Noname manuscript No.
(will be inserted by the editor)

Mathematical Programming Formulations for Piecewise Polynomial Functions

Bjarne Grimstad · Brage Rugstad Knudsen

Received: date / Accepted: date

Abstract This paper studies mathematical programming formulations for solving optimization problems with piecewise polynomial (PWP) constraint functions. We elaborate on suitable polynomial bases as a means of efficiently representing PWPs in mathematical programs, comparing and drawing connections between the monomial basis, the Bernstein basis, and B-splines. The theory is presented for both continuous and semi-continuous PWPs. Using a disjunctive formulation, we then exploit the characteristic of common polynomial basis functions to significantly reduce the number of nonlinearities, and to suggest a bound-tightening technique for PWP constraints. Upon a standard big-M reformulation yielding an MINLP model, we derive three extensions using logarithmic number of binary variables, Bernstein cuts and an expanded Bernstein basis. Numerical results from solving three test sets of MINLPs to global optimality compares the formulations. The proposed framework shows promising numerical performance, and facilitates the solution of PWP-constrained optimization problems using standard MINLP software.

Keywords Piecewise polynomials · Splines · Mixed integer programming · Disjunctions

1 Introduction

Modeling of optimization problems frequently involves representing functions that are piecewise, discontinuous or nonsmooth. This includes inherently piecewise economical and physical characteristics [32, 74, 8, 39], construction of surrogate models by sampling of simulators [37, 59, 41, 24, 72, 80, 46], and approxi-

B. Grimstad (✉), B. R. Knudsen
Department of Engineering Cybernetics
Norwegian University of Science and Technology
NO-7491 Trondheim, Norway
E-mail: bjarne.grimstad@gmail.com

mate or exact representation of nonconvex functions [15,3,54,66,13,51]. In this paper, we study the problem of efficiently representing and solving optimization problems containing piecewise polynomial (PWP) constraints. Piecewise polynomials are used in a wide range of disciplines, including efficiency curve modeling in electric-power unit commitment [45,55,60], rigid motion systems [48,16], image processing and data compression [17,58,71], flow networks [8, 24,33] and in optimal control [56,6,18].

We consider optimization problems where either or both of the objective function and a subset of the constraints are piecewise polynomial functions. Each polynomial may be nonconvex, and the piecewise polynomial function itself lower semi-continuous. There exists few targeted optimization methods for this class of optimization problems, while some approaches that exploit special structures of nonsmooth optimization problems are applicable, subject to certain modification methods: Womersley and Fletcher [84] developed a descent method for solving composite nonsmooth problems made up from a finite number of smooth functions. Conn and Mongeau [12] constructed a method based on non-differentiable penalty functions for solving discontinuous piecewise linear optimization problems, sketching an extension to problems with PWP constraints. Scholtes [66] developed an active-set method for dealing with nonlinear programs (NLPs) with underlying combinatorial structure in the constraints. Li [44] used a conjugated gradient method for minimizing an unconstrained, strictly convex, quadratic spline. None of these methods are currently available in standard optimization software.

From a broader perspective, applicable solution approaches to PWP optimization problems include methods based on general nonsmooth optimization, smoothing techniques and mixed integer programming (MIP). Bundle-type and subgradient methods [70,29], originally developed for nonsmooth convex optimization, may be applied to optimization problems with general nonsmooth structures such as PWPs through Clark's generalized gradients [67,66]. These generalized methods for nonsmooth optimization are known to have poor convergence properties for nonconvex structures [66]. Smoothing techniques for nonsmooth functions encompass a variety of techniques, seeking to ensure sufficient smoothness for gradient-based methods [86]. Many of these methods are, however, designed for optimizing a nonsmooth function on a convex set, e.g [11,53]. Meanwhile, smoothing techniques for discontinuities by means of step-function approximations (e.g. [86,10]) are known to be prone to numerical instabilities, particularly for increasing accuracies of the discontinuity [12,22,83]. Exploitation of MIP for solving PWP optimization problems beyond complete approximative linearization [51] and direct solution as a nonconvex mixed integer nonlinear programming (MINLP) problem appears to be limited.

We adopt disjunctive representations of PWP constraints, drawing upon the extensive work on disjunctive programming (DP) formulations and representation of piecewise linear (PWL) functions [2,54,69,76]. Modeling piecewise functions as disjunctions enables application of MIP techniques [78], or specialized branch-and-bound or branch-and-cut schemes with a set condition for

representing the piecewise constraints [3,43,54,38]. Still, using MIP to solve PWP-constrained optimization problems puts high requirement on the constraint formulation in order to overcome the inherent problem complexity. To this end, polynomial spline formulations [68], such as the B-spline, may be applied. Polynomial splines are constructed from overlapping (piecewise) polynomials with local support, and embodies a large and versatile set of techniques for modeling PWPs with favorable smoothness and numerical properties. For decades, polynomial splines, which we simply refer to as *splines* in this paper, have played an important role in function approximation and geometric modeling. In particular, they have been popular as nonlinear basis functions in regression problems [19,65,28], for example in kernel methods [20,85,30], and in finite element methods [31]. Yet, there are few references [24,8] using splines within mathematical programming beyond the optimization of spline design parameters [82,62,61].

The availability of spline-compatible optimization algorithms and codes is limited. In a recent work, [25] develop a spatial branch-and-bound (sBB) algorithm for global optimization of spline-constrained problems. While the algorithm is shown to be highly efficient, it is only available as a specialized code and requires software for spline generation [26]. To address the comparably high modeling and implementation effort required for using the specialized sBB algorithm of [25], [23] propose an explicit constraint-formulation for continuous splines, yielding a mixed-integer quadratically constrained program (MIQCP). In this paper, we build upon and extend the spline-constraint framework of [25] and [23]. Our main contribution is a general-purpose framework for mathematical programming formulations of piecewise polynomial constraints, subsuming spline constraints. We derive relations between Bernstein polynomial basis for PWPs and spline constraints, and extend this to lower semi-continuous PWPs. From a disjunctive PWP formulation, we construct an exact MINLP formulation with an associated bound-tightening technique, Bernstein cuts and reduced, logarithmic formulations to improve the computational efficiency.

The remainder of the paper is organized as follows. In Section 2, we present the theory of Bernstein polynomials, exploited as basis functions for piecewise polynomials in Section 3. Following this, we explore in Section 4 PWPs in the framework of the B-spline to highlight their applicability as a modeling tool for constraints. In Section 5, we present a disjunctive formulation of the PWPs. Using this disjunctive polynomial formulation, we present in Section 6 several MINLP formulations with cutting planes derived from the Bernstein polynomials. In Section 7, we present computational results from the proposed formulations, comparing the results with existing methods for optimizing PWP functions.

2 Polynomial bases

This section provides background material on polynomial functions to cover the theory needed for developing an optimization framework for piecewise

polynomial functions. The theory is presented as a series of propositions that summarize some classical results for polynomials. For brevity, most propositions are given without rigorous proofs; each proposition may, however, be proved by simple algebraic manipulations. To further simplify the disposition, we have put some computational details in Appendix A.

We begin by introducing the monomial and Bernstein basis for polynomials in one variable. Several propositions are provided that ultimately enables computation of lower and upper bounds on any polynomial. These results are then extended to the multivariate case.

2.1 Univariate polynomials and the Bernstein basis

Let $\mathbb{P}_p$ denote the vector space of polynomial functions in one real variable with degree less than or equal to $p \in \mathbb{N}$, i.e.

$$\mathbb{P}_p = \text{span}\{M_i\}_{i=0}^p, \quad M_i : \mathbb{R} \to \mathbb{R}, \quad M_i(x) = x^i, \quad 0 \leq i \leq p. \tag{1}$$

The set $\{M_i\}_{i=0}^p$ is commonly referred to as the *monomial basis* or *power basis* of $\mathbb{P}_p$. Any polynomial $f \in \mathbb{P}_p$ can be written as

$$f(x) = \sum_{i=0}^{p} a_i M_i(x) = \boldsymbol{a}^\mathsf{T} \boldsymbol{M}_p(x), \tag{2}$$

where the vector of coefficients $\boldsymbol{a} = [a_i]_{i=0}^p \in \mathbb{R}^{p+1}$, and the vector of monomial basis functions $\boldsymbol{M}_p(x) = [M_i(x)]_{i=0}^p \in \mathbb{R}^{p+1}$.

An alternative basis for $\mathbb{P}_p$ is the *Bernstein basis*

$$B_{i,p} = \binom{p}{i} x^i (1-x)^{p-i}, \quad 0 \leq i \leq p. \tag{3}$$

Since $\text{span}\{B_{i,p}\}_{i=0}^p = \mathbb{P}_p$, any polynomial $f \in \mathbb{P}_p$ may be expressed in the Bernstein basis as

$$f(x) = \sum_{i=0}^{p} c_i B_{i,p}(x) = \boldsymbol{c}^\mathsf{T} \boldsymbol{B}_p(x), \tag{4}$$

where the vector of coefficients $\boldsymbol{c} = [c_i]_{i=0}^p \in \mathbb{R}^{p+1}$, and the vector of Bernstein basis functions $\boldsymbol{B}_p(x) = [B_{i,p}(x)]_{i=0}^p \in \mathbb{R}^{p+1}$.

The monomial and Bernstein basis are related via the linear mapping $\boldsymbol{M}_p = Q_p \boldsymbol{B}_p$, where $Q_p \in \mathbb{R}^{(p+1)\times(p+1)}$ is the *transformation matrix* given in Appendix A.1. Consequently, $\boldsymbol{a}^\mathsf{T} \boldsymbol{M}_p(x) = \boldsymbol{c}^\mathsf{T} \boldsymbol{B}_p(x)$, given that $\boldsymbol{c} = Q_p^\mathsf{T} \boldsymbol{a}$.

The Bernstein polynomials possess several useful properties that facilitate the study of signs and bounds on polynomial functions. These properties, to be presented next, will later be utilized to devise bounds on PWPs.

Lemma 1 (Convex combination property of Bernstein polynomials) *The following holds true for a set of degree p Bernstein polynomials* $\{B_{i,p}\}_{i=0}^{p}$*:*

$$\begin{aligned} &B_{i,p}(x) \geq 0, \quad \forall x \in [0,1], \quad i = 0, \ldots, p \\ &\sum_{i=0}^{p} B_{i,p}(x) = 1, \quad \forall x \in \mathbb{R} \end{aligned} \tag{5}$$

Proof The lemma is proved by applying Newton's binomial identity to (3). □

Proposition 1 (Bounds on Bernstein polynomials) *Let* $f \in \mathbb{P}_p$ *be a polynomial expressed in the Bernstein basis* (4)*, and denote* $c^L = \min\{c_i\}_{i=0}^{p}$ *and* $c^U = \max\{c_i\}_{i=0}^{p}$*. Then, a valid bound on* f *is* $c^L \leq f(x) \leq c^U \ \forall x \in [0,1]$*.*

Proof It follows from Lemma 1 that

$$\begin{aligned} f(x) &= \sum_{i=0}^{p} c_i B_{i,p}(x) \leq \sum_{i=0}^{p} c^U B_{i,p}(x) = c^U, \\ c^L &= \sum_{i=0}^{p} c^L B_{i,p}(x) \leq \sum_{i=0}^{p} c_i B_{i,p}(x) = f(x), \end{aligned}$$

which proves the proposition. □

Observe that Proposition 1 holds for $x \in [0,1]$. To obtain a bounding box on a general domain $[x^L, x^U]$, we perform an affine change of variable.

Proposition 2 (Reparametrization of polynomial) *Let* $f \in \mathbb{P}_p$ *be a polynomial* $f(x) = \boldsymbol{a}^\intercal \boldsymbol{M}_p(x)$*, for* $x \in [x^L, x^U]$*. Consider the affine change of variables* $x = (x^U - x^L)u + x^L$*, with* $u \in [0,1]$*. The polynomial* f *can be reparametrized from* x *to* u *via the linear mapping* $\boldsymbol{M}_p(x) = R_p \boldsymbol{M}_p(u)$*, where* $R_p \in \mathbb{R}^{(p+1)\times(p+1)}$ *is the reparametrization matrix given in Appendix A.2.*

Proof Cf. [57]. □

By combining Propositions 1 and 2, we obtain a bound for polynomials on a general domain $x \in [x^L, x^U]$.

Proposition 3 (Polynomial bounds on general intervals) *Let* $\boldsymbol{a}$ *be the coefficients of a polynomial* $f \in \mathbb{P}_p$ *in the monomial basis. Furthermore, let* $\boldsymbol{c} = (R_p Q_p)^\intercal \boldsymbol{a}$ *be the coefficients obtained by first reparametrizing the monomial basis from* $x \in [x^L, x^U]$ *to* $u \in [0,1]$*, and then transforming the monomial basis to the Bernstein basis in* u*. From Proposition 1, it follows that*

$$c^L \leq f(x) \leq c^U \quad \forall x \in [x^L, x^U], \tag{6}$$

where $c^L = \min\{c_i\}_{i=0}^{p}$ *and* $c^U = \max\{c_i\}_{i=0}^{p}$*.*

Proof The result follows directly from Propositions 1 and 2. □

2.2 Multivariate polynomials

A vector space of multivariate polynomials $f : \mathbb{R}^d \to \mathbb{R}$ in the variables $\boldsymbol{x} = (x_1, \ldots, x_d) \subseteq \mathbb{R}^d$, can be constructed by taking the tensor product of univariate polynomial bases. Specifically, we construct a multivariate polynomial basis as

$$\boldsymbol{M}_p^d(\boldsymbol{x}) = \bigotimes_{j=1}^{d} \boldsymbol{M}_p(x_j), \tag{7}$$

where $\boldsymbol{M}_p(x_j) = [M_i(x_j)]_{i=0}^p$ is a vector of $p+1$ monomial basis functions in the variable x_j.

In (7), $\boldsymbol{M}_p^d$ is a vector of $n = (p+1)^d$ polynomials of degree less than or equal to dp in d variables: i.e. each multivariate basis function results from d products of univariate polynomial basis functions of degree less than or equal to p. The basis spans a (tensor product) vector space of multivariate polynomials, denoted $\mathbb{P}_p^d = \text{span}\{M_{i,p}^d\}_{i=0}^{n-1}$. For notational brevity, we assume in the above construction that the polynomial basis and degree are equal for all variables. This assumptions can be removed without loss of generality as the multivariate basis may be constructed from any combination of univariate polynomial bases of varying degrees. Subsequently, we consider also the multivariate Bernstein basis for $\mathbb{P}_p^d$, which we denote by $\boldsymbol{B}_p^d$.

Using the multivariate polynomial basis, we may express any polynomial $f \in \mathbb{P}_p^d$ as

$$f(\boldsymbol{x}) = \sum_{i=0}^{n-1} a_i M_{i,p}^d(\boldsymbol{x}) = \boldsymbol{a}^\mathsf{T} \boldsymbol{M}_p^d(\boldsymbol{x}). \tag{8}$$

The basis $\boldsymbol{M}_p^d$ is orthogonal and hence $\dim(\mathbb{P}_p^d) = (p+1)^d$. The exponential growth in the number of basis functions with the number of variables d, is a phenomenon often referred to as the curse of dimensionality [5], limiting most practical applications of tensor product bases to 5-6 variables.

The important property of the bounding box in Proposition (1) naturally extends to the multivariate case.

Proposition 4 (Multivariate polynomial bounds on the unit cube) *Let $f \in \mathbb{P}_p^d$ be a polynomial expressed in the multivariate Bernstein basis*

$$f(\boldsymbol{x}) = \boldsymbol{c}^\mathsf{T} \boldsymbol{B}_p^d(\boldsymbol{x}) = \boldsymbol{c}^\mathsf{T} \bigotimes_{j=1}^{d} \boldsymbol{B}_p(x_j). \tag{9}$$

Then, $c^L \leq f(\boldsymbol{x}) \leq c^U \ \forall \boldsymbol{x} \in [0,1]^d$, where $c^L = \min\{c_i\}_{i=0}^{n-1}$ and $c^U = \max\{c_i\}_{i=0}^{n-1}$.

Proof For any $1 \leq r \leq d$, let

$$\boldsymbol{B}_p^{(d,-r)}(\boldsymbol{x}) = \bigotimes_{j=1, j\neq r}^{d} \boldsymbol{B}_p(x_j). \tag{10}$$

Then, given $n = m^d = (p+1)^d$, it follows from Lemma 1 that

$$\begin{aligned}\sum_{i=0}^{m^d-1} B_{i,p}^d(\boldsymbol{x}) &= \left(\sum_{i=0}^{p} B_{i,p}(x_r)\right)\left(\sum_{i=0}^{m^{(d-1)}-1} B_{i,p}^{(d,-r)}(\boldsymbol{x})\right) \\ &= \sum_{i=0}^{m^{(d-1)}-1} B_{i,p}^{(d,-r)}(\boldsymbol{x}).\end{aligned} \tag{11}$$

The above relation implies that

$$\sum_{i=0}^{n-1} B_{i,p}^d(\boldsymbol{x}) = 1. \tag{12}$$

The identity in (12), combined with Lemma 1, ensures that for all $\boldsymbol{x} \in [0,1]^d$

$$\begin{aligned}f(\boldsymbol{x}) = \sum_{i=0}^{n-1} c_i B_{i,p}^d(\boldsymbol{x}) \leq c^U \sum_{i=0}^{n-1} B_{i,p}^d(\boldsymbol{x}) \leq c^U, \\ c^L \leq c^L \sum_{i=0}^{n-1} B_{i,p}^d(\boldsymbol{x}) \leq \sum_{i=0}^{n-1} c_i B_{i,p}^d(\boldsymbol{x}) = f(\boldsymbol{x}),\end{aligned} \tag{13}$$

which proves the proposition. □

Proposition 4 provides bounds on a polynomial expressed in the multivariate Bernstein basis for $\boldsymbol{x}$ constrained to the unit cube $[0,1]^d$. Analogous to the univariate case, we obtain bounds for general domains by reparametrizing the basis.

Proposition 5 (Multivariate polynomial bounds on general domains) *Let $f \in \mathbb{P}_p^d$ be a polynomial expressed in the multivariate monomial basis*

$$f(\boldsymbol{x}) = \boldsymbol{a}^T \bigotimes_{j=1}^{d} \boldsymbol{M}_p(x_j), \tag{14}$$

for $\boldsymbol{x} \in [x_1^L, x_1^U] \times \ldots \times [x_d^L, x_d^U] = [\boldsymbol{x}^L, \boldsymbol{x}^U]$. For each variable x_j, let $R_{p,j}$ be the reparametrization matrix that reparametrizes the monomial basis from $x_j \in [x_j^L, x_j^U]$ to $u_j \in [0,1]$, computed according to (44) in Appendix A.2. Furthermore, let Q_p be the transformation matrix computed as in (43) in Appendix A.1. Then, f may be mapped to the multivariate Bernstein basis as follows:

$$\begin{aligned}f(\boldsymbol{u}) &= \boldsymbol{a}^T \bigotimes_{j=1}^{d} R_{p,j} Q_p \boldsymbol{B}_p(u_j) \\ &= \boldsymbol{a}^T \left(\bigotimes_{j=1}^{d} R_{p,j} Q_p\right)\left(\bigotimes_{j=1}^{d} \boldsymbol{B}_p(u_j)\right) = \boldsymbol{c}^T \bigotimes_{j=1}^{d} \boldsymbol{B}_p(u_j)\end{aligned} \tag{15}$$

where

$$\boldsymbol{c}^T = \boldsymbol{a}^T \bigotimes_{j=1}^{d} R_{p,j} Q_p. \tag{16}$$

Finally, we may apply Proposition 4 to obtain the bounds

$$c^L \leq f(\boldsymbol{x}) \leq c^U \quad \forall \boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U], \tag{17}$$

where $c^L = \min\{c_i\}_{i=0}^{n-1}$ *and* $c^U = \max\{c_i\}_{i=0}^{n-1}$.

Proof The result follows directly from Proposition 4. □

3 Piecewise polynomial functions

In this section, we describe to piecewise polynomial functions (PWPs) to which we first give a formal definition for the *continuous* case. We then depart from the continuity requirement in order to consider the more general case of *lower semi-continuous* PWPs. The definitions given below provide a framework for the development of the mathematical programming formulations in Sec. 5 and 6.

Definition 1 (Continuous piecewise polynomial function) Let $D \in \mathbb{R}^d$ be a compact set. A function $f : D \subset \mathbb{R}^d \to \mathbb{R}$ is a *continuous piecewise polynomial* if and only if there exists a finite family of polytopes $\mathit{\Pi}$ such that $D = \bigcup_{P \in \mathit{\Pi}} P$ and

$$f(\boldsymbol{x}) := \{f_P(\boldsymbol{x}), \;\; \boldsymbol{x} \in P, \;\; \forall P \in \mathit{\Pi} \;, \tag{18}$$

where $f_P : P \subset \mathbb{R}^d \to \mathbb{R}$ and $f_P \in \mathbb{P}_p^d$ for all $P \in \mathit{\Pi}$, and some degree $p \in \mathbb{N}_0$.

Note that the domain D does not need to be connected or convex. Continuity of f on D is ensured since $f_{P_1}(\boldsymbol{x}) = f_{P_2}(\boldsymbol{x})$ if $\boldsymbol{x} \in P_1 \cap P_2$ for two adjacent polytopes $P_1, P_2 \in \mathit{\Pi}$.

In the above definition, the polynomial pieces $\{f_P\}_{P \in \mathit{\Pi}}$ are constructed from some basis that spans the space $\mathbb{P}_p^d$. A special case occurs when $d = 1$ and $p = 1$, for which $\{f_P\}_{P \in \mathit{\Pi}}$ are linear functions in one variable, and f a continuous piecewise linear function. Furthermore, for $d > 1$ and $p = 1$, f is a continuous piecewise multilinear function due to the tensor product construction of $\mathbb{P}_p^d$. In general, the polynomial pieces are of degree less than or equal to $dp \in \mathbb{N}_0$.

3.1 Polytopes on a Rectilinear Grid

Def. 1 of continuous PWPs does not prescribe the polytopes; they may for instance be given as the convex hull of a finite number of points, or as a system of linear inequalities. Some formulations for piecewise linear functions require the polytopes to be simplices resulting from a triangulation of D [77].

For most practical applications of PWPs, the polytopes are assumed to be n-orthotopes (hyperrectangles/boxes) arranged on an axis-aligned rectilinear grid.[1] In the rest of this paper, we will assume that the domain D is partitioned on such a rectilinear grid, for which the polytopes in $\mathit{\Pi}$ are characterized as follows.

For $i \in \{1, \dots, d\}$, let $\pi^i = \{\pi^i_0, \dots, \pi^i_{m_i}\} \in \mathbb{R}$ denote a strictly monotonically increasing sequence of m_i real numbers, e.g. $\pi^i_0 < \pi^i_1 < \ldots < \pi^i_{m_i}$. Let G denote the rectilinear grid with vertices $\mathcal{V}(G) = \{(v_1, \dots, v_d) : v_i \in \pi^i \; \forall i \in \{1, \dots, d\}\}$. Furthermore, let $\mathcal{V}(G^-) = \{(v_1, \dots, v_d) : v_i \in \pi^i \setminus \{\pi^i_{m_i}\} \; \forall i \in \{1, \dots, d\}\} \subset \mathcal{V}(G)$ denote the vertices of G, except the 'rightmost' vertices in the grid. We index a vertex $v \in \mathcal{V}(G)$ with $I(v) = k = (k_1, \dots, k_d)$ so that $v_1 = \pi^1_{k_1}$, …, $v_d = \pi^d_{k_d}$.

A box in G, identified by its 'leftmost' vertex $v \in \mathcal{V}(G^-)$, is given as

$$P_v = \{\boldsymbol{x} \in \mathbb{R}^d : \pi^i_{k_i} \leq x_i \leq \pi^i_{k_i+1}, \, \forall i \in \{1, \dots, d\} : k = I(v)\}. \tag{19}$$

In compliance with Def. 1, the n-orthotope P_v is a bounded polytope and $\mathit{\Pi}_G$ is given as the set of n-orthotopes on the grid G, i.e. $\mathit{\Pi}_G := \{P_v\}_{v \in V(G^-)}$. The number of boxes in $\mathit{\Pi}_G$ is $|\mathit{\Pi}_G| = m_1 \cdots m_d$. Subsequently, we will simply drop the subscript and write $\mathit{\Pi} = \mathit{\Pi}_G$. We will also denote with $\mathbb{P}^d_p(G)$ the space of piecewise degree p polynomials with a partition of the domain D given by the rectilinear grid G.

3.2 The epigraph of piecewise polynomials

A continuous PWP $f : D \subset \mathbb{R}^n \to \mathbb{R}$ may be modeled by its epigraph $\text{epi}(f) := \{(\boldsymbol{x}, z) \in D \times \mathbb{R} : f(\boldsymbol{x}) \leq z\}$. We assume that D is a bounded domain and that f participates in a constraint on the form $f(\boldsymbol{x}) \leq 0$ or in an objective function to be minimized. That is, the constraint $f(\boldsymbol{x}) \leq 0$ can be modeled as $(\boldsymbol{x}, z) \in \text{epi}(f)$, $z \leq 0$, and the objective f can be modeled as the minimization of z subject to $(\boldsymbol{x}, z) \in \text{epi}(f)$.

The epigraph of f can be expressed as the union of epigraphs of its pieces f_P, i.e.

$$\text{epi}(f) = \bigcup_{P \in \mathit{\Pi}} \text{epi}(f_P), \tag{20}$$

where $\text{epi}(f_P) := \{(\boldsymbol{x}, z) \in P \times \mathbb{R} : f_P(\boldsymbol{x}) \leq z\}$. As illustrated by Fig. 1, the epigraph of a piecewise function is in general a nonconvex set. Note that $\text{epi}(f_P)$ is convex if and only if f_P is convex on P. Furthermore, f may be nonconvex, even if f_P (and $\text{epi}(f_P)$) is convex for all $P \in \mathit{\Pi}$.

The theory developed by Jeroslow and Lowe [34–36] shows that $\text{epi}(f)$ can be modeled as a MILP if and only if f is piecewise linear and lower semi-continuous. Based on this theory, Vielma et al. [77,78] derived new MILP

[1] To understand why a rectilinear grid is practical, consider a domain partitioned into a set of non-regular polytopes (resembling a shattered window). Patching together higher order polynomials on these polytopes in order to ensure continuity on all faces is non-trivial.

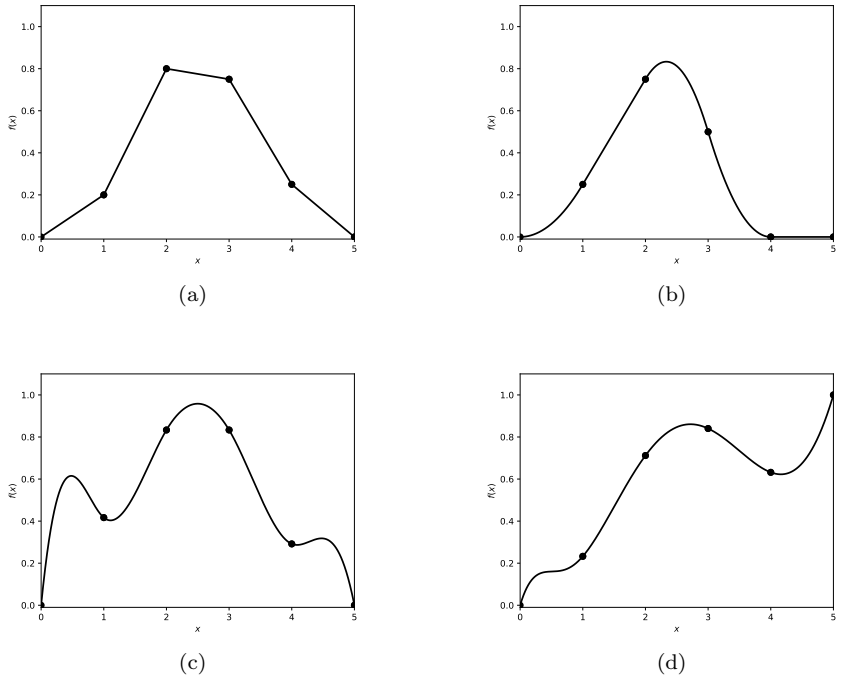


Fig. 1 Continuous piecewise polynomials and their epigraph (colored grey). The five pieces of the piecewise polynomial are linear in (a), quadratic in (b), cubic in (c), and quartic in (d).

models and presented a unifying framework for piecewise linear functions. To follow on these works, we continue by extending Def. 1 to handle lower semi-continuous piecewise polynomials.

3.3 Extension to lower semi-continuous piecewise polynomials

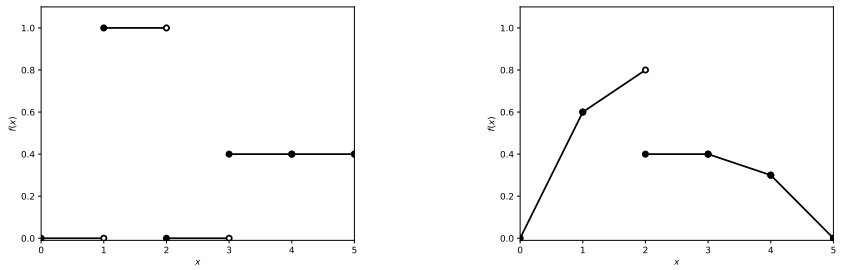
Below we provide a definition of PWPs that are not necessarily continuous. This allows us to analyze and integrate in the framework the subset of discontinuous PWPs that are *lower semi-continuous*. The extended definition lets us tie our PWP framework to the B-spline modeling framework, which facilitates construction of PWPs with predefined smoothness.

Before extending Definition 1 of PWPs, we consider the property of lower semi-continuity with some simple examples. Formally, a function f is lower semi-continuous if for any $\boldsymbol{x}_0 \in D$

$$\liminf_{\boldsymbol{x} \to \boldsymbol{x}_0} f(\boldsymbol{x}) \geq f(\boldsymbol{x}_0). \tag{21}$$

The importance of lower semi-continuity comes from the fact that the epigraph of a function is closed if and only if it is lower semi-continuous. It is hence a

requirement for the epigraph model in Sec. 3.2. Since a continuous function is lower semi-continuous, the requirement always holds for continuous PWPs. To illustrate this property, consider the two PWPs in Fig. 2. Both PWPs consist of five pieces defined on the intervals $[0, 1)$, $[1, 2)$, $[2, 3)$, $[3, 4)$, and $[4, 5]$; the open end of the intervals are marked with white-filled circles. The PWP in Fig. 2a is lower semi-continuous at $x = 2$, but not at points $x = 1$ and $x = 3$. Thus, it is not a lower semi-continuous PWP and its epigraph not closed. On the other hand, the PWP in Fig. 2b has one discontinuity ($x = 2$) at which it is lower semi-continuous. It is thus a lower semi-continuous function and its epigraph is closed. To summarize, we may model the PWP in Fig. 2b by its epigraph since it is a closed set, but not the epigraph of the PWP in Fig. 2a.



(a) (b)

Fig. 2 Discontinuous piecewise polynomials and their epigraph (colored grey). The five pieces of the piecewise polynomial are constant in (a), and linear in (b).

To allow discontinuities, Vielma, Ahmed, & Nemhauser [77] employed a characterization of the domain using *copolytopes* (sets defined by a finite set of strict and non-strict linear inequalities). Similarly, we use copolytopes to define not necessarily continuous PWPs as follows.

Definition 2 (Piecewise polynomial function) Let $D \in \mathbb{R}^d$ be a compact set. A (not necessarily continuous) function $f : D \subset \mathbb{R}^d \to \mathbb{R}$ is piecewise polynomial if and only if there exists a finite family of copolytopes $\mathit{\Pi}$ such that $D = \bigcup_{P \in \mathit{\Pi}} P$ and

$$f(\boldsymbol{x}) := \{ f_P(\boldsymbol{x}), \;\; \boldsymbol{x} \in P, \;\; \forall P \in \mathit{\Pi} \; , \tag{22}$$

where $f_P : P \subset \mathbb{R}^d \to \mathbb{R}$ and $f_P \in \mathbb{P}^d_p$ for all $P \in \mathit{\Pi}$, and some degree $p \in \mathbb{N}_0$.

With a minor adjustment to the continuous case, we may express the epigraph of a function f defined according to Def. 2, as the union of epigraphs of its pieces f_P. That is, we model

$$\text{epi}(f) = \bigcup_{P \in \mathit{\Pi}} \text{epi}(f_P), \tag{23}$$

where we now use $\text{epi}(f_P) := \left\{(\boldsymbol{x}, z) \in \bar{P} \times \mathbb{R} : f_P(\boldsymbol{x}) \leq z\right\}$ in which $\bar{P}$ is the closure of P. Recall that $\text{epi}(f)$ is closed if and only if f is lower semi-continuous.

Similar to continuous PWPs, we consider a partition of the domain on a rectilinear grid G. However, we now compose the grid of *left half-closed* boxes. A left half-closed box in $\mathit{\Pi}_G$, with leftmost vertex $v \in \mathcal{V}(G^-)$, is given as

$$P_v = \{\boldsymbol{x} \in \mathbb{R}^d : \pi^i_{k_i} \leq x_i < \pi^i_{k_i+1} \forall i \in \{1, \dots, d\} : k = I(v)\}. \tag{24}$$

Note that P_v, being a left half-closed box, is a special type of copolytope. Fig. 3 illustrates left half-closed boxes of dimensions one, two, and three, respectively.

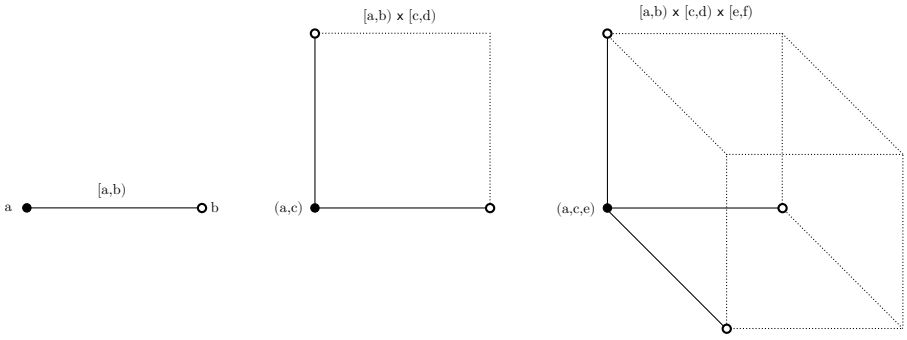


Fig. 3 Figure illustrating left half-closed boxes in one (line), two (rectangle), and three (cube) dimensions.

A technicality arises with this partitioning in that the rightmost boundaries of D are open, and hence D is open, which breaks compatibility with Def. 2. To close these boundaries we must require that the rightmost boundaries of the rightmost boxes are closed; i.e. in (24) we must replace $\pi^i_{k_i} \leq x_i < \pi^i_{k_i+1}$ with $\pi^i_{k_i} \leq x_i \leq \pi^i_{k_i+1}$ if $k_i + 1 = m_i$. The addition of this requirement on the partitioning ensures that the domain is closed and hence compatible with Def. 2.

4 B-splines

With piecewise linear functions one is often concerned with $\mathcal{C}^0$ continuity at intersections $\boldsymbol{x} \in P_1 \cap P_2$, where $P_1, P_2 \in \mathit{\Pi}$. For PWPs, $\mathcal{C}^1$, $\mathcal{C}^2$, or higher-order continuity at the intersections is an obtainable and often desired property. Def. 1 only guarantees $\mathcal{C}^0$ continuity, but does not exclude PWPs with higher order continuity. Below, we introduce the B-spline framework which enables construction of PWPs with a desired degree of smoothness.

A B-spline consists of overlapping polynomial basis functions, constructed under continuity constraints to control the order of continuity at the points

where the polynomials pieces connect (known as knots). The B-spline basis may be regarded as an extension of the Bernstein basis, generalizing the description of a single polynomial on a continuous interval to piecewise polynomial functions over partitioned domains, specified by a knot sequence [21]. It retains the non-negativity and partition-of-unity properties of the Bernstein basis; cf. [57,68].

In the following, we show how a B-spline may be brought to the form of piecewise polynomials in Def. 2, in which the polynomial pieces are non-overlapping. This procedure can be used to construct piecewise polynomial constraint functions from B-splines.

4.1 B-spline basis functions.

A B-spline of degree p in the variable x is expressed as

$$f(x) = \sum_{i=0}^{n-1} c_i N_{i,p}(x; \boldsymbol{t}), \tag{25}$$

where $\{c_i\}_{i=0}^{n-1} \in \mathbb{R}$ are coefficients and $\{N_{i,p}\}_{i=0}^{n-1}$ are B-spline basis functions defined as[2]

$$\begin{aligned} N_{i,p}(x; \boldsymbol{t}) &= \frac{x - t_i}{t_{i+p} - t_i} N_{i,p-1}(x; \boldsymbol{t}) + \frac{t_{i+p+1} - x}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(x; \boldsymbol{t}), \\ N_{i,0}(x; \boldsymbol{t}) &= \begin{cases} 1, & t_i \leq x < t_{i+1}, \\ 0, & \text{otherwise.} \end{cases} \end{aligned} \tag{26}$$

The parameter $\boldsymbol{t} = \{t_i\}_{i=0}^{n+p}$ is a non-decreasing sequence of real numbers, i.e. $t_0 \leq t_1 \leq \cdots \leq t_{n+p}$, referred to as a *knot sequence*. A knot sequence is said to be *regular* if it also abides to the requirement that $t_i < t_{i+p+1}$; i.e. the largest multiplicity of any knot is $p + 1$. Repetition of knots is used to control the continuity of the B-spline. At a single knot, $\mathcal{C}^{p-1}$ continuity is ensured. Generally, at a knot of multiplicity $m \geq p$, $\mathcal{C}^{p-m}$ continuity is ensured [7]. At knots of multiplicity $m = p + 1$, discontinuities may appear. Observe that a B-spline may be $\mathcal{C}^{p-1}$ continuous at any knot, even if the knot has multiplicity larger than one, as long as the coefficients $\boldsymbol{c}$ are chosen correctly. In the following, we will simplify the notation by omitting the dependence of the basis functions on $\boldsymbol{t}$ so that $N_{i,p}(x) \implies N_{i,p}(x; \boldsymbol{t})$.

The B-spline basis functions are piecewise polynomials by definition. Hence, the B-spline, being a linear combination of PWPs, is itself a piecewise polynomial function of degree equal to that of the basis functions. Furthermore, a B-spline f lies in the vector space $\mathbb{S}_p(\boldsymbol{t}) = \text{span}\{N_{i,p}\}_{i=0}^{n-1}$; i.e. f lies in the space spanned by degree p B-spline basis functions parametrized by the knot sequence $\boldsymbol{t}$. Note that $\mathbb{S}_p(\boldsymbol{t})$ is defined by the degree p and the partition of the domain, as prescribed by the knot sequence $\boldsymbol{t}$.

[2] Division by zero is handled by a '$0/0 = 0$' convention.

The following lemma gives some important properties of B-spline basis functions.

Lemma 2 (Properties of B-spline basis functions) *The following holds true for a set of degree p B-spline basis functions $\{N_{i,p}\}$:*

$$\begin{aligned} &N_{i,p}(x) \geq 0, \quad i = 1, \dots, n, \quad \forall x \in \mathbb{R} \\ &N_{i,p}(x) = 0, \quad \forall x \notin [t_i, t_{i+p+1}) \\ &\sum_{i=j-p}^{j} N_{i,p}(x) = 1, \quad \forall x \in [t_j, t_{j+1}) \end{aligned} \tag{27}$$

The three respective properties in Lemma 2 are referred to as the property of *nonnegativity*, *local support*, and *partition of unity*. These properties and the overlapping feature of the B-spline basis functions are displayed in Fig. 4.

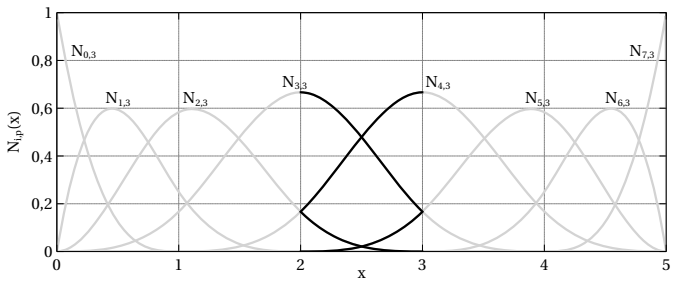


Fig. 4 B-spline basis functions for degree $p = 3$ and knot sequence $t = [0, 0, 0, 0, 1, 2, 3, 4, 5, 5, 5, 5]$. Basis functions that have support in the knot span $[t_5, t_6] = [2, 3]$, namely $\{N_{2,3}, N_{3,3}, N_{4,3}, N_{5,3}\}$, are accentuated.

From the properties in Lemma 2 it may not be immediately clear that B-spline basis functions are closely related to Bernstein polynomials. The relationship is nicely summarized by the following proposition.

Proposition 6 (Equivalence between B-spline and Bernstein basis) *The B-spline basis functions $\{N_{i,p}(x; \boldsymbol{t})\}_{i=0}^{p}$ in the variable $x \in [0, 1)$ defined by the regular knot sequence*

$$\boldsymbol{t} = \{\underbrace{0, \dots, 0}_{p+1}, \underbrace{1, \dots, 1}_{p+1}\}, \tag{28}$$

are equivalent to the Bernstein polynomials

$$N_{i,p}(x; \boldsymbol{t}) = \binom{p}{i} x^i (1 - x)^{p-i} = B_{i,p}(x). \tag{29}$$

Proof Cf. [57,68]. □

Another useful property of the B-spline basis is that it is invariant to an affine change of variable, as shown by Proposition 7 in Appendix B. By combining Proposition 6 and 7, we easily obtain the result in the following corollary, linking the B-spline basis function on $x \in [a, b)$ to the Bernstein basis on $y \in [0, 1)$.

Corollary 1 *Consider a B-spline basis* $\{N_{i,p}(x; \boldsymbol{t})\}_{i=0}^{p}$ *in the variable* $x \in [a, b)$ *defined by the regular knot sequence*

$$\boldsymbol{t} = \{\underbrace{a, \ldots, a}_{p+1}, \underbrace{b, \ldots, b}_{p+1}\}. \tag{30}$$

Then, for $i \in \{0, \ldots, p\}$ *and* $y \in [0, 1)$*, we have that* $N_{i,p}((b-a)y + a; \boldsymbol{t}) = B_{i,p}(y)$*, where* $B_{i,p}(y)$ *is the Bernstein basis.*

Proof The corollary follows immediately from Proposition 6 and 7. □

Finally, before proceeding to the multivariate case, we address the technicality that the support of a B-spline is restricted to the half-open domain $D_o = [t_0, t_{n+p})$. To close the domain we define

$$f(t_{n+p}) := \lim_{\substack{x \to t_{n+p} \\ x \leq t_{n+p}}} f(x), \tag{31}$$

and simply refer to the domain of a B-spline f as $D = \mathrm{cl}(D_o) = [t_0, t_{n+p}]$.

4.2 Multivariate B-splines

Multivariate B-splines, also known as tensor product B-splines, are constructed using the Kronecker product, analogous to the construction of multivariate polynomials in Section 2.2. Consider the multivariate B-spline basis functions

$$\boldsymbol{N}_p^d(\boldsymbol{x}; T) = \bigotimes_{j=1}^{d} \boldsymbol{N}_p(x_j; \boldsymbol{t}_j), \tag{32}$$

where $\boldsymbol{N}_p(x_j; \boldsymbol{t}_j) = [N_{i,p}(x_j; \boldsymbol{t}_j)]_{i=0}^{n_j - 1}$ is a vector of n_j degree p B-spline basis functions on the knot sequence $\boldsymbol{t}_j$ in the variable x_j, and $T = \{\boldsymbol{t}_j\}_{j=1}^{d}$ is the set of knot sequences. Similar to the description of multivariate polynomials, we have assumed without loss of generality that the univariate basis functions share a common degree p to ease the notation.

Using the basis in (32), we may construct a multivariate B-spline as

$$f(\boldsymbol{x}) = \sum_{i=0}^{N-1} c_i N_{i,p}^d(\boldsymbol{x}; T), \tag{33}$$

where the number of multivariate basis functions $N = n_1 \cdots n_d$. The space of B-splines spanned by the above basis is denoted $\mathbb{S}_p^d(T) = \mathbb{S}_p(\boldsymbol{t}_1) \times \cdots \times \mathbb{S}_p(\boldsymbol{t}_d)$, where the knot sequences T parametrize the partition of the domain.

From its construction by the Kronecker product of univariate B-spline basis functions, a special property of multivariate B-splines is that they are defined on a rectilinear grid of left half-closed boxes aligned to the variable axes. The partition is thus the same as the one we introduced for the piecewise polynmials in Sec. 3. This allows the space of piecewise polynomials $\mathbb{P}_p^d(G)$ to be represented in terms of B-splines $\mathbb{S}_p^d(T)$, given appropriate knot sequences T. This relationship was first stated for the univariate case in the Curry-Schoenberg theorem [14]. Using our notational framework, we restate this relationship for the multivariate case in the following lemma.

Lemma 3 (Relationship between B-splines and PWPs) *Given the space $\mathbb{P}_p^d(G)$ of piecewise degree p polynomials on a bounded domain D partitioned on a rectilinear grid G of left half-closed boxes. Then $\mathbb{S}_p^d(T) = \mathbb{P}_p^d(G)$, if the knot sequences $T = \{\boldsymbol{t}_i\}_{i=1}^d$ are given as*

$$\boldsymbol{t}_i = \{\underbrace{\pi_0^i, \dots, \pi_0^i}_{p+1}, \dots, \underbrace{\pi_k^i, \dots, \pi_k^i}_{p+1}, \dots, \underbrace{\pi_{m_i}^i, \dots, \pi_{m_i}^i}_{p+1}\} \quad \forall i \in \{1, \dots, d\}. \tag{34}$$

That is, the knot sequence $\boldsymbol{t}_i$ for variable x_i contains all partition points with multiplicity $p+1$.

Proof Consider the one-dimensional case first ($d = 1$). The specified knot sequence parametrizes the B-spline to have $p+1$ supported basis functions in each half-open knot interval $[\pi_k^1, \pi_{k+1}^1)$; the partition is thus equivalent to the partition of D by the grid G. According to Cor. 1, these basis functions are equivalent to the Bernstein basis functions, and hence spans the space $\mathbb{P}_p$ on D. For the general case of $d > 1$, we utilize the fact the multivariate B-spline basis functions are constructed using the Kronecker product in (32). D is thus partitioned into left half-open boxes $[\pi_{k_1}^1, \pi_{k_1+1}^1) \times \cdots \times [\pi_{k_d}^d, \pi_{k_d+1}^d)$, equivalent to the partition given by G. Furthermore, the multivariate B-spline basis is equivalent to the multivariate Bernstein basis since they both are constructed by the Kronecker product. It follows that the multivariate B-spline basis spans the space $\mathbb{P}_p^d$ on D, and by equivalence of the bases $\mathbb{S}_p^d(T) = \mathbb{P}_p^d(G)$. □

The knot sequences in Lemma 3 are special since all knots have multiplicity $p+1$. Consequently, any B-spline can be transformed to an equivalent B-spline with such knot sequences. This transformation is done by inserting knots into the knot sequences until all knots have multiplicity $p+1$, using a *knot insertion* method. The effect of raising the multiplicity of all knots to $p+1$ is that the B-spline is decomposed into a set of disjoint (non-overlapping) polynomial pieces, as illustrated by Figure 5. This procedure, provided in Appendix B, uses knot insertion to bring any B-spline to the PWP form in Def. 2.

In Lemma 3, the domain D is not required to be closed since the support of the B-spline is restricted to a half-open domain. However, a compact domain may be considered if the definition of the B-spline is extended to include support on the right boundary as described in Sec. 4.1, and the rightmost boxes in G are closed as in (31). To complete the comparison of B-splines with

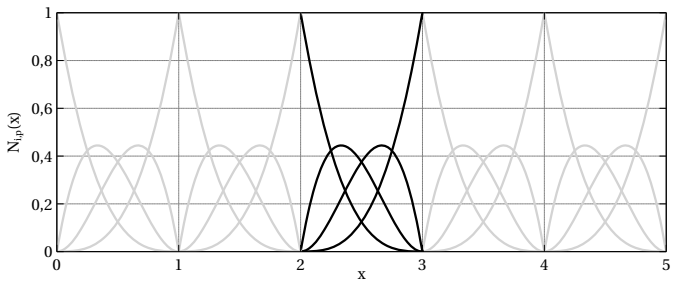


Fig. 5 B-spline basis functions for $p = 3$, $n = 8$, and knot sequence $\boldsymbol{t} = [0, 0, 0, 0, 1, 1, 1, 1, 2, 2, 2, 2, 3, 3, 3, 3, 4, 4, 4, 4, 5, 5, 5, 5]$. Each basis function spans only a single knot interval. In each knot interval, the cubic B-spline is a cubic polynomial expressed by four basis functions that are equivalent to the Bernstein polynomials (on the unit interval).

our definition of PWPs, we note that a B-spline is lower semi-continuous if no internal knot is repeated more than p times. Discontinuous B-splines, for which the multiplicity of one or more internal knots are $p + 1$, may still be lower semi-continuous.

5 Disjunctive formulations for piecewise polynomial functions

In this section, we apply disjunctive constraint formulation as a means of representing PWP constraints. Consider a piecewise polynomial $f : D \to \mathbb{R}$, defined in Def. 1, with a rectangular domain $D = \{\boldsymbol{x} : \boldsymbol{x}^L \leq \boldsymbol{x} \leq \boldsymbol{x}^U\} = \bigcup_{P \in \mathit{\Pi}} P \subset \mathbb{R}^d$. The epigraph of f, $\text{epi}(f) = \{(\boldsymbol{x}, z) \in D \times \mathbb{R} : f(\boldsymbol{x}) \leq z\}$, can be represented by the disjunction [1,34]

$$\bigvee_{P \in \mathit{\Pi}} \begin{bmatrix} \boldsymbol{x} \in P \\ f_P(\boldsymbol{x}) \leq z \end{bmatrix}. \tag{DP-1}$$

The disjunction DP-1 of $|\mathit{\Pi}|$ terms restricts $\boldsymbol{x}$ to exactly one polytope $P \in \mathit{\Pi}$, with each disjunctive term given by the epigraph $\text{epi}(f_P)$ of a piece f_P of f. The disjunction (DP-1) thus models $\text{epi}(f)$ as the union of epigraphs in (20). DP-1 is also a valid model for lower semi-continuous PWPs according to Def. 2, if we replace in each term the domain constraint with $\boldsymbol{x} \in \bar{P}$, where $\bar{P} = \mathbf{cl}(P)$. In both cases, DP-1 is a proper disjunction [2,75] in the sense that no single polytope covers the entire feasible region. Observe that Def. 1 ensures continuity in the overlap between the polytopes constituting the disjunction. In the derivations that follow we assume that f is a continuous PWP, but remark that the resulting formulations are also valid for lower semi-continuous PWPs subject to the mentioned domain-substitution.

When P is an n-orthotope, it may be expressed as $P = \{\boldsymbol{x} : \boldsymbol{x}_P^L \leq \boldsymbol{x} \leq \boldsymbol{x}_P^U\}$, where $\boldsymbol{x}_P^L$ and $\boldsymbol{x}_P^U$ denote lower and upper bounds on $\boldsymbol{x}$, respectively. The constraint $\boldsymbol{x} \in P$ then simplifies to the box constraints

$$\boldsymbol{x}_P^L \leq \boldsymbol{x} \leq \boldsymbol{x}_P^U. \tag{35}$$

The disjunction DP-1 contains a nonlinear, possibly nonconvex inequality for each term, thereby severely impeding the scalability of the formulation and hence its practical application. As a partial remedy, we may utilize that each polynomial f_P can be expressed as a linear combination of the basis functions spanning $\mathbb{P}_p^d$, that is, $f_P \in \mathbb{P}_p^d$, for all $P \in \mathit{\Pi}$. This salient characteristic allows us to exploit that the basis functions are independent of P, and hence be extracted outside the disjunction as a common set of nonlinearities. Together with the box constraints (35), this simplifies DP-1 to

$$\begin{aligned} &\bigvee_{P \in \mathit{\Pi}} \begin{bmatrix} \boldsymbol{x}_P^L \leq \boldsymbol{x} \leq \boldsymbol{x}_P^U \\ \boldsymbol{a}_P^\mathsf{T} \boldsymbol{\beta} \leq z \end{bmatrix}, \\ &\boldsymbol{\beta} = \boldsymbol{M}_p^d(\boldsymbol{x}), \end{aligned} \tag{DP-2}$$

where f_P is hence expressed as a linear combination of the $n = \dim(\mathbb{P}_p^d)$ multivariate monomial basis functions $\boldsymbol{\beta} = \boldsymbol{M}_p^d$. We stress that even though it is always possible to substitute nonlinearities with new variables to obtain linear disjunctions as in DP-2, the benefit comes solely from having common basis functions and hence reducing the number of nonlinear constraints.

Generally, DP-2 requires $n = \dim(\mathbb{P}_p^d) = (p+1)^d$ polynomial constraints to model a PWP, invariant to the discretization of the domain. Consider a PWP defined on a rectilinear grid of $|\mathit{\Pi}| = m^d$ boxes, resulting from a discretization with $m \geq 1$ intervals in each of the d variables. DP-1 requires m^d polynomial constraints to model this PWP. Thus, when $m > p+1$, $m^d > (p+1)^d$, and the formulation in DP-2 is likely preferable to DP-1. To summarize the above argument: modeling the polynomial function space $\mathbb{P}_p^d$ via its n basis functions, as opposed to modeling each of the $|\mathit{\Pi}|$ polynomial pieces separately, will generally result in fewer nonlinear constraints. It is worth noticing, however, that the exponential increase with d in the number of nonlinear constraints puts a practical limit on formulations derived from either DP-1 or DP-2.

Remark 1 Piecewise McCormick envelopes and other linear relaxations (e.g. [27,40]) for bilinear terms $f_P(x) = x_1 x_2$, with $(x_1, x_2) \in P$, can be derived from DP-1. This special case of DP-1, however, renders linear approximations, compared to DP-2 which is an exact formulation.

5.1 Reformulation of disjunction

Algorithmic approaches for mathematical programming problems with disjunctions either reformulates the disjunction to enable mixed-integer programming, or seeks to exploit the disjunctive constraints explicitly in a branch-and-bound or cutting-plane algorithm, possibly through combinations thereof [4, 73, 63]. Linear [2] and convex nonlinear disjunctions [9] may be reformulated either by its convex hull representations or by big-M reformulations. Pertaining to the linear disjunction in DP-2, the convex hull formulation [2] is at least as tight as big-M reformulations, though requiring more variables and

constraints. Big-M formulations, on the other hand, are known to be prone to the choice of the big-M parameters, and often yield weaker relaxations. For large nonlinear, possibly nonconvex disjunctive programming problems, it is important to keep the size of reformulation small, in which big-M reformulations may be advantageous [75]. A prerequisite for the numerical efficiency, however, is that strong big-M values can be derived.

5.2 Bounds on polynomial constraints

Deriving strong big-M values for DP-2 amounts to finding an upper bound on the constraint $\boldsymbol{a}_P^\mathsf{T}\boldsymbol{\beta} \leq z$ for some $P \in \mathit{\Pi}$. Suppose that $\boldsymbol{a}_P^\mathsf{T}\boldsymbol{\beta} \in [m_P^L, m_P^U]$ and that $z^L \leq z$ for any $\boldsymbol{x} \in D$. We may then define $M_P^U := m_P^U - z^L$ so that, for any $\boldsymbol{x} \in D$,

$$\boldsymbol{a}_P^\mathsf{T}\boldsymbol{\beta} - z \leq M_P^U. \tag{36}$$

To obtain a valid upper bound M_P^U, we must determine the values of m_P^U and z^L. We observe that any feasible solution must satisfy $f_P = \boldsymbol{a}_P^\mathsf{T}\boldsymbol{\beta} \leq z$ for some $P \in \mathit{\Pi}$. Thus, a valid lower bound on z is $z \geq z^L = \min\{m_P^L\}_{P\in\mathit{\Pi}}$, and we may rewrite the upper bound on the polynomial constraint to

$$M_P^U = m_P^U - \min\{m_P^L\}_{P\in\mathit{\Pi}}. \tag{37}$$

It is then obvious that computing M_P^U for each $P \in \mathit{\Pi}$ requires a lower and upper bound on all polynomials $\{f_P\}_{P\in\mathit{\Pi}}$.

Returning to DP-2, there is a subtlety due to the substitution of the nonlinearities that impedes the derivation of tight bounds on the polynomials. The issue is that the bounds on $f_P(\boldsymbol{x}) = \boldsymbol{a}_P^\mathsf{T}\boldsymbol{\beta} \in [m_P^L, m_P^U]$ must be valid for all $\boldsymbol{x} \in [\boldsymbol{x}^L, \boldsymbol{x}^U]$, not only for the subinterval $[\boldsymbol{x}_P^L, \boldsymbol{x}_P^U]$ on which the polynomial piece is modeled. This poses numerical problems since the piecewise polynomials may become prohibitively large on $[\boldsymbol{x}^L, \boldsymbol{x}^U]$, resulting in undesirably large bounds.

To solve this issue and obtain tighter bounds on the polynomials $\{f_P\}_{P\in\mathit{\Pi}}$, we utilize the procedure in Proposition 5 to perform a reparametrization before computing the polynomial bounds. Let $\boldsymbol{u} \in [\boldsymbol{0}, \boldsymbol{1}] \in \mathbb{R}^d$ and $f_P = \boldsymbol{a}_P^\mathsf{T}\boldsymbol{M}_p^d(\boldsymbol{x}) = \boldsymbol{c}_P^\mathsf{T}\boldsymbol{B}_p^d(\boldsymbol{u})$, where the coefficients of the reparametrized polynomial in Bernstein form are given as

$$\boldsymbol{c}_P^\mathsf{T} = \boldsymbol{a}_P^\mathsf{T} \bigotimes_{j=1}^{d} R_{p,j} Q_p. \tag{38}$$

Using the multivariate Bernstein basis $\boldsymbol{B}_p^d(\boldsymbol{u})$ for $\boldsymbol{0} \leq \boldsymbol{u} \leq \boldsymbol{1}$ enables reformulation of DP-2 to the disjunction

$$\begin{aligned} &\bigvee_{P\in\mathit{\Pi}} \begin{bmatrix} \boldsymbol{x}_P^L \leq \boldsymbol{x} \leq \boldsymbol{x}_P^U \\ \boldsymbol{x} = (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} + \boldsymbol{x}_P^L \\ \boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} \leq z \end{bmatrix}, \\ &\boldsymbol{\beta} = \boldsymbol{B}_p^d(\boldsymbol{u}), \\ &\boldsymbol{0} \leq \boldsymbol{u} \leq \boldsymbol{1}. \end{aligned} \tag{DP-3}$$

Within each term $P \in \mathit{\Pi}$ in DP-3, the variables $\boldsymbol{x}$ and $\boldsymbol{u}$ are linearly dependent. The reformulation DP-3 enables computation of bounds on the reparametrized polynomials. In parcticular, by invoking Proposition 4 we obtain $c_P^L \leq \boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} \leq c_P^U$, where $c_P^L = \min\{c_{i,P}\}_{i=0}^{n-1}$ and $c_P^U = \max\{c_{i,P}\}_{i=0}^{n-1}$. This allows us to redefine

$$M_P^U := c_P^U - \min\{c_P^L\}_{P\in\mathit{\Pi}}, \tag{39}$$

and thereby obtain a valid upper bound $\boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z \leq M_P^U$ for all $\boldsymbol{u} \in [\boldsymbol{0}, \boldsymbol{1}]$ and $P \in \mathit{\Pi}$.

Finally, we note that the bound in (39) corresponds to an upper bound on $f_P - z$ for $\boldsymbol{x} \in [\boldsymbol{x}_P^L, \boldsymbol{x}_P^U] \subset D$. This upper bound must be less than or equal to the upper bound on $f_P - z$ for $\boldsymbol{x} \in D = [\boldsymbol{x}^L, \boldsymbol{x}^U]$, as required for a reformulation of DP-2.

Remark 2 The special case of an equality constraint $\boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} = z$ can be handled by writing $0 \leq \boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z \leq 0$. Using the same arguments as above we obtain the bounds

$$\begin{aligned} \boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z &\leq M_P^U, \\ \boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z &\geq M_P^L, \end{aligned} \tag{40}$$

where $M_P^L := c_P^L - \max\{c_P^U\}_{P\in\mathit{\Pi}}$.

6 MINLP formulations for piecewise polynomial functions

In this section, we exploit the disjunctive formulation DP-3 of PWP functions with the associated strengthening procedure of big-M values described in Section 5.2 to construct mixed integer programming formulations. A basic MINLP formulation is obtained by performing elementary big-M reformulation of DP-3, yielding

$$\begin{aligned}
&\boldsymbol{x} \leq (\boldsymbol{x}_P^U - \boldsymbol{x}^U)y_P + \boldsymbol{x}^U, && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{x} \geq (\boldsymbol{x}_P^L - \boldsymbol{x}^L)y_P + \boldsymbol{x}^L, && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \leq (\boldsymbol{x}^U - \boldsymbol{x}_P^L)(1 - y_P), && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \geq (\boldsymbol{x}^L - \boldsymbol{x}_P^U)(1 - y_P), && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z \leq M_P^U(1 - y_P), && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{\beta} = \boldsymbol{B}_p^d(\boldsymbol{u}), \\
&\sum_{P\in\mathit{\Pi}} y_P = 1, \\
&\boldsymbol{0} \leq \boldsymbol{u} \leq \boldsymbol{1}, \\
&y_P \in \{0, 1\}, && \forall P \in \mathit{\Pi},
\end{aligned} \tag{MINLP-BM}$$

where $M_P^U = c_P^U - \min\{c_P^L\}_{P\in\mathit{\Pi}}$, and a binary variable y_P is introduced for each $P \in \mathit{\Pi}$. MINLP-BM has $n = \dim(\mathbb{P}_p^d)$ nonlinear constraints; all other

constraints are linear. The formulation requires $d + n$ continuous auxiliary variables $\boldsymbol{u}$ and $\boldsymbol{\beta}$, and $|\Pi|$ binary variables $\{y_P\}_{P\in\Pi}$.

Subsequently, we derive three variants of the MINLP-BM formulation before we investigate their efficiency in a numerical study.

6.1 Logarithmic number of binary variables

Compared to MINLP-BM, requiring a linear number of binary variables with respect to the number of pieces in the PWP constraint, we may obtain a formulation with logarithmic number of binary variables for PWP constraints by means of the MILP modeling technique proposed by [77,78] for piecewise linear functions. To this end, each polytope $P \in \Pi$ is identified by a binary vector in $\{0,1\}^{\lceil \log_2 |\Pi| \rceil}$ through an injective function $J : \Pi \to \{0,1\}^{\lceil \log_2 |\Pi| \rceil}$. We introduce $\lceil \log_2 |\Pi| \rceil$ binary variables $\boldsymbol{y} \in \{0,1\}^{\lceil \log_2 |\Pi| \rceil}$ to enforce the constraints in the disjunction when $\boldsymbol{y} = J(P)$. The resulting formulation has no requirement on the family of polytopes Π, and is given by

$$
\begin{aligned}
&\boldsymbol{x} \le (\boldsymbol{x}_P^U - \boldsymbol{x}^U)\nu_P + \boldsymbol{x}^U, && \forall P \in \Pi,\\
&\boldsymbol{x} \ge (\boldsymbol{x}_P^L - \boldsymbol{x}^L)\nu_P + \boldsymbol{x}^L, && \forall P \in \Pi,\\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \le (\boldsymbol{x}^U - \boldsymbol{x}_P^L)(1-\nu_P), && \forall P \in \Pi,\\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \ge (\boldsymbol{x}^L - \boldsymbol{x}_P^U)(1-\nu_P), && \forall P \in \Pi,\\
&\boldsymbol{c}_P^{\mathsf{T}}\boldsymbol{\beta} - z \le M_P^U(1-\nu_P), && \forall P \in \Pi,\\
&\boldsymbol{\beta} = \boldsymbol{B}_p^d(\boldsymbol{u}),\\
&\sum_{P\in\Pi} \nu_P = 1, && \text{(MINLP-LOG)}\\
&\sum_{P\in\Pi^1(J,l)} \nu_P \le y_l, && \forall l \in L(\Pi),\\
&\sum_{P\in\Pi^0(J,l)} \nu_P \le (1-y_l), && \forall l \in L(\Pi),\\
&\boldsymbol{0} \le \boldsymbol{u} \le \boldsymbol{1},\\
&y_l \in \{0,1\}, && \forall l \in L(\Pi),
\end{aligned}
$$

where $J : \Pi \to \{0,1\}^{\lceil \log_2 |\Pi| \rceil}$ is any injective function, $\Pi^0(J,l) := \{P \in \Pi : B(P)_l = 0\}$, $\Pi^1(J,l) := \{P \in \Pi : J(P)_l = 1\}$, and $L(\Pi) := \{1,\dots,\lceil \log_2 |\Pi| \rceil\}$. MINLP-LOG has $\lceil \log_2 |\Pi| \rceil$ binary variables $\{y_l\}_{l\in L(\Pi)}$. The reduction in number of binary variables comes at the cost of introducing $|\Pi|$ continuous variables $\{\nu_P\}_{P\in\Pi}$ and $2|L(\Pi)|$ additional linear constraints.

6.2 Bernstein cuts

The MINLP-BM formulation can be augmented with polyhedral cuts that may strengthen its relaxation and expedite the solution process. Here, we include cuts based on the inherent properties of the Bernstein polynomials similar to [23]. The nonnegativity of Bernstein basis functions in Lemma 1 and the identity in (12) are explicitly stated via *nonnegativity cuts* and *partition-of-unity cuts*, respectively. The cuts are expressed as $\boldsymbol{\beta} \geq \mathbf{0}$ and $\mathbf{1}^\mathsf{T}\boldsymbol{\beta} = 1$, where $\mathbf{0}$ and $\mathbf{1}$ are vectors of n zeros and ones.

$$
\begin{aligned}
&\boldsymbol{x} \leq (\boldsymbol{x}_P^U - \boldsymbol{x}^U)y_P + \boldsymbol{x}^U, && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{x} \geq (\boldsymbol{x}_P^L - \boldsymbol{x}^L)y_P + \boldsymbol{x}^L, && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \leq (\boldsymbol{x}^U - \boldsymbol{x}_P^L)(1 - y_P), && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \geq (\boldsymbol{x}^L - \boldsymbol{x}_P^U)(1 - y_P), && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z \leq M_P^U(1 - y_P), && \forall P \in \mathit{\Pi}, \\
&\boldsymbol{\beta} = \boldsymbol{B}_p^d(\boldsymbol{u}),\ \boldsymbol{\beta} \geq \mathbf{0},\ \mathbf{1}^\mathsf{T}\boldsymbol{\beta} = 1, \\
&\sum_{P \in \mathit{\Pi}} y_P = 1, \\
&\mathbf{0} \leq \boldsymbol{u} \leq \mathbf{1}, \\
&y_P \in \{0, 1\}, && \forall P \in \mathit{\Pi}.
\end{aligned}
\tag{MINLP-CUT}
$$

Compared to MINLP-BM, MINLP-CUT has $n + 1$ additional constraints for the linear cuts.

6.3 Expanded Bernstein basis

The preceding formulations do not utilize the structure of the tensor product basis functions in $\boldsymbol{B}_p^d$, cf. (9). The multivariate basis is formed by the product of all the univariate bases, resulting in constraints with polynomials of degree dp. In the following formulation, the multivariate basis is expanded to exploit the inherent structure due to the tensor product. Specifically, the univariate bases of degree p are assigned to continuous auxiliary variables. The multivariate basis is then formed as the product of these auxiliary variables, which results in polynomial constraints of degree d. The maximum degree of any polynomial in the set of constraints is thus reduced to $\max\{d, p\}$. Another benefit of this expansion is that it permits additional polyhedral cuts on the univariate basis functions.

To expand the multivariate Bernstein basis we introduce $p + 1$ auxiliary variables $\boldsymbol{\xi}_i = \boldsymbol{B}_p(u_i)$ for $i \in K = \{1, \ldots, d\}$. The total number of auxiliary variables is $d(p + 1)$. Using these variables we may express the multivariate basis as

$$
\boldsymbol{\beta} = \bigotimes_{i=1}^{d} \boldsymbol{\xi}_i. \tag{41}
$$

The auxiliary variables represent univariate Bernstein basis functions. We may add nonnegativity and partition-of-unity cuts, as was done for the multivariate basis MINLP-CUT. These cuts are given as

$$\begin{aligned} \boldsymbol{\xi}_i &\geq \mathbf{0}, \quad \forall i \in K, \\ \mathbf{1}^\mathsf{T}\boldsymbol{\xi}_i &= 1, \quad \forall i \in K, \end{aligned} \tag{42}$$

where $\mathbf{0}$ and $\mathbf{1}$ is a vector of $p+1$ zeros and ones, respectively.

The resulting formulation, with the expanded Bernstein basis and additional cuts, is given below.

$$\begin{aligned}
&\boldsymbol{x} \leq (\boldsymbol{x}_P^U - \boldsymbol{x}^U)y_P + \boldsymbol{x}^U, && \forall P \in \Pi, \\
&\boldsymbol{x} \geq (\boldsymbol{x}_P^L - \boldsymbol{x}^L)y_P + \boldsymbol{x}^L, && \forall P \in \Pi, \\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \leq (\boldsymbol{x}^U - \boldsymbol{x}_P^L)(1 - y_P), && \forall P \in \Pi, \\
&\boldsymbol{x} - (\boldsymbol{x}_P^U - \boldsymbol{x}_P^L)\boldsymbol{u} - \boldsymbol{x}_P^L \geq (\boldsymbol{x}^L - \boldsymbol{x}_P^U)(1 - y_P), && \forall P \in \Pi, \\
&\boldsymbol{c}_P^\mathsf{T}\boldsymbol{\beta} - z \leq M_P^U(1 - y_P), && \forall P \in \Pi, \\
&\boldsymbol{\xi}_i = \boldsymbol{B}_p(u_i),\ \boldsymbol{\xi}_i \geq \mathbf{0},\ \mathbf{1}^\mathsf{T}\boldsymbol{\xi}_i = 1, && \forall i \in K, \\
&\boldsymbol{\beta} = \bigotimes_{i=1}^{d} \boldsymbol{\xi}_i,\ \boldsymbol{\beta} \geq \mathbf{0},\ \mathbf{1}^\mathsf{T}\boldsymbol{\beta} = 1, && \\
&\sum_{P \in \Pi} y_P = 1, && \\
&\mathbf{0} \leq \boldsymbol{u} \leq \mathbf{1}, && \\
&y_P \in \{0, 1\}, && \forall P \in \Pi.
\end{aligned} \tag{MINLP-EXP}$$

Compared to MINLP-CUT, MINLP-EXP has $d(p+1)$ additional auxiliary variables, $d(p+1)+d$ additional linear cuts, and $d(p+1)$ extra nonlinear constraints representing the univariate basis functions.

6.4 Summary of formulations

The size of the formulations in terms of number of variables and constraints are summarized in Table 1.

Note that all formulations, except MINLP-EXP, have exactly $n = \dim(\mathbb{P}_p^d)$ nonlinear equality constraints. These constraints model the Bernstein basis functions that span $\mathbb{P}_p^d$, and are nonconvex since the Bernstein basis functions are polynomials of degree dp. MINLP-EXP has an additional $d(p+1)$ polynomial constraints of degree p. These constraints enable a reduction in the degree of the n polynomial constraints that model $\boldsymbol{\beta}$ to d (compared to dp for the other formulations).

Table 1 Size of formulations.

Formulation*	# Constraints		# Variables	
	Linear	Nonlin.	Cont.	Binary
MINLP-BM	$5\lvert\mathit{\Pi}\rvert + 2d + 1$	n	$n + 2d + 1$	$\lvert\mathit{\Pi}\rvert$
MINLP-LOG	$5\lvert\mathit{\Pi}\rvert + 2d + 1 + 2\lceil\log_2 \lvert\mathit{\Pi}\rvert\rceil$	n	$n+2d+1+\lvert\mathit{\Pi}\rvert$	$\lceil\log_2 \lvert\mathit{\Pi}\rvert\rceil$
MINLP-CUT	$5\lvert\mathit{\Pi}\rvert+2d+2+n$	n	$n + 2d + 1$	$\lvert\mathit{\Pi}\rvert$
MINLP-EXP	$5\lvert\mathit{\Pi}\rvert + d(p + 4) + n + 2$	$n + dp + d$	$n + dp + 3d + 1$	$\lvert\mathit{\Pi}\rvert$

* $n = \dim(\mathbb{P}_p^d)$.

7 Numerical study

To benchmark the performance of the proposed MINLP formulations, three sets of test problems were created by randomly generating cubic splines. The three sets contain problems with a piecewise polynomial objective function in one, two, and three variables, respectively. For the monovariable problems (*random1d*), the variable is discretized into 10 intervals, leading to 10 polynomial pieces. The set of problems in two variables (*random2d*) have an objective function defined on a 10×10 grid, leading to 100 polynomial pieces. Finally, the problems with three variables (*random3d*) have an objective function defined on a $6 \times 6 \times 6$ grid, for a total of 216 n-orthotopes. Thus, the hardest problems (*random3d*) contain 216 polynomial pieces of degree $dp = 3 \times 3 = 9$. Properties of the sets of test problems are summarized in Table 2.

Table 2 Test sets of piecewise polynomial problems.

Problem	d	p	pd	$\lvert\mathit{\Pi}\rvert$	Number of instances
random1d	1	3	3	10	100
random2d	2	3	6	100	100
random3d	3	3	9	216	100

The problems are on the epigraph form $\min_{\boldsymbol{x},z} \left\{ z \ : \ f(\boldsymbol{x}) \leq z, \ \boldsymbol{x} \in D \subset \mathbb{R}^d \right\}$, where $f : D \to \mathbb{R}$ is a piecewise polynomial function. Continuous cubic B-splines with equidistant knots are constructed by randomly drawing coefficients from a Gaussian distribution with zero mean and unity standard deviation, that is $c_i \sim \mathcal{N}(0, 1), \ \forall i = 0, \ldots, n-1$. Figure 6 shows one of the generated univariate cubic splines. Using the procedure in Sec. B, the B-spline is transformed into a piecewise polynomial f compatible with Def. 2.

The four formulations MINLP-BM, MINLP-LOG, MINLP-CUT, and MINLP-EXP were solved using the global optimization solver BARON [64]. We compare the computational performance of the proposed MINLP formulations with the two MIQCP formulations for spline-constrained problems proposed in [23].

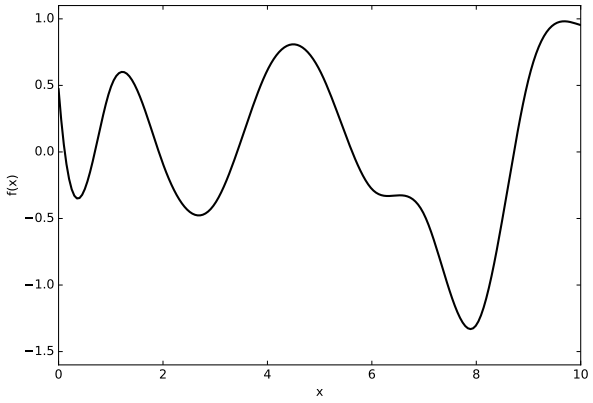


Fig. 6 A cubic spline with randomly generated coefficients. The function has several local minima. The global minimum for $x \in [0, 10]$ lies close to $x = 8$.

These two latter formulations, denoted MIQCP and MIQCP-CUT, were also solved using BARON. We further include the results from solving the test problems using the special-purpose spline solver CENSO, which solves spline constrained problems as NLPs using a spatial branch-and-bound algorithm [25].

All solvers were run with an absolute ϵ-convergence termination criteria of $\epsilon = 1 \cdot 10^{-6}$. Remaining settings were left at default values. The problems were solved on a laptop computer equipped with an Intel Core i7-5600U 2.6 GHz processor and 8 GB of RAM memory.

Results in terms of solve times are shown in Figures 7-9. The box plots show the median (in red) inside the box extended from the first to third quartile. The whiskers extending from the box represent the lower and upper value still within the lower and upper 1.5 interquartile range, respectively. An outlying value is indicated with a grey circle. A complete table of results can be found in Appendix C, Table 3.

Among the MINLP formulations, the MINLP-EXP formulation performed best overall. Comparing mean and median solve times, this formulation outperformed the MIQCP formulations on all three test sets. On the problems with three variables, however, the MIQCP-CUT formulation had only a slightly higher mean solve time than MINLP-EXP. CENSO [25] had the lowest solve time on all problems, and a mean solve time one-two orders of magnitude lower than MINLP-EXP.

Comparing the MINLP-BM and MINLP-LOG, it seems clear that reducing the number of binary variables at the cost of introducing additional constraints and variables yields little in terms of performance on the test sets. However, the introduction of cuts in MINLP-CUT and MINLP-EXP, significantly reduces the solve times. The benefit of utilizing the structure of the Kronecker product

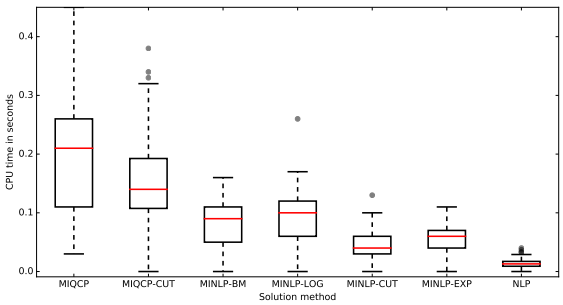


Fig. 7 A boxplot showing the results for random1d.

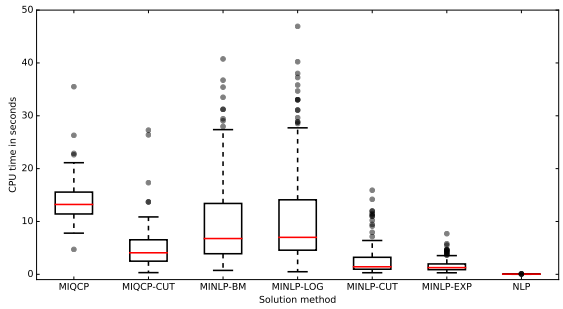


Fig. 8 A boxplot showing the results for random2d.

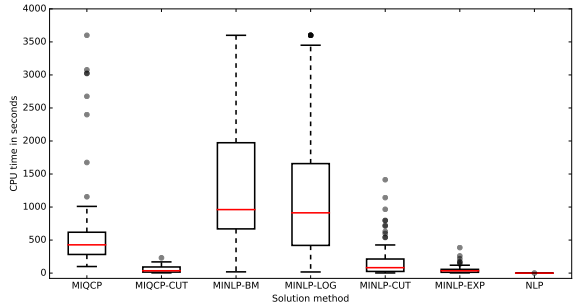


Fig. 9 A boxplot showing the results for random3d.

by modeling the univariate bases separately becomes evident for the 2-D and 3-D PWPs, as expected.

It seems that the main difficulty in solving these problems lies in the modeling of the polynomial basis $\mathbb{P}_p^d$, rather than the dichotomy of the grid. The

$d(p+1)$ additional nonlinear constraints in MINLP-EXP, which contains polynomials of degree p, made it possible to lower the degree of the remaining n nonlinear constraints that model the multivariate basis from dp to d. This had a positive effect on the solve times. In the extreme case, one may factorize the nonlinear constraints until all nonlinear constraints are bilinear, as done in the MIQCP formulations from [23].

8 Concluding remarks

This paper has presented a MIP modeling framework for solving mathematical programs with continuous and semi-continuous PWPs. Such problems have previously been restricted from mathematical optimization due lack of solver support. By enabling use of standard optimization-modeling software, the implementation efforts for spline- and PWP-constrained optimization problems are significantly reduced. Moreover, it facilitates exploitation of the advancement in global optimization solvers for MINLPs and MIQCPs [64, 49, 50, 79].

There is still a significant gap to the performance of the special-purpose spline-based solver CENSO, indicating that improvements can be made to the formulations herein. Ideas from polynomial optimization, such as sum-of-squares and semidefinite program relaxations [42, 81] and the Bernstein branch-and-prune algorithm [52], may be explored in combination with the proposed framework to improve the modeling of $\mathbb{P}_p^d$. We also emphasize that the formulations presented are exact; approximations of these formulations may hence aid efforts put in reducing the computational demand of solving PWP optimization problems

Finally, our strategy of modeling a PWP via its epigraph provides a general framework compatible with semi-continuous piecewise polynomials and B-splines, encompassing a broad set of spline modeling techniques. The modeling approach in this paper may hence enable formulation of a broad set of spline models in optimization problems, beyond the B-spline employed in this framework.

References

1. Balas, E.: Disjunctive Programming. Annals of Discrete Mathematics **5**, 3–51 (1979)
2. Balas, E.: Disjunctive programming and a hierarchy of relaxations for discret optimization problems. SIAM Journal on Algebraic and Discrete Methods **6**(3), 466–486 (1985)
3. Beale, E.M.L., Tomlin, J.A.: Special facilities in a general mathematical programming system for non-convex problems using ordered sets of variables. OR **69**(447-454), 99 (1970)
4. Beaumont, N.: An algorithm for disjunctive programs. European Journal of Operational Research **48**(3), 362–371 (1990)
5. Bellman, R.E.: Adaptive Control Processes: A Guided Tour. Princeton university press (1961)
6. Biegler, L.T.: Simultaneous Methods for Dynamic Optimization. In: Nonlinear Programming: Concepts, Algorithms, and Applications to Chemical Processes, chap. 10, pp. 287–324. SIAM (2010)

7. de Boor, C.: A Practical Guide to Splines, revised ed edn. Springer-Verlag New York, Inc. (2001)
8. Bragalli, C., D'Ambrosio, C., Lee, J., Lodi, A., Toth, P.: On the optimal design of water distribution networks: a practical MINLP approach. Optimization and Engineering **13**(2), 219–246 (2012)
9. Ceria, S., Soares, J.: Convex programming for disjunctive convex optimization. Mathematical Programming **86**(3), 595–614 (1999)
10. Chen, C., Mangasarian, O.L.: A class of smoothing functions for nonlinear and mixed complementarity problems. Computational Optimization and Applications **5**(2), 97–138 (1996)
11. Chen, X.: Smoothing methods for nonsmooth, nonconvex minimization. Mathematical Programming **134**(1), 71–99 (2012)
12. Conn, A.R., Mongeau, M.: Discontinuous piecewise linear optimization. Mathematical Programming **80**(3), 315–380 (1998)
13. Croxton, K.L., Gendron, B., Magnanti, T.L.: A Comparison of Mixed-Integer Programming Models for Nonconvex Piecewise Linear Cost Minimization Problems. Management Science **49**(9), 1268–1273 (2003)
14. Curry, H.B., Schoenberg, I.J.: On Pólya frequency functions IV: the fundamental spline functions and their limits. Journal d'Analyse Mathématique **17**(1), 71–107 (1966)
15. Dantzig, G.B.: On the significance of solving linear programming problems with some integer variables. Econometrica **28**(1), 30–44 (1960)
16. Demeulenaere, B., Pipeleers, G., De Caigny, J., Swevers, J., De Schutter, J., Vandenberghe, L.: Optimal splines for rigid motion systems: A convex programming framework. ASME Journal of Mechanical Design **131**(10), 101,004–101,004–11 (2009)
17. DeVore, R.A.: Nonlinear approximation. Acta Numerica **7**(November 2008), 51 (1998)
18. Egerstedt, M., Martin, C.: Control Theoretic Splines: Optimal Control, Statistics, and Path Planning. Princeton Series in Applied Mathematics (2009)
19. Eilers, P.H., Marx, B.D.: Flexible Smoothing with B-splines and Penalties. Statistical Science **11**(2), 89–102 (1996)
20. Evgeniou, T., Pontil, M., Poggio, T.: Regularization networks and support vector machines. Advances in Computational Mathematics **13**(1), 1–50 (2000). DOI 10.1023/a:1018946025316
21. Farouki, R.T.: The Bernstein polynomial basis: A centennial retrospective. Computer Aided Geometric Design **29**(6), 379–419 (2012)
22. Geißler, B., Martin, A., Morsi, A., Schewe, L.: Using piecewise linear functions for solving MINLPs. In: J. Lee, S. Leyffer (eds.) Mixed Integer Nonlinear Programming, *The IMA Volumes in Mathematics and its Applications*, vol. 154, pp. 287–314. Springer New York (2012)
23. Grimstad, B.: A MIQCP formulation for B-spline constraints. Optimization Letters (2017). DOI 10.1007/s11590-017-1190-1
24. Grimstad, B., Foss, B., Heddle, R., Woodman, M.: Global optimization of multiphase flow networks using spline surrogate models. Computers & Chemical Engineering **84**, 237–254 (2016)
25. Grimstad, B., Sandnes, A.: Global optimization with spline constraints: a new branch-and-bound method based on B-splines. Journal of Global Optimization **65**(3), 401–439 (2016)
26. Grimstad, B., et al.: SPLINTER: a library for multivariate function approximation with splines. http://github.com/bgrimstad/splinter (2015). Accessed: 2015-05-16
27. Hasan, M.M.F., Karimi, I.: Piecewise linear relaxation of bilinear programs using bivariate partitioning. AIChE Journal **56**(7), 1880–1893 (2010)
28. Hastie, T., Tibshirani, R., Friedman, J.: The Elements of Statistical Learning, *Springer Series in Statistics*, vol. 1, 2 edn. Springer New York, New York, NY (2009)
29. Hiriart-Urruty, J.B., Lemarechal, C.: Convex Analysis and Minimization Algorithms *II* - Advanced Theory and Bundle Methods. Springer-Verlag, Berlin (1993)
30. Hofmann, T., Schölkopf, B., Smola, A.J.: Kernel methods in machine learning. Annals of Statistics **36**(3), 1171–1220 (2008)
31. Höllig, K.: Finite Element Methods with B-Splines. Society for Industrial and Applied Mathematics, Philadelphia, PA (2003)

32. Holmberg, K.: Solving the staircase cost facility location problem with decomposition and piecewise linearization. European Journal of Operational Research **75**(1), 41–61 (1994)
33. Jahanshahi, E., Grimstad, B., Foss, B.: Spline fluid models for optimization. In: Proc. IFAC Symp. on Dynamics and Control of Process Systems, pp. 400–405. Trondheim, Norway (2016)
34. Jeroslow, R., Lowe, J.: Modelling with integer variables. Mathematical Programming Studies **22**, 167–184 (1984)
35. Jeroslow, R.G.: Representability in mixed integer programming, I: Characterization results. Discrete Applied Mathematics **17**, 223–243 (1987)
36. Jeroslow, R.G.: Representability of functions. Discrete Applied Mathematics **23**(2), 125–137 (1989)
37. Jones, D.: A Taxonomy of Global Optimization Methods Based on Response Surfaces. Journal of Global Optimization **21**(4), 345–383 (2001)
38. Keha, A.B., de Farias Jr, I.R., Nemhauser, G.L.: A branch-and-cut algorithm without binary variables for nonconvex piecewise linear optimization. Operations research **54**(5), 847–858 (2006)
39. Knudsen, B.R., Foss, B.: Shut-in based production optimization of shale-gas systems. Computers & Chemical Engineering **58**, 54–67 (2013)
40. Kolodziej, S., Castro, P.M., Grossmann, I.E.: Global optimization of bilinear programs with a multiparametric disaggregation technique. Journal of Global Optimization **57**(4), 1039–1063 (2013)
41. Kosmidis, V.D., Perkins, J.D., Pistikopoulos, E.N.: A mixed integer optimization formulation for the well scheduling problem on petroleum fields. Computers & Chemical Engineering **29**(7), 1523–1541 (2005)
42. Lasserre, J.B.: Global Optimization with Polynomials and the Problem of Moments. SIAM Journal on Optimization **11**(3), 796–817 (2001)
43. Lee, J., Wilson, D.: Polyhedral methods for piecewise-linear functions I: The lambda method. Discrete Applied Mathematics **108**(3), 269–285 (2001)
44. Li, W.: A conjugate gradient method for the unconstrained minimization of strictly convex quadratic splines. Mathematical Programming **72**(1), 17–32 (1996). DOI 10.1007/BF02592329
45. Lin, C., Viviani, G.: Hierarchical economic dispatch for piecewise quadratic cost functions. IEEE Transactions on Power Apparatus and Systems **PAS-103**(6), 1170–1175 (1984)
46. Luo, Y.: Simulation-based optimization over discrete sets with noisy constraints. Ph.D. thesis, University of Miami (2011)
47. Lyche, T., Cohen, E., Mørken, K.: Knot line refinement algorithms for tensor product B-spline surfaces. Computer Aided Geometric Design **2**(1-3), 133–139 (1985). DOI 10.1016/0167-8396(85)90016-0
48. Mermelstein, S.P., Acar, M.: Optimising cam motion using piecewise polynomials. Engineering with Computers **19**(4), 241–254 (2003)
49. Misener, R., Floudas, C.A.: GloMIQO: Global mixed-integer quadratic optimizer. Journal of Global Optimization **57**(1), 3–50 (2013)
50. Misener, R., Floudas, C.A.: Antigone: Algorithms for continuous / integer global optimization of nonlinear equations. Journal of Global Optimization **59**(2), 503–526 (2014)
51. Natali, J.a.M., Pinto, J.M.: Piecewise polynomial interpolations and approximations of one-dimensional functions through mixed integer linear programming. Optimization Methods and Software **24**(4-5), 783–803 (2009). DOI 10.1080/10556780802614507
52. Nataraj, P.S.V., Arounassalame, M.: Constrained global optimization of multivariate polynomials using Bernstein branch and prune algorithm. Journal of Global Optimization **49**, 185–212 (2011)
53. Nesterov, Y.: Smooth minimization of non-smooth functions. Mathematical Programming **152**, 127–152 (2005)
54. Padberg, M.: Approximating separable nonlinear functions via mixed zero-one programs. Operations Research Letters **27**(1), 1–5 (2000)
55. Park, J., Kim, Y., Eom, I., Lee, K.: Economic load dispatch for piecewise quadratic cost function using Hopfield neural network. IEEE Transactions on Power Systems **8**(3), 1030–1038 (1993)

56. Patrinos, P., Sarimveis, H.: Convex parametric piecewise quadratic optimization : Theory , Algorithms and Control Applications. Automatica **47**(8), 1770–1777 (2011)
57. Piegl, L.A., Tiller, W.: The NURBS book. Springer (1997)
58. Prandoni, P., Vetterli, M.: Approximation and compression of piecewise smooth functions. Philosophical Transactions: Mathematical, Physical and Engineering Sciences **357**(1760), 2573–2591 (1999)
59. Queipo, N.V., Haftka, R.T., Shyy, W., Goel, T., Vaidyanathan, R., Kevin Tucker, P.: Surrogate-based analysis and optimization. Progress in Aerospace Sciences **41**(1), 1–28 (2005)
60. Ratnam, E.L., Weller, S.R., Kellett, C.M., Murray, A.T.: Residential load and rooftop PV generation: an Australian distribution network dataset. International Journal of Sustainable Energy **36**(8), 787–806 (2017). DOI 10.1080/14786451.2015.1100196
61. Royset, J.O.: Approximations and solution estimates in optimization. Mathematical Programming (2017). DOI 10.1007/s10107-017-1165-0
62. Royset, J.O., Wets, R.J.B.: On univariate function identification problems. Mathematical Programming (2017). DOI 10.1007/s10107-017-1130-y
63. Ruiz, J.P., Grossmann, I.E.: Global optimization of non-convex generalized disjunctive programs : a review on reformulations and relaxation techniques. Journal of Global Optimization **67**(1), 43–58 (2017). DOI 10.1007/s10898-016-0401-0
64. Sahinidis, N.V.: BARON: A general purpose global optimization software package. Journal of Global Optimization **8**(2), 201–205 (1996)
65. Schmid, M., Hothorn, T.: Boosting additive models using component-wise P-Splines. Computational Statistics and Data Analysis **53**(2), 298–311 (2008)
66. Scholtes, S.: Nonconvex Structures in Nonlinear Programming. Operations Research **52**(3), 368–383 (2004)
67. Schramm, H., Zowe, J.: A version of the bundle idea for minimizing a nonsmooth function: conceptual idea, convergence analysis, numerical results. SIAM Journal on Optimization **2**(1), 121–152 (1992)
68. Schumaker, L.L.: Spline Functions: Basic Theory, 3 edn. Cambridge University Press (2007)
69. Sherali, H.D.: On mixed-integer zero-one representations for separable lower-semicontinuous piecewise-linear functions. Operations Research Letters **28**(4), 155–160 (2001)
70. Shor, N.: Minimization Methods for Non-Differentiable Functions. Springer-Verlag, Berlin (1985)
71. Shukla, R., Dragotti, P.L., Do, M.N., Vetterli, M.: Rate-Distortion Optimized Tree Structured Compression Algorithms for Piecewise Smooth Images. IEEE Transactions on Image Processing **14**(3), 343–359 (2005)
72. Silva, T.L., Camponogara, E.: A computational analysis of multidimensional piecewise-linear models with applications to oil production optimization. European Journal of Operational Research **232**(3), 630–642 (2014)
73. Stubbs, R.A., Mehrotra, S.: A branch-and-cut method for 0-1 mixed convex programming. Mathematical Programming **86**, 515–532 (1999)
74. Türkay, M., Grossmann, I.E.: Disjunctive Programming Techniques for the Optimization of Process Systems with Discontinuous Investment Costs - Multiple Size Regions. Industrial & Engineering Chemistry Research **35**, 2611–2623 (1996)
75. Vecchietti, A., Lee, S., Grossmann, I.E.: Modeling of discrete/continuous optimization problems: characterization and formulation of disjunctions and their relaxations. Computers & Chemical Engineering **27**(3), 433–448 (2003)
76. Vielma, J.P.: Mixed Integer Linear Programming Formulation Techniques. SIAM Review **57**(1), 3–57 (2015)
77. Vielma, J.P., Ahmed, S., Nemhauser, G.: Mixed-integer models for nonseparable piecewise-linear optimization: Unifying framework and extensions. Operations research **58**(2), 303–315 (2010)
78. Vielma, J.P., Nemhauser, G.L.: Modeling disjunctive constraints with a logarithmic number of binary variables and constraints. Mathematical Programming **128**(1-2), 49–72 (2011)

79. Vigerske, S., Gleixner, A.: Scip: global optimization of mixed-integer nonlinear programs in a branch-and-cut framework. Optimization Methods and Software pp. 1–31 (2017). DOI 10.1080/10556788.2017.1335312
80. Vu, K.K., D'Ambrosio, C., Hamadi, Y., Liberti, L.: Surrogate-based methods for black-box optimization. International Transactions in Operational Research **24**(3), 393–424 (2017). DOI 10.1111/itor.12292
81. Waki, H., Kim, S., Kojima, M., Muramatsu, M.: Sums of Squares and Semidefinite Program Relaxations for Polynomial Optimization Problems with Structured Sparsity. SIAM Journal on Optimization **17**(1), 218–242 (2006)
82. Wang, W., Pottmann, H., Liu, Y.: Fitting B-spline curves to point clouds by curvature-based squared distance minimization. ACM Transactions on Graphics **25**(2), 214–238 (2006)
83. Wechsung, A., Barton, P.I.: Global optimization of bounded factorable functions with discontinuities. Journal of Global Optimization **58**(1), 1–30 (2014). DOI 10.1007/s10898-013-0060-3
84. Womersley, R.S., Fletcher, R.: An algorithm for composite nonsmooth optimization problems. Journal of Optimization Theory and Applications **48**(3), 493–523 (1986). DOI 10.1007/BF00940574
85. Yuan, Y., Fan, W., Pu, D.: Spline function smooth support vector machine for classification. Journal of Industrial and Management Optimization **3**(3), 529–542 (2007)
86. Zang, I.: Discontinuous optimization by smoothing. Mathematics of Operations Research **6**(1), 140–152 (1981)

A Polynomials

In this appendix we provide some transformations that are useful for manipulating polynomials. These transformations can be found in most textbooks treating polynomials and are reproduced here without proofs. The interested reader may refer to the book by [57].

A.1 Transformation between the monomial and Bernstein basis

There exist a linear mapping $Q_p \in \mathbb{R}^{(p+1)\times(p+1)}$ that transforms a p-th degree Bernstein basis $\boldsymbol{B}_p = \{B_{i,p}\}_{i=0}^{p}$ to a p-th degree monomial basis $\boldsymbol{M}_p = \{M_i\}_{i=0}^{p}$; that is, $\boldsymbol{M}_p = Q_p \boldsymbol{B}_p$. The transformation matrix Q_p is an upper triangular matrix given as:

$$Q_p(i,j) = \begin{cases} 0 & , \ i > j \\ \binom{j}{i} / \binom{p}{i} & , \ i \leq j \end{cases} \tag{43}$$

Conversely, the inverse mapping $\boldsymbol{B}_p = Q_p^{-1} \boldsymbol{M}_p$ transforms a monomial basis to a Bernstein basis.

A.2 Reparametrization of the monomial basis

There exist a linear mapping $R_p \in \mathbb{R}^{(p+1)\times(p+1)}$ that reparametrizes a monomial basis $\boldsymbol{M}_p(u)$ on $u \in [0,1]$, to a monomial basis $\boldsymbol{M}_p(x)$ on $x \in [a,b]$; that is, $\boldsymbol{M}_p(x) = R_p \boldsymbol{M}_p(u)$. R_p is a lower triangular matrix given as:

$$R_p(i,j) = \begin{cases} 0 & , \ i < j \\ \binom{i}{j} (b-a)^j a^{i-j} & , \ i \geq j \end{cases} \tag{44}$$

Conversely, the inverse mapping $\boldsymbol{M}_p(u) = R_p^{-1} \boldsymbol{M}_p(x)$ reparametrizes a monomial basis from x to u.

B B-splines

B.1 Reparameterization of the B-spline basis

The B-spline basis is invariant to an affine change of variables, as shown by the following proposition.

Proposition 7 (Invariance to affine change of variables) *Consider a B-spline basis* $\{N_{i,p}(x;\boldsymbol{t})\}_{i=0}^{p}$ *in the variable* $x \in [a,b)$ *defined by the regular knot sequence*

$$\boldsymbol{t} = \{\underbrace{a,\dots,a}_{p+1},\underbrace{b,\dots,b}_{p+1}\}. \tag{45}$$

The B-spline is invariant to the affine change of variables $x = (b-a)y + a$, $y \in [0,1)$. *That is,* $N_{i,p}(x;\boldsymbol{t}) \equiv N_{i,p}(y;\boldsymbol{s})$, $\forall i,p$, *where* $\boldsymbol{s}$ *is the scaled knot sequence*

$$\boldsymbol{s} = \{\underbrace{0,\dots,0}_{p+1},\underbrace{1,\dots,1}_{p+1}\}, \tag{46}$$

obtained as $s_i = (t_i - a)/(b-a)$, $\forall i$.

Proof The proposition is proved by induction. Considering the base case for $p = 0$ in (26), simple algebra yields

$$N_{i,0}((b-a)y + a; (b-a)\boldsymbol{s} + a) = \begin{cases} 1, & s_i \le y < s_{i+1}, \\ 0, & \text{otherwise.} \end{cases} \tag{47}$$

proving that $N_{i,0}(x;\boldsymbol{t}) \equiv N_{i,0}(y;\boldsymbol{s})$, $\forall i,x$.

The next step of the proof is to show that, assuming $N_{i,p-1}(x;\boldsymbol{t}) \equiv N_{i,p-1}(y;\boldsymbol{s})$ $\forall i,x$, it will also be true that $N_{i,p}(x;\boldsymbol{t}) \equiv N_{i,p}(y;\boldsymbol{s})$ $\forall i,x$. Inserting the change of variable into (26) gives

$$\begin{aligned} N_{i,p}(x;\boldsymbol{t}) &= N_{i,p}((b-a)y + a; (b-a)\boldsymbol{s} + a) \\ &= \frac{y - s_i}{s_{i+p} - s_i} N_{i,p-1}(y;\boldsymbol{s}) + \frac{s_{i+p+1} - y}{s_{i+p+1} - s_{i+1}} N_{i+1,p-1}(y;\boldsymbol{s}) \\ &= N_{i,p}(y;\boldsymbol{s}), \end{aligned} \tag{48}$$

which concludes the proof. □

Remark that the special structure of the knot sequence $\boldsymbol{t}$ in Proposition 7 was not utilized in the proof. In fact, Proposition 7 holds for any regular knot sequence $\boldsymbol{t}$ [57].

B.2 Decomposition using knot insertion

Knot insertion, as described in the following lemma, can be used to decompose a B-spline into a set of disjoint (non-overlapping) polynomial pieces.

Lemma 4 (Knot insertion) *Consider the following two degree* p *B-spline bases* $N_p = [N_{i,p}(x;\boldsymbol{t})]_{i=0}^{n-1}$ *and* $\tilde{N}_p = [\tilde{N}_{i,p}(x;\boldsymbol{\tau})]_{i=0}^{\tilde{n}-1}$, *where* $\boldsymbol{\tau} = \{\tau_i\}_{i=0}^{\tilde{n}+p}$, $\boldsymbol{t} = \{t_i\}_{i=0}^{n+p}$, $n \le \tilde{n}$, *and* $\boldsymbol{\tau}$ *is a refinement of* $\boldsymbol{t}$ *(any real number occurs at least as many times in* $\boldsymbol{\tau}$ *as in* $\boldsymbol{t}$*). Then,*

$$span\{N_{i,p}(x;\boldsymbol{t})\}_{i=0}^{n-1} = \mathbb{S}_p(\boldsymbol{t}) \subseteq \mathbb{S}_p(\boldsymbol{\tau}) = span\{\tilde{N}_{i,p}(x;\boldsymbol{\tau})\}_{i=0}^{\tilde{n}-1}. \tag{49}$$

Furthermore, there exists an $\tilde{n} \times n$ *matrix* A *so that* $N_p = A^T\tilde{N}_p$. *And thus,*

$$f(x) = \sum_{i=0}^{n-1} c_i N_{i,p}(x;\boldsymbol{t}) = \boldsymbol{c}^T N_p(x;\boldsymbol{t}) = (A\boldsymbol{c})^T \tilde{N}_p(x;\boldsymbol{\tau}), \tag{50}$$

where $A\boldsymbol{c}$ *are the coefficients of the new basis. The matrix* A *is known as the knot insertion matrix and can be efficiently computed using the Oslo Algorithm 1 [47].*

Proof See [47] and related works. □

The practical interpretation of Lemma 4 is that knots may be inserted into the knot sequence to change the basis, without geometrically altering the spline. By combining knot insertion with the relationship between the B-spline and Bernstein basis functions, we may device a procedure that brings any B-spline to the form of a PWP in Def. 2. The algorithmic steps of the following procedure can also be found in [57], and are commonly used to decompose a B-spline into a Bézier curve.

1. Apply *knot insertion* as in Lemma 4, to increase the knot multiplicity of all knots to $p+1$. This effectively decomposes the B-spline into a set of non-overlapping degree p polynomials, with $p+1$ basis functions in each (non-empty) knot interval.
2. For each knot interval, select the coefficients corresponding to the $p+1$ supported basis functions. These coefficients are equivalent to the coefficients of the polynomial if it was expressed in Bernstein form on the unit interval.
3. (Optional) Perform a reparametrization of the Bernstein form from the unit interval to the range defined by the knot span using Proposition (2).

C Numerical results

The numerical results from the numerical study in Section 7 are listed in Table 3.

Table 3 Solve times on test problems.

Problem	Formulation	Solver	T_{med}	T_{mean}	T_{std}	T_{min}	T_{max}
random1d	MICQP	BARON	0.210	0.202	0.098	0.030	0.450
	MICQP-CUT	BARON	0.140	0.149	0.083	0.001	0.380
	MINLP-BM	BARON	0.090	0.082	0.042	0.001	0.160
	MINLP-LOG	BARON	0.100	0.092	0.047	0.001	0.260
	MINLP-CUT	BARON	0.040	0.046	0.024	0.001	0.130
	MINLP-EXP	BARON	0.060	0.057	0.027	0.001	0.110
	NLP	CENSO	0.013	0.014	0.009	0.001	0.040
random2d	MICQP	BARON	13.22	13.95	4.227	4.710	35.51
	MICQP-CUT	BARON	4.075	5.082	4.346	0.320	27.28
	MINLP-BM	BARON	6.770	10.73	9.700	0.740	40.76
	MINLP-LOG	BARON	6.995	11.87	10.89	0.490	46.92
	MINLP-CUT	BARON	1.425	2.955	3.457	0.300	15.91
	MINLP-EXP	BARON	1.270	1.789	1.392	0.290	7.690
	NLP	CENSO	0.092	0.098	0.053	0.003	0.228
random3d	MICQP	BARON	427.8	602.3	648.0	99.57	3600
	MICQP-CUT	BARON	35.44	54.46	47.21	3.670	231.6
	MINLP-BM	BARON	961.3	1364	1014	18.91	3600
	MINLP-LOG	BARON	912.6	1216	1018	17.15	3600
	MINLP-CUT	BARON	82.24	185.5	256.3	2.270	1414
	MINLP-EXP	BARON	34.91	47.29	55.71	3.790	386.3
	NLP	CENSO	0.226	0.246	0.124	0.039	0.877

Times are given in seconds.