Nonconvex piecewise linear functions: Advanced formulations and simple modeling tools

Joey Huchette
Operations Research Center, Massachusetts Institute of Technology, MA 02139, huchette@mit.edu

Juan Pablo Vielma
Sloan School of Management, Massachusetts Institute of Technology, MA 02139, jvielma@mit.edu

We present novel mixed-integer programming (MIP) formulations for (nonconvex) piecewise linear functions. Leveraging recent advances in the systematic construction of MIP formulations for disjunctive sets, we derive new formulations for univariate functions using a geometric approach, and for bivariate functions using a combinatorial approach. All formulations derived are small (logarithmic in the number of piecewise segments of the function domain) and strong, and we present extensive computational experiments in which they offer substantial computational performance gains over existing approaches. We characterize the connection between our geometric and combinatorial formulation approaches, and explore the benefits and drawbacks of both. Finally, we present PiecewiseLinearOpt, an extension of the JuMP modeling language in Julia that implements our models (alongside other formulations from the literature) through a high-level interface, hiding the complexity of the formulations from the end-user.

Key words: Piecewise linear, Integer programming

1. Introduction

Consider an optimization problem of the form \( \min_{x \in Q} f(x) \), where \( Q \subseteq \mathbb{R}^n \) and \( f: D \subseteq \mathbb{R}^n \rightarrow \mathbb{R} \) is a piecewise linear (PWL) function. That is, \( f \) can be described by a partition of the domain \( D \) into a finite family \( \{P^i\}_{i=1}^d \) of polyhedra, where for each domain piece \( P^i \) there is an affine function \( f^i: P^i \rightarrow \mathbb{R} \) whereby \( x \in P^i \implies f(x) = f^i(x) \). In the same way, we could imagine an optimization problem where \( Q \) is (partially) defined by a constraint of the form \( f(x) \leq 0 \), \( f \) is piecewise linear.

The potential applications for this class of optimization problems are legion. Piecewise linear functions arise naturally throughout operations (Croxton et al. 2003, 2007, Liu and Wang 2015) and engineering (Fügenschuh et al. 2014, Graf et al. 1990, Silva et al. 2012). They are a natural
choice for approximating nonlinear functions, as they often lead to optimization problems that are easier to solve than the original problem (Bergamini et al. 2005, 2008, Castro and Teles 2013, Geißler et al. 2012, Kolodziej et al. 2013, Misener et al. 2011, Misener and Floudas 2012). For example, there has been recently been significant interest in using piecewise linear functions for modeling gas network optimization problems (Codas and Camponogara 2012, Codas et al. 2012, Martin et al. 2006, Mahlke et al. 2010, Misener et al. 2009, Silva and Camponogara 2014).

If the function $f$ happens to be convex, it is possible to reformulate our optimization problem into an equivalent linear programming (LP) problem. However, if $f$ is nonconvex, this problem is NP-hard in general (Keha et al. 2006). A number of specialized algorithms for solving piecewise linear optimization problems have been proposed over the years (Beale and Tomlin 1970, de Farias Jr. et al. 2008, 2013, Keha et al. 2006, Tomlin 1981). Another popular approach is to use mixed-integer programming (MIP) to encode the logical constraints $x \in P^i \implies f(x) = f_i(x)$ using auxiliary integer variables. There are many possible ways to do this, and the MIP approach to modeling optimization problems containing piecewise linear functions has been an active and fruitful area of research for decades (Balakrishnan and Graves 1989, Croxton et al. 2003, D’Ambrosio et al. 2010, Dantzig 1960, Jeroslow and Lowe 1984, 1985, Keha et al. 2004, Lee and Wilson 2001, Magnanti and Stratila 2004, Markowitz and Manne 1957, Padberg 2000, Sherali and Wang 2001, Vielma and Nemhauser 2011, Vielma et al. 2010, Wilson 1998). This line of work has produced a large number of MIP formulations that exploit the high performance and flexibility offered by modern MIP solvers (Bixby and Rothberg 2007, Jünger et al. 2010), with varying degrees of success. The 2010 Operations Research paper of Vielma et al. (2010) compiled these formulations into a unified framework and provided extensive comparisons of their effectiveness. A notable conclusion of this study was the overwhelming computational advantage of the logarithmic formulations introduced in Vielma and Nemhauser (2011). These two papers have been extensively cited in the literature, and have sparked attempts to construct logarithmic formulations for other constraints beyond PWL functions (Huchette and Vielma 2016, Huchette et al. 2017, Vielma 2017). However, the complexity
of the logarithmic formulations has resulted in a relatively low rate of adoption in practice. In this paper, we present novel logarithmic formulations for PWL functions that improve on the state-of-the-art, and also provide accessible software tools for their use. In addition, while extending and improving the logarithmic formulation paradigm through the techniques of (Huchette and Vielma 2016, Vielma 2017), we study advanced formulation topics concerning redundancy and branching effectiveness. Specifically, the main contributions of this paper include:

1. **Improved binary and general integer formulations for univariate functions.** In Section 3 we use the geometric technique of Vielma (2017) to construct two new logarithmic formulations for univariate PWL functions. We show how these formulations computationally outperform all formulations (including the original logarithmic formulation) in regimes that are known to be problematic for the original logarithmic formulation. One of these formulations is the first effective formulation for PWL functions using general integer variables, instead of binary (i.e. 0/1) variables.

2. **Generalized hybrid formulations for bivariate functions.** In Section 4 we extend the applicability of the logarithmic formulation for bivariate functions, which was previously restricted to a very specific class of functions. We develop several families of formulations for general bivariate functions by combining the geometric and combinatorial characterizations of the logarithmic formulation developed in Vielma (2017) and Huchette and Vielma (2016), respectively. Furthermore, we show that for a vast class of disjunctive constraints (including this setting), the common loss of strength resulting from intersecting MIP formulations is entirely avoided. Finally, we show that the resulting formulations can provide a significant computational advantage over all known formulations.

3. **Accessible modeling library for advanced formulations.** In Section 5, we present a `PiecewiseLinearOpt`, an extension of the JuMP algebraic modeling language (Dunning et al. 2017) that offers a high-level way to model piecewise linear functions in practice. The package supports all the MIP formulations for piecewise linear functions discussed in this work, and generates them automatically and transparently. All computational experiments in the paper use this package.
4. Branching effectiveness and redundancy in advanced formulations. Finally, in Section 6 we consider two important technical aspects of effective formulations. First, we study the branching properties of the original logarithmic formulation, which has been observed to be less than ideal. In particular, we illustrate how the new formulations from Section 3 are specifically designed to produce superior branching behavior. Second, we consider the difference between the geometric and combinatorial formulation techniques of Vielma (2017) and Huchette and Vielma (2016), which do not always yield the same version of the logarithmic formulation. We give a precise connection between these techniques and use it to show how adding redundancy to disjunctive constraints can be exploited to construct smaller formulations.

We begin by reviewing standard formulation techniques for PWL functions, along with the advanced techniques of Vielma (2017) and Huchette and Vielma (2016).

2. Combinatorial disjunctive constraints of piecewise linear functions

In this work we will focus on continuous piecewise linear functions; for those interested in modeling discontinuous functions, we refer the reader to Vielma et al. (2010). Furthermore, we assume that our function $f : \mathbb{R}^n \to \mathbb{R}$ is non-separable and cannot be decomposed as the sum of piecewise linear functions over domains that are strict subsets of the components of the domain of $f$. This is without loss of generality, as if such a decomposition exists, we could apply our formulation techniques to the individual pieces separately. Finally, we will focus primarily on the regime where the dimension $n$ of the domain is relatively small: when $f$ is either univariate ($n = 1$) or bivariate ($n = 2$). Low dimensional piecewise linear functions are broadly applicable (especially with the non-separability assumption), and are sufficiently complex to warrant in-depth analysis.

Consider a continuous piecewise linear function $f : D \to \mathbb{R}$, where $D \subset \mathbb{R}^n$ is bounded and polyhedral. To embed $f$ in an optimization problem, we will construct a formulation for its graph $\text{gr}(f) \triangleq \{(x, f(x)) : x \in D\}$, which will couple the argument $x$ with the function output $f(x)$. We can view the graph disjunctively, as the union over the pieces of the graph, $\text{gr}(f) = \bigcup_{i=1}^{d} S^i$, where $S^i = \{(x, f^i(x)) : x \in P^i\}$ for each $i \in \{1, \ldots, d\}$. As each of the domain pieces $P^i$ is a polyhedra, and each $f^i$ is affine, we have a representation for $\text{gr}(f)$ as a union of polyhedra.
Example 1. Consider the univariate piecewise linear function $f : [0, 4] \to \mathbb{R}$ given as

\begin{align*}
x \in [0, 1] & \implies f(x) = 4x, & x \in [1, 2] & \implies f(x) = 3x + 1, \\
x \in [2, 3] & \implies f(x) = 2x + 3, & x \in [3, 4] & \implies f(x) = x + 6,
\end{align*}

along with its graph

$\text{gr}(f) = \{(x, 4x) : x \in [0, 1]\} \cup \{(x, 3x + 1) : x \in [1, 2]\} \cup \{(x, 2x + 3) : x \in [2, 3]\} \cup \{(x, x + 6) : x \in [3, 4]\}$.

It is possible to construct a MIP formulation for this disjunctive set directly. For example, the “multiple choice” (MC) formulations we will compare against in the computational experiments can be derived through generic MIP formulation techniques for disjunctive sets. Our approach will be to formulate the constraint in terms of the extreme points $\text{ext}(S^i)$ of each of the pieces $S^i$ of the graph. For univariate functions, this reduces to the classical special ordered set of type 2 (SOS2) constraint (Beale and Tomlin 1970). Notationally, for some finite set $V$, we define $\Delta^V \overset{\text{def}}{=} \{\lambda \in \mathbb{R}^V_+ : \sum_{v \in V} \lambda_v = 1\}$ and $P(T) \overset{\text{def}}{=} \{\lambda \in \Delta^V : \text{supp}(\lambda) \subseteq T\}$, where $\text{supp}(\lambda) \overset{\text{def}}{=} \{v \in V : \lambda_v \neq 0\}$.

Example 2. Let $f : [0, 4] \to \mathbb{R}$ be the piecewise linear function from Example 1 with $d = 4$ pieces. Observe that $(t_i)_{i=1}^5 = (0, 1, 2, 3, 4)$ is the set of all breakpoints of $f$; we take $V = [d+1]$ as the corresponding indices. Then $(x, z) \in \text{gr}(f)$ if and only if $(x, z) = \sum_{i=1}^{d+1} (t_i, f(t_i)) \lambda_i$ for some $\lambda \in \bigcup_{i=1}^d P(\{i, i+1\})$. We refer to $\bigcup_{i=1}^d P(\{i, i+1\})$ as the SOS2 constraint on $d$ pieces.

The standard “convex combination” (CC) formulation for SOS2 with $d = 4$ pieces is

$\lambda_1 \leq y_1, \quad \lambda_2 \leq y_1 + y_2, \quad \lambda_3 \leq y_2 + y_3, \quad \lambda_4 \leq y_3 + y_4, \quad \lambda_5 \leq y_4, \quad \sum_{i=1}^d y_i = 1, \quad (\lambda, y) \in \Delta^{d+1} \times \{0, 1\}^d$.

This formulation is sharp, as its LP relaxation transformed to the original $(x, z)$-space as in Example 2 is $\text{Conv}(\text{gr}(f))$, the tightest possible convex relaxation. However, it is not ideal, or as strong as possible, as not all extreme points of the LP relaxation naturally satisfy the integrality conditions on $y$. In contrast, the logarithmic independent branching (LogIB) formulation (Vielma and Nemhauser 2011) is ideal, and also much smaller than CC (and most other formulations), whose size
scales linearly in the number of pieces. As a result, the LogIB formulation can perform significantly better computationally than other existing formulations (Vielma et al. 2010).

For a generic PWL function \( f \) and disjunctive constraint \( \text{gr}(f) = \bigcup_{i=1}^{d} S_i \), take \( V = \bigcup_{i=1}^{d} \text{ext}(S_i) \) and the family \( \mathcal{T} = (T^i = \text{ext}(P^i))_{i=1}^{d} \), which describe the underlying combinatorial structure among the pieces of \( \text{gr}(f) \) induced by their shared breakpoints. Then we may express the graph in terms of \( \mathcal{T} \) as \( \text{gr}(f) = \left\{ \sum_{v \in V} \lambda_v (v, f(v)) : \lambda \in \bigcup_{i=1}^{d} P(T^i) \right\} \) and we can build a formulation for \( f \) through the combinatorial disjunctive constraint \( \lambda \in \bigcup_{i=1}^{d} P(T^i) \) (Huchette and Vielma 2016). We now detail the two formulation approaches that we use in this work. For conciseness, we will interchangeably refer to the combinatorial disjunctive constraint \( \bigcup_{i=1}^{d} P(T^i) \) by the associated family of sets \( \mathcal{T} = (T^i)_{i=1}^{d} \).

### 2.1. Independent branching formulations

The LogIB formulation is derived from the independent branching class of formulations, a combinatorial way of constructing formulations for disjunctive constraints. This approach was first developed for univariate and bivariate piecewise linear functions by Vielma and Nemhauser (2011), and a complete characterization of its expressive power is given by Huchette and Vielma (2016). One way to systematically construct independent branching formulations is through the following graphical procedure introduced by Huchette and Vielma (2016).

**Proposition 1 (Huchette and Vielma (2016)).** Let \( \mathcal{T} = (T^i \subseteq V)_{i=1}^{d} \) be a combinatorial disjunctive constraint and let \( \left\{ (A^k, B^k) \right\}_{k=1}^{r} \) be such that

- \( A^k, B^k \subseteq V \) and \( A^k \cap B^k = \emptyset \) for all \( k \in \llbracket r \rrbracket \), and
- for \( u, v \in T \) with \( u \neq v \), \( \{u, v\} \not \subseteq T^i \) for all \( i \in \llbracket d \rrbracket \) if and only if \( (u, v) \in (A^k \times B^k) \cup (B^k \times A^k) \) for some \( k \in \llbracket r \rrbracket \).

Then, an independent branching formulation for \( \bigcup_{i=1}^{d} P(T^i) \) is given by the ideal formulation

\[
\sum_{v \in A^k} \lambda_v \leq y_k, \quad \sum_{v \in B^k} \lambda_v \leq 1 - y_k \quad \forall k \in \llbracket r \rrbracket, \quad (\lambda, y) \in \Delta^V \times \{0, 1\}^r.
\]

We say that \( \left\{ (A^k, B^k) \right\}_{k=1}^{r} \) is a biclique representation of \( \mathcal{T} \) with \( r \) levels as it corresponds to a biclique cover of a graph associated to \( \mathcal{T} \) (see Huchette and Vielma (2016) for more details).
Example 3. The \( \text{LogIB} \) formulation for the SOS2 constraint with \( d = 4 \) arising in (1) is given by \( r = 2 \) and the sets \( A^1 = \{3\} \), \( B^1 = \{1, 5\} \), \( A^2 = \{4, 5\} \), and \( B^2 = \{1, 2\} \):

\[
\lambda_3 \leq y_1, \quad \lambda_1 + \lambda_5 \leq 1 - y_1, \quad \lambda_4 + \lambda_5 \leq y_2, \quad \lambda_1 + \lambda_2 \leq 1 - y_2, \quad (\lambda, y) \in \Delta^{d+1} \times \{0, 1\}^2.
\]

### 2.2. Embedding formulations

The embedding approach for formulating disjunctive constraints \( \bigcup_{i=1}^{d} S^i \) works by selecting an encoding of length \( r \), given by \( H \in \mathcal{H}_r(d) \) \( \defeq \{ H \in \{0, 1\}^{d \times r} : H_i \neq H_j \forall i \neq j \} \), where \( H_i \) denotes the \( i \)-th row of \( H \). Each alternative \( S^i \) is assigned its unique code \( H_i \), and the disjunctive set is “embedded” in a higher-dimensional space as \( \bigcup_{i=1}^{d} (S^i \times \{H_i\}) \). This easily leads to a MIP formulation for \( \bigcup_{i=1}^{d} S^i \) as follows (the restriction to the combinatorial case is for notational convenience).

**Proposition 2.** Let \( \mathcal{T} = (T^i \subseteq V)_{i=1}^{d} \) be a combinatorial disjunctive constraint and take \( r \geq \lfloor \log_2(d) \rfloor \) and \( H \in \mathcal{H}_r(d) \). Define the embedding \( \text{Em}(\mathcal{T}, H) \) \( \defeq \bigcup_{i=1}^{d} P(T^i) \times \{H_i\} \), and let \( Q(\mathcal{T}, H) \) \( \defeq \text{Conv}(\text{Em}(\mathcal{T}, H)) \). Then \( Q(\mathcal{T}, H) \) is a rational polyhedron and an ideal formulation for \( \bigcup_{i=1}^{d} P(T^i) \) is \( (\lambda, y) \in Q(\mathcal{T}, H) : y \in \mathbb{Z}^r \). We call this the embedding formulation of \( \mathcal{T} \) associated to \( H \).

In general, constructing a linear inequality description of \( Q(\mathcal{T}, H) \) is very difficult, the resulting representation may be exponentially large, and its structure is highly dependent on the interplay between the sets \( \mathcal{T} \) and the encoding \( H \). Fortunately, Vielma (2017) gave an explicit description of \( Q(\mathcal{T}, H) \) for the SOS2 constraint with any choice of encoding \( H \). This description is geometric, in terms of the differences \( \{e^i = H_{i+1} - H_i\}_{i=1}^{d-1} \) between adjacent codes.

**Proposition 3 (Proposition 2, Vielma (2017)).** For \( H \in \mathcal{H}_r(d) \), take \( H_0 = H_1 \) and \( H_{d+1} = H_d \). Define \( e^i \) \( \defeq H_{i+1} - H_i \) for each \( i \in [d - 1] \), and take \( L(H) = \text{aff}(H) - H_1 \), where \( \text{aff}(H) \) is the affine subspace spanned by the rows of \( H \). For \( b \in L(H) \setminus \{0\} \), let \( M(b) \) \( \defeq \{ y \in L(H) : b \cdot y = 0 \} \) be the hyperplane defined by \( b \). Finally, let \( \{b^i\}_{i=1}^{d} \subseteq L(H) \setminus \{0\} \) be such that \( \{M(b^i)\}_{i=1}^{d} \) is the set of hyperplanes spanned by \( \{e^i\}_{i=1}^{d-1} \) in \( L(H) \). If \( \mathcal{T} = \{(i, i+1)\}_{i=1}^{d} \) are the sets defining the SOS2 constraint on \( d + 1 \) breakpoints, then \( Q(\mathcal{T}, H) \) is equal to all \( (\lambda, y) \in \Delta^{d+1} \times \text{aff}(H) \) such that

\[
\sum_{v=1}^{d+1} \min \{ b^i \cdot H_v, b^i \cdot H_{v-1} \} \lambda_v \leq b^i \cdot y \leq \sum_{v=1}^{d+1} \max \{ b^i \cdot H_v, b^i \cdot H_{v-1} \} \lambda_v \quad \forall i \in [r].
\]
We obtain a variant of LogIB from Proposition 3 by using a class of encodings known as Gray codes (Savage 1997). These encodings are of the form \(K^r \in \{0,1\}^{d \times r}\) for \(r = \lfloor \log_2(d) \rfloor\), where \(\|K^r_{j+1} - K^r_j\|_1 = 1\) for all \(j \in [d-1]\). For the remainder, we will work with a particular Gray code known as the binary reflected Gray code (BRGC); see Appendix A.1 for a formal definition. We refer to the resulting formulation as Log and note that it coincides with LogIB when \(d\) is a power-of-two. We explore this relation further in Section 6.2.

3. A new embedding formulation for univariate piecewise linear functions

As stated above, the embedding approach works with encodings that are binary matrices. However, Proposition 3 also holds if the rows of \(H\) are in convex position, i.e., if \(\text{ext}(P(H)) = \{H_i\}_{i=1}^d\) for \(P(H) = \text{Conv}(\{H_i\}_{i=1}^d)\). Additionally, Proposition 2 holds if \(H\) is both in convex position and is lattice-empty: i.e., if \(\text{Conv}(H) \cap \mathbb{Z}^r = H\). Therefore, we will work with general integer encodings of the form \(\mathcal{H}_r^1(d) \triangleq \{H \in \mathbb{Z}^{d \times r} : H_i \neq H_j \forall i \neq j, \text{ext}(P(H)) = P(H) \cap \mathbb{Z}^r = \{H_i\}_{i=1}^d\}\). We will now construct two new encodings (and, therefore, two new formulations) for the SOS2 constraint by transforming the BRGC. For the remainder of the subsection, assume without loss of generality (w.l.o.g.) that \(d = 2^r\) is a power-of-two. Otherwise, construct the codes for \(\bar{d} = 2^\lceil \log_2(d) \rceil\) and take the first \(d\) elements of the sequence.

Take \(K^r\) as the BRGC for \(d = 2^r\) elements. Our first new code is the transformation of \(K^r\) to \(C^r \in \mathbb{Z}^{d \times r}\), where \(C^r_{k,i} = \sum_{j=2}^k |K^r_{j,i} - K^r_{j-1,i}|\) for each \(k \in [d]\) and \(i \in [r]\). In words, \(C^r_{k,i}\) is the number of times the sequence \((K^r_{1,i}, \ldots, K^r_{k,i})\) changes value, and is monotonic nondecreasing in \(k\). Our second encoding will be \(Z^r \in \{0,1\}^{d \times r}\), where \(Z^r_i = A(C^r_i)\) for the linear map \(A : \mathbb{R}^r \rightarrow \mathbb{R}^r\) given by \(A(y)_i = y_i - \sum_{k=i+1}^{r} y_k\) for each component \(i \in [r]\). We show the encodings for \(r = 3\) in Figure 1, and include formal recursive definitions for them in Appendix A.1. Applying Proposition 3 with the new encodings gives two new formulations for SOS2.

**Proposition 4.** For notational simplicity, take \(\alpha^j = C^r_j\) for each \(j \in [d]\), along with \(\alpha^0 = \alpha^1\) and \(\alpha^{d+1} = \alpha^d\). Then two ideal formulations for the SOS2 constraint on \(d+1\) breakpoints are given by

\[
(\lambda, y) \in \Delta^{d+1} \times \mathbb{Z}^r, \quad \sum_{v=1}^{d+1} \alpha_{i_v} \leq y_k \leq \sum_{v=1}^{d+1} \alpha_{i_v} \lambda_v, \quad \forall i \in [r], \tag{3}
\]
and
\[(\lambda, y) \in \Delta^{d+1} \times \{0, 1\}^r, \quad \sum_{v=1}^{d+1} \alpha_i^{v-1} \lambda_v \leq y_i + \sum_{k=i+1}^r 2^{k-2} y_i \leq \sum_{v=1}^{d+1} \alpha_i^v \lambda_v \quad \forall i \in [r].\] (4)

We dub (4) the binary zig-zag formulation (ZZB) for SOS2, as its associated binary encoding \(Z^r\) “zig-zags” through the interior of the unit hypercube (See Figure 1). We will refer to formulation (3) as the general integer zig-zag formulation (ZZI) because of its use of general integer encoding \(C^r\). We emphasize that ZZI and ZZB are logarithmically-sized in \(d\) and ideal: the same size and strength as the \(\text{LogIB}\) formulation of Vielma and Nemhauser (2011).

![Figure 1](image)

**Figure 1** Depiction of \(K^3\) (Left), \(C^3\) (Center), and \(Z^3\) (Right). The first row of each is marked with a dot, and the subsequent rows follow along the arrows. The axis orientation is different for \(Z^3\) for visual clarity.

### 3.1. Univariate computational experiments

To evaluate the new zig-zag formulations against the existing formulations for univariate piecewise linear functions, we reproduce the computational experiments of Vielma et al. (2010), with the addition of the ZZB and ZZI formulations. We compare against the MC, CC, \(\text{LogIB}\), and \(\text{Log}\) formulations mentioned previously, as well as the SOS2 native branching (SOS2) implementation, and the incremental (Inc) and disaggregated logarithmic (DLog) formulations as described by Vielma et al. (2010). The test instances are transportation problems whose objectives are the sum of 100 continuous nondecreasing concave univariate piecewise linear functions (see Vielma et al. (2010) for more details on the test instances). The instances are split into families where the piecewise linear functions have \(N \in \{8, 16, 32, 64\}\) pieces. We use CPLEX v12.7.0 with the JuMP algebraic
modeling library (Dunning et al. 2017) in the Julia programming language (Bezanson et al. 2017) for all computational trials, here and for the remainder of this work. All such trials were performed on an Intel i7-3770 3.40GHz Linux workstation with 32GB of RAM.

In Table 1, we present aggregated statistics for each formulation and each family of instances. The Inc and DLog formulations both work very well for smaller instances. The CC formulation is never competitive, while the SOS2 and MC are somewhat effective on smaller instances, but do not scale well beyond that. For larger instances ($N \in \{32, 64\}$), the logarithmic scaling of Log/LogIB, ZZI, and ZZB dominates, and one of them is the fastest formulation for over 85% of the instances in each family. The new formulations ZZB and ZZI are the winning formulations for roughly half of these larger instances.

In Table 2, we alter our univariate test cases such that the number of segments is not a power-of-two. In particular, we randomly drop $\log_2(N) - 1$ breakpoints from the interior, i.e. $\{t_2, \ldots, t_N\}$. We observe that the Inc formulation is superior for smaller instances, and that DLog performs relatively worse on these instances than when $N$ is a power-of-two. Additionally, the Log and LogIB formulations perform roughly the same on all families of instances, but are relatively much worse than before. In particular, we observe that the new ZZI and ZZB formulations are the best performers for larger instances, and one of the two is the fastest formulation for every instance with $N = 64$. Additionally, ZZI and ZZB both offer roughly a 1.5-2x speed-up in mean solve time over Log and LogIB for most families of instances ($N \in \{16, 32, 64\}$). We note that this relative degradation
in performance of the logarithmic formulations Log and LogIB for non powers-of-two has been observed previously in the literature (Vielma and Nemhauser 2011, Coppersmith and Lee 2005, Muldoon 2012, Muldoon et al. 2013). It is notable, then, that the new logarithmic formulations ZZI and ZZB do not exhibit this same performance penalty.

4. New hybrid formulations for bivariate piecewise linear functions

In Section 3, we modeled univariate functions by reducing them to the SOS2 constraint, which is a combinatorial description of the constraint in terms of the breakpoints. Since the combinatorial structure is sufficiently simple, it is possible to give an explicit geometric description for any possible embedding formulation of the constraint (Proposition 3). This allows us to design new formulations for univariate functions that are logarithmically-sized and display favorable computational performance properties. However, the combinatorial structure in the bivariate case is considerably more complex, and we do not have a geometric result analogous to Proposition 3 which characterizes all possible embedding formulations. Instead, we will see how we may apply the independent branching framework to construct ideal formulations combinatorially.

We will consider grid triangulations of a bounded rectangular domain \( D = [l^1, u^1] \times [l^2, u^2] \subset \mathbb{R}^2 \). We discretize each dimension \( s \in \{1, 2\} \) as \( l^s = t^s_1 < t^s_2 < \cdots < t^s_{d_s} = u^s \). This gives \( d = (d_1 + 1)(d_2 + 1) \) gridpoints, which we will represent by their indices as \( V = [d_1 + 1] \times [d_2 + 1] \). We then triangulate each subrectangle \( [t^s_{i1}, t^s_{i+1}] \times [t^s_{j1}, t^s_{j+1}] \) by partitioning it into two axis-aligned triangles. There are exactly two ways to triangulate each subrectangle in two dimensions. Then the combinatorial structure of the grid triangulation is given by the family of sets \( \mathcal{T} = \{T^i \subset V\}_{i=1}^d \), where each

| \( N \) | Metric | MC | CC | SOS2 | Inc | DLog | Log | LogIB | ZZB | ZZI |
|---|---|---|---|---|---|---|---|---|---|---|
| 6 | Mean (s) | 0.6 | 3.8 | 1.1 | 0.6 | 1.1 | 1.4 | 2.6 | 1.1 | 0.9 |
|   | Sample std | 0.3 | 4.1 | 1.5 | 0.3 | 1.0 | 1.2 | 2.4 | 0.9 | 0.5 |
|   | Win | 35 | 0 | 7 | 46 | 5 | 1 | 0 | 4 | 2 |
| 13 | Mean (s) | 3.0 | 71.2 | 4.5 | 1.7 | 4.6 | 4.4 | 4.2 | 2.4 | 2.6 |
|   | Sample std | 3.1 | 152.0 | 5.8 | 0.7 | 3.5 | 3.4 | 3.0 | 1.8 | 1.7 |
|   | Win | 11 | 0 | 9 | 47 | 11 | 0 | 0 | 15 | 7 |
| 28 | Mean (s) | 18.4 | 178.9 | 87.4 | 5.5 | 11.1 | 8.8 | 8.9 | 5.1 | 4.6 |
|   | Sample std | 26.0 | 359.3 | 309.3 | 4.4 | 8.1 | 5.6 | 5.4 | 3.7 | 2.7 |
|   | Win | 1 | 0 | 6 | 14 | 1 | 0 | 0 | 37 | 41 |
| 59 | Mean (s) | 348.7 | 541.0 | 664.3 | 17.1 | 19.1 | 16.3 | 16.0 | 9.8 | 9.3 |
|   | Sample std | 523.7 | 610.3 | 746.4 | 14.9 | 11.3 | 10.3 | 9.3 | 6.1 | 5.0 |
|   | Win | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 41 | 59 |

Table 2: Computational results for univariate transportation problems with non powers-of-two segments.
$T \in \mathcal{T}$ corresponds to the indices of the gridpoints incident to one of the triangles. The bivariate function is then taken to be piecewise linear on each of these triangles. We can model the graph of this function in the same way as for univariate functions as $\text{gr}(f) = \{(x_1, x_2, f(x_1, x_2)) : x \in D\} = \left\{ \sum_{v \in V} \lambda_v (t^1_{v_1}, t^2_{v_2}, f(t^1_{v_1}, t^2_{v_2})) : \lambda \in \bigcup_{i=1}^d P(T^i) \right\}$. As we will see, the combinatorial structure imparted by the triangulation will affect which formulations we may use. Furthermore, the choice of triangulation is not arbitrary, and affects the values the function takes over its domain. That is, bivariate piecewise linear functions are not uniquely determined by the breakpoints and the function values at those breakpoints, as is the case with univariate functions. See Figure 2 for an example.

![Figure 2](image)

**Figure 2** Two bivariate functions over $D = [0, 1]^2$ that match on the gridpoints, but differ on the interior of $D$.

### 4.1. Independent branching formulations for bivariate piecewise linear functions

Vielma and Nemhauser (2011) originally proposed an independent branching formulation for bivariate functions over a highly structured grid triangulation known as the J1 or Union Jack (Todd 1977). More recently, Huchette and Vielma (2016) propose an independent branching formulation under a strictly weaker structural condition involving the existence of certain graph colorings, as well as a (slightly larger) independent branching formulation for arbitrary grid triangulations.

All three formulation follow the same two-stage construction. The first stage enforces a “subrectangle selection” over the grid. For each axis (e.g. $x_2$), we group the elements of $V$ into axis-parallel sets $\{(u, 1), (u, 2), \ldots, (u, d_2 + 1)\}$ for each $u \in [d_1 + 1]$. We then apply an aggregated SOS2 constraint along the $d_1 + 1$ groups. That is, if $\{(A^k, B^k)\}_{k=1}^{r_1}$ is biclique representation for SOS2 on $d_1 + 1$
breakpoints, then we use \( \{(A^k \times \lfloor d_2 + 1 \rfloor, B^k \times \lfloor d_2 + 1 \rfloor)\}_{k=1}^{\lfloor d_1 \rfloor} \). We analogously construct a biclique representation \( \{(\lfloor d_1 + 1 \rfloor \times \tilde{A}^k, \lfloor d_1 + 1 \rfloor \times \tilde{B}^k)\}_{k=1}^{\lfloor d_1 \rfloor} \) for the aggregated SOS2 constraint aligned with the \( x_1 \) axis from a biclique representation \( \{(\tilde{A}^k, \tilde{B}^k)\}_{k=1}^{\lfloor d_1 \rfloor} \) for the SOS2 on \( d_2 + 1 \) breakpoints. After combining the aggregated SOS2 constraints along both axes, the resulting independent branching formulation (2) will ensure that \( \lambda \) will only have support over a single subrectangle in the grid, i.e. \( \text{supp}(\lambda) \subseteq \{i, i + 1\} \times \{j, j + 1\} \) for some \( i \in \lfloor d_1 \rfloor \) and \( j \in \lfloor d_2 \rfloor \). As SOS2 on \( k + 1 \) breakpoints admits biclique representation with \( \lceil \log(k) \rceil \) levels, the “subrectangle selection” step can be accomplished with a biclique representation with \( \lceil \log_2(d_1) \rceil + \lceil \log_2(d_2) \rceil \) levels.

The second stage constructs the bicliques \( \{(A^{3,k}, B^{3,k})\}_{k=1}^{\lfloor d_3 \rfloor} \) to enforce a “triangle selection” constraint. If we have already imposed that \( \text{supp}(\lambda) \subseteq \{i, i + 1\} \times \{j, j + 1\} \) with the rectangle selection constraint, presume that the triangulation of the subrectangle is oriented such that \( \{(i, j), (i + 1, j + 1)\} \in T \) for each \( T \in \mathcal{T} \). Then we would like that there is some \( k \in \lfloor r_3 \rfloor \) such that \( \{(i, j), (i + 1, j + 1)\} \in A^{3,k} \ast B^{3,k} \overset{\text{def}}{=} \{(u, v) : u \in A^{3,k}, v \in B^{3,k}\} \), and that \( \{(i, j + 1), (i + 1, j)\} \notin A^{3,k} \ast B^{3,k} \) for each \( k \in \lfloor r_3 \rfloor \). Vielma et al. (2010) observe that, when the triangulation is a Union Jack, this triangle selection step can be accomplished with \( r_3 = 1 \) levels. Huchette and Vielma (2016) show that, when a particular coloring exists for a graph corresponding to the triangulation \( \mathcal{T} \), it can be enforced with \( r_3 \leq 2 \). In the same work, the authors present a “stencil” approach that enforces triangle selection for an arbitrary grid triangulation with \( r_3 = 9 \).

In this work, we derive a construction that performs the triangle selection with \( r_3 = 6 \) levels. We defer a formal description to Appendix A.2, and instead illustrate it on the triangulation with \( d_1 = d_2 = 8 \) depicted in Figure 3. Let \( E \subset \lfloor 8 \rfloor^2 \) be the set of diagonal line segments that define the triangulation. In the top row, we show the first 3 levels, which enforce the triangle selection constraint on all subrectangles \( \{i, j\} \times \{j, j + 1\} \) whose orientation is from the southeast corner to the northwest corner (and so \( \{(i + 1, j), (i + 1, j)\} \in E \)). More specifically, for \( k \in \mathbb{Z} \) let the diagonal line offset by \( k \) be given by \( L^k = \{(i, j) \in \lfloor d_1 + 1 \rfloor \times \lfloor d_2 + 1 \rfloor : j = i + k\} \), along with the ordering \( \{u^i\}_{i=1}^{\lfloor L_k \rfloor} = L_k \) of the line (i.e. \( \|u^i - u^{i+1}\|_\infty = 1 \)). We first construct \( (\tilde{A}^k, \tilde{B}^k) \subset L^k \times L^k \) so that
(u^i, u^{i+1}) \in (\bar{A}^k \times \bar{B}^k) \cup (\bar{B}^k \times \bar{A}^k)$ if and only if $(u^i, u^{i+1}) \notin E$ (In Figure 3, the sets $\bar{A}^k$ and $\bar{B}^k$ are represented by blue squares and green diamonds, respectively). Finally, for each $k \in [3]$ we take $A^k$ (resp. $B^k$) as the union over all all $\bar{A}^{k'}$ (resp. $\bar{B}^{k'}$) with $k' \mod 3 \equiv k - 1$. From Figure 3 we can see how the lines in $E$ covered by $(A^k, B^k)$ are sufficiently far apart for it to satisfy the conditions of Proposition 1 (e.g for any $(u, v) \in A^k \times B^k$, $\{u, v\} \subseteq T$ for each $T \in \mathcal{T}$). The second row of Figure 3 shows the last 3 levels that are constructed analogously for segment crossing from the southeast to the northwest. As the complete construction has 6 levels and proceeds by translating a “stencil” diagonally through the grid, we refer to it as the 6-stencil triangle selection formulation.

![Figure 3](image-url)  
**Figure 3** An independent branching representation for triangle selection that uses 6 levels. The first row enforces triangle selection on all subrectangles whose triangulation goes from southwest to northeast; the bottom row enforces it for all subrectangles whose triangulation goes from southeast to northwest.

### 4.2. Combination of formulations

The modular nature of our bivariate independent branching formulations in terms of two (aggregated) SOS2 constraints and a biclique representation for the “triangle selection” hints at the fact that we could potentially replace the independent branching formulations for the two SOS2
constraints with any SOS2 formulation and maintain validity. This means that, for example, we can construct a formulation for bivariate functions over a grid triangulation by applying the ZZI formulation for the aggregated SOS2 constraint along the $x_1$ and the $x_2$ dimension, and the 6-stencil independent branching formulation to enforce triangle selection. However, in general the intersection of ideal formulations will not be ideal, with independent branching formulations being a notable exception. Fortunately, the following proposition (proven in Appendix A.3) shows that this preservation of strength is not restricted to independent branching formulations, but holds for any intersection of ideal formulations of combinatorial disjunctive constraints.

**Proposition 5.** For each $i \in [r]$, let $S_i = \bigcup_{j=1}^{s_i} P(T_i,j)$ for $T_i,j \subseteq V$, where $\bigcup_{j=1}^{s_i} T_i,j = V$. Furthermore, for each $i \in [r]$, let $R_i \subseteq \mathbb{R}^V \times \mathbb{R}^{r_i}$ be such that $(\lambda, y^i) \in R_i \cap (\mathbb{R}^V \times \mathbb{Z}^{r_i})$ is an ideal formulation of $S_i$. Then, an ideal formulation for $\bigcap_{i=1}^r S_i$ is

$$(\lambda, y^i) \in R_i \cap (\mathbb{R}^V \times \mathbb{Z}^{r_i}) \quad \forall i \in [r].$$

(5)

### 4.3. Computational experiments with bivariate piecewise linear functions

To study the computational efficacy of the 6-stencil approach for triangle selection, we perform a computational study on a series of two-commodity transportation problems studied in Section 5.2 of Vielma et al. (2010). The objective functions for these instances are the sum of 25 concave, nondecreasing bivariate piecewise linear functions over grid triangulations with $d_1 = d_2 = N \in \{4, 8, 16, 32\}$. The triangulation of each bivariate function is generated randomly, which is the only difference from (Vielma et al. 2010), where the Union Jack triangulation was used. To handle the generic triangulations, we apply the 6-stencil formulation for triangle selection, coupled with either the Log, ZZB, or ZZI formulation for the SOS2 constraints. We compare these new formulations against the CC, MC, and DLog formulations, which readily generalize to bivariate functions.

In Table 3, we see that the new approaches win on every instance in our test bed. For $N = 16$, we see a speed-up of almost an order of magnitude over the DLog formulation, the best of the existing approaches from the literature. We see that the Log formulation wins a plurality or majority of
Table 3  Computational results for transportation problems whose objective function is the sum of bivariate
piecewise linear objective functions on grids of size $N = d_1 = d_2$.

| $N$ | Metric | MC | CC | DLog | Log | ZZB | ZZI |
|-----|--------|----|----|------|-----|-----|-----|
|     | Mean (s) | 1.4 | 1.5 | 0.9  | 0.4 | 0.4 | 0.4 |
| 4   | Sample std | 1.3 | 1.5 | 0.6  | 0.2 | 0.2 | 0.2 |
|     | Win     | 0   | 0   | 0    | 29  | 31  | 40  |
| 8   | Mean (s) | 39.3| 97.2| 12.6 | 2.7 | 3.0 | 3.0 |
|     | Sample std | 75.0| 179.6| 9.8  | 2.2 | 2.4 | 2.9 |
|     | Win     | 0   | 0   | 0    | 51  | 17  | 32  |
| 16  | Mean (s) | 1370.9| 1648.1| 352.8| 24.6| 26.5| 35.2|
|     | Sample std | 670.4| 360.8| 499.4 | 24.5| 27.4| 40.4|
|     | Win     | 0   | 0   | 0    | 43  | 31  | 6   |
| 32  | Mean (s) | 1800.0| 1800.0| 1499.6| 133.5| 167.6| 246.5|
|     | Sample std | 0.0 | 0.0 | 475.2 | 162.7| 226.7| 306.6|
|     | Win     | 0   | 0   | 0    | 63  | 15  | 2   |

instances for $N \in \{8, 16, 32\}$, and that the ZZI formulation is outperformed by the ZZB formulation
by a non-trivial amount on larger instances.

For comparison, we also perform bivariate computational experiments where $N$ is not a power-of-two, now adding the LogIB formulation as an option for the SOS2 constraints. We use a similar
instance generation scheme as for the univariate non-power-of-two experiments in Section 3.1,
randomly dropping $\log_2(N) - 1$ gridpoints from the interior of the domain along each axis, i.e. from
both $\{t_1^1, t_3^1, \ldots, t_{d_1}^1\}$ and $\{t_2^2, t_3^2, \ldots, t_{d_2}^2\}$. We present the results in Appendix A.4. Qualitatively the
results are quite similar to those in Table 3, although the ZZB and ZZI formulations perform slightly
better on these instances relatively to when $N$ is a power-of-two. There is no significant difference
between the Log and LogIB 6-stencil formulations.

### 4.4. Optimal independent branching schemes

Although the independent branching constructions thus far have an interpretible, two-stage heuristic construction, the combinatorial framework of Huchette and Vielma (2016) admits the notion of “optimal” independent branching formulations. An optimal formulation is obtained from a biclique representation with the fewest possible levels. This value is universally lower bounded by $\lceil \log_2(d) \rceil$ where $d$ is the number of domain pieces, but it may not be attainable. The 6-stencil formulation coupled with two logarithmic representations for the SOS2 constraints requires $\lceil \log_2(d_1) \rceil + \lceil \log_2(d_2) \rceil + 6$ levels, and so is optimal up to a additive constant. However, this constant term can often have
significant impact on the overall computational performance. For example, if we have a grid triangulation with \( d_1 = d_2 = 8 \), there is a lower bound of 7 levels, while the 6-stencil approach gives a formulation with 12 levels, nearly twice the lower bound. Additionally, if the triangulation happens to be the Union Jack, the specialized formulation of Vielma and Nemhauser (2011) attains this lower bound of 7. Therefore, it stands to reason that there might be some remaining performance gains to be made by reducing this constant factor, particularly for smaller bivariate grid triangulations. As an illustration, we study the case where we compute an optimal biclique representation for the triangle selection constraint, and then combine it with the axis-aligned SOS2 approach for the rectangle selection portion. In preliminary experiments, we did not observe significant practical advantage for using an optimal representation for the complete triangulation.

In Table 4, we report computational experiments for this optimal triangle selection approach on our bivariate test problems with grids of size \( N = d_1 = d_2 \). We compare against the Log, ZZB, and ZZI formulations coupled with the 6-stencil. We compute an optimal triangle selection biclique representation using the MIP formulation of (Huchette and Vielma 2016, Proposition 7).

For \( N = 4 \), we observe that the optimal triangle selection formulations win on 65 of 100 instances, with relatively lower solve times, on average, than their 6-stencil counterparts. For the family of larger instances with \( N = 8 \), the optimal triangle selection formulations win on 51 of 100 instances. Interestingly, the optimal triangle selection formulations exhibit slightly higher average solve times than the 6-stencil ones, but with a lower variance in solve time.

The MIP formulation for computing optimal triangle selection representations does not scale for instances with \( N > 8 \), so the evaluation of this approach on larger instances will require new solution.
techniques for the minimum biclique cover problem. However, the subproblems to compute the optimal triangle selection formulations with $N = 4$ solved relatively quickly, on the order of a few seconds. We note that, with $N = 8$, the minimum triangle selection formulation had 3 levels on 54 instances and 4 levels on 46 instances. On those instances where the minimum size representation has 3 levels, the optimal triangle selection formulations had 175 binary variables and 350 general inequality constraints total, while the 6-stencil formulations has 250 binary variables and 500 general inequality constraints.

5. Computational tools for piecewise linear modeling: PiecewiseLinearOpt

Throughout this work, we have investigated a number of possible formulations for optimization problems containing piecewise linear functions. The performance of these formulations can be highly dependent on latent structure in the function and its domain, and there are potentially a number of formulations one may want to try on a given problem instance. However, these formulations can seem quite complex and daunting to a practitioner, especially one unfamiliar with the intricacies and idiosyncrasies of MIP modeling. Anecdotally, we have observed that the complexity of these formulations has driven potential users to simpler but less performant models, or to abandon MIP solutions altogether for other approaches.

This gap between high-performance and accessibility is fundamental throughout optimization. One essential tool to help close the gap is the algebraic modeling language, which allows the user to express an optimization problem in a user-friendly, pseudo-mathematical style, and obviates the need to interact with the underlying optimization solver directly. Because they offer a much more welcoming experience for the modeler, algebraic modeling languages have been widely used for decades, with AMPL (Fourer et al. 1989) and GAMS (Rosenthal 2014) being two particularly storied and successful commercial examples. JuMP (Dunning et al. 2017) is a recently developed open-source algebraic modeling language in the Julia programming language (Bezanson et al. 2017) which offers state-of-the-art performance and advanced functionality, and is readily extensible.

To accompany this work, we have created PiecewiseLinearOpt, a Julia package that extends JuMP to offer all the formulation options discussed herein through a simple, high-level modeling
using JuMP, PiecewiseLinearOpt, CPLEX
model = Model(solver=CplexSolver())
@variable(model, 0 <= x <= 4)
xval = [0,1,2,3,4]
fval = [0,4,7,9,10]
z = piecewiselinear(model, x, xval, fval, method=:Logarithmic)
@objective(model, Min, z)

Figure 4  PiecewiseLinearOpt code to set the univariate function (1) as the objective, using the Log formulation.

interface. The package supports continuous univariate piecewise linear functions, and bivariate piecewise linear functions over grid triangulations.

In Figure 4, we see sample code that sets the univariate piecewise linear function (1) as the objective function for a simple optimization problem. In the first line, we load the required packages: JuMP, PiecewiseLinearOpt, and the CPLEX package for the CPLEX solver. Next, we define the Model object, and add the x variable to it. Here, we express the piecewise linear function in terms of the breakpoints xval of the domain, and the corresponding function values fval at these breakpoints. We call the piecewiselinear function, which adds the Log formulation for the piecewise linear function to the model. It returns a JuMP variable z which is constrained to lie in the graph gr(f) of the function, and can then used anywhere in the model, e.g. in the objective function.

It is often most natural to express a piecewise linear function with a functional form, rather than the breakpoint representation. For instance, this is the case when you would like to use a piecewise linear function to approximate a general nonlinear function. In the sample code in Figure 5, we construct a grid discretization of a nonlinear bivariate function on the box domain [0, 1]². We construct a BivariatePWLFonction object to approximate it, choosing the triangulation such that it best approximates the function values at the midpoint of each subrectangle in the grid. We use the ZZI formulation along each axis x₁ and x₂; it will automatically use the 6-stencil triangle selection portion of the formulation, as the triangulation is unstructured.
using JuMP, PiecewiseLinearOpt, CPLEX

model = Model(solver=CplexSolver())
@variable(model, 0 <= x[1:2] <= 1)

f(u,v) = 2*(u-1/3)^2 + 3*(v-4/7)^4

dx = dy = linspace(0, 1, 9)
pwl = BivariatePWLFunction(dx, dy, f, pattern=:BestFit)

z = piecewiselinear(model, x[1], x[2], pwl, method=:ZigZagInteger)

Figure 5 PiecewiseLinearOpt code to set a bivariate function as the objective. The triangulation is selected to best approximate the function value at the midpoint of the subrectangles. The 6-stencil formulation is automatically selected, using with the ZZI formulation for both axis-aligned SOS2 constraints.

The PiecewiseLinearOpt package supports all the formulations presented in this work, and can handle the construction and formulation of both structured or unstructured grid triangulations. All this complexity is hidden from the user, who can embed piecewise linear functions in their optimization problem in a single line of code with the piecewiselinear function. We hope that this simple computational tool will make the advanced formulations available for modeling piecewise linear functions more broadly accessible to researchers and practitioners.

6. Analysis of the formulations

In the remainder, we analyze structural properties of our new logarithmic formulations. First, we study the branching behavior of the ZZI formulation, which is superior to that of Log and may explain its computational advantage. We also study a connection between the independent branching and the embedding perspectives through the notion of a “redundant embedding.” We see that, for more complex constraints such as the bivariate grid triangulation, redundancy may actually be necessary to construct small formulations through the embedding perspective.

6.1. Branching behavior of ZZI

As observed by Vielma et al. (2010) and in our computational experiments, the original logarithmic formulation LogIB can offer a considerable computational advantage over existing formulations,
particularly for univariate piecewise linear functions with many pieces \( d \). However, it has also been observed that variable branching with the logarithmic formulation can produce weaker dual bounds than other approaches such as the incremental Inc formulation (e.g. Yildiz and Vielma (2013)).

A traditional way to assess the strength of a MIP formulation is to study its LP relaxation, i.e. the formulation with integrality conditions relaxed. Tighter formulations with LP relaxations that closely approximate the convex hull of the set of interest (in this case, \( \text{gr}(f) \)) tend to lead to better computational performance with branch-and-bound-based methods that solve a series of these relaxations (with altered variable bounds and possibly valid inequalities added) as subproblems. All the formulations discussed in this work are sharp, and so their projection onto the original \( px,zq \)-space yields \( \text{Conv}(\text{gr}(f)) \). This is the best we may hope for a convex relaxation of a nonconvex set \( \text{gr}(f) \). However, the relaxations of the various formulations after branching on one of the integer variables \( y \) can be significantly different.

Returning to the SOS2 constraint with \( d = 4 \) from Example 2, The corresponding logarithmic formulation \( \text{Log} \) (also \( \text{LogIB} \)) is all \((\lambda, y) \in \Delta^5 \times \{0, 1\}^2 \) such that

\[
\lambda_3 \leq y_1 \leq \lambda_2 + \lambda_3 + \lambda_4, \quad \lambda_4 + \lambda_5 \leq y_2 \leq \lambda_3 + \lambda_4 + \lambda_5, \tag{6}
\]

while the ZZI formulation is all \((\lambda, y) \in \Delta^5 \times \{0, 1, 2\} \times \{0, 1\} \) such that

\[
\lambda_3 + \lambda_4 + 2\lambda_5 \leq y_1 \leq \lambda_2 + \lambda_3 + 2\lambda_4 + 2\lambda_5, \quad \lambda_4 + \lambda_5 \leq y_2 \leq \lambda_3 + \lambda_4 + \lambda_5. \tag{7}
\]

The corresponding mapping from the \( \lambda \) variables to the original \((x, z)\)-space is \((x, z) = (0, 0)\lambda_1 + (1, 4)\lambda_2 + (2, 7)\lambda_3 + (3, 9)\lambda_4 + (4, 10)\lambda_5 \). As mentioned, both formulations are sharp, and so their LP relaxation projected onto \((x, z)\)-space is \( \text{Conv} (\text{gr}(f)) \); see Figure 6. However, we will see that when \( f \) is a concave function such as (1), \( \text{Log} \) leads to relaxations after branching that are qualitatively and quantitatively worse than the corresponding relaxations after branching of ZZI.

To quantitatively assess relaxation strength after branching, we consider two metrics. The first is the volume of the projection of the LP relaxation onto \((x, z)\)-space. The second is the proportion of the domain where the LP relaxation after branching is stronger than the LP relaxation without branching. More formally, if \( F \) is the projection of the original LP relaxation
onto \((x,z)\)-space, and \(F'\) is the projection of the LP relaxation after branching, then we report
\[
\frac{1}{\text{Vol}(D)} \text{Vol}\left(\{x \in D : \min_{(x,z) \in F} z < \min_{(x,z) \in F'} z\}\right)
\]
which we dub the \textit{strengthened proportion}.

First, we consider the logarithmic formulation \(\text{Log}\). Down-branching on \(y_1\) (i.e. \(y_1 \leq 0\)) implies that \(\lambda_3 = 0\). Up-branching on \(y_1\) (i.e. \(y_1 \geq 1\)) implies that \(\lambda_1 = \lambda_5 = 0\). See in Figure 7 that the down branch produces an LP relaxation that is weak and qualitatively similar to the LP relaxation without branching. The strengthened proportion is 0, and so when minimizing \(f\), the dual bound will be the same after branching as for the original LP relaxation (assuming both are feasible).

For the general integer zig-zag formulation \(\text{ZZI}\), we have two possibilities for branching on \(y_1\), depicted in Figure 7. Branching \(y_1 \leq 0\) implies \(\lambda_3 = \lambda_4 = \lambda_5 = 0\), while the opposite branch \(y_1 \geq 1\) implies \(\lambda_1 \leq \lambda_4 + \lambda_5\). The second branching choice is between \(y_1 \leq 1\), which implies \(\lambda_5 \leq \lambda_1 + \lambda_2\), or \(y_1 \geq 2\), which implies that \(\lambda_1 = \lambda_2 = \lambda_3 = 0\). We note that after branching either \(y_1 \leq 0\) or \(y_1 \geq 2\), the relaxation is then exact, i.e. the relaxation is equal to exactly one of the pieces of the graph of \(f\). Furthermore, when branching either \(y_1 \leq 1\) or \(y_1 \geq 1\), we deduce a general inequality on the \(\lambda\) variables that improves the strengthened proportion relative to \(\text{Log}\).

As we see qualitatively in Figures 6 and 7 and quantitatively in Table 5, the \(\text{ZZI}\) formulation yields LP relaxations after branching that are both stronger and more balanced than the \(\text{Log}\) formulation. In Appendix B, we offer a more complex example with an 8-piece concave piecewise linear function where this effect is even more pronounced.

An instructive way to interpret the branching of \(\text{ZZI}\) is as emulating the “incremental” branching of \(\text{Inc}\). The \(\text{Inc}\) formulation has \(d-1\) binary variables \(y\), and produces very balanced branch-and-bound trees (Yildiz and Vielma 2013, Vielma 2015). Branching down on \(y_k\) yields a projected relaxation of \(\text{Conv}((x,z) \in \text{gr}(f) : t_1 \leq x \leq t_{k+1})\) of all the pieces of the graph to the left of the \((k+1)\)-th breakpoint \(t_{k+1}\); similarly, up-branching giving the convex hull of all pieces to the right of \(t_{k+1}\). Hence, the \(\text{Inc}\) formulation has \textit{hereditary sharpness} (Jeroslow 1988), as it preserves sharpness

| Statistic       | LP Relaxation | Log 0 ↓ | Log 1 ↑ | ZZI 0 ↓ | ZZI 1 ↑ | ZZI 1 ↓ | ZZI 2 ↑ |
|-----------------|---------------|---------|---------|---------|---------|---------|---------|
| Volume          | 6             | 0       | 5.5     | 0       | 3.5     | 3.5     | 0       |
| Strengthened Prop. | 0          | 0       | 1       | 1       | 0.5     | 1       | 0.5     | 1       |

Table 5 Metrics for each possible branching decision on \(z_1\) for \(\text{Log}\) and \(\text{ZZI}\) applied to (1).
after variable branching. The combination of a balanced branch-and-bound tree and hereditary sharpness allows Inc to perform very well for small $d$, before the linear scaling in $d$ takes over (see the computational results in Section 3.1). As we see in Figure 7, the ZZI formulation mimics this incremental branching, but only approximates the hereditary sharpness of Inc. In this sense, we have constructed ZZI to combine the small size of the logarithmic formulations Log and LogIB, with the superior incremental branching behavior of Inc.

![Figure 6](image-url) Projection of the LP relaxation of both (6) and (7) onto the $(x,z)$-space.

![Figure 7](image-url) Feasible region in the $(x,z)$-space for the Log formulation (6) after: down-branching $y_1 \leq 0$ (top left), and up-branching $y_1 \geq 1$ (bottom left); and for the ZZI formulation (7) after: down-branching on $y_1 \leq 0$ (top center), up-branching on $y_1 \geq 1$ (bottom center), down-branching on $y_1 \leq 1$ (top right), and up-branching on $y_1 \geq 2$ (bottom right).

### 6.2. Redundant representations of disjunctive sets

As mentioned previously, the logarithmic independent branching formulation LogIB of Vielma et al. (2010) coincides with the logarithmic embedding formulation Log constructed via Proposition 3
when $d$ is a power-of-two. However, this is not the case otherwise. For a concrete example, take $d = 3$. The Log formulation is

$$\lambda_3 + \lambda_4 \leq y_1 \leq \lambda_2 + \lambda_3 + \lambda_4, \quad \lambda_4 \leq y_2 \leq \lambda_3 + \lambda_4, \quad (\lambda, y) \in \Delta^4 \times \{0, 1\}^2. \quad (8)$$

Contrastingly, the LogIB formulation is

$$\lambda_3 \leq y_1 \leq \lambda_2 + \lambda_3 + \lambda_4, \quad \lambda_4 \leq y_2 \leq \lambda_3 + \lambda_4, \quad (\lambda, y) \in \Delta^4 \times \{0, 1\}^2, \quad (9)$$

after a suitable affine transformation of the variables. The formulations are identical, save an extra $\lambda_4$ term that appears in the first constraint of the Log formulation. Furthermore, take the encoding $H \in \mathcal{H}_2(4)$ where $(H_i)^4_{i=1} \overset{\text{def}}{=} ((0, 0), (1, 0), (1, 1), (0, 1))$. For each row $i \in \llbracket 4 \rrbracket$, there is a corresponding feasible solution $(\lambda, H_i)$ for (9). This is not the case for the embedding formulation, as there does not exist any $\lambda' \in \Delta^4$ such that $(\lambda', H_4)$ is feasible for (8). Furthermore, both formulations are embedding formulations in the following sense. Take $(T^i)_{i=1}^4 = \{i, i+1\}_{i=1}^4$. Then the feasible set with respect to (8) and (9) are $\text{Em}((T^i)^3_{i=1}, H_{1;3})$ and $\text{Em}((T^i)^4_{i=1}, H_{1;4})$, respectively. Furthermore, $Q((T^i)^4_{i=1}, H_{1;4})$ is equal to the LP relaxation of (9). That is, the independent branching formulation (9) is an embedding formulation for $\bigcup_{i=1}^4 P(T^i)$, which is identical to $\bigcup_{i=1}^3 P(T^i)$ as $P(T^4) \subseteq P(T^3)$.

As the following proposition shows, this is true in general: any independent branching formulation is an embedding formulation for a (potentially redundant) representation of the PWL function.

**Proposition 6.** Let $\mathcal{T} = (T^i \subseteq V)_{i=1}^d$ be a combinatorial disjunctive constraint and $\{(A^k, B^k)\}_{k=1}^L$ be a biclique representation of $\mathcal{T}$. If there is no $i \neq j$ such that $T^i \subsetneq T^j$, then there exists an encoding $\overline{H} \in \mathcal{H}_L(2^L)$ and a combinatorial disjunctive constraint $\overline{T} = (\overline{T}^i \subseteq V)_{i=1}^{2L}$ such that

- for all $i \in \llbracket 2^L \rrbracket$, there exists $j \in \llbracket d \rrbracket$ such that $T^i \subseteq T^j$,
- for all $j \in \llbracket d \rrbracket$, there exists $i \in \llbracket 2^L \rrbracket$ such that $T^i = T^j$,
- $(\lambda, y) \in Q(\overline{T}, \overline{H}) \cap (\mathbb{R}^V \times \mathbb{Z}^L)$ is an ideal formulation for $\mathcal{T}$, and
- $(\lambda, y) \in Q(\overline{T}, \overline{H})$ if and only if

$$(\lambda, y) \in \Delta^V \times \{0, 1\}^L, \quad \sum_{v \in A^k} \lambda_v \leq y_k, \quad \sum_{v \in B^k} \lambda_v \leq 1 - y_k \quad \forall k \in \llbracket L \rrbracket.$$  

We refer to $\overline{T}$ as a redundant representation of $\mathcal{T}$. 


6.3. Redundancy can lead to smaller formulations

We have observed that there is a divergence of the embedding formulation $\log$ and the independent branching formulation $\logIB$ for the SOS2 constraint when the number of segments $d$ is not a power-of-two. However, the difference between $\log$ and $\logIB$ seems to be practically inconsequential: it only manifested through certain coefficient changes in the constraints, and the results in Sections 3.1 and Appendix A.4 suggest that there is little computational difference between the two. However, we now show that for more complex constraints, redundant representations can be strictly more powerful than non-redundant representations, in the sense that redundant representations can yield ideal formulations that are strictly smaller than any ideal non-redundant embedding formulation. Furthermore, we see in the following example that this can arise naturally when using the combinatorial independent branching approach.

Take the $d_1 = d_2 = 2$ grid triangulation depicted in Figure 8, given by $T = (T^i)_{i=1}^8$, where

$$T^1 = \{(1,1), (1,2), (2,1)\}, \quad T^2 = \{(1,2), (2,1), (2,2)\}, \quad T^3 = \{(2,1), (3,1), (2,2)\},$$

$$T^4 = \{(3,1), (3,2), (2,2)\}, \quad T^5 = \{(1,2), (2,2), (1,3)\}, \quad T^6 = \{(2,2), (2,3), (1,3)\},$$

$$T^7 = \{(2,2), (2,3), (3,2)\}, \quad T^8 = \{(2,3), (3,2), (3,3)\}.$$

Using the approach of Huchette and Vielma (2016), one can show that the minimum size biclique representation for $T$ has 4 levels, which is strictly greater than the $\log_2(|T|) = 3$ lower bound. One such formulation is

$$\lambda_{(1,1)} + \lambda_{(3,3)} \leq 1 - y_1,$$

$$\lambda_{(1,2)} + \lambda_{(2,1)} \leq 1 - y_2,$$

$$\lambda_{(1,1)} + \lambda_{(2,1)} + \lambda_{(3,1)} \leq 1 - y_3,$$

$$\lambda_{(1,1)} + \lambda_{(1,2)} + \lambda_{(1,3)} \leq 1 - y_4,$$

$$\lambda_{(1,3)} + \lambda_{(2,2)} + \lambda_{(3,1)} \leq y_1,$$

$$\lambda_{(2,3)} + \lambda_{(3,2)} \leq y_2,$$

$$\lambda_{(1,3)} + \lambda_{(2,3)} + \lambda_{(3,3)} \leq y_3,$$

$$\lambda_{(3,1)} + \lambda_{(3,2)} + \lambda_{(3,3)} \leq y_4,$$

$$(\lambda, y) \in \Delta^V \times \{0, 1\}^4.$$
Take $H \in \{0,1\}^{8\times 4}$, where

\[
H_1 = (0,0,0,0), \quad H_2 = (1,0,0,0), \quad H_3 = (1,0,0,1), \quad H_4 = (1,1,0,1),
\]
\[
H_5 = (1,0,1,0), \quad H_6 = (1,1,1,0), \quad H_7 = (1,1,1,1), \quad H_8 = (0,1,1,1),
\]

If $(\lambda, H_i)$ is feasible for (10), then $\lambda \in P(T^i)$. Furthermore, $\text{Proj}_y(\{(\lambda, y) : (10)\}) = \{0,1\}^4$. That is, there are feasible points $(\lambda, y)$ for (10) where $y$ is not a row of $H$. Therefore, it is not an embedding formulation for $\mathcal{T}$ with respect to $H$. However, upon further inspection, we could also take the sets

\[
T^9 = \{(1,1)\}, \quad T^{10} = \{(2,1)\}, \quad T^{11} = \{(2,2)\}, \quad T^{12} = \{(2,2)\},
\]
\[
T^{13} = \{(1,2)\}, \quad T^{14} = \{(2,3)\}, \quad T^{15} = \{(2,3)\}, \quad T^{16} = \{(3,3)\}
\]

along with $\overline{H} \in \{0,1\}^{16\times 4}$, where $\overline{H}_i = H_i$ for $i \in [8]$ and

\[
\overline{H}_9 = (0,1,0,0), \quad \overline{H}_{10} = (0,0,0,1), \quad \overline{H}_{11} = (1,1,0,0), \quad \overline{H}_{12} = (1,0,1,1),
\]
\[
\overline{H}_{13} = (0,0,1,0), \quad \overline{H}_{14} = (0,1,0,0), \quad \overline{H}_{15} = (0,1,1,0), \quad \overline{H}_{16} = (0,0,1,1).
\]

If $(\lambda, \overline{H}_i)$ is feasible for (10) for some $i \in \{9,\ldots,16\}$, then $\lambda \in P(T^i) \subseteq P(T^{i-8})$ (see Figure 8). Furthermore, the LP relaxation of (10) is precisely $Q(\overline{\mathcal{T}}, \overline{H})$, where $\overline{\mathcal{T}} = (T^i)_{i=1}^{16}$. By introducing the artificial redundant sets $(T^i)_{i=9}^{16}$ in this particular way, we can construct an embedding formulation for $\text{Em}(\overline{\mathcal{T}}, \overline{H})$ that, as $\overline{\mathcal{T}}$ is a redundant representation for $\mathcal{T}$, is also a valid formulation for the original constraint $\bigcup_{i=1}^{8} P(T^i)$. This formulation has 4 auxiliary binary variables and 8 facet-defining general inequality constraints. In contrast, if we construct the embedding formulation $\text{Em}(\mathcal{T}, H)$ for the non-redundant representation, we would have an ideal formulation, given by $Q(\mathcal{T}, H)$, with 3 auxiliary binary variables, but 19 general inequality constraints. Moreover, any choice of binary embedding (i.e. assignment of $\{0,1\}^3$ to the sets $\mathcal{T}$) must have at least 9 facet-defining general inequalities, strictly more than required for the redundant representation $\overline{\mathcal{T}}$. That is, by adding an additional binary variable and the redundant sets $(T^i)_{i=9}^{16}$, we have constructed a strictly smaller formulation than would be possible using non-redundant representations.
We close by noting that constructing a formulation through the embedding approach requires us to carefully select the encoding vectors $H$ for each of the sets in $P$. In the case of more complex constraints such as the example above, we may additionally have to consider redundant representations as well. In contrast to the univariate case, we have no principled way to go about this beyond brute force enumeration. In contrast, (10) arises naturally through the combinatorial independent branching approach, and can be constructed directly from the coloring formulation of Huchette and Vielma (2016). Therefore, we offer this, along with the computational experiments in Section 4.3, as evidence that the combinatorial nature of the independent branching framework, while more restrictive than the geometric embedding approach, is still capable of producing small and high-performing formulations for disjunctive constraints, and can be considerably simpler to apply.

![Figure 8](image)

**Figure 8** A K1 grid triangulation with $d_1 = d_2 = 2$. The numbers inside the triangles are the codes $H_i$ for the corresponding triangles $T_i$. The numbers in the cells are the codes $H_i$ corresponding to the redundant sets $T_i$, which are the singleton grid points over which the cells are positioned.

### 7. Conclusion

We present this work as a natural sequel to the work of Vielma et al. (2010), applying modern advanced MIP formulation techniques developed in the intervening 10 years to piecewise linear functions. We close this paper by stating some remaining open questions and research directions.

First, in Section 4.4 we saw that the MIP formulation of Huchette and Vielma (2016) for computing minimum size biclique covers does not scale beyond small problem instances. A specialized algorithm or heuristic for computing these biclique covers would be of significant interest for computing “optimal” formulations for piecewise linear functions, and for a host of other disjunctive
constraints. Second, lower bounds on the minimum size of independent branching representations for both grid triangulation, as well as for the triangle selection constraint, would be of interest, along with simple constructions akin to the 6-stencil that attain this lower bound. Based on our computational observations, we conjecture that any grid triangulation on $V = [d_1 + 1] \times [d_2 + 1]$ admits an independent branching formulation with $\lfloor \log_2(d_1) \rfloor + \lfloor \log_2(d_2) \rfloor + 3$ levels, and that the triangle selection constraint for the same grid can always be enforced with 4 levels. Third, our work here focuses on univariate and bivariate piecewise linear functions. The case where the domain is higher-dimensional is also of interest, and considerably less well-understood. Finally, we only consider bivariate functions over a regular grid triangulation. There may be cases (e.g. in an adaptive grid refinement scheme) where irregular triangulations are desirable. It is worthwhile investigating how the formulation approaches studied in this work would generalize to these other settings.

Acknowledgments

This material is based upon work supported by the National Science Foundation under Grant CMMI-1351619.

References

Balakrishnan, A., S. C. Graves. 1989. A composite algorithm for a concave-cost network flow problem. *Networks* **19** 175–202.

Beale, E. M. L., J. A. Tomlin. 1970. Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. J. Lawrence, ed., *OR 69: Proceedings of the Fifth International Conference on Operational Research*. Tavistock Publications, 447–454.

Bergamini, M. L., P. Aguirre, I. Grossmann. 2005. Logic-based outer approximation for globally optimal synthesis of process networks. *Computers and Chemical Engineering* **29**(9) 1914–1933.

Bergamini, M. L., I. Grossmann, N. Scenna, P. Aguirre. 2008. An improved piecewise outer-approximation algorithm for the global optimization of MINLP models involving concave and bilinear terms. *Computers and Chemical Engineering* **32**(3) 477–493.

Bezanson, J., A. Edelman, S. Karpinski, V. B. Shah. 2017. Julia: A fresh approach to numerical computing. *SIAM Review* **59**(1) 65–98.
Bixby, R., E. Rothberg. 2007. Progress in computational mixed integer programming—A look back from the other side of the tipping point. *Annals of Operations Research* **149** 37–41.

Castro, P. M., J. P. Teles. 2013. Comparison of global optimization algorithms for the design of water-using networks. *Computers and Chemical Engineering* **52** 249–261.

Codas, A., E. Camponogara. 2012. Mixed-integer linear optimization for optimal lift-gas allocation with well-separator routing. *European Journal of Operational Research* **217**(1) 222–231.

Codas, A., S. Campos, E. Camponogara, V. Gunnerud, S. Sunjerga. 2012. Integrated production optimization of oil fields with pressure and routing constraints: The Urucu field. *Computers and Chemical Engineering* **46** 178–189.

Coppersmith, D., J. Lee. 2005. Parsimonious binary-encoding in integer programming. *Discrete Optimization* **2** 190–200.

Croxton, K. L., B. Gendron, T. L. Magnanti. 2003. A comparison of mixed-integer programming models for nonconvex piecewise linear cost minimization problems. *Management Science* **49**(9) 1268–1273.

Croxton, K. L., B. Gendron, T. L. Magnanti. 2007. Variable disaggregation in network flow problems with piecewise linear costs. *Operations Research* **55**(1) 146–157.

D’Ambrosio, C., A. Lodi, S. Martello. 2010. Piecewise linear approximation of functions of two variables in MILP models. *Operations Research Letters* **38**(1) 39–46.

Dantzig, G. B. 1960. On the significance of solving linear programming problems with some integer variables. *Econometrica, Journal of the Econometric Society* 30–44.

de Farias Jr., I. R., E. Kozyreff, R. Gupta, M. Zhao. 2013. Branch-and-cut for separable piecewise linear optimization and intersection with semi-continuous constraints. *Math. Prog. Comp.* **5** 75–112.

de Farias Jr., I., M. Zhao, H. Zhao. 2008. A special ordered set approach for optimizing a discontinuous separable piecewise linear function. *Operations Research Letters* **36**(2) 234–238.

Dunning, I., J. Huchette, M. Lubin. 2017. JuMP: A modeling language for mathematical optimization. *SIAM Review* **59**(2) 295–320.

Fourer, R., D. M. Gay, B. Kernighan. 1989. *AMPL: a mathematical programming language*. Springer-Verlang.
Fügenschuh, A., C. Hayn, D. Michaels. 2014. Mixed-integer linear methods for layout-optimization of screening systems in recovered paper production. *Optimization and Engineering* **15** 533–573.

Geißler, B., A. Martin, A. Morsi, L. Schewe. 2012. *Using piecewise linear functions for solving MINLPs*. Springer, 287–314.

Graf, T., P. V. Hentenryck, C. Pradelles-Lasserre, L. Zimmer. 1990. Simulation of hybrid circuits in constraint logic programming. *Computers and Mathematics with Applications* **20**(9–10) 45–56.

Huchette, J., S. S. Dey, J. P. Vielma. 2017. Strong mixed-integer formulations for the floor layout problem. *To appear in INFOR: Information Systems and Operational Research* doi:10.1080/03155986.2017.1346916.

Huchette, J., J. P. Vielma. 2016. A combinatorial approach for small and strong formulations of disjunctive constraints. *arXiv preprint arXiv:1607.04803*.

Jeroslow, R. G., J. K. Lowe. 1985. Experimental results on the new techniques for integer programming formulations. *The Journal of the Operational Research Society* **36**(5) 393–403.

Jeroslow, R., J. Lowe. 1984. Modelling with integer variables. *Mathematical Programming Study* **22** 167–184.

Jeroslow, R. G. 1988. Alternative formulations of mixed integer programs. *Annals of Operations Research* **12** 241–276.

Jünger, M., T. Liebling, D. Naddef, G. Nemhauser, W. Pulleyblank, G. Reinelt, G. Rinaldi, L. Wolsey. 2010. *50 years of integer programming 1958-2008*. Springer.

Keha, A. B., I. R. de Farias Jr., G. L. Nemhauser. 2004. Models for representing piecewise linear cost functions. *Operations Research Letters* **32**(1) 44–48.

Keha, A. B., I. R. de Farias Jr., G. L. Nemhauser. 2006. A branch-and-cut algorithm without binary variables for nonconvex piecewise linear optimization. *Operations Research* **54**(5) 847–858.

Kolodziej, S., P. M. Castro, I. E. Grossmann. 2013. Global optimization of bilinear programs with a multiparametric disaggregation technique. *Journal of Global Optimization* **57** 1039–1063.

Lee, J., D. Wilson. 2001. Polyhedral methods for piecewise-linear functions I: the lambda method. *Discrete Applied Mathematics* **108** 269–285.

Liu, H., D. Z. Wang. 2015. Global optimization method for network design problem with stochastic user equilibrium. *Transportation Research Part B: Methodological* **72** 20–39.
Magnanti, T. L., D. Stratila. 2004. Separable concave optimization approximately equals piecewise linear optimization. Daniel Bienstock, George Nemhauser, eds., Lecture Notes in Computer Science, vol. 3064. Springer, 234–243.

Mahlke, D., A. Martin, S. Moritz. 2010. A mixed integer approach for time-dependent gas network optimization. Optimization Methods and Software 25(4) 625–644.

Markowitz, H. M., A. S. Manne. 1957. On the solution of discrete programming problems. Econometrica 25(1) 84–110.

Martin, A., M. Möller, S. Moritz. 2006. Mixed integer models for the stationary case of gas network optimization. Math. Program., Ser. B 105 563–582.

Misener, R., C. Floudas. 2012. Global optimization of mixed-integer quadratically-constrained quadratic programs (MIQCP) through piecewise-linear and edge-concave relaxations. Mathematical Programming, Series B 136 155–182.

Misener, R., C. E. Gounaris, C. A. Floudas. 2009. Global optimization of gas lifting operations: A comparative study of piecewise linear formulations. Ind. Eng. Chem. Res. 48(13) 6098–6104.

Misener, R., J. P. Thompson, C. A. Floudas. 2011. APOGEE: Global optimization of standard, generalized, and extended pooling problems via linear and logarithmic partitioning schemes. Computers and Chemical Engineering 35 876–892.

Muldoon, F. 2012. Polyhedral approximations of quadratic semi-assignment problems, disjunctive programs, and base-2 expansions of integer variables. Ph.D. thesis, Clemson University, Clemson, SC.

Muldoon, F. M., W. P. Adams, H. D. Sherali. 2013. Ideal representations of lexicographic orderings and base-2 expansions of integer variables. Operations Research Letters 41 32–39.

Padberg, M. 2000. Approximating separable nonlinear functions via mixed zero-one programs. Operations Research Letters 27 1–5.

Rosenthal, R. 2014. GAMS - A User’s Guide. GAMS Development Corporation.

Savage, C. 1997. A survey of combinatorial Gray codes. SIAM Review 39(4) 605–629.

Sherali, H. D., H. Wang. 2001. Global optimization of nonconvex factorable programming problems. Math. Program., Ser. A 89 459–478.
Silva, T. L., A. Codas, E. Camponogara. 2012. A computational analysis of convex combination models for multidimensional piecewise-linear approximation in oil production optimization. *Proceedings of the 2012 IFAC Workshop on Automatic Control in Offshore Oil and Gas Production*. 292–298.

Silva, T. L., E. Camponogara. 2014. A computational analysis of multidimensional piecewise-linear models with applications to oil production optimization. *European Journal of Operational Research* 232(3) 630–642.

Todd, M. J. 1977. Union Jack triangulations. *Fixed Points: Algorithms and Applications* 315–336.

Tomlin, J. 1981. A suggested extension of special ordered sets to non-separable non-convex programming problems. *North-Holland Mathematics Studies* 59 359–370.

Vielma, J. P. 2015. Mixed integer linear programming formulation techniques. *SIAM Review* 57(1) 3–57.

Vielma, J. P. 2017. Embedding formulations and complexity for unions of polyhedra. *To appear in Management Science* .

Vielma, J. P., S. Ahmed, G. Nemhauser. 2010. Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. *Operations Research* 58(2) 303–315.

Vielma, J. P., G. Nemhauser. 2011. Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. *Mathematical Programming, Series A* 128 49–72.

Wilson, D. L. 1998. Polyhedral methods for piecewise-linear functions. Ph.D. thesis, University of Kentucky, Lexington, Kentucky.

Yildiz, S., J. P. Vielma. 2013. Incremental and encoding formulations for mixed integer programming. *Operations Research Letters* 41 654–658.
Appendix A: Omitted proofs and additional results for Sections 3 and 4

A.1. Binary reflected Gray codes, related encodings, and proof of Proposition 4

For the proof of Proposition 4, we first note that Proposition 3 holds for \( \mathcal{H}^c_r(d) \equiv \{ H \in \mathbb{R}^{d \times r} : H_i \neq H_j \forall i \neq j, \text{ext}(P \{ H_i \}_{i=1}^d) = \{ H_i \}_{i=1}^d \} \) through a direct adaptation of the proof of Proposition 2 in Vielma (2017). Hence, Proposition 3 also holds for \( \mathcal{H}^r_r(d) \). The additional requirement of \( \mathcal{H}^r_r(d) \) over \( \mathcal{H}^c_r(d) \) is just to ensure the validity of the MIP formulation described in Proposition 2. Finally, the following straightforward lemma gives a recursive construction for \( K^r, C^r, \) and \( Z^r \).

**Lemma 1.** \( K^k = C^1 = Z^1 \equiv (0, 1)^T \), and for \( r \in \mathbb{Z}_{++} \) (and \( d = 2^r \)):

\[
K^{r+1} \equiv \begin{pmatrix} K^r \\ \text{rev}(K^r) \end{pmatrix}, \quad C^{r+1} \equiv \begin{pmatrix} C^r \\ C^r + 1^r \otimes C^r \end{pmatrix}, \quad \text{and} \quad Z^{r+1} \equiv \begin{pmatrix} Z^r \\ Z^r \end{pmatrix},
\]

where \( 0^r, 1^r \in \mathbb{R}^r \) are the vectors with all components equal to 0 or 1, respectively, \( u \otimes v = uv^T \in \mathbb{R}^{m \times n} \) for any \( u \in \mathbb{R}^m \) and \( v \in \mathbb{R}^n \), and \( \text{rev}(A) \) reverses the rows of the matrix \( A \).

**Proof of Proposition 4** First, we observe that \( K^r, Z^r \in \{0, 1\}^{d \times r} \) and that \( \mathcal{A} \) is an invertible linear map. Therefore, for each \( r \in \mathbb{Z}_{++}, K^r, C^r, \) and \( Z^r \) are in \( \mathcal{H}^c_r(d) \). The additional requirements for inclusion in \( \mathcal{H}^r_r(d) \) are trivially satisfied for \( K^r \) and \( Z^r \) as they are binary matrices. Inclusion in \( \mathcal{H}^r_r(d) \) is inherited by \( C^r \) because \( \mathcal{A} \) is invertible and linear, and both \( \mathcal{A} \) and \( \mathcal{A}^{-1} \) are unimodular (\( \mathcal{A}(w) \in \mathbb{Z}^r \) if and only if \( w \in \mathbb{Z}^r \)).

Now the result is direct from Proposition 3, as \( \{ e^i = \alpha^{i+1} - \alpha^i \}_{i=1}^{d-1} = \{ e^i \}_{i=1}^{r} \), where \( e^i \) is the canonical unit vector with support on component \( i \), and that the inverse of \( \mathcal{A} \) is \( \mathcal{A}^{-1}(y)_i = y_i + \sum_{k=i+1}^r 2^{k-2} y_i \) for each \( i \in [r] \). Formulations (3) and (4) correspond to encodings \( C^r \) and \( Z^r \), respectively. \( \Box \)

A.2. Triangle selection with 6 levels

We can give a precise description of the 6-level triangle-selection step illustrated in Section 4.1 through the formal procedure introduced in Section 8.4.1 of Huchette and Vielma (2016). For this let \( G = (V, \bar{E}) \) be the graph given by \( V \equiv \{ [d_1 + 1] \times [d_2 + 1] \} \) and \( \bar{E} \equiv \{ \{ u, v \} \in [V]^2 : \{ u, v \} \subseteq T \} \) where \( [V]^2 \equiv \{ \{ u, v \} : u, v \in V, u \neq v \} \). Edges \( \bar{E} \) are the complement of \( E \) mentioned in Section 4.1 so that \( \bar{E} \) corresponds precisely to elements of \( [V]^2 \) that do not have an associated line segment in a drawing of the triangulation.

We obtain the 6-stencil triangle selection formulation by a direct adaptation of Theorem 4 in Huchette and Vielma (2016) by constructing \( \{(A^{3,k}, B^{3,k})\}_{k=1}^6 \) so that each \( (A^{3,k}, B^{3,k}) \) induces a complete bipartite graph (biclique) on \( G \), and and that combined edges cover the “triangle-selection” edges in \( \bar{E}^T = \{ \{ u, v \} \in \bar{E} : ||u - v||_2 = 1 \} \). That is, we show that \( \bar{E}^T \subseteq \bigcup_{k=1}^6 (A^{3,k} \ast B^{3,k}) \subseteq \bar{E} \).
To construct this biclique cover, consider all “diagonal lines” in the grid, offset by some $i \in \mathbb{Z}$:

\[
DL(i) \overset{\text{def}}{=} \{(u, v) \in [d_1 + 1] \times [d_2 + 1] : u + i = v\},
\]

along with all “anti-diagonal lines” offset by $i \in \mathbb{Z}$:

\[
ADL(i) \overset{\text{def}}{=} \{(u, v) \in [d_1 + 1] \times [d_2 + 1] : u + i = d_1 + 2 - v\}.
\]

Take $\tilde{E}^{DL}(i) = \{(u, v) \in \tilde{E}^T : u, v \in DL(i)\}$ and $\tilde{E}^{ADL}(i) = \{(u, v) \in \tilde{E}^T : u, v \in ADL(i)\}$ as all edges in $\tilde{E}^T$ where both ends lie on the diagonal line $DL(i)$ or the anti-diagonal line $ADL(i)$, respectively. Then $\tilde{E}^T = \bigcup_{i \in \mathbb{Z}}(\tilde{E}^{DL}(i) \cup \tilde{E}^{ADL}(i))$.

As the $DL(i)$ are all lines, it is straightforward to see that there exists a simple linear ordering on their elements. Therefore, a corresponding ordering exist on $ADJ^{DL}(i) \overset{\text{def}}{=} \bigcup\{(u, v) \in \tilde{E}^{DL}(i)\} \subseteq DL(i)$ of the form $\{u^1\}_{i=1}^{\text{adj}^{DL}(i)}$. We take $(\bar{A}^{DL(i)}, \bar{B}^{DL(i)})$ such that $\bar{A}^{DL(i)} \cup \bar{B}^{DL(i)} = ADJ^{DL}(i)$, and such that adjacent elements $u^k, u^{k+1} \in ADJ^{DL}(i)$ are separated by the biclique (i.e. $\{u^k, u^{k+1}\} \in \bar{A}^{DL(i)} \ast \bar{B}^{DL(i)})$ if and only if the pair is in the edge set $\{(u^k, u^{k+1}) \in \tilde{E}^T\}$.

Applying an identical argument to the anti-diagonal edges, we can produce $(\bar{A}^{ADL(i)}, \bar{B}^{ADL(i)})$ such that

\[
\tilde{E}^T \subseteq \bigcup_{i \in \mathbb{Z}}(\bar{A}^{ADL(i)} \ast \bar{B}^{ADL(i)}) \subset \tilde{E}.
\]

It just remains to show that we can aggregate these sets together into 6 groups, while maintaining this inclusion property. For this note that for any $i, j \in \mathbb{Z}$ with $|i - j| \geq 2$, we have that $||u - v||_\infty \geq 2$ for each $u \in DL(i)$ and $v \in DL(j)$. Furthermore, $\{u, v\} \in \tilde{E}$ for any $u, v \in V$ such that $||u - v||_\infty \geq 2$. Then for any $u \in \bar{A}^{DL(i)}$ and $v \in \bar{B}^{DL(j)}$, we have $\{u, v\} \in \tilde{E}$. The same property holds for the anti-diagonal lines, so if we define

\[
A^{DL,\alpha} = \bigcup_{i \in (3Z+\alpha)} \bar{A}^{DL(i)}, \quad B^{DL,\alpha} = \bigcup_{i \in (3Z+\alpha)} \bar{B}^{DL(i)}
\]

\[
A^{ADL,\alpha} = \bigcup_{i \in (3Z+\alpha)} \bar{A}^{ADL(i)}, \quad B^{ADL,\alpha} = \bigcup_{i \in (3Z+\alpha)} \bar{B}^{ADL(i)}
\]

for each $\alpha \in \{0, 1, 2\}$ we have

\[
\tilde{E}^T \subseteq \bigcup_{\alpha \in \{0, 1, 2\}} ((A^{DL,\alpha} \ast B^{DL,\alpha}) \cup (A^{ADL,\alpha} \ast B^{ADL,\alpha})) \subset \tilde{E}.
\]

**A.3. Proof of Proposition 5**

For simplicity, assume w.l.o.g. that $V = [n]$. Let $R = \{ (\lambda, y^i, \ldots, y^r) \in \mathbb{R}^{n + \sum_i r_i} : (\lambda, y^i) \in R^i \forall i \in [r] \}$ be the LP relaxation of (5). Because the original formulations are ideal (and in particular, sharp), we have

\[
\text{Proj}_\lambda(R) = \bigcap_{i=1}^r \text{Proj}_\lambda(R^i) = \bigcap_{i=1}^r \text{Conv}(S^i) \subseteq \Delta^n = \text{Conv} \left( \bigcap_{i=1}^r S^i \right),
\]
and hence (5) is sharp, as \( \text{Proj}_{\lambda}(R) = \Delta^n \).

To show (5) is also ideal, consider any point \((\hat{\lambda}, \hat{y}^1, \ldots, \hat{y}^r) \in R\). First, we show that if this point is extreme, then \( \hat{\lambda} = e^v \) for some \( v \in [n] \). Consider some point where \( \hat{\lambda} \) is fractional; w.l.o.g., presume that \( 0 < \hat{\lambda}_1, \hat{\lambda}_2 < 1 \). Define \( \lambda^+ \overset{\text{def}}{=} \lambda + \epsilon \text{e}^1 - \epsilon \text{e}^2 \) and \( \lambda^- \overset{\text{def}}{=} \lambda - \epsilon \text{e}^1 + \epsilon \text{e}^2 \) for sufficiently small \( \epsilon > 0 \); clearly \( \hat{\lambda} = \frac{1}{2} \lambda^+ + \frac{1}{2} \lambda^- \). We would like to construct points \( y^{i,+} \) and \( y^{i,-} \) for each \( i \in \{r\} \) such that \( \hat{y}^i = \frac{1}{2} y^{i,+} + \frac{1}{2} y^{i,-} \), and such that \((\lambda^+, y^{i,+}), (\lambda^-, y^{i,-}) \in R^i \). Then \((\hat{\lambda}, \hat{y}^1, \ldots, \hat{y}^r) = \frac{1}{2}(\lambda^+, \hat{y}^{1,+}, \ldots, \hat{y}^{r,+}) + \frac{1}{2}(\lambda^-, \hat{y}^{1,-}, \ldots, \hat{y}^{r,-}) \) is the convex combination of two other feasible points for \( R \), and so is not extreme.

For a given \( i \in \{r\} \), define \( E^i = \{ (k, h) : (e^k, h) \in \text{ext}(R^i) \} \), which is equivalent to the set of all extreme points of \( R^i \). As \((\hat{\lambda}, \hat{y}^i) \in R^i \), there must exist some \( \gamma^i \in \Delta E^i \) where \((\hat{\lambda}, \hat{y}^i) = \sum_{(k, y) \in E^i} \gamma^i_{(k, y)} (e^k, h) \). As \( 1, 2 \in \text{supp}(\hat{\lambda}) \), there must exist some \( \hat{h}^i \) and \( \hat{h}^i \) wherein \((1, \hat{h}^i), (2, \hat{h}^i) \in E^i \) and \( 0 < \gamma^i_{(1, \hat{h}^i)}, \gamma^i_{(2, \hat{h}^i)} < 1 \). Now define

\[
\gamma^i_{(k, h)} = \begin{cases} 
\gamma^i_{(k, h)} \pm \epsilon & k = 1, h = \hat{h}^i \\
\gamma^i_{(k, h)} + \epsilon & k = 2, h = \hat{h}^i \\
\gamma^i_{(k, h)} & \text{o.w.}
\end{cases}
\]

Note that, as \( \gamma^i \in \Delta E^i \), so is \( \gamma^i_{(k, h)} \in \Delta E^i \). Therefore, we may take

\[
y^{i,+} \overset{\text{def}}{=} \sum_{(k, h) \in E^i} \gamma^i_{(k, h)} h = \epsilon \hat{h}^i - \epsilon \hat{h}^i + \sum_{(k, h) \in E^i} \gamma^i_{(k, h)} h \\
y^{i,-} \overset{\text{def}}{=} \sum_{(k, h) \in E^i} \gamma^i_{(k, h)} h = -\epsilon \hat{h}^i + \epsilon \hat{h}^i + \sum_{(k, h) \in E^i} \gamma^i_{(k, h)} h.
\]

Then we may observe that \( y^{i,+}, y^{i,-} \in R^i \), and that \( \hat{y}^i = \frac{1}{2} y^{i,+} + \frac{1}{2} y^{i,-} \). Now see that

\[
\lambda^\pm = \sum_{(k, h) \in E^i} \gamma^i_{(k, h)} e^k = \sum_{(k, h) \in E^i} \gamma^i_{(k, h)} e^k + \epsilon e^1 \pm \epsilon e^2 = \hat{\lambda} \pm \epsilon e^1 \pm \epsilon e^2
\]

Therefore, for each \( i \in \{r\} \), we have that \((\lambda^+, y^{i,+}), (\lambda^-, y^{i,-}) \in R^i \), and that \((\hat{\lambda}, \hat{y}^i) = \frac{1}{2}(\lambda^+, y^{i,+}) + \frac{1}{2}(\lambda^-, y^{i,-}) \). This implies that \((\lambda^+ + h^{i,+}, \ldots, h^{r,+}), (\lambda^+, h^{1,+}, \ldots, h^{r,-}) \in R \) and that \((\hat{\lambda}, \hat{y}^1, \ldots, \hat{y}^r) = \frac{1}{2}(\lambda^+, h^{1,+}, \ldots, h^{r,+}) + \frac{1}{2}(\lambda^-, h^{1,-}, \ldots, h^{r,-}) \). Therefore, as our original point is a convex combination of two distinct points also feasible for \( R \), it cannot be extreme. Therefore, we must have that \( \lambda = e^v \) for some \( v \in [n] \) for any extreme point of \( R \).

Now, assume for contradiction that \( R \) has a fractional extreme point. Using property of extreme points just stated, we may assume without loss of generality that this fractional extreme point is of the form \((e^1, \tilde{y}^1, \ldots, \tilde{y}^r) \) with \( \tilde{y}^i \notin \mathbb{Z}^r \). As \((e^1, \tilde{y}^1) \in R^1 \), then \((e^1, \tilde{y}^1) = \sum_{(v, h) \in E^1} \gamma_{(v, h)} (e^v, h) \) for some \( \gamma \in \Delta E^1 \). Also, as \( R^3 \) is ideal and \( \tilde{y}^1 \) is fractional, \((e^1, \tilde{y}^1) \notin \text{ext}(\text{Conv}(R^3)) \), and so \( \gamma \) must have at least two non-zero components. But then

\[
(\hat{\lambda}, \hat{y}^1, \hat{y}^2, \ldots, \hat{y}^r) = \sum_{(v, h) \in E^1} \gamma_{(v, h)} (e^1, h, \hat{y}^2, \ldots, \hat{y}^r),
\]

a contradiction of the points extremality. Therefore, \( R \) is ideal.
Table 6  Computational results for transportation problems whose objective function is the sum of bivariate piecewise linear objective functions on grids of size $N = d_1 = d_2$, when $N$ is not a power-of-two.

### A.4. Non-power-of-two bivariate computational results

See Table 6.

**Appendix B: 8-segment piecewise linear function formulation branching**

Consider the univariate piecewise linear function $f : [0, 8] \to \mathbb{R}$ given by

$$f(x) = \begin{cases} 
8x & 0 \leq x \leq 1 \\
7x + 1 & 1 \leq x \leq 2 \\
6x + 3 & 2 \leq x \leq 3 \\
5x + 6 & 3 \leq x \leq 4 \\
4x + 10 & 4 \leq x \leq 5 \\
3x + 15 & 5 \leq x \leq 6 \\
2x + 21 & 6 \leq x \leq 7 \\
x + 28 & 7 \leq x \leq 8.
\end{cases} \quad (11)$$

The corresponding LogIB/Log formulation is

$$\begin{align*}
\lambda_2 + 2\lambda_3 + 3\lambda_4 + 4\lambda_5 + 5\lambda_6 + 6\lambda_7 + 7\lambda_8 + 8\lambda_9 &= x \\
8\lambda_2 + 15\lambda_3 + 21\lambda_4 + 26\lambda_5 + 30\lambda_6 + 33\lambda_7 + 35\lambda_8 + 36\lambda_9 &= z \\
\lambda_3 + \lambda_7 &\leq y_1 \leq \lambda_2 + \lambda_3 + \lambda_4 + \lambda_6 + \lambda_7 \\
\lambda_4 + \lambda_5 + \lambda_6 &\leq y_2 \leq \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 + \lambda_7 \\
\lambda_6 + \lambda_7 + \lambda_8 + \lambda_9 &\leq y_3 \leq \lambda_5 + \lambda_6 + \lambda_7 + \lambda_8 + \lambda_9 \\
(\lambda, y) &\in \Delta^9 \times \{0, 1\}^3, 
\end{align*} \quad (12)$$

and the corresponding ZZI formulation is

$$\begin{align*}
\lambda_2 + 2\lambda_3 + 3\lambda_4 + 4\lambda_5 + 5\lambda_6 + 6\lambda_7 + 7\lambda_8 + 8\lambda_9 &= x \\
8\lambda_2 + 15\lambda_3 + 21\lambda_4 + 26\lambda_5 + 30\lambda_6 + 33\lambda_7 + 35\lambda_8 + 36\lambda_9 &= z
\end{align*} \quad (13)$$
In Table 7, we show statistics for the relaxations of both. We observe that the ZZI formulation yields more balanced branching, with the volume and strengthened proportion more equal between the resulting two branches.

\[
\begin{align*}
\lambda_3 + \lambda_4 + 2\lambda_5 + 2\lambda_6 + 3\lambda_7 + 3\lambda_8 + 4\lambda_9 & \leq y_1 \leq \lambda_2 + \lambda_3 + 2\lambda_4 + 2\lambda_5 + 3\lambda_6 + 3\lambda_7 + 4\lambda_8 + 4\lambda_9 \quad (13c) \\
\lambda_4 + \lambda_5 + \lambda_6 + \lambda_7 + 2\lambda_8 + 2\lambda_9 & \leq y_2 \leq \lambda_3 + \lambda_4 + \lambda_5 + \lambda_6 + 2\lambda_7 + 2\lambda_8 + 2\lambda_9 \quad (13d) \\
\lambda_5 + \lambda_7 + \lambda_8 + \lambda_9 & \leq y_3 \leq \lambda_5 + \lambda_6 + \lambda_7 + \lambda_8 + \lambda_9 \quad (13e) \\
(\lambda, y) & \in \Delta^9 \times \{0, 1, 2, 3, 4\} \times \{0, 1, 2\} \times \{0, 1\} \quad (13f)
\end{align*}
\]

In Table 7, we show statistics for the relaxations of both. We observe that the ZZI formulation yields more balanced branching, with the volume and strengthened proportion more equal between the resulting two branches.

| Statistic | Log 0 ↓ | Log 1 ↑ | ZZI 0 ↓ | ZZI 1 ↑ | ZZI 1 ↓ | ZZI 2 ↑ | ZZI 2 ↓ | ZZI 3 ↑ | ZZI 3 ↓ | ZZI 4 ↑ |
|-----------|---------|---------|---------|---------|---------|---------|---------|---------|---------|---------|
| Volume    | 41      | 17      | 0       | 38.5    | 11.5    | 27      | 27      | 11.5    | 38.5    | 0       |
| Strengthened Prop. | 0 1 | 1 0.25 | 0.75 0.5 | 0.5 0.75 | 0.25 1 |

Table 7 Metrics for each possible branching decision on $z_1$ for Log and ZZI applied to (11).
Appendix C: Proof of Proposition 6

Let $\overline{\mathcal{H}} \in \mathcal{H}_{L}(2^{L})$, $A_{1}^{k} = A^{k}$, and $A_{2}^{k} = B^{k}$ for each $k \in [L]$. For each $i \in [2^{L}]$, let $h_{i} \overset{\text{def}}{=} \overline{\mathcal{H}}_{i}$ and $\mathcal{T}_{i} \overset{\text{def}}{=} \bigcap_{k=1}^{L} \left(V \setminus A_{h_{i}^{k}}^{k}\right)$, and define $\mathcal{T} \overset{\text{def}}{=} \left\{ \mathcal{T}_{i} \right\}_{i=1}^{2^{L}}$. Then

$$(\lambda, y) \in \Delta^{V} \times [0,1]^{L}, \quad \sum_{v \in A^{k}} \lambda \leq y_{k}, \quad \sum_{v \in B^{k}} \lambda_{v} \leq 1 - y_{k} \quad \forall k \in [L]$$ (14)

is a formulation of $\mathcal{T}$ and $\overline{\mathcal{T}}$, and hence $\bigcup_{i=1}^{d} P(T_{i}) = \bigcup_{i=1}^{d} P(\overline{T}_{i})$. Then

1. for all $i \in [2^{L}]$, there exists $j \in [d]$ such that $\mathcal{T}_{i} \subseteq T_{j}$, and

2. for all $j \in [d]$, there exists $i \in [2^{L}]$ such that $T_{j} \subseteq \overline{T}_{i}$,

as $P(T) \cup P(S)$ is non-convex for any $T, S \subset V$ such that $S \nsubseteq T$ and $T \nsubseteq S$. The containment on the second point cannot always be strict because of the non-containment assumption on $\mathcal{T}$. The fact that (14) is equivalent to $(\lambda, y) \in Q(\overline{\mathcal{T}}, \overline{\mathcal{H}})$ follows by noting that $(\lambda, y) \in \mathbb{R}^{V} \times \{0,1\}^{L}$ satisfies (14) if and only if $(\lambda, y) \in \text{Em}(\overline{\mathcal{T}}, \overline{\mathcal{H}})$, and that the polytope defined by (14) has extreme points with $y \in \{0,1\}^{L}$. 