Piecewise-Linear Approximations of Multidimensional Functions

R. Misener • C. A. Floudas

Abstract We develop explicit, piecewise-linear formulations of functions $f(x) : \mathbb{R}^n \mapsto \mathbb{R}$, $n \leq 3$, that are defined on an orthogonal grid of vertex points. If mixed-integer linear optimization problems (MILPs) involving multidimensional piecewise-linear functions can be easily and efficiently solved to global optimality, then non-analytic functions can be used as an objective or constraint function for large optimization problems. Linear interpolation between fixed gridpoints can also be used to approximate generic, nonlinear functions, allowing us to approximately solve problems using mixed-integer linear optimization methods. Towards this end, we develop two different explicit formulations of piecewise-linear functions and discuss the consequences of integrating the formulations into an optimization problem.

Keywords Approximate optimization • Linear interpolation • Simplices • EPA Complex Emissions Model

1 Introduction

1.1 Optimizing Piecewise-Linear Functions

The field of global optimization has advanced significantly during the last two decades from theoretical, algorithmic, and application viewpoints (*e.g.*, [1] – [11]). As an example application domain, consider the advanced oil recovery technique of optimally allocating compressed natural gas, called *lift gas*, into a large, interdependent set of oil wells. Kosmidis et al. [12, 13] optimized the profitability of a petroleum field with lift gas, integrating factors such as pressure drop across tubing, line merging, multiphase flow, and reservoir pressure into their optimization model. Because they used hydraulic lookup tables to relate gas injection ($Q_{GAS,i}$) to oil production ($Q_{OIL,i}$), the objective and constraints in their model are non-smooth, piecewise-affine functions defined by linear interpolation between vertex points. Piecewise-

The authors thankfully acknowledge support from the National Science Foundation through the first author's graduate research fellowship and the second author's grant (CBET – 0827907).

R. Misener • C. A. Floudas
Department of Chemical Engineering, Princeton University, Princeton, New Jersey
e-mail: floudas@titan.princeton.edu

linear functions in one dimension have been used to formulate the gas lift problem since the work of Buitrago et al. [14].

In a recent comparative study of formulations for the gas lift problem, Misener et al. [15] addressed the gas lift problem using four different representations of piecewise-affine functions in one dimension proposed by Nemhauser and Wolsey [16], Floudas [17], Sherali [18], and Keha et al. [19]. Each of the the four algorithms was sufficient to solve the mixed integer linear program (MILP) to global optimality. However, the tests we reported in Misener et al. [15] revealed that the special structure method from Keha et al. [19] consistently out-performed the three other algorithms. Based on these results, we recommended that industrially-relevant piecewise-linear optimization problems be solved using the Keha et al. [19] formulation.

This paper develops explicit, piecewise-linear formulations of functions $f(x) : \mathbb{R}^n \mapsto \mathbb{R}$, $n \leq 3$, that are defined on an orthogonal grid of vertex points. If MILPs involving multidimensional piecewise-linear functions can be easily and efficiently solved to global optimality, then non-analytic functions (*e.g.*, the pointwise-defined functions in the gas lifting problem) can be used as an objective or constraint function for large optimization problems. Towards this end, we develop two different explicit formulations of piecewise-linear functions and discuss the consequences of integrating the formulations into an optimization problem.

Linear interpolation between fixed gridpoints can also be used to approximate generic, nonlinear functions, allowing us to approximately solve problems using linear, rather than nonlinear, programming techniques. The potential of this method is twofold. First, in cases where efficient solution time is of paramount importance, a local search near the optimal point of the approximation will yield a good feasible point of the original nonlinear problem. Second, the solution of the piecewise-linear problem can be used as a *warm start* for a global optimization algorithm by generating a good initial upper bound.

In this paper, we begin in Section 1.2 by discussing previous applications of piecewise-linear functions to optimization problems. Section 2 introduces the approximation algorithm. Section 3 discusses interpolation within a simplex. Section 4 formulates explicit, piecewise-linear formulations for two and three dimensions which confine a point to a simplex. Section 5 explicitly presents the equations used in the approximation algorithm. Section 6 provides illustrative examples on a set of functions and analyzes the associated error. Finally, Section 7 concludes the paper.

1.2 Literature Review

Williams [20] used linear interpolation to convert separable nonlinear programs (NLPs) into piecewise-defined linear programs (MILPs). Kosmidis et al. [13] constructed a two-dimensional piecewise linear function using a hydraulic lookup table in their study of gas lifting and well scheduling for enhanced oil recovery. Zhang and Wang [21] solved an approximation of a nonlinear objective function with linear constraints using a series of linear programs. Magnani and Boyd [22] developed an NLP that can be used to fit a convex piecewise-linear function to a given set of data.

In addition to approximate methods, other groups have studied piecewise underestimation of nonlinear functions to expedite the global solution of large-scale problems. Rosen and Pardalos [23] and Pardalos and Rosen [24] addressed large-scale concave programming problems using piecewise linearization techniques. Meyer and Floudas [25] and Karuppiah and Grossmann [26] took advantage of the special structure of bilinear terms to partition the domain and construct piecewise-linear underestimators that strengthened the lower bound on the generalized pooling and integrated water systems problems, respectively. Based on their success, Wicaksono and Karimi [27] and Gounaris et al. [28] thoroughly studied piecewise-linear relaxations of bilinear functions and suggested formulations that could improve the computational times of Meyer and Floudas [25] and Karuppiah and Grossmann [26]. Recognizing that solution times are sometimes more important than certificates of optimality, Pham et al. [29] designed a piecewise bilinear programming algorithm that quickly obtains a good feasible point for large-scale pooling problems.

For more generic functions, Mangasarian et al. [30] discussed a succession of piecewise-linear underestimators converging to the global minimum of an NLP, a technique similar to the algorithm designed by Gounaris and Floudas [31, 32], which converges on the convex envelope of a function through a piecewise combination of convex and linear functions.

Dividing a domain into non-overlapping simplices has been previously discussed by Chien and Kuh [33] in the context of linearly interpolating nonlinear electrical networks. Simplex division also played a key role in the development of convex envelopes for trilinear terms and edge concave functions [34, 35, 36]. The technique we use in this study, linearly interpolating vertex points within non-overlapping simplices, generates an easily-solved approximation of the original problem.

2 Introduction to the Approximation Algorithm

2.1 Lookup Tables

Given a continuous function $\Omega(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$, an approximation function $\hat{\Omega}(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$ can be constructed using a lookup table and an interpolation algorithm. For the purposes of this study, a lookup table consists of function values $\Omega(\mathbf{x}) \in \mathbb{R}$ and associated domain points $\mathbf{x} \in \mathbb{R}^n$ that are recorded at orthogonal gridpoints.

Function interpolation between the lookup table gridpoints can be performed using a variety of algorithms, but this paper will study linear interpolation through a convex combination of the gridpoints Chien and Kuh [33]. As will be shown in Section 3, the linear interpolation function is uniquely defined only if each point in the domain is restricted to a single simplex. Therefore, the approximation function will interpolate function values within a tessellation of simplices.

2.2 Justification of Lookup Table Dimensions

Because the domain space of function $\Omega(\mathbf{x})$ is partitioned into orthogonal gridpoints and then tessellated with a pattern of simplices, one of the sub-problems associated with this study is division of a hypercube into simplices. Hughes and Anderson [37] summarized the minimum number of simplices needed to triangulate an n-dimension hypercube and developed results for dimensions six and seven. Dimensions one, two, and three can be triangulated with as few as one, two, and five simplicies, respectively, but four, five, six, and seven-dimensional hypercubes require 16, 67, 308, and 1493 simplices [37].

Noting the large number of simplices needed to partition hypercubes of dimension greater than three, this study restricts lookup tables to no more than three dimensions. In other words, the algorithm developed in this study uses lookup tables to construct an approximation $\hat{\Omega}(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$ of function $\Omega(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$ when $n \leq 3$.

2.3 Functional Form

Using the lookup tables introduced in Section 2.1, the interpolation algorithm that will be described in Section 3, and the explicit piecewise linearization for two and three dimensions presented in Section 4, functions $\Omega(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$ of dimension three or lower can be approximated as an affine equation $\hat{\Omega}(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$. Higher dimension functions consisting of a summation that can be separated into terms of three or fewer terms can be approximated by constructing a number of lookup tables. Functions $\Psi(\mathbf{x}) : \mathbb{R}^n \mapsto \mathbb{R}$ with $n \leq 6$ that cannot be separated into low-order terms can be written as a summation of bilinear terms:

$$\Psi(x_1, \ldots, x_6) = \sum_i \Omega_i^1(x_1, x_2, x_3) \cdot \Omega_i^2(x_4, x_5, x_6) \approx \sum_i \hat{\Omega}_i^1(x_1, x_2, x_3) \cdot \hat{\Omega}_i^2(x_4, x_5, x_6). \quad (1)$$

Equation (1) can be relaxed using the convex envelope developed by McCormick [38] and Al-Khayyal and Falk [39]. The resulting bilinear envelope can be tightened using one of the piecewise approaches of Wicaksono and Karimi [27] and Gounaris et al. [28].

Similarly, a function $\Phi(\mathbf{x}) : \mathbb{R}^9 \mapsto \mathbb{R}$ with $n \leq 9$ can be separated into a summation of trilinear terms and relaxed using the convex envelopes determined by Meyer and Floudas [34, 35] or the looser relaxation of Maranas and Floudas [40] and Ryoo and Sahinidis [41]. Although the convex envelope represents the tightest possible relaxation, determining the convex envelopes of trilinear functions requires *a priori* permutation of the variables [34, 35]. The relaxation of Maranas and Floudas [40] and Ryoo and Sahinidis [41], which recursively applies bilinear underestimators, permits tight relaxations through piecewise partitioning of the variable domains.

This study introduces piecewise linear approximations and exploits lookup tables of dimension one, two, and three to construct approximation functions that can be written as a summation of linear, bilinear, and trilinear terms. Note that the algorithm's generic nature permits approximation of arbitrary functions which can be written as a summation of nonlinear terms with up to nine dimensions each.

3 Interpolation within a Simplex

After a one-, two-, or three-dimensional domain X is partitioned into an orthogonal grid that spans the domain, any point $x \in X$ can be written as a convex combination of the gridpoints. But the convex combination of the gridpoints is not necessarily unique. Carathéodory [42] showed that every element of compact, convex set $X \subset \mathbb{R}^n$ can be written as a convex combination of at most $n + 1$ points of X. If the domain of X is convex, then any point in the domain space can be written as a convex combination of two points (when $X \subset \mathbb{R}$), three points (when $X \subset \mathbb{R}^2$), or four points (when $X \subset \mathbb{R}^3$).

Although at most $n + 1$ gridpoints are needed to express each point $x \in X \subset \mathbb{R}^n$, there are many more than $n + 1$ gridpoints in any reasonable representation of the domain space. Because there are many gridpoints, unique interpolation of function values using a convex combination of gridpoints is unlikely. To guarantee a deterministic interpolation outcome, only $n + 1$ gridpoints are activated at one time. Section 4 describes appropriately activating $n + 1$ gridpoints for each point in the domain space.

Assuming that the $n + 1$ appropriate gridpoints $x_0, \ldots, x_n \in X$ for domain point $x \in X \subset \mathbb{R}^n$ are activated and that we wish to approximate function $f(x) : X \mapsto \mathbb{R}$, consider the system of equations:

$$
\begin{aligned}
&\hat{f}(x) = w_0 \cdot f(x_0) + \ldots + w_n \cdot f(x_n),\\
&x = w_0 \cdot x_0 + \ldots + w_n \cdot x_n,\\
&\sum_{i=0}^{n} w_i = 1,\\
&w_i \geq 0, \quad \forall\, i = 0, \ldots, n.
\end{aligned}
\tag{2}
$$

In the above linear system of equations (2), there are $n + 2$ unknowns (the $n + 1$ convex combination weights w_i and the value of the approximation function $\hat{f}(x)$) and $n+2$ equations (the interpolation equation for function $\hat{f}(x)$, the n-dimensional equation for x, and the summation of the convex combination weights). This system is uniquely determined when x is in the interior of $n + 1$ gridpoints, allowing us to interpolate the function $f(x) : X \mapsto \mathbb{R}$ between the $n + 1$ appropriate gridpoints.

4 Restriction to a Simplex: Explicit Formulations for Two and Three Dimensions

To uniquely represent each point in the domain as a convex combination of gridpoints, we follow Zhang and Wang [21] in partitioning the domain space into small boxes (rectangles and rectangular prisms in two and three dimensions, respectively) and partitioning each of the boxes into non-overlapping simplices. Section 4.1 describes the set of constraints that restricts each point in domain space to a small box. Sections 4.2 – 4.5 introduce the equations that uniquely confine each point to a single simplex and Section 4.6 describes the interpolation between the simplex vertices. The algorithm described in Sections 4.1 – 4.6 generalizes the one-dimensional piecewise-linear approximation from Floudas [17] and Nemhauser and Wolsey [16] to two and three dimensions.

4.1 Box Constraints

After variable set X is partitioned into the orthogonal grid, any point $x \in X$ is within a line segment (when $X \subset \mathbb{R}$), a rectangle (when $X \subset \mathbb{R}^2$), or a rectangular prism (when $X \subset \mathbb{R}^3$) defined by the gridpoints. The equations introduced in this section activate only the gridpoints at the vertices of the small box that contains x. Figure 1 diagrams an activated line, rectangle, and rectangular prism within the domain space for dimensions one, two and three, respectively. All other vertices are deactivated.

If the variable is one dimensional, the domain set is X partitioned into $X_i \in \mathbb{R}$, $i = 0, \ldots, N_1$, where

N_1 represents the number of segments. To activate a single line segment, a set of variables, $\lambda_i^1 \in \{0, 1\}$ s.t. $i = 1, \ldots, N_1$, is introduced and declared as a Special Ordered Set of type 1 ($SOS1$). Special ordered sets, proposed by Beale and Tomlin [43] and initially implemented by Forrest et al. [44], are sets with at most one nonzero component. We use the $SOS1$ concept because advanced mixed-integer linear programming solvers such as CPLEX [45] efficiently exploit special ordered sets [15]:

$$\sum_{i=1}^{N_1} \lambda_i^1 = 1, \quad \lambda_i^1 \in [0,\, 1], \quad \lambda_i^1 \; SOS1 \quad \forall\, i = 1, ..., N_1. \tag{3}$$

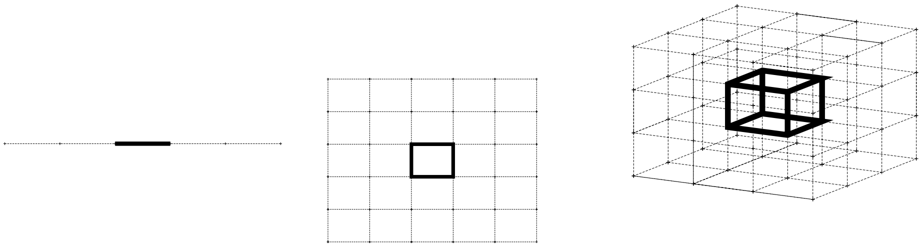


Figure 1: A single active line, square, and box in the domain space for dimensions 1-3

Only the vertices of the single active line segment are allowed to contribute to the interpolation, so continuous variables $w_i \in [0,\, 1]$ s.t. $i = 0, \ldots, N_1$, which act as convex combination weights, are constrained by the activated line segment [16, 17]:

$$\begin{aligned} & w_0 \leq \lambda_1^1, \\ & w_i \leq \lambda_i^1 + \lambda_{i+1}^1, \quad \forall\, i = 1, ..., N_1 - 1, \\ & w_{N_1} \leq \lambda_{N_1}^1. \end{aligned} \tag{4}$$

When the domain X has two dimensions, it is partitioned into $X_{i,j} \in \mathbb{R}^2 \;\; i = 0, \ldots, N_1, \;\; j = 0, \ldots, N_2$. Two sets of variables, $\lambda^1 \in [0,\, 1]$ s.t. $i = 1, \ldots, N_1$ and $\lambda^2 \in [0,\, 1]$ s.t. $j = 1, \ldots, N_2$, activate each rectangle. As in the one dimension case, these variables are $SOS1$:

$$\sum_{i=1}^{N_1} \lambda_i^1 \;=\; 1, \quad \lambda_i^1 \in [0,\, 1], \quad \lambda_i^1 \; SOS1 \quad \forall\, i = 1, ..., N_1, \tag{5}$$

$$\sum_{j=1}^{N_2} \lambda_j^2 \;=\; 1, \quad \lambda_j^2 \in [0,\, 1], \quad \lambda_j^2 \; SOS1 \quad \forall\, j = 1, ..., N_2. \tag{6}$$

Only the vertices of a single activated rectangle contribute to the interpolation within that rectangle, so the convex combination of continuous weights $w_{i,j} \in [0,\, 1]$ s.t. $i = 0, \ldots, N_1,\;\; j = 0, \ldots, N_2$ are constrained as follows:

$$\text{Variable 1}\begin{cases} \sum\limits_{j=0}^{N_2} w_{0,j} & \leq \lambda_1^1, \\ \sum\limits_{j=0}^{N_2} w_{i,j} & \leq \lambda_i^1 + \lambda_{i+1}^1, \quad \forall\, i = 1, ..., N_1 - 1, \\ \sum\limits_{j=0}^{N_2} w_{N_1,j} & \leq \lambda_{N_1}^1, \end{cases} \tag{7}$$

$$\text{Variable 2}\begin{cases} \sum\limits_{i=0}^{N_1} w_{i,0} & \leq \lambda_1^2, \\ \sum\limits_{i=0}^{N_1} w_{i,j} & \leq \lambda_j^2 + \lambda_{j+1}^2, \quad \forall\, j = 1, ..., N_2 - 1, \\ \sum\limits_{i=0}^{N_1} w_{i,N_2} & \leq \lambda_{N_2}^2. \end{cases} \tag{8}$$

Finally, when the domain X has three dimensions, it is partitioned into $X_{i,j,k} \in \mathbb{R}^3$ s.t. $i = 0, \ldots, N_1, j = 0, \ldots, N_2, k = 0, \ldots, N_3$. Three sets of variables: $\lambda^1 \in [0,\, 1]$ s.t. $i = 1, \ldots, N_1$, $\lambda^2 \in [0,\, 1]$ s.t. $j = 1, \ldots, N_2$ and $\lambda^3 \in [0,\, 1]$ s.t. $k = 1, \ldots, N_3$, denote the active rectangular prism. As in the other two cases, these variable sets are $SOS1$:

$$\sum_{i=1}^{N_1} \lambda_i^1 \quad = \quad 1, \quad \lambda_i^1 \in [0,\, 1], \quad \lambda_i^1 \; SOS1 \quad \forall\, i = 1, ..., N_1, \tag{9}$$

$$\sum_{j=1}^{N_2} \lambda_j^2 \quad = \quad 1, \quad \lambda_j^2 \in [0,\, 1], \quad \lambda_j^2 \; SOS1 \quad \forall\, j = 1, ..., N_2, \tag{10}$$

$$\sum_{k=1}^{N_3} \lambda_k^3 \quad = \quad 1, \quad \lambda_k^3 \in [0,\, 1], \quad \lambda_k^3 \; SOS1 \quad \forall\, k = 1, ..., N_3. \tag{11}$$

Only the vertices of the activated rectangular prism contribute to the interpolation of points within that prism, so convex combination weights $w_{i,j,k} \in [0,\, 1]$ s.t. $i = 0, \ldots, N_1$, $j = 0, \ldots, N_2$, $k = 0, \ldots, N_3$ are constrained as follows:

$$\text{Variable 1}\begin{cases} \sum\limits_{j=0}^{N_2} \sum\limits_{k=0}^{N_3} w_{0,j,k} & \leq \lambda_1^1, \\ \sum\limits_{j=0}^{N_2} \sum\limits_{k=0}^{N_3} w_{i,j,k} & \leq \lambda_i^1 + \lambda_{i+1}^1, \quad \forall\, i = 1, ..., N_1 - 1, \\ \sum\limits_{j=0}^{N_2} \sum\limits_{k=0}^{N_3} w_{N_1,j,k} & \leq \lambda_{N_1}^1, \end{cases} \tag{12}$$

$$\text{Variable 2}\begin{cases}\sum_{i=0}^{N_1}\sum_{k=0}^{N_3} w_{i,0,k} & \leq \lambda_1^2, \\ \sum_{i=0}^{N_1}\sum_{k=0}^{N_3} w_{i,j,k} & \leq \lambda_j^2 + \lambda_{j+1}^2, \quad \forall\, j = 1, ..., N_2 - 1, \\ \sum_{i=0}^{N_1}\sum_{k=0}^{N_3} w_{i,N_2,k} & \leq \lambda_{N_2}^2, \end{cases} \tag{13}$$

$$\text{Variable 3}\begin{cases}\sum_{i=0}^{N_1}\sum_{j=0}^{N_2} w_{i,j,0} & \leq \lambda_1^3, \\ \sum_{i=0}^{N_1}\sum_{j=0}^{N_2} w_{i,j,k} & \leq \lambda_k^3 + \lambda_{k+1}^3, \quad \forall\, k = 1, ..., N_3 - 1, \\ \sum_{i=0}^{N_1}\sum_{j=0}^{N_2} w_{i,j,N_3} & \leq \lambda_{N_3}^3. \end{cases} \tag{14}$$

The constraints in this section restrict each point in the domain to a line segment (defined by 2 points), rectangle (defined by 4 points), or rectangular prism (defined by 8 points). But, as described in Section 3, convex combinations of points in the interior of the small two and three dimensional shapes will not be unique. To achieve a unique interpolation, we partition the rectangles and rectangular prisms into non-overlapping simplices. Sections 4.2 through 4.5 divide the shapes into simplices and, for each point in the domain, activate only the vertices of the appropriate simplex.

4.2 Triangulation Classes

In two dimensions, $X \subset \mathbb{R}^2$, there is one representative triangulation class with two distinct orientations that divide a rectangle into non-overlapping simplices. The two possible triangulation orientations of a rectangle are shown in Figure 2.

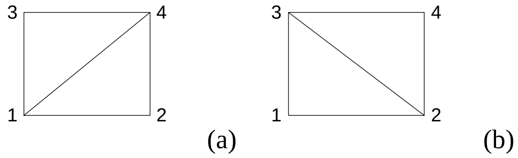
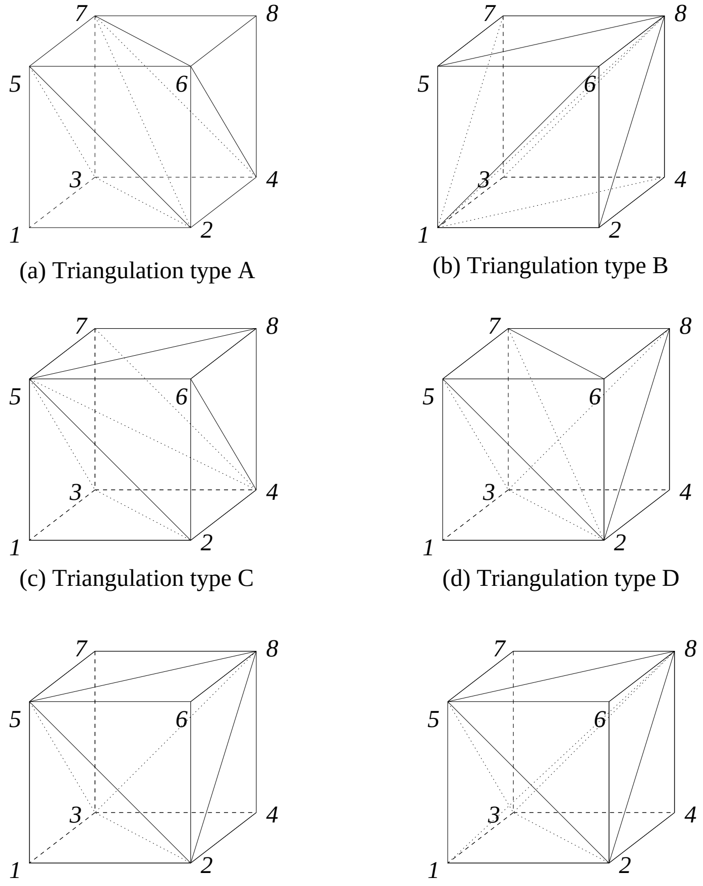


Figure 2: The two possible divisions into nonoverlapping simplices for a 2-cube

The three dimensional case, $X \subset \mathbb{R}^3$, has six representative classes that divide the rectangular prism into non-overlapping simplices [36]. The six standard representatives are diagrammed in Figure 3. Each triangulation type has multiple orientations.

To partition the domain space X, we choose a particular triangulation class and triangulation orientation. Section 4.3 justifies choosing triangulation type B (see Figure 3) as the representative triangulation. After choosing a triangulation type, the specific triangulation orientation for each variable set can be selected to reduce interpolation error. The triangulation type and orientation is tessellated across the entire domain, as shown in Figure 4 [33].



(a) Triangulation type A

(b) Triangulation type B

(c) Triangulation type C

(d) Triangulation type D

(e) Triangulation type E

(f) Triangulation type F

Figure 3: Triangulation types of the 3-cube [36]

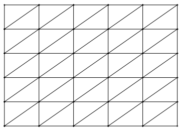


Figure 4: Chosen triangulation type and orientation tessellated over domain space (two dimensions)

4.3 Justification of Triangulation Type B

There are two major advantages to using triangulation B, shown in Figure 3(b), as the representative triangulation for three dimensional domains ($X \subset \mathbb{R}^3$). First, only the three planes illustrated in Figure 5 need to be considered to isolate a point in a small rectangular prism into a particular simplex. Second, each of the six simplices in triangulation type B have equal volume in the case of uniform partitioning, increasing the accuracy of the interpolation. The three planes which partition the rectangular prism into simplices, Y_{1368}, Y_{1458} and Y_{1278}, are defined by the numbers of their vertex points.

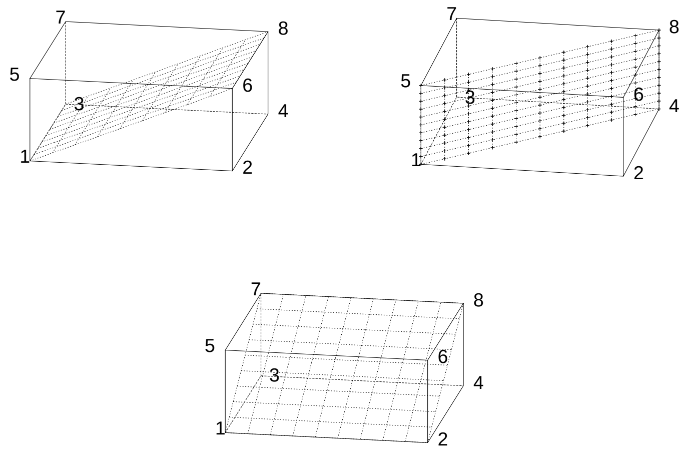


Figure 5: Three planes define triangulation: Y_{1368}, Y_{1458} *&* Y_{1278}

Figure 6 diagrams each of the six simplices after the three planes are used to divide the prism. Because the six simplices are of equal volume in the case of uniform partitioning, the relative error from the approximation will be relatively uniform across the rectangular prism.

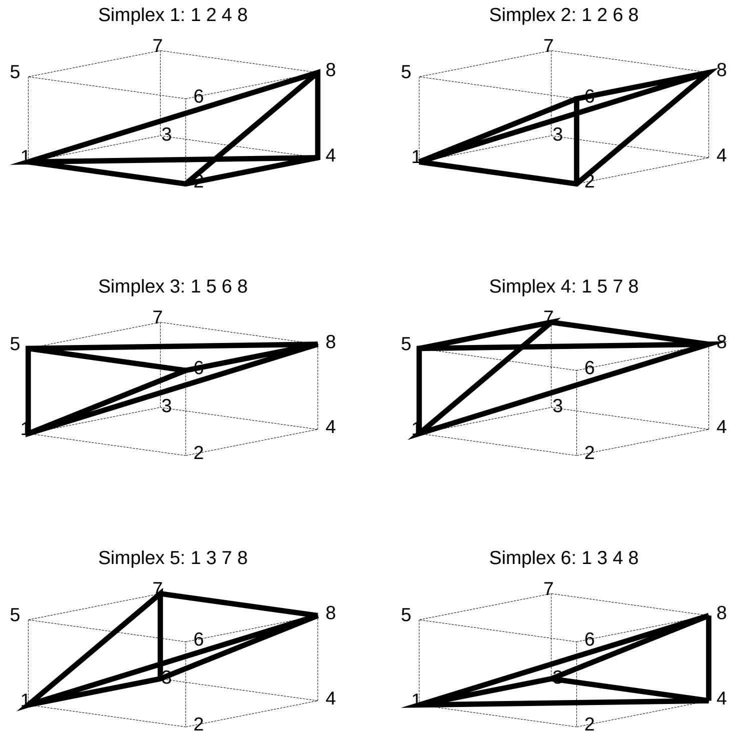


Figure 6: Six simplices result from triangulation type B using 3 planes

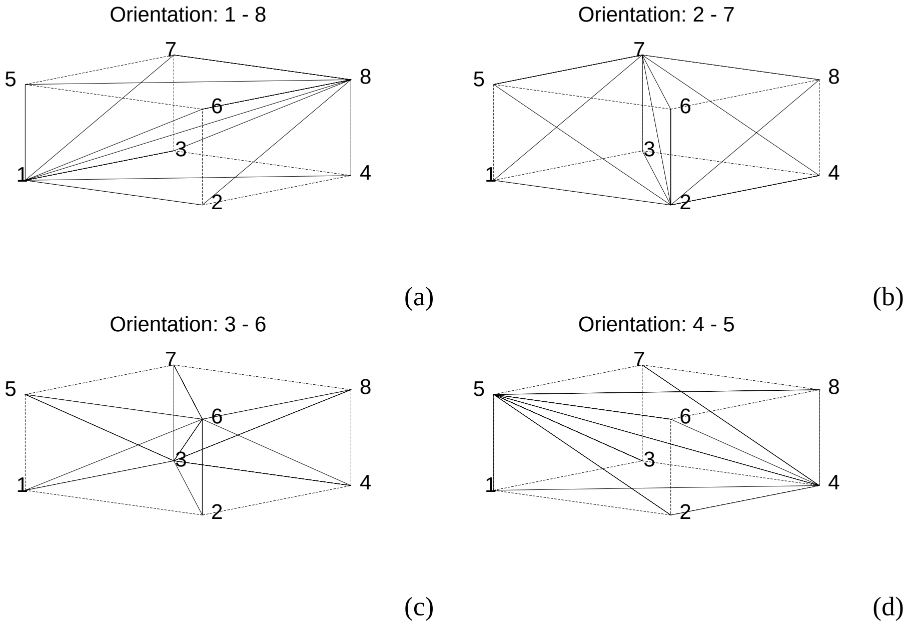


Figure 7: The four orientations of triangulation type B are defined by their primary diagonal

Figure 7(a) illustrates the triangulation orientation diagrammed in Figure 6, while Figures 7(b) through 7(d) delineate the three alternative orientations for triangulation type B. Each of the four type B triangulation orientations is defined by one of the four rectangular prism diagonals: 1 – 8, 2 – 7, 3 – 6, or 4 – 5. Analogous to the division by planes scheme in Figure 5, the three other triangulation orientations each contain three planes which share a common diagonal. For clarity, Sections 4.4 and 4.5 will introduce vertex activation using the triangulation orientation along diagonal 1 – 8 that is illustrated in Figure 7(a), but different triangulation orientations can be considered to reduce the error of approximating a given function.

4.4 Isolating a Simplex in a Single Box

Given an isolated box of two / three dimensions, this section introduces equations that partition the rectangle and rectangular prism into two and six simplices, respectively, and, for a given point in the box, activate only the relevant simplex vertices. In Section 4.5, the results of this section are generalized to the case of a box situated inside of a domain of gridpoints.

4.4.1 Isolating a Simplex in a Rectangle

For rectangle A of two dimensions:

$$(a,\ b) \in A \subset \mathbb{R}^2 \text{ such that } a \in \left[a^L,\, a^U\right] \text{ and } b \in \left[b^L,\, b^U\right], \tag{15}$$

each simplex is defined by its relation to one of the lines shown in Figure 2 of Section 4.2. Assume for the purpose of illustration that the rectangle is divided using the orientation shown in Figure 2(a). Defining $\Delta a = a^U - a^L$ and $\Delta b = b^U - b^L$ to be the two side lengths of the rectangle, the equation of the diagonal line in Figure 2(a) is:

$$\left(a - a^U\right) \cdot \Delta b - \left(b - b^U\right) \cdot \Delta a = 0. \tag{16}$$

To activate the vertices of the appropriate simplex within the rectangle, define binary variable $Y_{14} \in \{0, 1\}$ that is activated on one side of the line and deactivated on the other. Figure 8 delineates the region where Y_{14} is activated. In the following two inequalities, the binary variable representing the line is explicitly determined according to the scheme shown in Figure 8,

$$-\Delta a \cdot \Delta b \cdot (1 - Y_{14}) \leq \left(a - a^U\right) \cdot \Delta b - \left(b - b^U\right) \cdot \Delta a \leq \Delta a \cdot \Delta b \cdot Y_{14}. \tag{17}$$

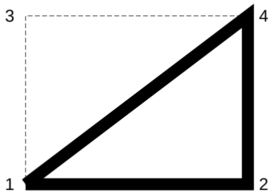


Figure 8: Region of activated binary variable: $Y_{14} = 1$

All four of the rectangle vertices diagrammed in Figure 8 are activated, so the remaining task is to set the convex combination weight of either vertex 3 (w_{a^L,b^U}) to zero when $Y_{14} = 1$ or vertex 2 (w_{a^U,b^L}) to zero when $Y_{14} = 0$:

$$w_{a^L,b^U} \leq 1 - Y_{14},\ w_{a^U,b^L} \leq Y_{14}, \tag{18}$$

so that, given any point $[a,\ b] \in A$, the point can be expressed as a unique convex combination of the three activated vertices.

4.4.2 Isolating a Simplex in a Rectangular Prism

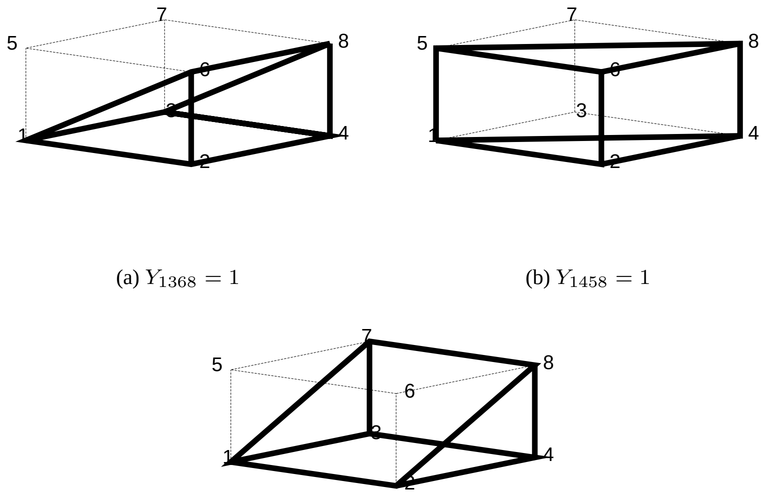
Consider point $[a,\ b,\ c]$ in rectangular prism A of three dimensions:

$$(a,\ b,\ c) \in A \subset \mathbb{R}^3 \text{ such that } a \in [a^L,\ a^U],\ b \in [b^L,\ b^U]\ and\ c \in [c^L,\ c^U], \tag{19}$$

where each simplex in A is defined by its relation to the three planes shown in Figure 5 of Section 4.3. Defining $\Delta a = a^U - a^L$, $\Delta b = b^U - b^L$ and $\Delta c = c^U - c^L$ to be the three orthogonal lengths of the rectangular prism, the equations of the planes in Figure 5 are

$$\begin{aligned} \text{Plane } Y_{1368} \quad &\rightarrow \quad (a - a^U) \cdot \Delta c - (c - c^U) \cdot \Delta a = 0, && (20) \\ \text{Plane } Y_{1458} \quad &\rightarrow \quad (a - a^U) \cdot \Delta b - (b - b^U) \cdot \Delta a = 0, && (21) \\ \text{Plane } Y_{1278} \quad &\rightarrow \quad (b - b^U) \cdot \Delta c - (c - c^U) \cdot \Delta b = 0. && (22) \end{aligned}$$

To activate the vertices of the appropriate simplex, define three binary variables, $Y_{1368} \in \{0, 1\}$, $Y_{1458} \in \{0, 1\}$ and $Y_{1278} \in \{0, 1\}$ that are each activated on one side of the plane and deactivated on the other. Figure 9 highlights the regions where Y_{1368}, Y_{1458} and Y_{1278} are activated. The following set of three sets of inequalities, Equations (23) to (25), explicitly determine the binary variables representing the planes corresponding to the scheme shown in Figure 9.



(a) $Y_{1368} = 1$

(b) $Y_{1458} = 1$

(c) $Y_{1278} = 1$

Figure 9: Regions of activated binary variable: $Y_{1368}, Y_{1458}, Y_{1278}$

$$\begin{aligned} -\Delta a \cdot \Delta c \cdot (1 - Y_{1368}) \quad &\le (a - a^U) \cdot \Delta c - (c - c^U) \cdot \Delta a \quad \le \Delta a \cdot \Delta c \cdot Y_{1368}, && (23) \\ -\Delta a \cdot \Delta b \cdot (1 - Y_{1458}) \quad &\le (a - a^U) \cdot \Delta b - (b - b^U) \cdot \Delta a \quad \le \Delta a \cdot \Delta b \cdot Y_{1458}, && (24) \end{aligned}$$

$$-\Delta b \cdot \Delta c \cdot (1 - Y_{1278}) \leq (b - b^U) \cdot \Delta c - (c - c^U) \cdot \Delta b \leq \Delta b \cdot \Delta c \cdot Y_{1278}. \tag{25}$$

Now, the equations in Section 4.1 restrict the active gridpoints to eight box vertices, so the remaining task is to allow only the relevant simplex vertices to remain active. For example, if $Y_{1278} = 1$, then the convex combination weights of vertices 5 and 6 are set to zero. Table 1 lists the vertices that are deactivated for each value of the binary variables. Notice that, in every case, the vertices deactivated by each binary variable lie on a single edge.

Table 1: Deactivated vertices according to the 3 binary variables representing the planes in Figure 9

	Value	Deactivated Vertices	
$Y_{1368} =$	0	$2 \rightarrow (a^U, b^L, c^L)$	$4 \rightarrow (a^U, b^U, c^L)$
	1	$5 \rightarrow (a^L, b^L, c^U)$	$7 \rightarrow (a^L, b^U, c^U)$
$Y_{1458} =$	0	$2 \rightarrow (a^U, b^L, c^L)$	$6 \rightarrow (a^U, b^L, c^U)$
	1	$3 \rightarrow (a^L, b^U, c^L)$	$7 \rightarrow (a^L, b^U, c^U)$
$Y_{1278} =$	0	$3 \rightarrow (a^L, b^U, c^L)$	$4 \rightarrow (a^U, b^U, c^L)$
	1	$5 \rightarrow (a^L, b^L, c^U)$	$6 \rightarrow (a^U, b^L, c^U)$

$$\begin{aligned}
&\left(w_{a^U,b^L,c^L} + w_{a^U,b^U,c^L}\right) \leq Y_{1368},\\
&\left(w_{a^L,b^L,c^U} + w_{a^L,b^U,c^U}\right) \leq 1 - Y_{1368},\\
&\left(w_{a^U,b^L,c^L} + w_{a^U,b^L,c^U}\right) \leq Y_{1458},\\
&\left(w_{a^L,b^U,c^L} + w_{a^L,b^U,c^U}\right) \leq 1 - Y_{1458},\\
&\left(w_{a^L,b^U,c^L} + w_{a^U,b^U,c^L}\right) \leq Y_{1278},\\
&\left(w_{a^L,b^L,c^U} + w_{a^U,b^L,c^U}\right) \leq 1 - Y_{1278}.
\end{aligned} \tag{26}$$

4.5 Activating a Single Simplex in a Domain of Gridpoints

Sections 4.5.1 and 4.5.2 use the equations introduced in Section 4.4 and re-cast them in an entire domain of points. Notice that this method requires constraints of order N^2 for both domains of two and three dimensions. The constraints are of order N^2 for two dimensional domains because the equations consider each of the gridpoints ($N_1 \times N_2$) and turn off exactly one of the vertices. The method is order N^2 for three dimension domains because the constraint equations select three lines to switch off (there are approximately $N_1 \times N_2 + N_1 \times N_3 + N_2 \times N_3$ lines). Additionally, the method in Sections 4.5.1 and 4.5.2 requires equal partitioning of each variable. Because Δx_1, Δx_2, and Δx_3 are constants, there is no room to use finer partitioning in a portion of a variable's domain.

Section 4.5.3 and 4.5.4 introduce a method for activating a simplex in a domain of gridpoints that is based on work by Kosmidis et al. [13], who use triangulation orientation 1 – 4 to tessellate a function with a two dimensional domain. Section 4.5.4 generalizes their results to a three dimension domain. In this method, the constraints are of order N for a domain of two dimensions and of order N^2 for a domain of three dimensions.

Because the method described in Section 4.5.3 and 4.5.4 does not require the definition of constants like Δx_1, it can be used to irregularly partition the variable domains. Regions of higher curvature can be partitioned with more gridpoints, while nearly-linear regions can be coarsely partitioned.

4.5.1 2D Domain with 1 Binary Variable

When the domain has two dimensions ($x \in X \subset \mathbb{R}^2$), define vectors $e_1 = [1,\ 0]^{\mathrm{T}}$ and $e_2 = [0,\ 1]^{\mathrm{T}}$ which select the first and second component respectively from variable x (*i.e*, the first element of x is $x \cdot e_1$). Then, assuming even grid spacing, the distances between the gridpoints (represented as Δa and Δb in Section 4.4.1) are:

$$\begin{aligned} \Delta x_1 &= (x_{i+1,1} - x_{i,1}) \cdot e_1, \quad \forall\, i = 0, \ldots, N_1, && (27) \\ \Delta x_2 &= (x_{1,j+1} - x_{1,j}) \cdot e_2, \quad \forall\, j = 0, \ldots, N_2. && (28) \end{aligned}$$

Notice that both the second index of x in Equation (27) and the first index of x in Equation (28) are irrelevant. Since the grid spacing of the first index is independent of the grid spacing of the second index, the index 1 is used arbitrarily. The distances between the gridpoints could be equivalently defined using other placeholder indices.

Equation (16) of the diagonal line connecting vertices 1 and 3 of the rectangle becomes:

$$\left(x - \sum_{i=1}^{N_1} x_{i,1} \cdot \lambda_i^1 \right) \cdot e_1 \cdot \Delta x_2 - \left(x - \sum_{j=1}^{N_2} x_{1,j} \cdot \lambda_j^2 \right) \cdot e_2 \cdot \Delta x_1 = 0. \qquad (29)$$

The $SOS1$ variables λ_i and λ_j are only activated for a single $i \in 0, \ldots, N_1$ and $j \in 0, \ldots, N_2$, so Equation (29) describes exactly one line for a given x.

To activate the vertices of the appropriate simplex, define binary variable $Y_{14} \in \{0, 1\}$ that is activated as in Section 4.4.1:

$$-\Delta x_1 \cdot \Delta x_2 \cdot (1 - Y_{14}) \leq \left(x - \sum_{i=1}^{N_1} x_{i,1} \cdot \lambda_i^1 \right) \cdot e_1 \cdot \Delta x_2 - \left(x - \sum_{j=1}^{N_2} x_{1,j} \cdot \lambda_j^2 \right) \cdot e_2 \cdot \Delta x_1 \leq \Delta x_1 \cdot \Delta x_2 \cdot Y_{14}. \tag{30}$$

Finally, the convex combination weight of either vertex 3 (when $Y_{14} = 1$) or vertex 2 (when $Y_{14} = 0$) is set to zero:

$$w_{i-1,j} \leq 3 - Y_{14} - \lambda_i^1 - \lambda_j^2, \quad \forall\, i = 1, \ldots, N_1,\ j = 1, \ldots, N_2, \tag{31}$$

$$w_{i,j-1} \leq 2 + Y_{14} - \lambda_i^1 - \lambda_j^2, \quad \forall\, i = 1, \ldots, N_1,\ j = 1, \ldots, N_2. \tag{32}$$

The variables λ_i and λ_j ensure that only one vertex within the entire domain is deactivated.

4.5.2 3D Domain with 3 Binary Variables

When the domain has three dimensions ($x \in X \subset \mathbb{R}^3$), define vectors $e_1 = [1,\ 0,\ 0]^{\mathrm{T}}$, $e_2 = [0,\ 1,\ 0]^{\mathrm{T}}$ and $e_3 = [0,\ 0,\ 1]^{\mathrm{T}}$ to isolate each individual component of the vector x. Then, assuming even grid spacing, the distances between the gridpoints are:

$$\Delta x_1 = (x_{i+1,1,1} - x_{i,1,1}) \cdot e_1, \quad \forall\, i = 0, \ldots, N_1, \tag{33}$$

$$\Delta x_2 = (x_{1,j+1,1} - x_{1,j,1}) \cdot e_2, \quad \forall\, j = 0, \ldots, N_2, \tag{34}$$

$$\Delta x_3 = (x_{1,1,k+1} - x_{1,1,k}) \cdot e_3, \quad \forall\, k = 0, \ldots, N_3. \tag{35}$$

As in Section 4.5.1, the indices labeled 1 are only placeholders. The grid spacing for each variable is independent of the other two variables, so any arbitrary placeholder could have been used.

Equations (20) – (22) of the planes in Figure 5 become:

$$\text{Plane } Y_{1368} \qquad \left(x - \sum_{i=1}^{N_1} x_{i,1,1} \cdot \lambda_i^1 \right) \cdot e_1 \cdot \Delta x_3 - \left(x - \sum_{k=1}^{N_3} x_{1,1,k} \cdot \lambda_k^3 \right) \cdot e_3 \cdot \Delta x_1 = 0, \tag{36}$$

$$\text{Plane } Y_{1458} \qquad \left(x - \sum_{i=1}^{N_1} x_{i,1,1} \cdot \lambda_i^1 \right) \cdot e_1 \cdot \Delta x_2 - \left(x - \sum_{j=1}^{N_2} x_{1,j,1} \cdot \lambda_j^2 \right) \cdot e_2 \cdot \Delta x_1 = 0, \tag{37}$$

$$\text{Plane } Y_{1278} \qquad \left(x - \sum_{j=1}^{N_2} x_{1,j,1} \cdot \lambda_j^2\right) \cdot e_2 \cdot \Delta x_3 - \left(x - \sum_{k=1}^{N_3} x_{1,1,k} \cdot \lambda_k^3\right) \cdot e_3 \cdot \Delta x_2 = 0. \quad (38)$$

As in Section 4.5.1, the variables λ_i, λ_j and λ_k are active for just one value of $i \in 0, \dots, N_1$, $j \in 0, \dots, N_2$ and $k \in 0, \dots, N_3$ so that only the three appropriate planes are defined.

To activate the vertices of the appropriate simplex, define three binary variables, $Y_{1368} \in \{0,1\}$, $Y_{1458} \in \{0,1\}$ and $Y_{1278} \in \{0,1\}$ that are each activated as in Section 4.4.2:

$$\begin{aligned} -\Delta x_1 \cdot \Delta x_3 \cdot (1 - Y_{1368}) \leq {} & \left(x - \sum_{i=1}^{N_1} x_{i,1,1} \cdot \lambda_i^1\right) \cdot e_1 \cdot \Delta x_3 \\ & - \left(x - \sum_{k=1}^{N_3} x_{1,1,k} \cdot \lambda_k^3\right) \cdot e_3 \cdot \Delta x_1 \leq \Delta x_1 \cdot \Delta x_3 \cdot Y_{1368}, \quad (39) \\ -\Delta x_1 \cdot \Delta x_2 \cdot (1 - Y_{1458}) \leq {} & \left(x - \sum_{i=1}^{N_1} x_{i,1,1} \cdot \lambda_i^1\right) \cdot e_1 \cdot \Delta x_2 \\ & - \left(x - \sum_{j=1}^{N_2} x_{1,j,1} \cdot \lambda_j^2\right) \cdot e_2 \cdot \Delta x_1 \leq \Delta x_1 \cdot \Delta x_2 \cdot Y_{1458}, \quad (40) \\ -\Delta x_2 \cdot \Delta x_3 \cdot (1 - Y_{1278}) \leq {} & \left(x - \sum_{j=1}^{N_2} x_{1,j,1} \cdot \lambda_j^2\right) \cdot e_2 \cdot \Delta x_3 \\ & - \left(x - \sum_{k=1}^{N_3} x_{1,1,k} \cdot \lambda_k^3\right) \cdot e_3 \cdot \Delta x_2 \leq \Delta x_2 \cdot \Delta x_3 \cdot Y_{1278}. \quad (41) \end{aligned}$$

The remaining equations allow only the relevant simplex vertices to remain active according to the scheme in Table 1. Noticing that the vertices inactivated lie on a single edge, the following constraints deactivate the entire edge:

$$\begin{aligned} \sum_{i=0}^{N_1} w_{i,j,k-1} & \leq 2 + Y_{1278} - \lambda_j^2 - \lambda_k^3, \quad \forall\, j = 1, \dots, N_2,\ k = 1, \dots, N_3, & (42) \\ \sum_{i=0}^{N_1} w_{i,j-1,k} & \leq 3 - Y_{1278} - \lambda_j^2 - \lambda_k^3, \quad \forall\, j = 1, \dots, N_2,\ k = 1, \dots, N_3, & (43) \\ \sum_{j=0}^{N_2} w_{i,j,k-1} & \leq 2 + Y_{1368} - \lambda_i^1 - \lambda_k^3, \quad \forall\, i = 1, \dots, N_1,\ k = 1, \dots, N_3, & (44) \\ \sum_{j=0}^{N_2} w_{i-1,j,k} & \leq 3 - Y_{1368} - \lambda_i^1 - \lambda_k^3, \quad \forall\, i = 1, \dots, N_1,\ k = 1, \dots, N_3, & (45) \\ \sum_{k=0}^{N_3} w_{i,j-1,k} & \leq 2 + Y_{1458} - \lambda_i^1 - \lambda_j^2, \quad \forall\, i = 1, \dots, N_1,\ j = 1, \dots, N_2, & (46) \end{aligned}$$

$$\sum_{k=0}^{N_3} w_{i-1,j,k} \quad \leq \quad 3 - Y_{1458} - \lambda_i^1 - \lambda_j^2, \quad \forall\, i = 1, \ldots, N_1,\; j = 1, \ldots, N_2. \tag{47}$$

As in Section 4.5.1, the variables λ_i, λ_j and λ_k allow only one line to be deactivated in the domain space for each plane.

4.5.3 2D Domain with SOS2 Index

In this section, we describe domain partitioning for orientation 1 – 4, but note that this method can be similarly developed for any triangulation orientation. When triangulation orientation 1 – 4 (shown in Figure 2) is tessellated across the domain as shown in Figure 10, we can define a new index t such that $t \in \{0, 1, \ldots, N_1 - 1, N_1, N_1 + 1, \ldots, N_1 + N_2 - 1, N_1 + N_2\}$.

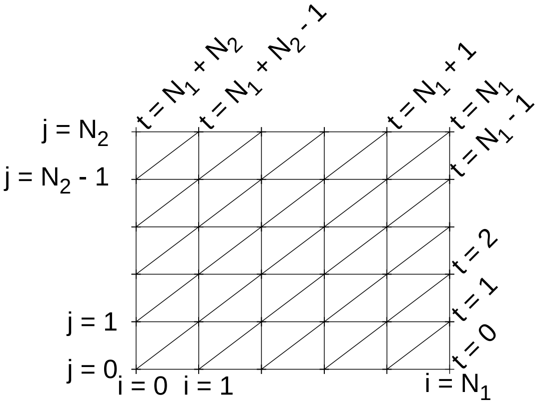


Figure 10: Triangulation tessellation in two dimensions.

Each t represents a diagonal of gridpoints, so that each gridpoint along a given diagonal t can be represented by $x_{i,j} = x_{i,i-N_1+t}$ where $\max\{0,\, N_1 - t\} \leq i \leq \min\{N_2 + N_1 - t,\, N_1\}$. Since two adjacent diagonals activate a simplex, we can define $SOS2$ variable set Ω_t such that:

$$\Omega_t = \sum_i w_{i,i-N_1+t}, \quad \forall\, t, \tag{48}$$

$$\sum_t \Omega_t = 1, \tag{49}$$

$$\Omega_t \quad SOS2. \tag{50}$$

where $w_{i,j}$ is the convex combination weight associated with gridpoint $x_{i,j}$.

Notice that this change in formulation has changed the order of the constraints from N^2 to N. Using the index t requires us to introduce $N_1 + N_2 + 1$ new $SOS2$ variables, but it also eliminates the binary variable representing the switch between the two triangles in the rectangle.

4.5.4 3D Domain with 3 SOS2 Indices

Triangulation orientation 1 – 8 partitions domains of three dimensions into six equal simplices using the planes shown in Figure 5. As in Section 4.5.3, we introduce new indices t_1, t_2, t_3, each representing a diagonal of gridpoints;

$$\begin{aligned} t_1 \quad &\in \quad \{0,1,\dots,N_1,\dots,N_1+N_2-1,N_1+N_2\} \rightarrow x_{i,j,k} = x_{i,i-N_1+t_1,k} \\ &\quad\text{where } \max\{0,\, N_1-t_1\} \le i \le \min\{N_2+N_1-t_1,\, N_1\}, \quad \forall\, k, \end{aligned} \tag{51}$$

$$\begin{aligned} t_2 \quad &\in \quad \{0,1,\dots,N_1,\dots,N_1+N_3-1,N_1+N_3\} \rightarrow x_{i,j,k} = x_{i,j,i-N_1+t_2} \\ &\quad\text{where } \max\{0,\, N_1-t_2\} \le i \le \min\{N_3+N_1-t_2,\, N_1\}, \quad \forall\, j, \end{aligned} \tag{52}$$

$$\begin{aligned} t_3 \quad &\in \quad \{0,1,\dots,N_2,\dots,N_2+N_3-1,N_2+N_3\} \rightarrow x_{i,j,k} = x_{i,j,j-N_2+t_3} \\ &\quad\text{where } \max\{0,\, N_2-t_3\} \le j \le \min\{N_3+N_2-t_3,\, N_2\}, \quad \forall\, i, \end{aligned} \tag{53}$$

and we can use these indices to define three new $SOS2$ variable sets Ω_{t_1}, Ω_{t_2}, and Ω_{t_3}. As in Section 4.5.3, these new variables and constraints represent the observation that each point in the domain can be written as a convex combination of points on two adjacent planes:

$$\text{Plane} \quad Y_{1458} \begin{cases} \Omega_{t_1} = \sum\limits_i w_{i,i-N_1+t_1,k}, \quad \forall\, t_1,\ k, \\ \sum\limits_{t_1} \Omega_{t_1} = 1, \\ \Omega_{t_1}\ SOS2, \end{cases} \tag{54}$$

$$\text{Plane} \quad Y_{1368} \begin{cases} \Omega_{t_2} = \sum\limits_i w_{i,j,i-N_1+t_2}, \quad \forall\, t_2,\ j, \\ \sum\limits_{t_2} \Omega_{t_2} = 1, \\ \Omega_{t_2}\ SOS2, \end{cases} \tag{55}$$

$$\text{Plane}\quad Y_{1278}\begin{cases}\Omega_{t_3}=\sum\limits_j w_{i,j,i-N_2+t_3}, & \forall\, t_3,\ i,\\ \sum\limits_{t_3}\Omega_{t_3}=1, & \\ \Omega_{t_3}\ SOS2. & \end{cases} \tag{56}$$

Like the previously-presented method (Sections 4.5.1 and 4.5.2), these constraints are order N^2. But this method requires introduction of $[2\times(N_1+N_2+N_3)+3]$ continuous variables as opposed to the 3 binary variables that the other method needs to deactivate the appropriate vertices. So, when the distances between the gridpoints are constant and the partitioning is relatively fine, it is better to use the previous method for deactivating vertices. But, if there is a need to use non-equal partitioning of the domain, then it may be advantageous to use this new method, as it does not rely on equal grid spacing.

4.6 Interpolation within a Simplex

With the box and simplex constraints described in Sections 4.1 to 4.5, the appropriate $n+1$ vertices discussed in Section 3 have been activated for domain $X\in\mathbb{R}^n$. Interpolation of function $f(x): X\mapsto\mathbb{R}$ is performed by taking the convex combination of the activated vertices. For approximation of an one dimensional variable, linear system of equations (2) described in Section 3 becomes:

$$X\subset\mathbb{R}\begin{cases}\hat{f}(x) & =\sum\limits_{i=0}^{N_1} w_i\cdot f(x_i),\\ x & =\sum\limits_{i=0}^{N_1} w_i\cdot x_i,\\ \sum\limits_{i=0}^{N_1} w_i & =1,\\ w_i & \geq 0,\quad \forall\, i=0,\ldots,N_1.\end{cases} \tag{57}$$

For approximation of a function of two dimensions, the interpolation is:

$$X\subset\mathbb{R}^2\begin{cases}\hat{f}(x) & =\sum\limits_{i,j=0}^{N_1,N_2} w_{i,j}\cdot f(x_{i,j}),\\ x & =\sum\limits_{i,j=0}^{N_1,N_2} w_{i,j}\cdot x_{i,j},\\ \sum\limits_{i,j=0}^{N_1,N_2} w_{i,j} & =1,\\ w_{i,j} & \geq 0,\quad \forall\, i=0,\ldots,N_1,\ j=0,\ldots,N_2.\end{cases} \tag{58}$$

Finally, for a three dimensional function, the interpolation is:

$$X \subset \mathbb{R}^3 \begin{cases} \hat{f}(x) & = \sum\limits_{i,j,k=0}^{N_1,N_2,N_3} w_{i,j,k} \cdot f(x_{i,j,k}), \\ x & = \sum\limits_{i,j,k=0}^{N_1,N_2,N_3} w_{i,j,k} \cdot x_{i,j,k}, \\ \sum\limits_{i,j,k=0}^{N_1,N_2,N_3} w_{i,j,k} & = 1, \\ w_{i,j,k} & \geq 0, \quad \forall\, i = 0, \ldots, N_1,\ j = 0, \ldots, N_2,\ k = 0,\, \ldots,\, N_3. \end{cases} \tag{59}$$

5 Summary of Explicit Equations

This section presents explicit equations for two- and three-dimensional domains to summarize the development in Sections 3 – 4. Equation Sets (62) and (67) – (69) use the simplex activation technique described in Sections 4.5.3 – 4.5.4. An alternative method is described in Sections 4.5.1 – 4.5.2.

5.1 2D Domain

$$\text{Variable 1} \begin{cases} \sum_{i=1}^{N_1} \lambda_i^1 & = 1, \quad \lambda_i^1 \in [0,\, 1], \quad \lambda_i^1\ SOS1 \quad \forall\, i = 1, ..., N_1, \\ \sum\limits_{j=0}^{N_2} w_{0,j} & \leq \lambda_1^1, \\ \sum\limits_{j=0}^{N_2} w_{i,j} & \leq \lambda_i^1 + \lambda_{i+1}^2, \quad \forall\, i = 1, ..., N_1 - 1, \\ \sum\limits_{j=0}^{N_2} w_{N_1,j} & \leq \lambda_{N_1}^1, \end{cases} \tag{60}$$

$$\text{Variable 2} \begin{cases} \sum_{j=1}^{N_2} \lambda_j^2 & = 1, \quad \lambda_j^2 \in [0,\, 1], \quad \lambda_j^2\ SOS1 \quad \forall\, j = 1, ..., N_2, \\ \sum\limits_{i=0}^{N_1} w_{i,0} & \leq \lambda_1^2, \\ \sum\limits_{i=0}^{N_1} w_{i,j} & \leq \lambda_j^2 + \lambda_{j+1}^2, \quad \forall\, j = 1, ..., N_2 - 1, \\ \sum\limits_{i=0}^{N_1} w_{i,N_2} & \leq \lambda_{N_2}^2. \end{cases} \tag{61}$$

$$\text{Simplex Line} \begin{cases} \Omega_t = \sum\limits_i w_{i,i-N_1+t}, \quad \forall\, t, \\ \sum\limits_t \Omega_t = 1, \\ \Omega_t\ SOS2, \end{cases} \tag{62}$$

$$X \subset \mathbb{R}^2 \begin{cases} \hat{f}(x) & = \sum\limits_{i,j=0}^{N_1,N_2} w_{i,j} \cdot f(x_{i,j}), \\ x & = \sum\limits_{i,j=0}^{N_1,N_2} w_{i,j} \cdot x_{i,j}, \\ \sum\limits_{i,j=0}^{N_1,N_2} w_{i,j} & = 1, \\ w_{i,j} & \geq 0, \quad \forall\, i = 0, \ldots, N_1,\ j = 0, \ldots, N_2. \end{cases} \tag{63}$$

5.2 3D Domain

$$\text{Variable 1} \begin{cases} \sum\limits_{i=1}^{N_1} \lambda_i^1 & = 1, \quad \lambda_i^1 \in [0,\, 1], \quad \lambda_i^1\ SOS1 \quad \forall\, i = 1, ..., N_1 - 1, \\ \sum\limits_{j=0}^{N_2} \sum\limits_{k=0}^{N_3} w_{0,j,k} & \leq \lambda_1^1, \\ \sum\limits_{j=0}^{N_2} \sum\limits_{k=0}^{N_3} w_{i,j,k} & \leq \lambda_i^1 + \lambda_{i+1}^1, \quad \forall\, i = 1, ..., N_1 - 1, \\ \sum\limits_{j=0}^{N_2} \sum\limits_{k=0}^{N_3} w_{N_1,j,k} & \leq \lambda_{N_1}^1, \end{cases} \tag{64}$$

$$\text{Variable 2} \begin{cases} \sum\limits_{j=1}^{N_2} \lambda_j^2 & = 1, \quad \lambda_j^2 \in [0,\, 1], \quad \lambda_j^2\ SOS1 \quad \forall\, j = 1, ..., N_2 - 1, \\ \sum\limits_{i=0}^{N_1} \sum\limits_{k=0}^{N_3} w_{i,0,k} & \leq \lambda_1^2, \\ \sum\limits_{i=0}^{N_1} \sum\limits_{k=0}^{N_3} w_{i,j,k} & \leq \lambda_j^2 + \lambda_{j+1}^2, \quad \forall\, j = 1, ..., N_2 - 1, \\ \sum\limits_{i=0}^{N_1} \sum\limits_{k=0}^{N_3} w_{i,N_2,k} & \leq \lambda_{N_2}^2, \end{cases} \tag{65}$$

$$\text{Variable 3} \begin{cases} \sum\limits_{k=1}^{N_3} \lambda_k^3 & = 1, \quad \lambda_k^3 \in [0,\, 1], \quad \lambda_k^2\ SOS1 \quad \forall\, k = 1, ..., N_3 - 1, \\ \sum\limits_{i=0}^{N_1} \sum\limits_{j=0}^{N_2} w_{i,j,0} & \leq \lambda_1^3, \\ \sum\limits_{i=0}^{N_1} \sum\limits_{j=0}^{N_2} w_{i,j,k} & \leq \lambda_k^3 + \lambda_{k+1}^3, \quad \forall\, k = 1, ..., N_3 - 1, \\ \sum\limits_{i=0}^{N_1} \sum\limits_{j=0}^{N_2} w_{i,j,N_3} & \leq \lambda_{N_3}^3, \end{cases} \tag{66}$$

$$\text{Plane} \quad Y_{1458} \begin{cases} \Omega_{t_1} = \sum\limits_{i} w_{i,i-N_1+t_1,k}, \quad \forall\, t_1,\ k, \\ \sum\limits_{t_1} \Omega_{t_1} = 1, \\ \Omega_{t_1}\ SOS2, \end{cases} \tag{67}$$

$$\text{Plane} \quad Y_{1368} \begin{cases} \Omega_{t_2} = \sum\limits_i w_{i,j,i-N_1+t_2}, \quad \forall\, t_2,\ j, \\ \sum\limits_{t_2} \Omega_{t_2} = 1, \\ \Omega_{t_2}\ SOS2, \end{cases} \tag{68}$$

$$\text{Plane} \quad Y_{1278} \begin{cases} \Omega_{t_3} = \sum\limits_j w_{i,j,i-N_2+t_3}, \quad \forall\, t_3,\ i, \\ \sum\limits_{t_3} \Omega_{t_3} = 1, \\ \Omega_{t_3}\ SOS2, \end{cases} \tag{69}$$

$$X \subset \mathbb{R}^3 \begin{cases} \hat{f}(x) & = \sum\limits_{i,j,k=0}^{N_1,N_2,N_3} w_{i,j,k} \cdot f(x_{i,j,k}), \\ x & = \sum\limits_{i,j,k=0}^{N_1,N_2,N_3} w_{i,j,k} \cdot x_{i,j,k}, \\ \sum\limits_{i,j,k=0}^{N_1,N_2,N_3} w_{i,j,k} & = 1, \\ w_{i,j,k} & \geq 0, \quad \forall\, i = 0, \ldots, N_1,\ j = 0, \ldots, N_2,\ k = 0,\ \ldots,\ N_3. \end{cases} \tag{70}$$

6 Applying the Approximation and Estimating the Error

This section applies the approximation algorithm to three examples of increasing difficulty. The function in Section 6.1 has two variables, so it can be directly approximated using a two-dimensional grid. The multilinear function in Section 6.2 has four dimensions, but the function can be separated into a sum of one two- and one three-dimensional function, so the example is approximated using two lookup tables. Finally, the example in Section 6.3 represents an 11-dimensional model that had to be approximated using the extensions discussed in Section 2.3. The approximated models were solved using CPLEX (version 9.0.2) [45] within the modeling language GAMS [46] on a Pentium 4 running Linux.

6.1 Six-Hump Camelback Function

This example is taken from Problem 8.2.5 in Floudas et al. [47]. The objective function is:

$$f_2(x, y) = 4 \cdot x^2 - 2.1 \cdot x^4 + \frac{1}{3} \cdot x^6 + x \cdot y - 4 \cdot y^2 + 4 \cdot y^4. \tag{71}$$

The domains of the 2 continuous variables are $x \in [-2,\ 2]$ and $y \in [-1,\ 1]$. The 6-hump camelback function has 6 local solutions and1 global optimum with $f_2(x = 0.08984,\ y = -0.71266) = -1.03163$.

The average value of the function across the 1×10^7 sample points is 1.128 and the maximum function value is $f_2(x = -2,\ y = -1) = 5.733$. Table 2 records five partitioning schemes for the two variables and Table 3 displays the associated errors.

Table 2: Partitions CB-1 through CB-5 represent the number of segments in the two variable domains. Estimated error associated with each partitioning scheme shown in Table 3.

Partition	x	y
CB-1	4	4
CB-2	8	8
CB-3	16	16
CB-4	32	32
CB-5	64	64

Table 3: Estimate of the errors associated with each of the partitioning schemes introduced in Table 2.

Partition	Absolute	Absolute	Absolute
	Max Error	Ave Error	Std Dev
CB-1	1.934	0.589	0.499
CB-2	1.108	0.175	0.311
CB-3	0.416	0.046	0.092
CB-4	0.126	0.012	0.024
CB-5	0.035	0.003	0.006

Replacing the 6-hump camelback function with the piecewise-linear approximations, the approximation functions were solved to global optimality. Table 4 compares optimizing the piecewise-linear approximations with optimizing the actual nonlinear function. Note that, in this case, the approximation algorithm finds one of the non-global local minima, which is generated quickly.

Table 4: Optimizing the camelback function using each of the Table 2 partitioning schemes.

Partition	Obj Value	$f_2(x,y)$	x	y	# Vars	# of Nodes	# Iter	CPU (s)
$f_2(x,y)$	-1.032	-1.032	0.090	-0.713	3	495	0	0.41
CB-1	-0.750	-0.752	0.000	-0.500	45	0	2	0.00
CB-2	-0.984	-0.984	0.000	-0.750	117	0	3	0.00
CB-3	-0.984	-0.984	0.000	-0.750	357	0	4	0.00
CB-4	-1.021	-1.022	-0.125	0.688	1221	0	9	0.01
CB-5	-1.028	-1.029	-0.063	0.719	4457	0	9	0.05

6.2 Multilinear Function

This example is taken from the second problem in Table 2 of Gounaris and Floudas [32]. The objective function is

$$f_1(x_1,\ x_2,\ x_3,\ x_4) = x_1 \cdot x_2 - x_2 \cdot x_3 - x_3 \cdot x_4 + x_1 \cdot x_2 \cdot x_3 - x_1 + x_4. \tag{72}$$

Table 5: Partitions MF-1 through MF-3 represent the number of segments in each of the four variable domains.

Partition	x_1	x_2	x_3	x_4
MF-1	4	4	4	4
MF-2	8	8	8	8
MF-3	16	16	16	16

The domains of the 4 continuous variables are $x_i \in [0,\ 1] \quad \forall\, i.$ The average function value across the 1×10^7 sample points is -0.125 and there are infinitely many globally optimal solutions with $f_1 = -1$. Examples include $f_1(1,\ 0,\ \alpha,\ 0) = -1 \quad \forall\, \alpha \in [0,\ 1]$ and $f_1(1,\ 0,\ 1,\ \beta) = -1 \quad \forall\, \beta \in [0,\ 1]$. The global maximum of the function is 2. Table 5 records partitioning schemes for the 4 variables and Table 6 displays the associated errors. Finally, Table 7 shows that the approximations were able to capture the important features of the function well enough to reach the global optimum.

Table 6: Estimate of the errors associated with each of the partitioning schemes introduced in Table 5.

Partition	Absolute	Absolute	Absolute
	Max Error	Ave Error	Std Dev
MF-1	0.040	0.0026	0.0083
MF-2	0.011	0.0007	0.0021
MF-3	0.003	0.0002	0.0005

Table 7: Optimizing the multilinear function using each of the Table 5 partitioning schemes.

Partition	Obj Value	x_1	x_2	x_3	x_4	# Vars	# of Nodes	# Iter	CPU (s)
$f_1(x, y)$	-1.00	1.00	0.00	0.00	0.00	5	1	0	0.02
MF-1	-1.00	0.00	1.00	1.00	0.75	188	0	102	0.01
MF-2	-1.00	1.00	0.00	0.75	0.00	872	0	155	0.03
MF-3	-1.00	1.00	0.00	1.00	0.00	5312	0	329	0.27

6.3 EPA Complex Emissions Model

Environmental Protection Agency (EPA) *Title 40 Code of Federal Regulations Part 80.45: Complex Emissions Model* [48] codifies a mathematical model of gasoline emissions for both reformulated and conventional gasoline. The model calculates three emissions types based on the eleven fuel qualities recorded in Table 8: volatile organic (VOC), NO$_X$ (NOX) and toxics (TOX).

EPA Complex Emissions Model legally certifies the emissions performance of gasoline within the bounds specified by Table 8, providing the basis for other legislation, such as *Title 40 Code of Federal Regulations Part 80.41: Standards and requirements for compliance* [49], to set emissions standards. Final products exiting an oil refinery must comply with these standards, or upper bounds, on volatile organic (VOC_{MAX}), NO$_X$ (NOX_{MAX}) and toxics (TOX_{MAX}) emissions.

Although the EPA Model can be formulated as an MINLP [50], it is difficult to solve even a small problem using the EPA Model as a constraint set for optimal reformulated gasoline blending. But, by approximating the functions representing each of the emissions types, we can construct a reasonable substitute for the EPA Model that can be integrated into an optimization algorithm for process improvement.

Table 8: Variables Relevant to the EPA Model [48]. The bounds are the limits of model accuracy for reformulated gasoline. The different upper bounds for conventional gasoline are bracketed.

	Variable	Fuel Quality	Applicable Bounds		Measuring Units
1	OXY	Oxygen content	0.0-4.0		Weight %
2	SUL	Sulfur content	0.0-500.0	[1000.0]	Parts per million
3	RVP	Reid Vapor Pressure	6.4-10.0	[11.0]	Pounds per square inch
4	E200	200^oF distillation fraction	30.0-70.0		Volume %
5	E300	300^oF distillation fraction	70.0-100.0		Volume %
6	ARO	Aromatics content	0.0-50.0	[55.0]	Volume %
7	BEN	Benzene content	0.0-2.0		Volume %
8	OLE	Olefins content	0.0-25.0		Volume %
9	MTB	Methyl tertiary butyl ether			Weight % oxygen
10	ETB	Ethyl tertiary butyl ether			Weight % oxygen
11	ETH	Ethanol content			Weight % oxygen

Because the three components of the EPA emissions model have up to 9 variables in each term, we used the techniques discussed in Section 2.3 to decompose the problem into bilinear and trilinear terms of approximate functions. To estimate the error introduced by the approximations, we randomly sample a large number of domain points (1×10^7) and compute the difference between the interpolated estimate and the actual function value for each sample point.

To reduce the error, we carefully choose the lookup table partitioning and triangulation orientation for the EPA Complex Emissions Model functions. We balance the increased accuracy of finer partitioning with the higher computational times required for many gridpoints. Tables 9 and 10 estimate the error associated with a number of different partitioning schemes. Partitions EPA-1 though EPA-5 in Table 9 represent increasingly accurate approximations.

Table 9: Partitioning schemes EPA-1 through EPA-5 represent the number of segments that partition the eleven variable domains. Estimated error associated with each partitioning scheme shown in Table 10.

Partition	OXY	SUL	RVP	E200	E300	ARO	BEN	OLE	MTB	ETB	ETH
EPA-1	1	1	1	1	1	1	1	1	1	1	1
EPA-2	4	4	4	4	4	4	4	4	4	4	4
EPA-3	4	4	4	4	4	4	4	4	4	8	8
EPA-4	8	4	8	8	8	8	4	4	8	8	8
EPA-5	8	8	8	8	8	8	8	8	8	8	8

Notice from Table 10 that dividing each of the variables into eight segments (Partition EPA-5) is significantly more accurate than dividing each of the variables into only four segments (Partition EPA-2). Partitions EPA-3 and EPA-4 combine the accuracy of a finer partition with the computational practicality of a coarser mesh. The variables associated with ethyl tertiary butyl ether content (ETB) and ethanol content (ETH) are refined in Partition EPA-3 because the constant coefficients associated with these variables in the calculation of acetaldehyde emissions (a component of the toxics emissions) are approximately an order of magnitude larger than the constant coefficients of the other variables. Refining the mesh of these two high-impact variables leads to the significant gains in toxics emissions accuracy of Partition EPA-3 over Partition EPA-2. Using similar logic, Partition EPA-4 further refines specific variables for increased accuracy.

Table 10: Estimate of the errors associated with each of the partitioning schemes introduced in Table 9.

Partition		Absolute Max Error	Relative Max Error	Absolute Ave Error	Relative Ave Error	Absolute Std Dev	Relative Std Dev
EPA-1	VOC	318.87	17.78%	105.28	8.16%	45.74	3.82%
	NOX	99.65	7.80%	30.14	2.34%	19.44	1.58%
	TOX	155.62	164.48%	66.46	66.10%	27.08	25.98%
EPA-2	VOC	44.03	3.53%	8.71	0.66%	5.87	0.45%
	NOX	20.30	1.57%	3.42	0.27%	3.12	0.25%
	TOX	29.36	12.23%	4.15	3.66%	3.13	1.58%
EPA-3	VOC	44.26	3.47%	8.71	0.66%	5.87	0.45%
	NOX	20.22	1.58%	3.42	0.27%	3.12	0.25%
	TOX	11.76	4.75%	1.97	1.84%	1.07	0.60%
EPA-4	VOC	24.75	1.96%	2.49	0.19%	2.29	0.17%
	NOX	15.39	1.20%	2.16	0.17%	2.02	0.16%
	TOX	9.46	3.59%	1.43	1.31%	0.87	0.43%
EPA-5	VOC	24.43	1.96%	2.42	0.18%	2.27	0.17%
	NOX	11.70	0.90%	1.11	0.09%	1.60	0.13%
	TOX	8.76	3.58%	1.05	0.92%	0.80	0.40%

7 Conclusions

The explicit, piecewise-linear functions for two and three dimensions developed in this paper can be easily integrated into an MILP model, allowing us to approximately solve large-scale problems. As we show in Section 6, the algorithm produces good approximations for large, industrially relevant models.

References

[1] Tuy, H.: Convex Analysis and Global Optimization. Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1998.

[2] Sherali, H. D., Adams, W. P.: A Reformulation-Linearization Technique for Solving Discrete and Coninuous Nonconvex Problems. Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 1999.

[3] Floudas, C. A.: Deterministic Global Optimization: Theory, Methods and Applications. Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, Dordrecht, Netherlands, 2000.

[4] Horst, R., Pardalos, P. M., Thoai, N. V.: Introduction to Global Optimization. Nonconvex Optimization and its Applications. Kluwer Academic Publishers, 2000.

[5] Tawarmalani, M., Sahinidis, N. V.: Convexification and Global Optimization in Continuous and Mixed-Integer Nonlinear Programming: Theory, Applications, Software, and Applications. Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[6] Horst, R., Tuy, H.: Global Optimization: Deterministic Approaches. Springer, 2003.

[7] Floudas, C. A., Pardalos, P. M.: State of the art in global optimization: Computational methods and applications – preface. J. Glob. Optim., 7(2):113-113, 1995.

[8] Floudas, C. A., Pardalos, P. M., editors: State of the Art In Global Optimization: Computational Methods and Applications. Nonconvex Optimization and Its Applications, Kluwer Academic Publishers, 1996.

[9] Floudas, C. A., Pardalos, P. M., editors: Frontiers in Global Optimization. Nonconvex Optimization and Its Applications, Kluwer Academic Publishers, 2004.

[10] Floudas, C. A., Akrotirianakis, I. G., Caratzoulas, S., Meyer, C. A., Kallrath, J.: Global optimization in the 21st century: Advances and challenges. Comput. Chem. Eng., 29: 1185–1202, 2005.

[11] Floudas. C. A., Gounaris, C. E.: A review of recent advances in global optimization. J. of Glob. Optim., 45:3–38, 2009.

[12] Kosmidis, V. D., Perkins, J. D., Pistikopoulos, E. N.: Optimization of well oil rate allocations in petroleum fields. Ind. Eng. Chem. Res., 43(14):3513–3527, 2004.

[13] Kosmidis, V. D., Perkins, J. D., Pistikopoulos, E. N.: A mixed integer optimization formulation for the well scheduling problem on petroleum fields. Comput. & Chem. Eng., 29(7):1523–1541, 2005.

[14] Buitrago, S., Rodríguez, E., Espin, D.: Global optimization techniques in gas allocation for continuous flow gas lift systems. In SPE Gas Technology Symposium, Calgary, Alberta, Canada, 1996. Society of Petroleum Engineers. SPE 35616.

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

[16] Nemhauser, G. L., Wolsey, L. A.: Integer and Combinatorial Optimization. J. Wiley, New York, 1988.

[17] Floudas, C. A.: Nonlinear and Mixed-Integer Optimization: Fundamentals and Applications. Oxford University Press, New York, NY, 1995.

[18] Sherali, H. D.: On mixed-integer zero-one representations for seperable lower-semicontinuous piecewise-linear functions. Oper. Res. Letters, 28(4):155–160, 2001.

[19] Keha, A. B., de Farias Jr., I. R., Nemhauser, G. L.: Models for representing piecewise linear cost functions. Oper. Res. Letters, 32(1):44–48, 2004.

[20] Williams, H. P.: Model Building in Mathematical Programming. John Wiley & Sons, Chichester, Great Britian, 1978.

[21] Zhang, H., Wang, S.: Linearly constrained global optimization via piecewise-linear approximation. J. of Computational and Applied Mathematics, 214(1):111–120, April 2008.

[22] Magnani, A., Boyd, S. P.: Convex piecewise-linear fitting. Optim. Eng., 10:1 – 17, 2009.

[23] Rosen J. B., Pardalos, P. M.: Global minimization of large-scale constrained concave quadratic problems by separable programming. Math. Program., 34(2):163 – 174, 1986.

[24] Pardalos, P. M., Rosen J. B.: Constrained Global Optimization: Algorithms and Applications. Lecture Notes in Computer Science. Springer-Verlag, Germany, 1987.

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

[26] Karuppiah, R., Grossmann, I. E.: Global optimization for the synthesis of integrated water systems in chemical processes. Comput. & Chem. Eng., 30:650–673, 2006.

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

[28] Gounaris, C. E., Misener, R., Floudas, C. A.: Computational comparison of piecewise-linear relaxations for pooling problems. Ind. Eng. Chem. Res., 48(12):5742–5766, 2009.

[29] Pham, V., Laird, C., El-Halwagi, M.: Convex hull discretization approach to the global optimization of pooling problems. Ind. Eng. Chem. Res., 48:1973 – 1979, 2009.

[30] Mangasarian, O. L., Rosen, J. B., Thompson, M. E.: Global minimization via piecewise-linear underestimation. J. of Glob. Optim., 32(1):1 – 9, 2005.

[31] Gounaris, C. E., Floudas, C A.: Tight convex underestimators for $\mathcal{C}^2$-continuous problems: I. univariate functions. J. of Glob. Optim., 42(1):51–67, 2008.

[32] Gounaris, C. E., Floudas, C A.: Tight convex underestimators for $\mathcal{C}^2$-continuous problems: II. multivariate functions. J. of Glob. Optim., 42(1):69–89, 2008.

[33] Chien, M., Kuh, E.: Solving nonlinear resistive networks using piecewise-linear analysis and simplicial subdivision. Circuits and Systems, IEEE Transactions on, 24(6):305–317, 1977.

[34] Meyer, C. A., Floudas, C. A.: Trilinear monomials with positive or negative domains: Facets of the convex and concave envelopes. In C. A. Floudas and P. M. Pardalos, editors, Frontiers in Global Optimization, pages 327–352. Kluwer Academic Publishers, 2003.

[35] Meyer, C. A., Floudas, C. A.: Trilinear monomials with mixed sign domains: Facets of the convex and concave envelopes. J. of Glob. Optim., 29(2):125–155, 2004.

[36] Meyer, C. A., Floudas, C. A.: Convex envelopes for edge-concave functions. Math. Program., 103(2):207–224, 2005.

[37] Hughes, R. B., Anderson, M. R.: Simplexity of the cube. Discrete Math, 158(1-3):99–150, 1996.

[38] McCormick, G. P.: Computability of global solutions to factorable nonconvex programs: Part 1-convex underestimating problems. Math. Program., 10(1):147–175, December 1976.

[39] Al-Khayyal, F. A., Falk, J. E.: Jointly constrained biconvex programming. Mathematics of Operations Research, 8(2):273–286, 1983.

[40] Maranas, C. D., Floudas, C. A.: Finding all solutions of nonlinearly constrained systems of equations. J. of Glob. Optim., 7(2):143–182, 1995.

[41] Ryoo, H. S., Sahinidis, N. V.: Analysis of bounds for multilinear functions. J. of Glob. Optim., 19(4):403–424, 2001.

[42] Carathéodory, C.: Über den Variabilitätsbereich der Fourierschen Konstanten von positiven harmonischen Funktionen. Rend. Circ. Mat. Palermo, 32:193–217, 1911.

[43] Beale, E. M. L., Tomlin, J. A.: Special Facilities in a General Mathematical Programming System for Non-convex Problems Using Ordered Sets of Variables. In J. Lawrence, ed., Proceedings of the Fifth International Conference on Operational Research. **1970**, 447-454.

[44] Forrest, J. J. H., Hirst, J. P. H., Tomlin, J. A.: Practical Solution of Large Mixed Integer Programming Problems with Umpire. Manage. Sci., 20:736–773, 1974.

[45] ILOG CPLEX 9.0.2 User's Manual; ILOG, INC.; Mountain View, CA, 2005.

[46] Brooke, A., Kendrick, D., Meeraus, A.: GAMS: A User's Guide. GAMS Development Corporation, 2005.

[47] Floudas, C. A., Pardalos, P. M., Adjiman, C. S., Esposito W. R., Gümüs Z. H., Harding, S. T., Klepeis, J. L Meyer, C. A., Schweiger, C.A.: Handbook of Test Problems in Local and Global Optimization. Nonconvex Optimization and Its Applications. Kluwer Academic Publishers, Dordrecht, Netherlands, 1999.

[48] 40CFR80.45. Code of Federal Regulations: Complex emissions model, July 2007. http://frwebgate.access.gpo.gov/cgi-bin/get-cfr.cgi.

[49] 40CFR80.41. Code of Federal Regulations: Standards and requirements for compliance, June 2008. http://frwebgate.access.gpo.gov/cgi-bin/get-cfr.cgi.

[50] Furman, K. C., Androulakis, I. P.: A novel MINLP-based representation of the original complex model for predicting gasoline emissions. Comput. & Chem. Eng., 32:2857–2876, 2008.