in Q, k \in K, \ell \in L$$

$$\sum_{k=0}^{9} \hat{x}_{ijk\ell} = x_i \quad \forall\ (i,j) \in BL_q, q \in Q, k \in K, \ell \in L$$

$$\sum_{k=0}^{9} z_{jk\ell} = 1 \quad \forall\ j \in \{j|(i,j) \in BL_q, q \in Q\}, k \in K, \ell \in L$$

$$z_{jk\ell} \in \{0,1\} \quad \forall\ j \in \{j|(i,j) \in BL_q, q \in Q\}, k \in K, \ell \in L$$

$$x \in S \cap \Omega \subset \mathbb{R}^n$$

where x_j and w_{ij} represents the discrete and continuous approximations to x_j and w_{ij}, respectively, in the constraints (9)-(12) and (6). Note that if the convex functions $h_q(x)$ are linear, problem (**P′**) represents a mixed integer linear program (MILP). Otherwise it corresponds to an MINLP.

Further, note that problem (**P′**) is a restricted version of problem (**P**), or equivalently problem (**P**) is a relaxation of problem (**P′**). Thus, if the solution of problem (**P′**) is feasible then it is also feasible for problem (**P**). It then follows that the solution of problem (**P′**) either yields an upper bound on problem (**P**) such that $z' = z^U \geq z$, or else problem (**P′**) is infeasible. This restricted feasible region can be seen in Figures 1 and 2.

2.1. Infeasibilities in the Discretized Problem

The mixed-integer approximation problem (**P′**) can be infeasible even if the original problem (**P**) is feasible. For example, if bounds such as $10^{p-1} \leq x_j \leq 2 \cdot 10^{p-1}$ are enforced, (**P′**) will be infeasible, while such a constraint is completely valid and will not necessarily result in an infeasible problem (**P**). Thus, the parameters p and P must be chosen appropriately to avoid such infeasibilities.

Some general guidelines for ensuring precision-based feasibility can be established. For example, the largest power of 10 (P) must be large enough such that 10^P is at least as large as the upper bound on x_j: $P \geq \lceil \log_{10} x_j^U \rceil$

Additionally, p must be small enough to ensure at least one (and preferably many) discretization points lie between the lower and upper bounds for x_j. Thus, $p \leq P$ is the absolute minimum requirement, but feasibility is more likely as p is decreased. Note that these guidelines do not

guarantee feasibility of (**P′**) in all cases, but represent the minimum level of precision needed given reasonable bounds on x_j. The following proposition can then be established:

Property 1: If Problem (**P′**) is feasible, its solution yields an upper bound for problem (**P**).

Proof: If (**P′**) is feasible, its solution is feasible in (**P**), as (**P**) is a relaxation of (**P′**). Since this solution is not necessarily optimal in (**P**) it will yield an upper bound. □

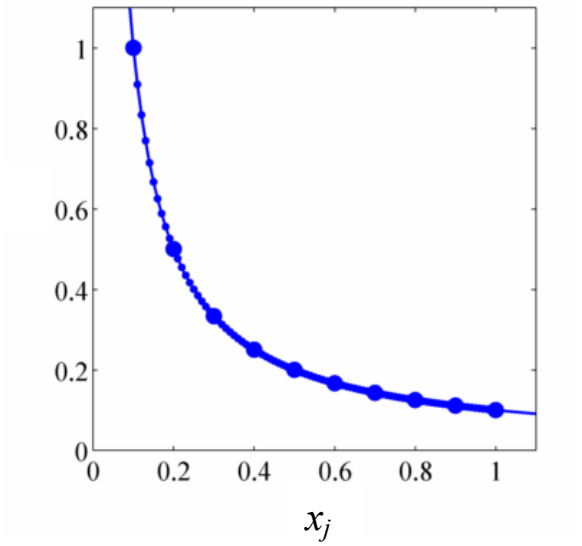

Figure 1: Feasible region for a bilinear curve $x_i \cdot x_j = 0.1$, $p = -2$, $P = -1$. The solid curve represents the exact bilinear curve, while the dots represent the approximated (and incomplete) feasible region resulting from the upper bounding formulation.

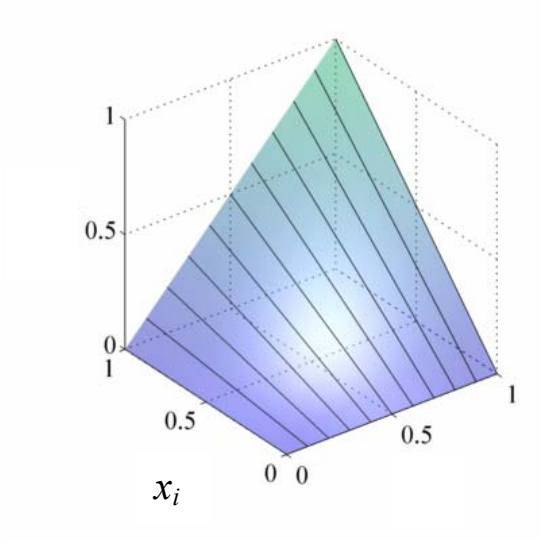

Figure 2: Feasible region for a bilinear surface. The surface is the exact bilinear curve, while the solid lines represent the reduced feasible region resulting from the upper bounding formulation for $p = P = -1$.

3. Lower Bounding with Multiparametric Disaggregation

To obtain a lower bounding problem using multiparametric disaggregation, we first note that in the discretized problem, there always exists a gap between discretization points for a finite p. Thus, we can introduce a slack variable Δx_j such that $x_j^R = x_j' + \Delta x_j$, where x_j' is the discretized representation of x_j from Section 2, and x_j^R is the continuous representation of x_j.

Again switching to the notation $w = u \cdot v$ for the bilinear term, we begin with the discretization of v from the truncated set of constraints (13):

$$v = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell}$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall \ \ell \in L$$

$$z_{k,\ell} \in \{0,1\} \quad \forall \ \ell \in L, k \in K$$

We can add the slack term Δv (bounded between 0 and 10^p) to represent a continuous v, denoted as v^R:

$$v^R = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} + \Delta v$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall \ \ell \in L \tag{14}$$

$$z_{k,\ell} \in \{0,1\} \quad \forall \ \ell \in L, k \in K$$

$$0 \leq \Delta v \leq 10^p$$

We can rewrite Δv using a form similar to the discretization scheme already described. Given the following relationship,

$$0 \leq \sum_{k=0}^{1} 10^p \cdot k \cdot \tilde{z}_k \leq 10^p$$

$$0 \leq \tilde{z}_k \leq 1$$

we use this expression to represent Δv:

$$\Delta v = \sum_{k=0}^{1} 10^p \cdot k \cdot \tilde{z}_k$$

$$0 \leq \tilde{z}_k \leq 1$$

The constraints (14) can now be rewritten as

$$v^R = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} + \sum_{k=0}^{1} 10^p \cdot k \cdot \tilde{z}_k$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall \ \ell \in L \tag{15}$$

$z_{k,\ell} \in \{0,1\} \quad \forall\ \ell \in L, k \in K$

$0 \leq \tilde{z}_k \leq 1 \quad \forall\ k \in \{0,1\}$

Furthermore, by substituting constraints (15) into the bilinear term $w = u \cdot v$ as done in the prior derivation of the discretized reformulation, and introducing a new variable $\tilde{u}_k = u \cdot \tilde{z}_k$, we obtain

$$w^R = u \cdot v^R$$

$$= u \cdot \left[\sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{z}_k \right]$$

$$= \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot u \cdot z_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot u \cdot \tilde{z}_k$$

$$= \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{u}_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{u}_k$$

Analogous constraints can be derived in the same fashion as before in the discretized formulation, yielding the final set of mixed-integer constraints for the bilinearity:

$$w^R = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{u}_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{u}_k$$

$$v^R = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{z}_k$$

$$\hat{u}_{k,\ell} \leq u^U \cdot z_{k,\ell} \quad \forall\ \ell \in L, k \in K$$

$$\hat{u}_{k,\ell} \geq u^L \cdot z_{k,\ell} \quad \forall\ \ell \in L, k \in K$$

$$\tilde{u}_k \leq u^U \cdot \tilde{z}_k \quad \forall\ k \in \{0,1\}$$

$$\tilde{u}_k \geq u^L \cdot \tilde{z}_k \quad \forall\ k \in \{0,1\}$$

$$\sum_{k=0}^{9} \hat{u}_{k,\ell} = u \quad \forall\ \ell \in L$$

$$\sum_{k=0}^{1} \tilde{u}_k = u$$

$$\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall\ \ell \in L$$

$$\sum_{k=0}^{1} \tilde{z}_k = 1$$

$$z_{k,\ell} \in \{0,1\} \quad \forall\ \ell \in L, k \in K$$

$0 \le \tilde{z}_k \le 1 \quad \forall\ k \in \{0,1\}$

Introducing these constraints into Problem (**P**), and expressing the variables in terms of the original variables $w_{ij} = x_i \cdot x_j$, we obtain the new optimization problem, (**PR**):

$$\text{Min } z^R = f_0 = \sum_{(i,j)\in BL_0} a_{ij0} w_{ij} + h_0(x)$$

subject to

$$f_q(x) = \sum_{(i,j)\in BL_q} a_{ijq} w_{ij} + h_q(x) \le 0 \quad q \in Q\backslash\{0\}$$

$$w_{ij} = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{x}_{ijk\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{x}_{ijk} \quad \forall\ (i,j) \in BL_q, q \in Q$$

$$x_j = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{ijk\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{z}_{ijk} \quad \forall j \in \{j|(i,j) \in BL_q, q \in Q\}$$

$$\hat{x}_{ijk\ell} \le x_i^U \cdot z_{ijk\ell} \quad \forall\ (i,j) \in BL_q, q \in Q, \ell \in L, k \in K$$

$$\hat{x}_{ijk\ell} \ge x_i^L \cdot z_{ijk\ell} \quad \forall\ (i,j) \in BL_q, q \in Q, \ell \in L, k \in K$$

$$\tilde{x}_{ijk} \le x_i^U \cdot \tilde{z}_{ijk} \quad \forall\ (i,j) \in BL_q, q \in Q, k \in \{0,1\}$$

$$\tilde{x}_{ijk} \ge x_i^L \cdot \tilde{z}_{ijk} \quad \forall\ (i,j) \in BL_q, q \in Q, k \in \{0,1\} \qquad \textbf{(PR)}$$

$$\sum_{k=0}^{9} \hat{x}_{ijk\ell} = x_i \quad \forall\ (i,j) \in BL_Q, q \in Q, \ell \in L$$

$$\sum_{k=0}^{1} \tilde{x}_{ijk} = x_i \quad \forall\ (i,j) \in BL_Q, q \in Q$$

$$\sum_{k=0}^{9} z_{ijk\ell} = 1 \quad \forall\ (i,j) \in BL_q, q \in Q, \ell \in L$$

$$\sum_{k=0}^{1} \tilde{z}_{ijk} = 1 \quad \forall (i,j) \in BL_q, q \in Q$$

$$z_{ijk\ell} \in \{0,1\} \quad \forall\ (i,j) \in BL_q, q \in Q, \ell \in L, k \in K$$

$$0 \le \tilde{z}_{ijk} \le 1 \quad \forall\ (i,j) \in BL_q, q \in Q, k \in \{0,1\}$$

$$x \in S \cap \Omega \subset \mathbb{R}^n$$

While (**PR**) does not exactly represent the product $w_{ij} = x_i \cdot x_j$ and is feasible for values of w_{ij}, x_i, and x_j that do not satisfy $w_{ij} = x_i \cdot x_j$, the bilinear term is feasible in (**PR**). Thus, (**PR**) is a relaxation of (**P**). The relaxed feasible region resulting from (**PR**) can be seen in Figures 3 and 4.

The following property can be readily established:

Property 2: The solution of problem (**PR**) yields a lower bound for problem (**P**), i.e. $z^R \le z$.

Proof: This directly follows from the fact that the constraints in (**PR**) represent a relaxation of problem (**P**), with which it clearly follows that the optimal solution of (**PR**), z^R, represents a lower bound to the solution of the original problem (**P**). □

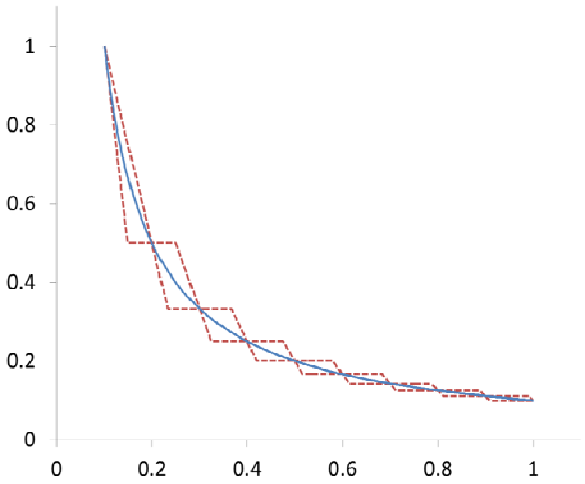

Figure 3: Plot of the lower bounding feasible region for w = 0.1. The solid line is the true curve, while the dotted lines represent the boundaries of the relaxed feasible region of (**PR**). Note the similarities to piecewise McCormick envelopes.

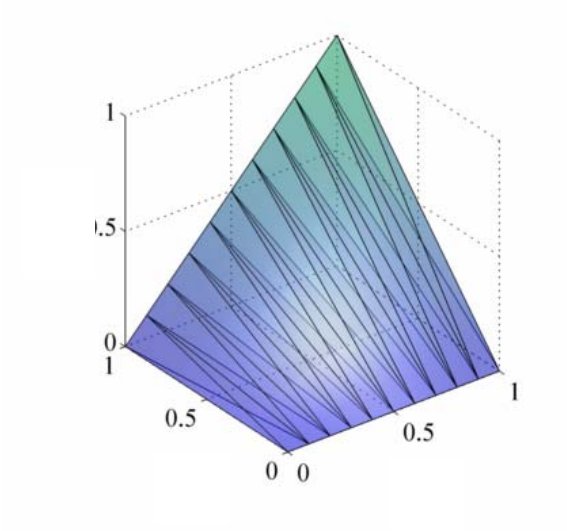

Figure 4: Feasible region of the relaxed problem (**PR**). The surface is the exact bilinear curve, while the envelopes represent the relaxed feasible region resulting from the lower bounding formulation for p = P = -1.

4. Discussion of Global Optimization Algorithms

The upper and lower bounding schemes described can be combined into a global optimization algorithm. First, the following property can be established:

Property 3: As p approaches $-\infty$, z' approaches z^R.

Proof: As p approaches $-\infty$ in (**PR**), 10^p approaches 0, which implies

$$\lim_{p\to-\infty} \sum_{k=0}^{1} 10^p \cdot k \cdot \tilde{z}_{ijk} = 0$$

$$\lim_{p\to-\infty} \sum_{k=0}^{1} 10^p \cdot k \cdot \tilde{u}_{ijk} = 0$$

Thus, since the variables $\tilde{z}_{ijk}$ and $\tilde{u}_{ijk}$ are eliminated, this yields problem (**P′**), and hence z' approaches z^R. □

From Property 3, we can establish that as precision is increased (i.e. p approaches $-\infty$), both (**P′**) and (**PR**) converge to the same value. Assuming P is large enough such that $10^P \geq x_j^U$, we can further state that (**P′**) and (**PR**) converge such that $z' = z^R = z$.

4.1. Algorithm 1

The first global optimization algorithm that can be established from the aforementioned upper and lower bounding is as follows. First, we start at some coarse level of discretization such that $P \geq p$, and solve both (**PR**) and (**P′**). If the difference in solutions to the upper and lower bounding problems is sufficiently small, then the algorithm terminates; otherwise, precision is increased and the problems are resolved. The algorithm is then as follows:

Algorithm 1

Step 0. Choose $p = P \geq \lceil log_{10} x_j^U \rceil$

Step 1. Solve (**PR**) to obtain the lower bound z^R.

Step 2. Solve (**P′**) to obtain the upper bound z'. If (**P′**) is infeasible, let $z' = +\infty$.

Step 3. If $(z' - z^R)/z^R \leq \varepsilon$, STOP, the solution is globally optimal. Otherwise, set $p = p - 1$, and return to step 1.

4.2. Algorithm 2

While Algorithm 1 follows most naturally from the problems (**P′**) and (**PR**), it has several shortcomings. Notably, because (**P′**) and (**PR**) are fairly similar and increase similarly in problem size as precision is added, solving (**P′**) and (**PR**) repeatedly becomes increasingly expensive. Thus, an alternative method for obtaining an upper bound instead solving of (**P′**) is to use a local NLP algorithm in place of solving the problem (**P′**).

Algorithm 2

Step 0. Choose $p = P \geq \lceil log_{10} x_j^U \rceil$

Step 1. Solve (**PR**) to obtain the lower bound z^R.

Step 2. Solve (**P**) using a local NLP algorithm to obtain some upper bound z using the solution to (**PR**) as a starting point.

Step 3. If $(z - z^R)/z^R \leq \varepsilon$, STOP, the solution is globally optimal. Otherwise, set $p = p - 1$ and return to step 1.

Algorithm 2 is generally more computationally efficient than Algorithm 1, as the solution of (**P**) using a local NLP algorithm is more efficient than the increasingly large MILP that (**P′**) becomes as $P - p$ grows.

4.3. Extensions to MINLP

The algorithms in Sections 4.1 and 4.2 can be readily extended for solving MINLPs with the following general form:

$$\begin{aligned} &\text{Min } z = f_0(x,y) \\ &\text{subject to} \\ &f_q(x,y) \leq 0 \quad q \in Q\backslash\{0\} \\ &f_q(x,y) = \sum_{(i,j)\in BL_q} a_{ijq} x_i x_j + h_q(x,y) \quad q \in Q \\ &x \in S \cap \Omega \subset \mathbb{R}^n \\ &y \in \{0,1\} \end{aligned} \qquad \textbf{(P-MINLP)}$$

where $h_q(x,y)$ is jointly convex in x and y. For most cases in practice the variables y appear in linear form in these terms [34].

Analogous problems (**P′-MINLP**) and (**PR-MINLP**) can be derived, and Algorithm 1 can be used without modification. However, Algorithm 2 requires some modification, as (**P-MINLP**) cannot be solved using a local NLP algorithm because it is an MINLP. A heuristic can be used to compute the upper bound as in Algorithm 2:

Algorithm 3

Step 0. Choose $p = P \geq \lceil log_{10} x_j^U \rceil$

Step 1. Solve (**PR-MINLP**) to obtain z^R.

Step 2. Fix the binary variables y in (**P-MINLP**) to the values found by the solution of (**PR-MINLP**) in Step 1, reducing it to an NLP. Solve (**P-MINLP**) with these fixed binary variables using a local NLP algorithm to obtain some z using the solution to (**PR-MINLP**) as a starting point.

Step 3. If $(z - z^R)/z^R \leq \varepsilon$, STOP, the solution is globally optimal. Otherwise, set $p = p - 1$ and return to step 1.

A disadvantage of this algorithm is that it could take, in the worst case, an infinite number of iterations to converge. By fixing the binary variables to that of the solution to (**PR-MINLP**), (**P-MINLP**) can be rendered infeasible, and the algorithm would continue until (**PR-MINLP**) has enough discretization points to exactly represent (**P-MINLP**). If this occurs, heuristics or other approaches may be utilized to obtain an upper bound in place of solving (**P-MINLP**) with fixed binary variables. However, in practice, this is unlikely to occur.

5. Comparison with Piecewise McCormick Envelopes

A common approach to solving bilinear programs of the form (**P**) is to reformulate the bilinear terms using McCormick convex envelopes [35]. This reformulation results in an LP if the only nonlinearities are bilinear, or a convex NLP if there are remaining convex nonlinearities, either of which are easily solved to global optimality. For each bilinearity $w_{ij} = x_i \cdot x_j$, we introduce instead the following constraints:

$$\begin{aligned} w_{ij} &\geq x_i \cdot x_j^L + x_i^L \cdot x_j - x_i^L \cdot x_j^L \\ w_{ij} &\geq x_i \cdot x_j^U + x_i^U \cdot x_j - x_i^U \cdot x_j^U \\ w_{ij} &\leq x_i \cdot x_j^L + x_i^U \cdot x_j - x_i^U \cdot x_j^L \\ w_{ij} &\leq x_i \cdot x_j^U + x_i^L \cdot x_j - x_i^L \cdot x_j^U \end{aligned} \qquad \forall\ (i,j) \in BL_q, q \in Q \tag{16}$$

This relaxation yields a lower bound on problem (**P**). However, this lower bound can be weak depending on the bounds on x_i and x_j. To improve the quality of the lower bound, these convex envelopes can be used on discretized portions of the variable range. Thus, piecewise McCormick envelopes can be introduced in (**P**) to obtain a tighter lower bound at the cost of becoming an MILP [10, 17, 36]. The envelopes, defined over a set of N points on the variable x_i, can be represented by the following disjunctive constraints:

$$\begin{bmatrix} y_{jn} \\ w_{ij} \ge x_i \cdot x_{jn}^L + x_i^L \cdot x_j - x_i^L \cdot x_{jn}^L \ \forall\{i \mid (i,j) \in BL_q, q \in Q\} \\ w_{ij} \ge x_i \cdot x_{jn}^U + x_i^U \cdot x_j - x_i^U \cdot x_{jn}^U \ \forall\{i \mid (i,j) \in BL_q, q \in Q\} \\ w_{ij} \le x_i \cdot x_{jn}^L + x_i^U \cdot x_j - x_i^U \cdot x_{jn}^L \ \forall\{i \mid (i,j) \in BL_q, q \in Q\} \\ w_{ij} \le x_i \cdot x_{jn}^U + x_i^L \cdot x_j - x_i^L \cdot x_{jn}^U \ \forall\{i \mid (i,j) \in BL_q, q \in Q\} \\ x_{jn}^L \le x_j \le x_{jn}^U \end{bmatrix}_{n=1}^{N} \forall\{j \mid (i,j) \in BL_q, q \in Q\}$$

$$\left.\begin{array}{c} x_i^L \le x_i \le x_i^U \ \forall i \\ x_{jn}^L = x_j^L + \dfrac{x_j^U - x_j^L}{N} \cdot (n-1) \\ x_{jn}^U = x_j^L + \dfrac{x_j^U - x_j^L}{N} \cdot n \end{array}\right\} \forall\{j \mid (i,j) \in BL_q, q \in Q\}, n = 1,2..N \tag{17}$$

Applying the convex hull reformulation [33] for the above disjunctive constraints yields

$$\begin{aligned}
& w_{ij} \ge \sum_n (\hat{x}_{ijn} \cdot x_{jn}^L + x_i^L \cdot \hat{x}_{jn} - x_i^L \cdot x_{jn}^L \cdot y_{jn}) \ \forall (i,j) \in BL_q, q \in Q \\
& w_{ij} \ge \sum_n (\hat{x}_{ijn} \cdot x_{jn}^U + x_i^U \cdot \hat{x}_{jn} - x_i^U \cdot x_{jn}^U \cdot y_{jn}) \ \forall (i,j) \in BL_q, q \in Q \\
& w_{ij} \le \sum_n (\hat{x}_{ijn} \cdot x_{jn}^L + x_i^U \cdot \hat{x}_{jn} - x_i^U \cdot x_{jn}^L \cdot y_{jn}) \ \forall (i,j) \in BL_q, q \in Q \\
& w_{ij} \le \sum_n (\hat{x}_{ijn} \cdot x_{jn}^U + x_i^L \cdot \hat{x}_{jn} - x_i^L \cdot x_{jn}^U \cdot y_{jn}) \ \forall (i,j) \in BL_q, q \in Q \\
& x_i = \sum_n \hat{x}_{ijn} \ \forall (i,j) \in BL_q, q \in Q \\
& x_j = \sum_n \hat{x}_{jn} \ \forall\{j \mid (i,j) \in BL_q\}, q \in Q \\
& x_i^L \cdot y_{jn} \le \hat{x}_{ijn} \le x_i^U \cdot y_{jn} \ \forall (i,j) \in BL_q, q \in Q, n = 1,2..N \\
& x_{jn}^L \cdot y_{jn} \le \hat{x}_{jn} \le x_{jn}^U \cdot y_{jn} \ \forall\{j \mid (i,j) \in BL_q\}, q \in Q, n = 1,2..N \\
& \sum_n y_{jn} = 1 \ \forall\{j \mid (i,j) \in BL_q\}, q \in Q \\
& y_{jn} \in \{0,1\} \ \forall\{j \mid (i,j) \in BL_q\}, q \in Q, n = 1,2..N
\end{aligned} \tag{18}$$

By adding these piecewise McCormick envelope constraints (18) to problem (**P**), we can define a new relaxed MILP, (**PR-PCM**). Furthermore, we can compare the performance of (**PR-PCM**) to the lower bounding problem derived from multiparametric disaggregation, (**PR**).

6. Illustrative example (P1)

We first consider as an example the bilinear program by Quesada and Grossmann [37] and originally reported by Al-Khayyal and Falk [18]:

$$\begin{aligned} &\text{Min } f = -x_1 + x_1 x_2 - x_2 \\ &\text{subject to} \\ &\quad -6x_1 + 8x_2 \le 3 \\ &\quad 3x_1 - x_2 \le 3 \\ &\quad 0 \le x_1, x_2 \le 1.5 \end{aligned} \qquad \textbf{(P1)}$$

The global optimum of this bilinear program is f = -1.0833 at (1.167, 0.5). Two other local minima correspond to: f = -1.0 at (1, 1), and f = -1.005 at (0.917, 1.062).

6.1. Lower bounding problems

In order to compare the lower bounds predicted by multiparametric disaggregation and piecewise McCormick envelopes, we solve the relaxation problems (**PR**) and (**PR-PCM**) resulting from multiparametric disaggregation and piecewise McCormick, respectively. For the specific example Problem (**P1**), we can derive analogous relaxed problems (**P1R**) and (**P1R-PCM**) using the multiparametric disaggregation technique and piecewise McCormick envelopes, respectively. Note that x_1 is the variable being discretized.

$\begin{aligned} &\text{Min } f = -x_1 + \mathrm{w}_{12} - x_2 \\ &\text{subject to} \\ &\quad -6x_1 + 8x_2 \le 3 \\ &\quad 3x_1 - x_2 \le 3 \\ &\quad 0 \le x_1, x_2 \le 1.5 \end{aligned}$	
$w_{12} = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot \hat{x}_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{x}_k$	$w_{12} \le \sum_{n=1}^{N} \hat{x}_{1n} x_2^U + \hat{x}_{2n} x_{1n}^L - x_{1n}^L x_2^U y_n$

$x_1 = \sum_{\ell=p}^{P} \sum_{k=0}^{9} 10^{\ell} \cdot k \cdot z_{k,\ell} + \sum_{k=0}^{1} 10^{p} \cdot k \cdot \tilde{z}_k$ $\hat{x}_{k,\ell} \leq 1.5 \cdot z_{k,\ell} \quad \forall\ \ell \in L, k \in K$ $\tilde{x}_k \leq 1.5 \cdot \tilde{z}_k \quad \forall\ k \in \{0,1\}$ $\sum_{k=0}^{9} \hat{x}_{k,\ell} = x_2 \quad \forall\ \ell \in L$ $\sum_{k=0}^{1} \tilde{x}_k = x_2$ $\sum_{k=0}^{9} z_{k,\ell} = 1 \quad \forall\ \ell \in L$ $\sum_{k=0}^{1} \tilde{z}_k = 1$ $z_{k,\ell} \in \{0,1\} \quad \forall\ \ell \in L, k \in K$ $0 \leq \tilde{z}_k \leq 1 \quad \forall\ k \in \{0,1\}$ $0 \leq \hat{x}_{k,\ell} \leq 1.5 \quad \forall\ \ell \in L, k \in K$ $0 \leq \tilde{x}_k \leq 1.5 \quad \forall\ k \in K$ $0 \leq \hat{x}_{k,\ell} \leq 1.5 \quad \forall\ \ell \in L, k \in K$ $0 \leq \tilde{x}_k \leq 1.5 \quad \forall\ k \in K$	$w_{12} \leq \sum_{n=1}^{N} \hat{x}_{1n} x_2^L + \hat{x}_{2n} x_{1n}^U - x_{1n}^U x_2^L y_n$ $w_{12} \geq \sum_{n=1}^{N} \hat{x}_{1n} x_2^L + \hat{x}_{2n} x_{1n}^L - x_{1n}^L x_2^L y_n$ $w_{12} \geq \sum_{n=1}^{N} \hat{x}_{1n} x_2^U + \hat{x}_{2n} x_{1n}^U - x_{1n}^U x_2^U y_n$ $x_1 = \sum_{n=1}^{N} \hat{x}_{1n}$ $x_2 = \sum_{n=1}^{N} \hat{x}_{2n}$ $\hat{x}_{1n} \leq y_n x_{1n}^U \quad \forall\ n = 1,2..N$ $\hat{x}_{1n} \geq y_n x_{1n}^L \quad \forall\ n = 1,2..N$ $\hat{x}_{2n} \leq y_n x_2^U \quad \forall\ n = 1,2..N$ $\hat{x}_{2n} \geq y_n x_2^L \quad \forall\ n = 1,2..N$ $\sum_{n=1}^{N} y_n = 1$ $x_{1n}^L = x_1^L + \frac{x_n^U - x_n^L}{N} \cdot (n-1) \quad \forall\ n = 1,2..N$ $x_{1n}^U = x_1^L + \frac{x_n^U - x_n^L}{N} \cdot n \quad \forall\ n = 1,2..N$ $y_n \in \{0,1\} \quad \forall\ n = 1,2..N$
(**P1R**)	(**P1R-PCM**)

Problem size and computational results are shown in Table 1 using GAMS 23.8.2 [38] for the lower bounds predicted by (**P1R**) at P = 0 and p = {0, -1,…-6} and (**P1R-PCM**) at N = {1, 10, 100, 1000, 10000} being solved by CPLEX 12.4 [39]. Solving (**P1R-PCM**) at various levels of discretization, it becomes clear that problem size increases approximately exponentially with each order of magnitude decrease in the optimality gap. However, solving (**P1R**), this is not the case. As precision is added to (**P1R**), i.e. p is decreased; a linear relationship holds with each order of magnitude added. Note that due to the range of discretization [0, 1.5], the precision of PCM at N = 10 is between that of MDT at p = 0 and p = -1. Likewise, PCM at N = 100 is between that of MDT at p = -1 and p = -2. Additionally, upper bounds are reported using CONOPT 3 [40], by using as a starting point the solution of the lower bounding relaxation problem as in Algorithm 2. Results for

BARON [41] solving the original NLP (**P1**) are also reported. Note also that the computational time reported for MDT is for each subproblem, and these times would need to be accumulated to compare Algorithm 2 directly to BARON, unless the discretization level (p) is chosen a priori.

As seen in Table 1, the relaxed problems (**P1R**) and (**P1R-PCM**) are considerably larger than the original NLP (**P1**) since the addition of both continuous and binary variables increases problem size. However, note that the multiparametric disaggregation problem (**P1R**) requires fewer additional continuous variables and fewer constraints than when utilizing piecewise McCormick envelopes in (**P1R-PCM**). For a single McCormick envelope, a relatively weak lower bound of -1.5 (38.5% gap) is obtained. Multiparametric disaggregation with $P = 0$ and $p = -4$ approaches an optimality gap of 0.003%, while piecewise McCormick envelopes, even with 1000 partitions, only approach an optimality gap of 0.046%. Further discretization and refinement of the solution is quickly solved using multiparametric disaggregation, as the $p = -5$ and $p = -6$ problems are solved in less than 1 second and fewer than 200 variables. In contrast, adding more partitions in (**P1R-PCM**) leads to intractable problems with more than 10000 variables and requiring significantly more computational time. Notice also that the results reported in the columns MDT from p = 0 to p = -6, are equivalent to the iterations of Algorithm 2 in Section 4.2.

7. Numerical Experiments

The performance of the underestimating problems from multiparametric disaggregation (**PR**) and piecewise McCormick (**PR-PCM**) is evaluated through the solution of four additional small test problems from the literature for different accuracy levels. Since in all problems the functions $h_q(x)$ in (**P**) are linear, the resulting bounding MILP problems were solved in GAMS 23.7.1 using CPLEX 12.3 (1 thread). Default options were used except the relative optimality tolerance, equal to 10^{-6} and a maximum computational effort equal to 3600 CPU seconds. Similarly as in problem (**P1**), the original nonlinear programs were solved by CONOPT 3, following initialization with the values from the MILP, and by BARON 9.3.1. We also report results for larger problems in the

areas of water networks and multiperiod blending. These problems were also solved by GLoMIQO 1.0.0. The computational experiments were performed on an Intel i7 950 processor running at 3.07 GHz, with 8 GB of RAM, running Windows 7.

7.1. Small Test Problems

7.1.1. P2

Problem (**P2**) is originally from the compilation of test problems by Rijckaert and Martens (1978) [42] but has been converted to a bilinear program following simple transformations and addition of new variables. Variables x_1, x_2 and x_3 are selected for parameterization/partitioning.

$$
\begin{aligned}
\min\ & 5.3578x_3^2 + 0.8357x_1x_5 + 37.2392x_1 \\
s.t.\ & 0.00002584x_3x_5 - 0.00006663x_2x_5 - 0.0000734x_1x_4 \le 1 \\
& 0.000853007x_2x_5 + 0.00009395x_1x_4 - 0.00033085x_3x_5 \le 1 \\
& 1330.3294x_6 - 0.42x_1 - 0.30586x_7 \le x_5 \\
& 0.00024186x_2x_5 + 0.00010159x_1x_2 + 0.00007379x_3^2 \le 1 \\
& 2275.1327x_8 - 0.2668x_1 - 0.40584x_4 \le x_5 \\
& 0.00029955x_3x_5 + 0.00007992x_1x_3 + 0.00012157x_3x_4 \le 1 \\
& 1 = x_2x_6 \\
& x_7x_2 = x_3^2 \\
& 1 = x_8x_3 \\
& 78 \le x_1 \le 102, \quad 33 \le x_2 \le 45, \quad 27 \le x_3 \le 45, \quad 27 \le x_4 \le 45, \quad 27 \le x_5 \le 45 \\
& \frac{1}{45} \le x_6 \le \frac{1}{33}, \quad \frac{27^2}{45} \le x_7 \le \frac{45^2}{33}, \quad \frac{1}{45} \le x_8 \le \frac{1}{27}.
\end{aligned}
$$

7.1.2. P3

Test problem (**P3**) corresponds to Problem 106 in Hock and Schittowski (1981) [43]. While the objective function and the first three constraints are linear, there are three bilinear inequalities. To study how the choice of parameterized/partitioning variables affects computational performance, three versions are considered: (a) 3 parameterized/partitioned variables, x_1, x_2 and x_3; (b) 4 variables, x_2, x_5, x_6 and x_8; (c) 5 variables, x_4, x_5, x_6, x_7 and x_8.

$$
\begin{aligned}
\min \quad & x_1 + x_2 + x_3 \\
s.t. \quad & 0.0025(x_4 + x_6) - 1 \le 0 \\
& 0.0025(-x_4 + x_5 + x_7) - 1 \le 0 \\
& 0.01(-x_5 + x_8) - 1 \le 0 \\
& 100x_1 - x_1 x_6 + 833.33252 x_4 - 83333.333 \le 0 \\
& x_2 x_4 - x_2 x_7 - 1250 x_4 + 1250 x_5 \le 0 \\
& x_3 x_5 - x_3 x_8 - 2500 x_5 + 1250000 \le 0 \\
& 100 \le x_1 \le 10000, 1000 \le x_2, x_3 \le 10000, 10 \le x_4, x_5, x_6, x_7, x_8 \le 100
\end{aligned}
$$

7.1.3. P4

Test problem (**P4**) is taken from Shen and Zhang et al. (2004) [44] and after a few transformations the following bilinear problem results. Variables x_2, x_5 and x_6 are chosen as the parameterized variables.

$$
\begin{aligned}
\min \quad & 168 x_1 x_2 + 3651.2 x_5 + 40000 x_6 \\
s.t. \quad & 1.0425 x_1 - x_2 \le 0 \\
& 0.00035 x_1 x_2 - 1 \le 0 \\
& 1.25 x_4 - x_1 + 41.63 \le 0 \\
& x_1 x_2 - x_3 x_5 = 0 \\
& x_4 x_6 - 1 = 0 \\
& 40 \le x_1 \le 44, 40 \le x_2 \le 45, 60 \le x_3 \le 70, 0.1 \le x_4 \le 1.4 \\
& 22.85714 \le x_5 \le 33, 0.714286 \le x_6 \le 10
\end{aligned}
$$

7.2. Computational statistics

Table 1 to Table 4 give the computational results for problems (**P1**)-(**P4**) as a function of the discretization level, and correspond to Algorithm 2, where the relaxation problem is either generated by multiparametric disaggregation (**PR**), MDT columns, or by the piecewise McCormick envelopes (**PR-PCM**), PCM columns. We provide the problem size, lower bound from the relaxation problem, and upper bound from a local NLP solver (the values of the latter remain the same independent of the accuracy level). The optimality gap and total computational effort (CPLEX plus CONOPT, the latter being almost negligible) are also reported. Note that the triplets in Table 2 are related to the number of discrete points for the three variables selected. The n-tuples in Tables 3-

4 have a similar meaning. We also show the results obtained with BARON. In problems (**P2**)-(**P4**), the discretization points of PCM match exactly those being used by MDT (e.g. second column of PCM should be compared with second column of MDT). Also, as the overall McCormick envelope can be added to the MDT constraints for each bilinearity to tighten their relaxation, problems (**P2**)-(**P4**) were solved in this manner. For example, the LP relaxations of the MILP model from (**P2**) are equal to 9983.61 and 9001.46, with and without the standard McCormick envelopes, respectively. The solution from the MILP is however the same in both cases (e.g. 10121.33 for p=(0,0,0)) with minor changes in the computational time. Notice that the relaxation from piecewise McCormick (PCM) is often tighter than MDT for the same accuracy level. For (**P4**), the lower bounds of x_5 and x_6 are not integer values and so, while the number of discrete points is roughly the same, their location is not, making it possible for the relaxation from MDT to be better than the one from PCM. The exception is PCM = (50,100,100) and MDT = (-1,-1,-1). Due to the better performance of the new method, there are more columns for MDT than for PCM, meaning that higher accuracy levels, i.e. lower optimality gaps, can be achieved by the former for a given computational time.

While MDT is less tight than PCM, the latter generates considerably larger MILP problems for the same accuracy level. More specifically, with PCM we get exactly an order of magnitude increase in the number of binary variables for each new significant digit and roughly the same behavior with respect to the number of total variables and constraints. In contrast, for MDT we get a linear increase, keeping problems tractable for a wider accuracy range. Notice that with MDT all problems can be solved with an optimality gap of less then 0.01% in under one hour, while PCM closes only to a gap of 6.63% for (**P3**).

Since problem size is related to the number of parameterized/partitioned variables, one might be tempted to keep this number as low as possible. However, the results for (**P3**) give opposite results, since the worst performance is obtained for 3 parameterized variables (gap = 0.009%), followed by

4 (gap = 0.0067% in 2144 CPUs) and then 5 (gap = 0.0071 % in 591 CPUs, which improves to just 0.0020% for p = (-3,…,-3), calculated using the best possible solution at time of termination).

7.2.1. Evaluation of Algorithm 1

Compared to Algorithm 2 in Tables 2, 3, and 4, Algorithm 1 generally leads to larger optimality gaps, particularly in the first iterations at coarse discretization levels, and it also requires more CPU time since two MILPs are solved at each iteration. Comparatively, solving (**P**) with a local solver can be done almost instantaneously. The lower bounding problem is the same as in Algorithm 2, while the upper bounding problem (**P′**) yields considerably worse bounds than the ones returned from the local NLP solver (recall that these were always global optimal solutions). As is illustrated in Figure 5 for test problem (**P3a**), the solutions from both (**P′**) and (**PR**) typically become closer to the optimum with an increase in accuracy. However, it is clear that using a local NLP solver, as in Algorithm 2, is a superior approach. While it is possible for the solution of (**P′**) to be closer to the global optimum than the solution of (**P**) using a local NLP algorithm, in practice, this generally does not occur, especially when using the solution of (**PR**) as a starting point.

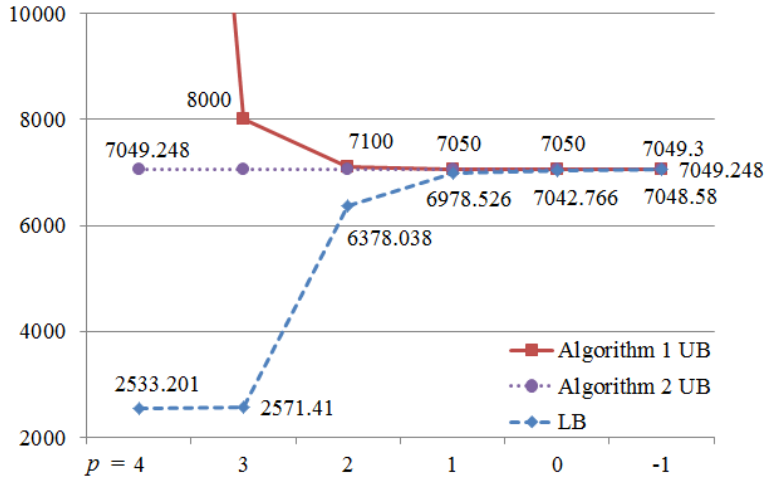

Figure 5: Comparison of Algorithm 1 and 2 for problem (**P3a**). The upper bound from Algorithm 1 for p = 4 is equal to 30000 (off scale).

7.3. Results for larger problems

As seen in Tables 1-4, when compared to the commercial solver BARON, multiparametric disaggregation is competitive in (**P1**), (**P2**) and (**P4**) but is orders of magnitude slower in (**P3**). This is to be expected given the very small size of problems and the reduced number of bilinear terms, which facilitates the spatial branch-and-bound procedure in BARON. In order to evaluate how both methods scale with problem size, we solved the five most difficult (higher optimality gap at termination when solved with BARON) water-using network design problems in Teles et al. (2012) [11], which also correspond to bilinear programs. In such problems, the discretized variables are concentrations featuring $P = 3$ in Ex17 and $P = 2$ in the other examples. Given the larger size, it is not possible to be as demanding in terms of accuracy, with difficulties arising already for $p = 0$ (MDT). However, the results in Table 5 show that the optimality gaps for MDT are at least one third those of BARON. In particular, Ex17 and Ex18 can be solved to global optimality in a few seconds due to the solution from the relaxed problem (**PR**) being equal to that of (**P**). It should be highlighted that in the full set of problems [11], MDT performs better than BARON 90% of the time. GLoMIQO can also solve two problems to global optimality (Ex14 and Ex15), interestingly different problems than what MDT can solve, and is always better than BARON.

In Table 6, several multiperiod blending problems [45, 46] are solved using BARON, GloMIQO, and Algorithm 3 at a single level of discretization. These problems are multiperiod blending problems with varying numbers of tanks, time periods, and product qualities. Each problem is identified such that 6T-3P-2Q-029 is a 6 tank, 3 time period, 2 quality problem, with a unique 3-digit identifier to distinguish it from other problems of the same size. The results for MDT are reported for $P = 0$ and $p = -3$ and were solved using Gurobi [47] using 12 threads. As these problems are originally MINLPs, are still larger than those in Table 5, and 12 threads are utilized, the computational difference is much more significant. GloMIQO and BARON both use MILP solvers in their algorithms which can utilize multiple threads, but this effect is not significant as the

subproblems being solved are generally very small. Because of the use of multiple threads and multiple CPUs in this comparison, wall times are reported for all three methods. While GloMIQO and BARON are unable to converge to an optimality gap of 0.1% within the time limit of 2 hours, the multiparametric disaggregation technique is able to close the gap in all but two cases. In these two cases an additional level of discretization would close the gap, but these results show that if a reasonably fine discretization is chosen a priori, the optimality gaps can be significantly less than those of commercial global optimization solvers. For the one problem that BARON and GloMIQO were able to close the gap within the time limit, MDT outperformed them by ˜17 and ˜1400 times, respectively, although the problem is small enough and is solved fast enough that meaningful conclusions should not be drawn from this result alone.

Table 1. Comparison between multiparametric disaggregation (MDT) and piecewise McCormick (PCM), problem (**P1**). For MDT, $P = 0$ is used.

Method	**BARON***	**MDT**	**MDT**	**MDT**	**MDT**	**MDT**	**MDT**	**MDT**	**PCM**	**PCM**	**PCM**	**PCM**	**PCM**
p* or *N	-	0	-1	-2	-3	-4	-5	-6	1	10	100	1000	10000
Binary Variables	0	10	20	30	40	50	60	70	1	10	100	1000	10000
Total Variables	3	28	48	68	88	108	128	148	7	34	304	3004	30004
Equations	3	21	33	45	57	69	81	93	14	50	410	4010	40010
Lower Bound (CPLEX)	-1.08334	-1.3333	-1.1167	-1.0867	-1.0837	-1.08337	-1.08334	-1.08333	-1.5	-1.13077	-1.08830	-1.08383	-1.08338
Upper Bound (CONOPT)	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333	-1.08333
Optimality Gap	0.001%	23.074%	3.080%	0.311%	0.034%	0.003%	0.000%	0.000%	38.462%	4.379%	0.459%	0.046%	0.005%
CPU Time (s)	0.232	0.084	0.245	0.231	0.239	0.457	0.377	0.465	0.229	0.211	0.488	6.898	673.503

*****Applied directly to (**P1**).

Table 2. Comparison between multiparametric disaggregation (MDT) and piecewise McCormick (PCM), problem (**P2**). For MDT, $P = (2,1,1)$ is used.

Method	**BARON**	**MDT**	**MDT**	**MDT**	**MDT**	**MDT**	**PCM**	**PCM**	**PCM**
p* or *N	-	(0,0,0)	(-1,-1,-1)	(-2,-2,-2)	(-3,-3,-3)	(-4,-4,-4)	(24,12,18)	(240,120,180)	(2400,1200,1800)
Binary variables	0	47	77	107	137	167	54	540	5400
Total variables	9	252	392	532	672	812	296	2780	27620
Equations	10	470	704	938	1172	1406	513	4509	44469
Lower Bound (CPLEX)	10122.48	10121.33	10121.33	10122.21	10122.46	10122.49	10121.84	10121.84	10122.29
Upper Bound (CONOPT)	10122.49	10122.49	10122.49	10122.49	10122.49	10122.49	10122.49	10122.49	10122.49
Optimality gap	0.0001%	0.011%	0.011%	0.003%	0.0003%	0.0000%	0.006%	0.006%	0.002%
CPU Time (s)	1.73	0.26	0.44	0.54	0.56	0.67	0.35	1.66	284

Table 3. Comparison between multiparametric disaggregation (MDT) and piecewise McCormick (PCM), problem (**P3**).

		BARON*	**MDT**	**MDT**	**MDT**	**MDT**	**PCM**	**PCM**
***P*=(4,4,4)**	***p* or *N***		(2,2,2)	(1,1,1)	(0,0,0)	(-1,-1,-1)	(99,90,90)	(990,900,900)
(P3a)	**Lower Bound (CPLEX)**	7049.128	6378.038	6978.526	7042.766	7048.580	6378.038	4750.289
	Upper Bound (CONOPT)	7049.135	7049.248	7049.248	7049.248	7049.248	7049.248	7049.248
	Optimality gap	0.0001%	10.5%	1.01%	0.09%	0.009%	10.5%	48.4%
	CPU Time (s)	1.07	1.20	22.9	234	3190	8.20	*3600*
***P*=(4,3,3,3)**	***p* or *N***		(1,1,1,1)	(0,0,0,0)	(-1,-1,-1,-1)	(-2,-2,-2,-2)	(900,99,99,99)	(9000,990,990,990)
(P3b)	**Lower Bound (CPLEX)**		6610.973	7002.303	7044.518	7048.775	6610.973	2732.460
	Upper Bound (CONOPT)		7049.248	7049.248	7049.248	7049.248	7049.248	7049.248
	Optimality gap		6.63%	0.67%	0.067%	0.0067%	6.63%	158%
	CPU Time (s)		1.81	31.8	288	2144	310	*3600*
***P*=(3,3,3,3,3)**	***p* or *N***		(1,1,1,1,1)	(0,0,0,0,0)	(-1,-1,-1,-1,-1)	(-2,-2,-2,-2,-2)	(99,99,99,99,99)	(990,990,990,990,990)
(P3c)	**Lower Bound (CPLEX)**		6591.393	6999.466	7044.224	7048.745	6591.393	2884.311
	Upper Bound (CONOPT)		7049.248	7049.248	7049.248	7049.248	7049.248	7049.248
	Optimality gap		6.95%	0.71%	0.07%	0.007%	6.95%	144%
	CPU Time (s)		1.20	10.3	53.1	591	16.0	*3600*

*****Applied directly to (**P3**).

Table 4. Comparison between multiparametric disaggregation (MDT) and piecewise McCormick (PCM), problem (**P4**). For MDT, $P = (1,1,1)$ is used.

Method	**BARON***	**MDT**	**MDT**	**MDT**	**MDT**	**MDT**	**PCM**	**PCM**	**PCM**	**PCM**
p* or *N		(0,0)	(-1,-1)	(-2,-2)	(-3,-3)	(-4,-4)	(5,10,10)	(50,100,100)	(500,1000,1000)	(5000,10000,10000)
Binary variables	0	35	65	95	125	155	25	250	2500	25000
Total variables	7	92	152	212	272	332	85	760	7510	75010
Equations	6	124	190	256	322	388	127	1027	10027	100027
Lower Bound (CPLEX)	460211.8	458712.10	459917.9	460177.9	460209.7	460211.8	457162.4	459976.0	460171.7	460209.0
Upper Bound (CONOPT)	460212.3	460212.30	460212.3	460212.3	460212.3	460212.3	460212.3	460212.3	460212.3	460212.3
Optimality gap	0.0001%	0.33%	0.06%	0.007%	0.0006%	0.0001%	0.67%	0.05%	0.009%	0.0007%
CPU Time (s)	1.10	0.23	0.87	1.09	1.20	1.10	0.23	0.56	3.64	866

*****Applied directly to (**P4**).

Table 5. Comparison for water-using networks design problems [11]. For MDT, $P = 2$ is used, except for Ex17, where $P = 3$.

	Original NLP		BARON		GloMIQO		MDT ($p = 1$)					MDT ($p = 0$)				
Problem	**Variables**	**Equations**	**CPU Time (s)**	**Gap**	**CPU Time (s)**	**Gap**	**Binary Variables**	**Total Variables**	**Equations**	**CPU Time (s)**	**Gap**	**Binary Variables**	**Total Variables**	**Equations**	**CPU Time (s)**	**Gap**
Ex14	123	81	*3600*	2.00%	2803	0.00%	263	3138	1973	25.2	2.00%	503	5298	2429	*3600*	0.65%
Ex15	136	87	*3600*	1.73%	2047	0.00%	334	3952	2482	44.6	1.77%	634	6652	3052	*3600*	0.59%
Ex17	74	38	*3600*	3.02%	*3600*	2.85%	182	1600	857	2.27	0.00%	-	-	-	-	-
Ex18	60	37	*3600*	1.11%	*3600*	0.74%	124	1020	677	1.10	0.40%	244	1740	869	4.67	0.00%
Ex20	171	84	*3600*	3.08%	*3600*	2.57%	263	3724	2085	1191	0.38%	463	5924	2505	*3600*	0.46%

Table 6. Comparison for multiperiod blending problems [45, 46]. For MDT, $P = 1$ is used.

	Original MINLP			BARON		GloMIQO		MDT ($p = -3$)				
Problem	**Binary Variables**	**Total Variables**	**Equations**	**Wall Time (s)**	**Gap**	**Wall Time (s)**	**Gap**	**Binary Variables**	**Total Variables**	**Equations**	**Wall Time (s)**	**Gap**
6T-3P-2Q-029	36	103	214	21.12	0.10%	1771	0.10%	420	1819	1990	1.25	0.00%
8T-3P-2Q-146	87	223	624	*7200*	18.0%	*7200*	1.96%	855	5743	7441	870.88	0.16%
8T-3P-2Q-718	87	223	607	*7200*	62.8%	*7200*	76.2%	855	5569	7151	97.69	0.00%
8T-3P-2Q-721	87	223	628	*7200*	8.82%	*7200*	1.04%	855	5743	7444	11.78	0.00%
8T-4P-2Q-480	124	313	885	*7200*	134%	*7200*	0.39%	1148	8233	10093	741.28	0.41%
8T-4P-2Q-531	104	273	737	*7200*	12.1%	*	*	1128	7933	9577	31.84	0.00%
8T-4P-2Q-852	120	305	861	*7200*	11.3%	*7200*	0.26%	1144	8225	10069	22.10	0.00%

*Solver Failure

Conclusions

This paper has presented a derivation of the multiparametric disaggregation technique for solving bilinear programming problems. Lower bounding properties have also been established for the corresponding relaxation problem. As has been shown with the smaller test problems, the relaxation from the multiparametric disaggregation technique was shown to scale more favorably than the relaxation based on piecewise McCormick envelopes (PCM). Although the MDT is generally not as tight as the PCM relaxation, the MDT relaxation technique outperforms the PCM relaxation in terms of computational time due to the more favorable scaling of problem size as discretization is increased. For large problems it was shown that multiparametric disaggregation can outperform general global solvers such as BARON and GloMIQO.

Acknowledgments

Pedro Castro gratefully acknowledges financial support from the Luso-American Foundation, under the 2011 Portugal-U.S. Research Networks Program. Ignacio Grossmann and Scott Kolodziej acknowledge financial support from the National Science Foundation under Grant OCI-0750826

References

[1] Misener, R., Thompson, J.P., Floudas, C.A. (2011). APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes, Computers & Chemical Engineering, 35(5), 876-892 (2011)

[2] Misener, R., and Floudas, C. A.: Advances for the pooling problem: Modeling, global optimization, and computational studies. Applied and Computational Mathematics 8(1), 3-22 (2009)

[3] Bagajewicz, M.: A review of recent design procedures for water networks in refineries and process plants. Computers & Chemical Engineering 24(9–10), 2093-2113 (2000)

[4] Jeżowski, J.: Review of Water Network Design Methods with Literature Annotations. Industrial & Engineering Chemistry Research 49 (10), 4475-4516 (2010)

[5] Haverly, C. A.: Studies of the behavior of recursion for the pooling problem. SIGMAP Bull. 25, 19-28 (1978)

[6] Quesada, I., Grossmann, I.E.: Global optimization of bilinear process networks with multicomponent flows. Computers & Chemical Engineering, 19, Issue 12, 1219-1242 (1995)

[7] Tawarmalani, M., and Sahinidis, N. V.: Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming. Kluwer Academic Publishers. 254–284 (2002)

[8] Meyer, C. A., and Floudas, C. A.: Global optimization of a combinatorially complex generalized pooling problem. AIChE Journal 52(3), 1027–1037 (2006)

[9] Misener, R., and Floudas, C. A.: Global optimization of large-scale generalized pooling problems: Quadratically constrained MINLP models. Industrial & Engineering Chemistry Research 49(11):5424–5438 (2010)

[10] Karuppiah, R., Grossmann, I.E.: Global optimization for the synthesis of integrated water systems in chemical processes. Computers and Chemical Engineering 30, 650-673 (2006).

[11] Teles, J.P., Castro, P.M., Matos, H.A.: Global Optimization of Water Networks Design using Multiparametric Disaggregation. Computers and Chemical Engineering 40, 132-147 (2012).

[12] Ahmetović, E. and Grossmann, I. E.: Global superstructure optimization for the design of integrated process water networks. AIChE Journal. 57 (2), 434-457 (2010)

[13] Sherali, H.D., Alameddine, A.: A new reformulation linearization technique for bilinear programming problems. Journal of Global Optimization 2, 379-410 (1992)

[14] Liberti, L., Cafieri, S., Tarissan, F.: Reformulations in Mathematical Programming: A Computational Approach. In Abraham, A., Hassanien, A., Siarry, P., and Engelbrecht, A. (eds.) Foundations of Computational Intelligence Volume 3, pp. 153-234. Springer Berlin, Heidelberg (2009)

[15] Liberti, L., Pantelides, C.C.: An Exact Reformulation Algorithm for Large Nonconvex NLPs Involving Bilinear Terms. Journal of Global Optimization, 36, 161 (2006)

[16] Ruiz, J.P., Grossmann, I.E.: Exploiting vector space properties to strengthen the relaxation of bilinear programs arising in the global optimization of process networks. Optimization Letters, 5, 1 (2011)

[17] Wicaksono, D. S. and Karimi, I. A.: Piecewise MILP under- and overestimators for global optimization of bilinear programs. AIChE Journal. 54 (4), 991-1008 (2008)

[18] Al-Khayyal, F. A., and Falk J. E.: Jointly Constrained Biconvex Programming. Mathematics of Operations Research, 8 (2), 273-286 (1983)

[19] Smith, E. M. B., and Pantelides, C. C.: Global optimisation of nonconvex MINLPs. Computers & Chemical Engineering. 21, S791-S796 (1997)

[20] Horst, R., and Tuy, H.: Global Optimization: Deterministic Approaches. Springer (1996)

[21] Floudas, C. A. and Visweswaran, V.: Quadratic Optimization. In Horst, R. and Pardalos, P. M. (eds.) Handbook of Global Optimization. Kluwer (1995)

[22] Shor, N.: Dual quadratic estimates in polynomial and Boolean programming. Annals of Operations Research. 25 (1), 163-168 (1990)

[23] Xu, H.K.: An Iterative Approach to Quadratic Optimization. Journal of Optimization Theory and Applications. 116 (3), 659- 678 (2003)

[24] Yu, N.: Semidefinite relaxation and nonconvex quadratic optimization. Optimization Methods and Software. 9 (1-3), 141-160 (1998)

[25] Ye, Y.: Approximating quadratic programming with bound and quadratic constraints. Mathematical Programming. 84 (2), 219-226 (1999)

[26] Adhya, N., Tawarmalani, M., and Sahinidis, N.V.: A Lagrangian Approach to the Pooling Problem. Industrial & Engineering Chemistry Research. 38 (5), 1956-1972 (1999)

[27] Bergamini, M.L., Aguirre, P., Grossmann, I.E.: Logic-based outer approximation for globally optimal synthesis of process networks. Computers and Chemical Engineering. 29, 1914-1933 (2005).

[28] Vielma, J. P., Ahmed, S., Nemhauser, G.: Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. Operations Research, 58, 303–315 (2010)

[29] Vielma, J. P., Nemhauser, G.: Modeling disjunctive constraints with a logarithmic number of binary variables and constraints, Mathematical Programming In Press (2010). DOI: 10.1007/s10107–009-0295–4

[30] Teles, J.P., Castro, P.M., Matos, H.A.: Multiparametric disaggregation technique for global optimization of polynomial programming problems. Journal of Global Optimization (2011). DOI: 10.1007/s10898-011-9809-8.

[31] Grossmann, I.E. and Ruiz, J.P.: Generalized Disjunctive Programming: A Framework for Formulation and Alternative Algorithms for MINLP Optimization. IMA Volume 154, Mixed Integer Nonlinear Programming, (eds., Jon Lee and Sven Leyffer) (2011)

[32] Oral, M., and Kettani, O.: A Linearization Procedure for Quadratic and Cubic Mixed-Integer Problems. Operations Research. 40, Supplement 1: Optimization, S109-S116 (1992)

[33] Balas, E.: Disjunctive Programming and a Hierarchy of Relaxations for Discrete Optimization Problems. SIAM Journal of Algebraic and Discrete Mathematics 6, 466-486 (1985)

[34] Grossmann, I. E.: Review of Nonlinear Mixed-Integer and Disjunctive Programming Techniques. Optimization and Engineering. 3 (3), 227-252 (2002)

[35] McCormick, G.P.: Computability of global solutions to factorable nonconvex programs. Part I. Convex underestimating problems. Mathematical Programming 10, 146 (1976).

[36] Gounaris, C. E., Misener, R., and Floudas, C. A.: Computational Comparison of Piecewise−Linear Relaxations for Pooling Problems. Industrial & Engineering Chemistry Research. 48 (12), 5742-5766 (2009)

[37] Quesada, I., and Grossmann, I. E.: A global optimization algorithm for linear fractional and bilinear programs. Journal of Global Optimization. 6 (1), 39-76 (1995)

[38] Brook, A., Kendrick, D., and Meeraus, A.: GAMS, a user's guide. ACM SIGNUM Newsletter. 23 (3-4), (1988)

[39] IBM. IBM ILOG CPLEX V12.1—User's Manual for CPLEX, IBM (2009)

[40] Drud, A. S.: CONOPT—A Large-Scale GRG Code. INFORMS Journal on Computing. 6 (2), 207-216 (1994)

[41] Sahinidis, N.: BARON: A general purpose global optimization software package. Journal of Global Optimization. 8, 201-205 (1996)

[42] Rijckaert, M., and Martens, X.: Comparison of generalized geometric programming algoritms. J. Opt. Th Appl. 26, 205 (1978)

[43] Hock, W., and Schittkowski, K.: Test examples for nonlinear programming codes. Vol. 187 of Lecture Notes in Economics and Mathematical Systems. Springer Verlag. (1981)

[44] Shen, P., and Zhang, K.: Global optimization of signomial geometric programming using linear relaxation. Applied Mathematics and Computation. 150 (1), 99–114 (2004)

[45] Kolodziej, S. P.: Global Optimization of the Multiperiod Blending Problem. Master's Thesis. Carnegie Mellon University, Pittsburgh, PA. (2012)

[46] Kolodziej, S. P., Grossmann, I.E., Furman, K.C., and Sawaya, N.W.: A Novel Global Optimization Approach to the Multiperiod Blending Problem. Unpublished.

[47] Gurobi Optimizer Reference Manual Version 4.5. Gurobi Optimization. http://www.gurobi.com/doc/45/refman/ (2011)