A MIQCP formulation for B-spline constraints

Bjarne Grimstad
Department of Engineering Cybernetics,
Norwegian University of Science and Technology, NO-7491 Trondheim, Norway
bjarne.grimstad@ntnu.no

September 4, 2017

This paper presents a mixed-integer quadratically constrained programming (MIQCP) formulation for B-spline constraints. The formulation can be used to obtain an exact MIQCP reformulation of any spline-constrained optimization problem, provided that the polynomial spline functions are continuous. This reformulation allows practitioners to use a general-purpose MIQCP solver, instead of a special-purpose spline solver, when solving B-spline constrained problems.

B-splines are a powerful and widely used modeling tool, previously restricted from optimization due to lack of solver support. This contribution may encourage practitioners to use B-splines to model constraint functions. However, as the numerical study suggests, there is still a large gap between the solve times of the general-purpose solvers using the proposed formulation, and the special-purpose spline solver CENSO, the latter being significantly lower.

1 Introduction

Consider the optimization problem

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{a}^\mathsf{T} \boldsymbol{x} \\ \text{subject to} \quad & \boldsymbol{g}(\boldsymbol{x}) \leq \boldsymbol{0} \\ & \boldsymbol{x} \in X \end{aligned} \tag{$\mathbf{P}$}$$

where $\boldsymbol{a} \in \mathbb{R}^n$, $g_i : D_i \to \mathbb{R}$ for $i = 1, \dots, m$ are m nonlinear (possibly nonconvex) constraint functions, and $X \subseteq \mathbb{R}^n$ is a convex polyhedron bounding the n variables $\boldsymbol{x}$. Without loss of generality, we assume that the objective function is linear; a problem with a nonlinear objective function can always be brought to the form in $\mathbf{P}$ by using the epigraph reformulation. A problem that has convex constraint functions $\boldsymbol{g}$ is said to be a *convex optimization problem*. For convex problems, any local minimum is a global minimum, which simplifies the solution process significantly. In contrast, finding a global minimum of a *nonconvex* problem is generally $\mathcal{NP}$-hard [10].

There exist general-purpose optimization solvers that can solve a broad class of nonconvex programming problems to global optimality. Generally speaking, these solvers handle constraint functions composed by binary operations such as summation, subtraction, multiplication, and division, and unary operations such as the exponential and logarithm [7].

Furthermore, these solvers may solve mixed-integer nonlinear programming (MINLP) problems, in which some of the variables are required to be integers.

In this paper, we are concerned with problems where the constraint functions are polynomial spline functions, which we refer to as spline-constrained optimization problems. Piecewise polynomial functions are included in this general class of functions. A spatial branch-and-bound algorithm for spline-constrained problems was recently published [3]. The algorithm utilizes the properties of B-splines to ensure ϵ-convergence for problems on the form in $\mathbf{P}$. An implementation of the algorithm is available in the experimental, open-source solver CENSO, which currently is the only optimization solver that naturally handles spline constraints.

To the author's knowledge, there do not exist any formulations or recipes that allow spline constrained problems to be solved by a general-purpose solver. To this end, the author presents a mixed-integer quadratically constrained programming formulation for spline constraints. The formulation can be used to reformulate and solve $\mathbf{P}$ using any MIQCP solver, such as GLOMIQO [4] or BARON [7].

This work is motivated by the many applications of polynomial splines in engineering and mathematics, some listed in the next section, and the surprising lack of optimization tools for spline-constrained optimization problems. Several works exist in the literature on optimization with polynomial splines, however, the majority of them address the problem of finding a spline parametrization that is optimal in some sense; cf. [1, 2, 5, 11, 12]. There are few works on constrained optimization of spline functions with a fixed parametrization. The most relevant work in this area are listed in [3]. The formulation proposed in this paper is based on the widely used B-spline framework, which is introduced in the next section.

2 B-splines

A widely used class of piecewise functions is the polynomial splines, henceforth referred to simply as splines. Splines are used extensively in function approximation, curve fitting, data smoothing, signal and image processing, and modelling of geometric shapes in computer-aided design (CAD), among many other applications [6, 9]. The most common framework for constructing and representing splines is the *B-spline*. A B-spline is constructed by overlapping polynomial functions: this construction allows for a high order of continuity at the points where the polynomials pieces connect (known as knots). In this section we give a brief introduction to the B-spline and some of its basic properties.

The theory presented in this chapter can be found in most text books on Spline Theory; cf. [6, 8].

2.1 Univariate B-splines

A B-spline $f : \mathbb{R} \to \mathbb{R}$ in the variable x is expressed as

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

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

$$
\begin{aligned}
N_{i,p}(x;\boldsymbol{t}) &= \frac{x - t_i}{t_{i+p} - t_i} N_{i,p-1}(x;\boldsymbol{t}) + \frac{t_{i+p+1} - x}{t_{i+p+1} - t_{i+1}} N_{i+1,p-1}(x;\boldsymbol{t}), \\
N_{i,0}(x;\boldsymbol{t}) &= \begin{cases} 1, & t_i \leq x < t_{i+1}, \\ 0, & \text{otherwise.} \end{cases}
\end{aligned} \tag{2}
$$

The parameter $\boldsymbol{t} = \{t_i\}_{i=0}^{n+p}$ is called a *knot sequence* and it is required to be a non-decreasing sequence of real numbers called *knots*. In this work we assume that the knot sequence is *regular* according to the following definition.

Definition 1 (Regular knot sequence)**.** *A knot sequence $\boldsymbol{t}$ is said to be regular iff*

1. *it is a non-decreasing sequence of real numbers, i.e.* $t_0 \leq t_1 \leq \cdots \leq t_{n+p}$*;*
2. *no knot is repeated more than $p+1$ times, i.e.* $t_i < t_{i+p+1}$*;*
3. *end knots are repeated $p+1$ times, i.e.* $t_0 = t_p$, $t_n = t_{n+p}$.

Note that the basis functions are piecewise polynomials by definition: the 0-th degree basis functions are piecewise constant, the 1-st degree basis functions are piecewise linear, the 2-nd degree basis funtions are piecewise quadratic, and so on. This is a consequence of the construction of degree p basis function as a convex combination of two degree $p-1$ basis functions. The B-spline function f, being a linear combination of piecewise polynomial functions, is itself a piecewise polynomial function with polynomial pieces of degree p.

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

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

$$N_{i,p}(x;\boldsymbol{t}) \geq 0, \quad \forall x \in \mathbb{R} \tag{3}$$

$$N_{i,p}(x;\boldsymbol{t}) = 0, \quad \forall x \notin [t_i, t_{i+p+1}) \tag{4}$$

$$\sum_{i=j-p}^{j} N_{i,p}(x;\boldsymbol{t}) = 1, \quad \forall x \in [t_j, t_{j+1}) \tag{5}$$

Proof. A proof can be found in [6, 8]. □

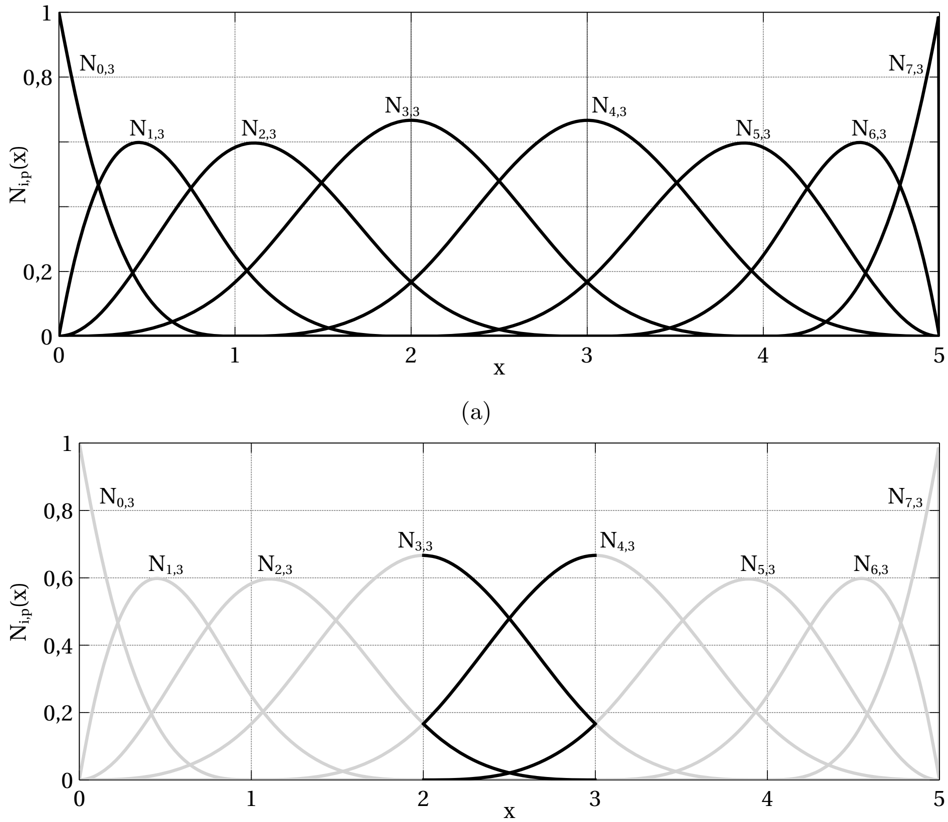
The three properties (3)–(5) in Lemma 2, are referred to as the properties of nonnegativity, local support, and partition of unity, respectively. These properties and the feature of overlapping B-spline basis functions are displayed in Fig. 1.

The next corollary extends the partition of unity property to the supported interval $[t_0, t_{n+p})$.

Corollary 2.1 (Partition of unity)**.** *The following holds true for a set of degree p B-spline basis functions $\{N_{i,p}\}$ defined by a regular knot vector $\boldsymbol{t}$:*

$$\sum_{i=0}^{n-1} N_{i,p}(x;\boldsymbol{t}) = 1, \quad \forall x \in [t_0, t_{n+p}) \tag{6}$$

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



(a)

(b)

Figure 1: B-spline basis functions for $p = 3$, $n = 8$, and knot sequence $t = [0, 0, 0, 0, 1, 2, 3, 4, 5, 5, 5, 5]$ are plotted in (a). Basis functions that are non-zero 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 in (b).

Proof. The corollary follows directly from Lemma 2 □

Corollary 2.1 tells us that, for $x \in [t_0, t_{n+p})$, the basis function of degree p sum to one. Notice that the proof does not put any restrictions on the basis degree p, so that Corollary 2.1 holds true for any $p \geq 0$.

Subsequently, when there is no ambiguity, we simplify the notation by omitting the dependence on x and $\boldsymbol{t}$ so that $N_{i,p} \implies N_{i,p}(x) \implies N_{i,p}(x; \boldsymbol{t})$.

2.2 Multivariate B-splines

Multivariate B-splines, also known as tensor product B-splines, can be constructed by taking the Kronecker product of univariate B-splines bases. Consider d variables $\boldsymbol{x} = (x_1, \ldots, x_d) \subseteq \mathbb{R}^d$. Let

$$\boldsymbol{N}_{p_k}(x_k; \boldsymbol{t}_k) = [N_{i,p_k}(x_k; \boldsymbol{t}_k)]_{i=0}^{n_k-1}, \tag{7}$$

be a vector of n_k univariate, degree p_k B-spline basis functions in the variable x_k given by the knot sequence $\boldsymbol{t}_k = \{t_{0,k}, \ldots, t_{n_k+p_k,k}\}$. Then, a multivariate B-spline basis is constructed as

$$\boldsymbol{\Phi}(\boldsymbol{x}) = \bigotimes_{k=1}^{d} \boldsymbol{N}_{p_k}(x_k; \boldsymbol{t}_k), \tag{8}$$

and a multivariate B-spline as

$$f(\boldsymbol{x}) = \sum_{i=0}^{N-1} c_i \Phi_i(\boldsymbol{x}), \tag{9}$$

where

$$N = \prod_{k=1}^{d} n_k \tag{10}$$

is the number of multivariate basis functions.

The properties of the univariate B-spline basis functions in Lemma 2 extend naturally to the multivariate case. Below we provide an extension of Corollary 2.1 to the multivariate case.

Proposition 3 (Partition of unity). *The following holds true for a multivariate B-spline basis $\boldsymbol{\Phi}(\boldsymbol{x})$:*

$$\sum_{i=0}^{N-1} \Phi_i(\boldsymbol{x}) = 1, \quad \forall x_k \in [t_{0,k}, t_{n_k+p_k,k}), k = 1, \ldots, d. \tag{11}$$

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

$$\boldsymbol{\Phi}^{(-r)}(\boldsymbol{x}) = \bigotimes_{k=1, k\neq r}^{d} \boldsymbol{N}_{p_k}(x_k). \tag{12}$$

Furthermore, let

$$N^{(-r)} = \prod_{k=1, k\neq r}^{d} n_k. \tag{13}$$

Then, it follows from Lemma 2 that

$$\sum_{i=0}^{N-1} \Phi_i(\boldsymbol{x}) = \left(\sum_{i=0}^{n_r-1} N_{i,p_r}(x_r) \right) \left(\sum_{i=0}^{N^{(-r)}-1} \Phi_i^{(-r)}(\boldsymbol{x}) \right) = \sum_{i=0}^{N^{(-r)}-1} \Phi_i^{(-r)}(\boldsymbol{x}), \tag{14}$$

for $x_r \in [t_{0,r}, t_{n_r+p_r,r})$. The above relation implies that

$$\sum_{i=0}^{N-1} \Phi_i(\boldsymbol{x}) = 1, \quad \forall x_k \in [t_{0,k}, t_{n_k+p_k,k}), k = 1, \ldots, d. \tag{15}$$

□

Notice that the proof of Proposition 3, as in the univariate case, does not put any restrictions on the basis degrees $\{p_k\}_{k=1}^{d}$.

3 A MIQCP formulation for B-splines

In this section we present a MIQCP formulation for $\mathbf{P}$, where the constraint functions $\boldsymbol{g}$ are multivariate (tensor product) B-spline functions defined as in (9). Each spline constraint $g_i \leq 0$, where $g_i : D_i \to \mathbb{R}$ and D_i is a compact domain, is modeled using its epigraph by introducing an auxiliary variable z_i. That is, we model $g_i \leq 0$ as $(\boldsymbol{x}, z) \in \text{epi}(g_i) = \{(\boldsymbol{x}, z) \in D_i \times \mathbb{R} : g_i(\boldsymbol{x}) \leq z_i, z_i \leq 0\}$. To ensure that the epigraph is closed, we require that g_i is continuous by restricting the knot sequence to be regular according to Def. 1 and the degree $p \geq 1$.

Subsequently, we focus on the model of a single constraint $g_i \leq 0$, first by considering the univariate case and then extending to the multivariate case.

3.1 Univariate spline constraint formulation

We can model the B-spline basis functions in (2) by replacing the piecewise constant (0-th degree) basis functions with binary variables, and higher degree basis functions with continuous variables. We introduce $n + p$ binary auxiliary variables $\boldsymbol{\beta}_0 \in \{0, 1\}^{n+p}$ to represent the degree 0 basis functions. Specifically, we want $\beta_{i,0} = N_{i,0}$, for $i = 0, \ldots, n + p - 1$. Next we introduce continuous auxiliary variables $\boldsymbol{\beta}_1 \in \mathbb{R}^{n+p-1}$, ..., $\boldsymbol{\beta}_p \in \mathbb{R}^n$ to represent the basis functions of degree $1, 2, \ldots, p$, respectively. This amounts to a total of $\sum_{i=1}^{p}(n + p - i) = p(n + \frac{1}{2}(p + 1))$ continuous auxiliary variables.

The relation between the knots and basis functions of increasing degree is illustrated in Fig. 2. A key observation is that, in accordance with (2), a basis function of degree $p \geq 1$ is defined as a convex combination of two degree $p - 1$ basis functions.

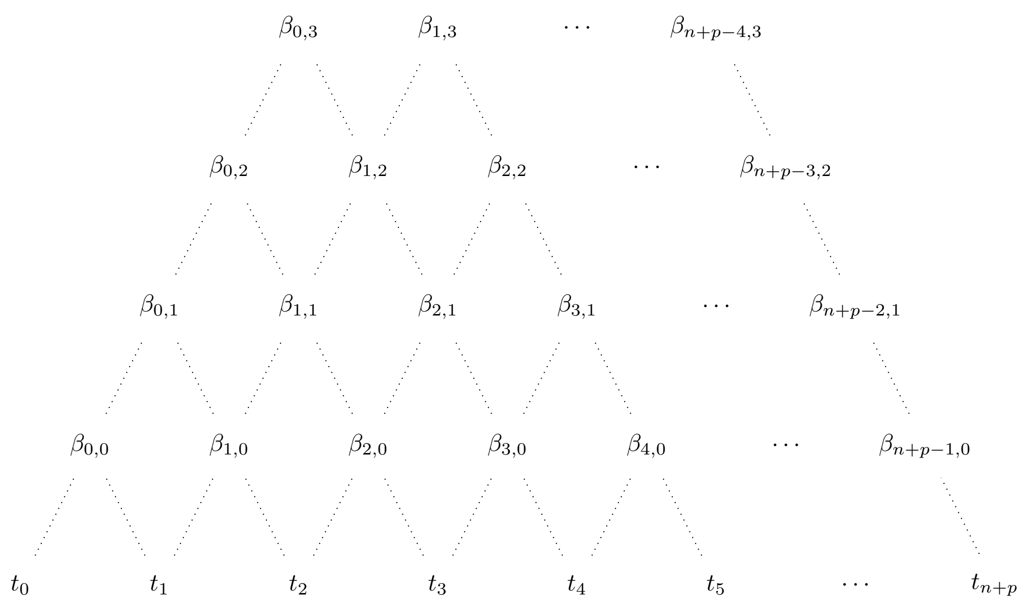


Figure 2: Illustration of the tree structure in which knots and B-spline basis functions relate. From the bottom up, the levels represent the knot sequence $\boldsymbol{t}$, the 0-th, 1-st, 2-nd, and 3-rd degree basis functions $\boldsymbol{\beta}_0$, $\boldsymbol{\beta}_1$, $\boldsymbol{\beta}_2$, and $\boldsymbol{\beta}_3$, respectively.

$\boldsymbol{\beta}_0 = [\beta_{i,0}]_{i=0}^{n+p-1}$ represents the 0th-degree basis functions $\{N_{0,0}, \dots, N_{n+p-1,0}\}$, which are piecewise constant functions. We model this as follows:

$$\begin{aligned}
& x \geq (t_i - t_0)\beta_{i,0} + t_0, && i = 0, \dots, n+p-1 \\
& x \leq (t_{i+1} - t_{n+p})\beta_{i,0} + t_{n+p}, && i = 0, \dots, n+p-1 \\
& \sum_{i=0}^{n+p-1} \beta_{i,0} = 1, && \\
& \beta_{i,0} = \{0, 1\}, && i = 0, \dots, n+p-1
\end{aligned} \tag{16}$$

The first two constraint sets models the bounds imposed on x when a basis function is nonzero; the bounds represent the knot interval in which the basis function supports the spline. The third and fourth constraints ensure that exactly one basis function is nonzero. Readers familiar with the B-spline may notice that, since only one 0th-degree basis function may be active at a time, the disjunction of the half-open domains of (2) is precisely modeled if the B-spline is continuous, which is guaranteed by virtue of the regular knot sequence when $p \geq 1$. A detail that is easily overlooked, however, is that the support of the B-spline basis functions is the half-open interval $[t_0, t_{n+p})$; see Corollary 2.1. With the above formulation, the support becomes the closure $[t_0, t_{n+p}]$; and for the univariate B-spline we set $D_i = [t_0, t_{n+p}]$.

$\boldsymbol{\beta}_j$, for $j = 1, \dots, p$, represents the jth-degree basis functions $\{N_{0,j}, \dots, N_{n+p-1-j,j}\}$. These basis functions are modeled analogous to the definition in (2). That is,

$$\beta_{i,j} = \frac{x - t_i}{t_{i+j} - t_i}\beta_{i,j-1} + \frac{t_{i+j+1} - x}{t_{i+j+1} - t_{i+1}}\beta_{i+1,j-1}, \quad i = 0, \dots, n+p-1-j. \tag{17}$$

Finally, we complete the model of $(x, z) \in \operatorname{epi}(g_i) = \{(x, z) \in D_i \times \mathbb{R} : f(x) \leq z, z \leq 0\}$ by adding the constraints

$$\begin{aligned}
\boldsymbol{c}^\mathsf{T}\boldsymbol{\beta}^p &\leq z, \\
z &\leq 0,
\end{aligned} \tag{18}$$

where $\boldsymbol{c}$ are the B-spline coefficients.

In the univariate case above, the formulation introduces $(n+p)$ binary variables, $\sum_{i=1}^{p}(n+p-i)$ continuous variables, $2(n+p)+3$ linear constraints, and $\sum_{i=1}^{p}(n+p-i)$ quadratic (bilinear) equality constraints.

3.2 Multivariate spline constraint formulation

To obtain a multivariate formulation we first build each univariate basis, then we apply the Kronecker product as in (8). However, successive Kronecker products cannot be applied directly as it would produce terms of degree higher than two, resulting in a MINLP formulation rather than a MIQCP formulation. To retain a MIQCP formulation, we need to introduce additional auxiliary variables and constraints that allow us to "store" the result in each step of the successive Kronecker products.

Let $\boldsymbol{\beta}_{p_k}^k$ be a vector of n_k degree p_k basis functions in variable x_k, modeled as in the previous section. As explained above, we wish to model the multivariate basis by using bilinear constraints only. To achieve this, we introduce additional auxiliary variables $\boldsymbol{\xi}^k$, $k \in K$, and the bilinear constraints

$$\begin{aligned}
\boldsymbol{\xi}^1 &= \boldsymbol{\beta}_{p_1}^1, \\
\boldsymbol{\xi}^{k+1} &= \boldsymbol{\beta}_{p_k}^{k+1} \otimes \boldsymbol{\xi}^k, \quad k \in K \setminus d,
\end{aligned} \tag{19}$$

where we have introduced the set $K = \{1, \ldots, d\}$ to ease notation.

The number of auxiliary variables in $\boldsymbol{\xi}^1$ is n_1, $\boldsymbol{\xi}^2$ is $n_2 n_1$, $\boldsymbol{\xi}^3$ is $n_3 n_2 n_1$, and so on. In total

$$\sum_{k=1}^{d} \prod_{j=1}^{k} n_j, \tag{20}$$

additional auxiliary variables are required (note that the n_1 auxiliary variables $\boldsymbol{\xi}^1$ are redundant and may be eliminated). An equal amount of additional constraints are added to the problem formulation.

The variables $\boldsymbol{\beta}^k_{p_k}$ and $\boldsymbol{\xi}^k$ represent B-spline basis functions and products of B-spline basis functions, respectively. Thus, according to Lemma 2 they are both bounded between 0 and 1, and we may add the bounding constraints

$$\begin{aligned} \mathbf{0} &\leq \boldsymbol{\xi}^k \leq \mathbf{1}, && k \in K \\ \mathbf{0} &\leq \boldsymbol{\beta}^k_j \leq \mathbf{1}, && j = 1, \ldots, p_k, k \in K \end{aligned} \tag{21}$$

With the multivariate basis constructed, we may model $(\boldsymbol{x}, z) \in \operatorname{epi}(g_i) = \{(\boldsymbol{x}, z) \in D_i \times \mathbb{R} : f(\boldsymbol{x}) \leq z, z \leq 0\}$ by adding the constraints

$$\begin{aligned} \boldsymbol{c}^\mathsf{T} \boldsymbol{\xi}^d &\leq z, \\ z &\leq 0. \end{aligned} \tag{22}$$

As above, for each variable x_k, let $\boldsymbol{\beta}^k_j = [\beta^k_{i,j}]_{i=0}^{n_k+p_k-1-j}$ represent the degree j B-spline basis functions defined by the knot sequence $\boldsymbol{t}_k = [t_{i,k}]_{i=0}^{n_k+p_k}$. Using this notation, we form the complete MIQCP model for a multivariate B-spline:

$$\begin{aligned}
& x_k \geq (t_{i,k} - t_{0,k})\beta^k_{i,0} + t_{0,k}, && i = 0, \ldots, n_k + p_k - 1, k \in K \\
& x_k \leq (t_{i+1,k} - t_{n_k+p_k,k})\beta^k_{i,0} + t_{n_k+p_k,k}, && i = 0, \ldots, n_k + p_k - 1, k \in K \\
& \sum_{i=0}^{n_k+p_k-1} \beta^k_{i,0} = 1, && k \in K \\
& \beta^k_{i,0} = \{0, 1\}, && i = 0, \ldots, n_k + p_k - 1, k \in K \\
& \beta^k_{i,j} = \frac{x_k - t_{i,k}}{t_{i+j,k} - t_{i,k}} \beta^k_{i,j-1} + \frac{t_{i+j+1,k} - x_k}{t_{i+j+1,k} - t_{i+1,k}} \beta^k_{i+1,j-1}, && \begin{cases} i = 0, \ldots, n_k + p_k - 1 - j, \\ j = 1, \ldots, p_k, \\ k \in K \end{cases} \\
& \boldsymbol{\xi}^1 = \boldsymbol{\beta}^1_{p_1}, \\
& \boldsymbol{\xi}^{k+1} = \boldsymbol{\beta}^{k+1}_{p_k} \otimes \boldsymbol{\xi}^k, && k \in K \setminus d \\
& \mathbf{0} \leq \boldsymbol{\xi}^k \leq \mathbf{1}, && k \in K \\
& \mathbf{0} \leq \boldsymbol{\beta}^k_j \leq \mathbf{1}, && j = 1, \ldots, p_k, k \in K \\
& z \leq 0, \\
& \boldsymbol{c}^\mathsf{T} \boldsymbol{\xi}^d \leq z.
\end{aligned} \tag{MIQCP}$$

The number of linear constraints in MIQCP is (not counting the redundant constraints $\boldsymbol{\xi}^1 = \boldsymbol{\beta}_{p_1}^1$ and the variable bounds)

$$2 + d + 2\sum_{k=1}^{d}(n_k + p_k) + 2\sum_{k=1}^{d}\sum_{j=1}^{p_k}(n_k + p_k - j) + 2\sum_{k=1}^{d}\prod_{j=1}^{k} n_j. \tag{23}$$

The number of bilinear constraints is

$$\sum_{k=1}^{d}\sum_{j=1}^{p_k}(n_k + p_k - j) + \sum_{k=1}^{d}\prod_{j=1}^{k} n_j, \tag{24}$$

which simplifies to

$$d\sum_{j=1}^{p}(n + p - j) + \sum_{k=1}^{d} n^k = dp(n + \frac{1}{2}(p+1)) + \sum_{k=1}^{d} n^k = \mathcal{O}(dpn + dp^2 + n^d), \tag{25}$$

if all univariate bases are of degree p and have n basis functions.

3.3 Relaxation strengthening cuts

The formulation in MIQCP may be augmented with *valid cuts* to reduce solve times. Valid cuts are constraints that do not alter the optimal solution to **P**, but that may strenghten the relaxations produced by the solver. Below, we consider two classes of cuts based on B-spline properties, that are valid for the formulation in MIQCP. We augment the MIQCP formulation with these cuts and denote the resulting formulation MIQCP-CUT.

3.3.1 Partition-of-unity cuts

Consider the following cuts

$$\begin{aligned} \mathbf{1}^\mathsf{T}\boldsymbol{\xi}^k &= 1, \quad k \in K, \\ \mathbf{1}^\mathsf{T}\boldsymbol{\beta}_{p_k}^k &= 1, \quad k \in K, \end{aligned} \tag{26}$$

where $\mathbf{1}$ is a vector of ones of appropriate size. These cuts explicitly model the partition-of-unity property of multivariate B-splines in Proposition 3; we refer to them here as partition-of-unity cuts.

According to Proposition 3, the $2d$ linear equality constraints above are *valid cuts* for **P** given that $x_k \in [t_{0,k}, t_{n_k+p_k,k}]$ for $k \in K$. This restriction on $\boldsymbol{x}$ is already programmed in MIQCP. Thus, we may add the above cuts to MIQCP without altering the optimal solution to **P**.

3.3.2 Local support cuts

It follows directly from the local support property in (4) that

$$\beta_{i,0}^k = \ldots = \beta_{i+p_k,0}^k = 0 \implies \beta_{i,p_k}^k = 0, \quad i = 0, \ldots, n_k - 1, k \in K. \tag{27}$$

This implication, which is easily verified by looking at Fig. 2, may be programmed explicitly as follows

$$\beta_{i,p_k}^k \leq \sum_{j=i}^{i+p_k} \beta_{j,0}^k, \quad i = 0, \ldots, n_k - 1, k \in K, \tag{28}$$

where we have utilized the fact that $0 \leq \beta_{i,j}^k \leq 1$ for all i, j, k.

As with the partition-of-unity property, the local support property is already programmed implicitly by MIQCP, and thus, the cuts in (28) are valid for $\mathbf{P}$.

4 Numerical study

To assess the performance of the two formulations, three sets of randomly generated test problems were created. The three sets contain problems with one cubic spline constraint in one, two, and three variables, respectively. The problems are on the form

$$\min_{\boldsymbol{x},z} \left\{ z \ : \ g(\boldsymbol{x}) \leq z, \ \boldsymbol{x} \in D \subset \mathbb{R}^d \right\} \tag{29}$$

where g is a B-spline with domain D. g was constructed with fixed equidistant knots and coefficients randomly drawn from a Gaussian distribution with zero mean and unity standard deviation. That is, $c_i \sim \mathcal{N}(0, 1), \ \forall i = 0, \ldots, N-1$.

The monovariable problems (*random1d*) have a cubic spline constraint with 13 basis functions, giving 10 knot intervals with cubic polynomial pieces of degree 3.[2] The set of problems in two variables (*random2d*) have a bicubic spline constraint with $13 \times 13 = 169$ basis functions, equivalent to a 10×10 rectilinear grid of 100 polynomial pieces of degree $pd = 6$. And finally, the problems with three variables (*random3d*) have a tricubic spline constraint with $9 \times 9 \times 9$ basis functions. This is equivalent to a $6 \times 6 \times 6$ rectilinear grid of 216 polynomial pieces of degree $pd = 9$. The problem sets are summarized in Table 1.

Table 1: Test sets of piecewise polynomially constrained problems.

Problem	d	p	pd	N	#poly	#inst
random1d	1	3	3	13	10	100
random2d	2	3	6	169	100	100
random3d	3	3	9	729	216	100

#poly: Number of polynomial pieces. #inst: Number of problem instances.

The reformulated problems, denoted MIQCP and MIQCP-CUT, were solved by GLOMIQO [4] and BARON [7]. For comparison, the problems were also solved by the specialized spline solver CENSO [3]. All solvers were run with an absolute ϵ-convergence termination criteria of $\epsilon_A = 1 \cdot 10^{-6}$, a relative ϵ-convergence termination criteria of $\epsilon_R = 1 \cdot 10^{-8}$, and a time limit of 3600 seconds. All other settings were left on 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 given in Table 2.

5 Discussion

As the results in Sec. 4 show, MIQCP-CUT is superior to the MIQCP formulation on all three sets of test problems, regardless of which solver is used. The performance gain

[2] Consider a B-spline defined by a regular knot sequence with $n + p + 1 = 17$ knots and degree $p = 3$. This B-spline has $n = 13$ basis functions and $n - p = 10$ knot intervals. The reader may inspect Figure 1 to verify these numbers.

Table 2: Solve times on test problems.

Problem	Formulation	Solver	T_{med}	T_{mean}	T_{std}	T_{min}	T_{max}
random1d	MICQP	GLOMIQO	0.539	0.511	0.289	0.012	1.364
	MICQP	BARON	0.210	0.202	0.098	0.030	0.450
	MICQP-CUT	GLOMIQO	0.494	0.500	0.287	0.011	1.316
	MICQP-CUT	BARON	0.140	0.149	0.083	0.001	0.380
	NLP	CENSO	0.013	0.014	0.009	0.001	0.040
random2d	MICQP	GLOMIQO	104.3	118.2	66.9	11.1	335.4
	MICQP	BARON	13.22	13.95	4.227	4.710	35.51
	MICQP-CUT	GLOMIQO	33.80	38.04	27.22	0.044	136.1
	MICQP-CUT	BARON	4.075	5.082	4.346	0.320	27.28
	NLP	CENSO	0.043	0.043	0.017	0.002	0.098
random3d	MICQP	GLOMIQO	3600	3600	0	3600	3600
	MICQP	BARON	427.8	602.3	648.0	99.57	3600
	MICQP-CUT	GLOMIQO	-	-	-	-	-
	MICQP-CUT	BARON	35.44	54.46	47.21	3.670	231.6
	NLP	CENSO	0.228	0.246	0.124	0.039	0.877

Times are given in seconds.

obtained by adding cuts is especially evident on the *random3d* test set, where the MIQCP-CUT formulation is a order of magnitude faster than the MIQCP formulation.

On all test sets, BARON outperforms GLOMIQO, which is a bit surprising considering that GLOMIQO is a state-of-the-art solver for quadratically constrained problems. On the *random3d* test set, GLOMIQO failed to solve the MIQCP formulations within the time limit of 3600 seconds, and terminated with suboptimal solutions on many of the MIQCP-CUT formulations (these results were omitted for this reason).

Of more interest, is the large gap between the performance of the general-purpose solvers, using the proposed MIQCP and MIQCP-CUT formulations, and the special-purpose solver CENSO, which solves the problems as nonconvex NLPs. The gap is most pronounced on the *random3d* problems, where CENSO solved all problems within 1 second, while BARON used 54.46 seconds on average to solve the MIQCP-CUT formulations. The author believes that this gap may be reduced by further research into more efficient reformulations that require less variables and constraints than the MIQCP formulations presented here. Specifically, due to the exponential growth, $N \propto n^d$, in the number of bilinear constraints with number of variables d, it is advantageous to keep the number n as low as possible. As indicated in Table 1, $N = 729$ basis functions are required to model 216 polynomial pieces.

The success of the B-spline as a modeling tool is partly due to its local support property, which allows for fast fitting and evaluation. The local support property induces a sparsity pattern in that at any point $\boldsymbol{x}$ in the domain of a B-spline, at most $(p+1)^d$ basis functions are non-zero. This sparsity pattern could be utilized to obtain leaner formulations for spline constraints. Consider as an example the case above, in which $N = 729$ basis functions are used to model 216 polynomial pieces. Hypothetically, with a formulation that exploits the sparsity pattern it would suffice with $(p+1)^3 = 64$ polynomial basis functions to model the space of polynomial functions in which all of the 216 polynomial pieces lie. By using logic constraints to select a partition of the domain, the corresponding polynomial piece could be linearly mapped to this space.

6 Conclusion

The merit of this paper is a MIQCP formulation that allows practitioners to solve spline-constrained problems with a general-purpose MIQCP solver. The numerical study shows that the MIQCP-CUT formulation allows practitioners to solve problems with spline constraints in 1-3 variables in reasonable time using the BARON solver. However, there is still a large gap to the performance of the special-purpose spline solver CENSO. Until better formulations are available, practitioners should resort to specialized solution methods for problems with higher-dimensional spline constraints. A promising path towards better formulations is the pursuit of formulations that exploit the sparsity pattern of B-splines.

References

[1] B. Demeulenaere, G. Pipeleers, J. De Caigny, J. Swevers, J. De Schutter, and L. Vandenberghe. Optimal splines for rigid motion systems: A convex programming framework. *Journal of Mechanical Design*, 131(10):101004, 2009.

[2] David R Ferguson, Paul D Frank, and Alan K Jones. Surface shape control using constrained optimization on the B-spline representation. *Computer Aided Geometric Design*, 5:87–103, 1988.

[3] Bjarne Grimstad and Anders Sandnes. Global optimization with spline constraints: a new branch-and-bound method based on B-splines. *Journal of Global Optimization*, 65(3):401–439, 2016.

[4] Ruth Misener and Christodoulos A. Floudas. GloMIQO: Global mixed-integer quadratic optimizer. *Journal of Global Optimization*, 57(1):3–50, sep 2013.

[5] Chong Jin Ong, Yoke San Wong, Han Tong Loh, and Xin Gui Hong. An optimization approach for biarc curve-fitting of B-spline curves. *Computer Aided Design*, 28(12):951–959, 1996.

[6] Les A Piegl and Wayne Tiller. *The NURBS book*. Springer, 1997.

[7] Nikolaos V. Sahinidis. BARON: A general purpose global optimization software package. *Journal of Global Optimization*, 8(2):201–205, mar 1996.

[8] Larry L Schumaker. *Spline Functions: Basic Theory*. Cambridge University Press, 3 edition, 2007.

[9] M. Unser. Splines: a perfect fit for signal and image processing, 1999.

[10] Stephen A. Vavasis. *Complexity Issues in Global Optimization: A Survey*, pages 27–41. Springer US, Boston, MA, 1995.

[11] Wenping Wang, Helmut Pottmann, and Yang Liu. Fitting B-spline curves to point clouds by curvature-based squared distance minimization. *ACM Transactions on Graphics*, 25(2):214–238, 2006.

[12] Xiuyang Zhao, Caiming Zhang, Bo Yang, and Pingping Li. Adaptive knot placement using a GMM-based continuous optimization algorithm in B-spline curve approximation. *CAD Computer Aided Design*, 43(6):598–604, 2011.