Numer. Math. Theor. Meth. Appl.
doi: 10.4208/nmtma.OA-2018-0118

Vol. **13**, No. 1, pp. 150-175
February 2020

Spline R-Function and Applications in FEM

Tianhui Yang, Ammar Qarariyah and Jiansong Deng*

School of Mathematical Sciences, University of Science and Technology of China, Hefei 230026, Anhui, China

Received 21 November 2018; Accepted (in revised version) 29 March 2019

Abstract. R-function is a widely used tool when considering objects obtained through the Boolean operations start from simple base primitives. However, there is square root operation in the representation. Considering that the use of splines will facilitate the calculations within the CAD system, in this paper, we propose a system of R-functions represented in spline form called Spline R-function (SR). After transforming the function ranges of two base primitives to a new coordinate system, a series of sign constraints following a specific Boolean operation are derived and the spline R-function can be formulated as a piecewise function. Representation of SR in both Bézier form and B-spline form have been given. Among which the Bézier ordinates are determined with the help of the B-net method through setting up a series of relations according to the sign constraints and properties of R-functions. The construction processes for both Boolean intersection and union operations with different smoothness are discussed in detail. Numerical experiments are conducted to show the potential of the proposed spline R-function.

AMS subject classifications: 68U07, 65D18, 65D30

Key words: R-functions, CSG, Boolean operations, B-splines, B-net method, WEB method.

1. Introduction

Recent years have witnessed a rapid development of 3D printing together with the related studies derived in topics such as modeling and analysis [1–7]. Due to the several modeling requirements of 3D printing, implicit representation forms an ideal choice for such technology. Implicit representation possesses a natural point classification (i.e., inside/outside the geometry) and enables various topologies, which is convenient to describe the slicing structures of different objects [8, 9]. These properties among others have increased the attention for using the implicit form to model geometries for 3D printing [10–12]. In the context of implicit solids and function representation, Constructive Solid Geometry (CSG) is adopted in the modeling process for objects with complex structures (e.g., [13, 14]).

*Corresponding author. *Email address:* `dengjs@ustc.edu.cn` (J. S. Deng)

CSG is a technique used to get complex solid objects starting from simple primitives by a series of Boolean operations (union, intersection and difference) [15]. Sophisticated objects are produced by combining a relatively small number of primitives such as spheres, cones and cuboids. The primary tool used when dealing with Boolean operations in the literature is the R-function, which is employed to produce the final algebraic representation [16, 17].

V. L. Rvachev, originally introduced R-functions, or Rvachev functions, in the 1960s by combining logic, geometry and analysis under a unified framework [17, 18]. The method gives an algebraic representation for shapes obtained from the combinations of simple shapes through set-theoretic operations. Since then, the related theory, as well as the applications, have been extensively explored [19–22]. R-functions are applied in many areas including computer graphics, engineering design, analysis and optimization [23–26]. In geometric modeling, R-functions are used when constructing objects with complex structures (e.g., in [13, 16]). Moreover, these functions are adopted to define the geometry when solving boundary-value problems in computational physics [18, 27, 28].

Among different systems of R-functions, R_0-function is the most popular one regarding applications [17, 18]. In [19], higher smoothness of R-functions has been attained by developing a generalized R_0^m system. However, there are square root operations in such representations. Furthermore, addition and multiplication are not enough to construct R-function systems of polynomials that are sufficiently complete. According to [19], a sufficiently complete system do not need to use the root operation and R-functions can be constructed from piecewise polynomials without the need for the root operation. This leads to a good direction to present systems that includes complete Boolean operations and avoids the square root operations at the same time.

In the Computer-Aided Design (CAD) system, the dominant geometrical representation uses spline functions [29]. The growing interest in splines is due to their simple implementation and nice properties. Different types of splines have been introduced in the recent years with various studies for their proprties [30–33]. In this paper, a R-function system represented in spline form is proposed. The starting point is based on the definition of R-function, where specific Boolean operations have fixed signs that depend on the signs of their base primitives. Afterward, a straightforward procedure to map such fixed signs into a new coordinate system is conducted. One of the motivations for this work is to adopt such simple mapping in order to represent the R-function by splines. Moreover, the composition operation with splines is also convenient and can be handled well by the de Boor algorithm [29]. A detailed construction of spline R-function is presented for both Boolean intersection and union operations. In each case, different smoothness are illustrated to indicate the simplicity and practicality of the new mechanism. Results show that this new system of R-functions satisfies the requirements of the original R-functions. Additionally, the proposed spline R-functions prevent the ambiguity zeros inside the interior of the domain since no singularities appear in the representation.

The remainder of the paper is organized as follows. Section 2 presents the preliminary information, including R-functions, the B-net method, B-splines and a brief introduction of WEB-Spline FEM. Section 3 is dedicated to the construction of spline R-function. Experimental results and comparisons are presented in Section 4. Finally, Section 5 concludes the paper and suggests possible future topics.

2. Preliminaries

In this section, we review the definition of R-functions with some properties as mentioned in [17, 18]. Also, we briefly review the core ideas related to B-net method, B-splines and the finite element solution structure for WEB-splines due to their relationship with the topic presented in this work. The interested reader is referred to [25, 29, 30, 34, 35] and references therein for more details on such topics.

2.1. R-function

In this subsection, we recall the definition of the R-function and some accompanying properties as mentioned in [18]. A standard R-function can be defined as follows:

Definition 2.1 ([18]). *A function $y = f(x_1, \cdots, x_n)$ is called an R-function if in the specification of a group, the signs of its arguments completely determine the function sign, i.e., if there exists a Boolean function $Y = F(X_1, \cdots, X_n)$ that determines the dependence of the sign of function $f(x_1, \cdots, x_n)$ on the group of the signs of its arguments. Namely, if*

$$S_2(x) = \begin{cases} 0, & x < 0, \\ 1, & x > 0, \end{cases}$$

then

$$F(S_2(x_1), S_2(x_2), \cdots, S_2(x_n)) = S_2(f(x_1, x_2, \cdots, x_n)).$$

F is called an accompanying function for f.

Following the definition, the sign of the real-valued function will stay unchanged unless the arguments change signs.

Now, given two base primitives x and y represented by inequalities of implicit functions, it should be clear that the R-function is not unique and exhibits a rich variety of differential properties. Several systems of R-functions constructed by the founder of the theory are used in various applications, such as the work in [13, 16, 18–20, 36]. There exists a system of R-functions that is defined as follows:

$$R_\alpha : \quad \frac{1}{1+\alpha}\left(x + y \pm \sqrt{x^2 + y^2 - 2\alpha xy}\right),$$

where $\alpha = \alpha(x, y)$ is an arbitrary continuous function satisfying $-1 < \alpha(x, y) \leq 1$ and $(\pm)$ indicates the formulation for the Boolean intersection $(-)$ and union case $(+)$, respectively.

By setting different values for α, we get different systems of R_α-functions. Among them, the R_0 and R_1 functions are the most widely used ones in applications. The two functions are defined respectively such that:

$$\begin{aligned} R_0: &\quad x+y\pm\sqrt{x^2+y^2}, \\ R_1: &\quad x+y\pm|x-y|, \quad (i.e.,\ \min(x,y),\max(x,y)). \end{aligned}$$

In both cases, the existence of derivative is not always guaranteed. For example, while R_1 has the simplest form, yet it is not differentiable along the line $x=y$. Likewise, R_0 is not differentiable when $x=y=0$. R_0 is more usable in practice because of its higher smoothness than R_1.

In [37], a modified R_0 function is proposed and to overcome the differentiation weakness of the original R_0 and increases the smoothness. The differentiation properties of the function is discussed further in [20]. R_0^m is m-times differentiable everywhere and can be defined as

$$R_0^m: \quad (x+y\pm\sqrt{x^2+y^2})(x^2+y^2)^{\frac{m}{2}}, \ \ m \text{ is any even positive integer.}$$

Another type of generalized R-functions discussed in [20] and references therein. R_p can be written as:

$$R_p: \quad x+y\pm(x^p+y^p)^{\frac{1}{p}}, \ \ p \text{ is an even positive integer.}$$

The several types of R-functions are used in many areas of applications due to their mathematical properties and the natural ability to represent complex geometrical models. Partial differential equations are a crucial part of engineering problems where R-functions can serve as a tool to model sophisticated geometries before solving the related Dirichlet boundary problems. For example, the R-function Method (RFM) introduced in [21, 22] proposes a solution structure to satisfy the exact boundary condition. Also, in the Weighted Extended B-spline Method (WEB) [25, 35], R-function is adopted as the weight function when concerning problems over complex domains. Moreover, in artificial intelligence, R-functions can be used in pattern recognition [26] to handle design specifications. Additional details of R-function and its applications can be found in [17, 19] and the references therein.

2.2. B-net method

When considering multivariate splines, the B-net method is considered one of the most important approaches especially after G. Farin [29] established the relationship between smoothness conditions and the Bézier coordinates. In this subsection, we briefly describe the B-net method. For a more detailed discussion, the interested reader can turn to [38].

Let $f_1(x,y)$ and $f_2(x,y)$ be two functions with bidegree (m,n), defined over two adjacent domains D_1 and D_2, respectively, where $D_1=[x_0,x_1]\times[y_0,y_1]$, $D_2=[x_1,x_2]\times$

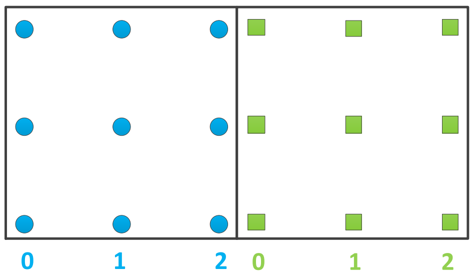


Figure 1: The Bézier ordinates of two biquadratic polynomials.

$[y_0, y_1]$. Then they can be represented in the Bernstein-Bézier form as [39]:

$$f_l(x,y) = \sum_{i=0}^{m}\sum_{j=0}^{n} b_{ij}^l B_i^m\left(\frac{x-x_{k-1}}{x_l-x_{l-1}}\right) B_j^n\left(\frac{y-y_0}{y_1-y_0}\right), \tag{2.1}$$

where $B_i^m(t)$ and $B_j^n(t)$ are the Bernstein polynomials, $\{b_{i,j}^l, l=1,2\}$ are called the Bézier ordinates of $f_1(x,y)$ and $f_2(x,y)$, respectively.

It is well known that $f_1(x,y)$ and $f_2(x,y)$ are r-time differentiable across their common boundary if and only if

$$\frac{1}{(x_1-x_0)^i}\Delta^{i,0}b_{m-i,j}^1 = \frac{1}{(x_2-x_1)^i}\Delta^{i,0}b_{0,j}^2, \tag{2.2}$$

where $j=0,\cdots,n$, $i=0,\cdots,r$, $\Delta^{i,0}b_{j,k} = \Delta^{i-1,0}b_{j+1,k} - \Delta^{i-1,0}b_{j,k}$ with $\Delta^{0,0}b_{j,k} = b_{j,k}$.

Fig. 1 shows an example to illustrate the geometric meaning. Two biquadratic Bézier functions are defined over adjacent domains, which are shown with markers of blue circle and green square, respectively. Following the Eq. (2.2), if $f_1(x,y)$ and $f_2(x,y)$ are C^1 continuous across their common boundary, then the 2nd column of the circular ordinates should be coincide with the 0th square ordinates and the 1st square ordinates are determined by the 1st and 2nd ordinates of the circular ordinates.

2.3. B-splines

Definition 2.2 (B-spline Basis Function). *Let $T = \{t_0,\cdots,t_{n+k}\}$ be a nondecreasing sequence of real numbers, i.e., $t_i \le t_{i+1}$, $i=0,\cdots,n+k-1$. The t_i are called knots and T is the knot vector. The ith B-spline basis function of order k (of degree $k-1$) defined recursively using the Cox-de Boor recursion formula [30] is as follows :*

$$N_i^1(t) = \begin{cases} 1, & \textit{for} \quad t_i \le t < t_{i+1}, \\ 0, & \textit{otherwise}, \end{cases} \tag{2.3}$$

for $k=1$ and

$$N_i^k(t) = \frac{t-t_i}{t_{i+k-1}-t_i}N_i^{k-1}(t) + \frac{t_{i+k}-t}{t_{i+k}-t_{i+1}}N_{i+1}^{k-1}(t), \tag{2.4}$$

for $k>1$ and $i=0,1,2,\cdots,n$.

Several basic properties of B-spline basis function [30] related are listed below:

(a) Local support: $N_i^k(t) \equiv 0, t \notin [t_i, t_{i+k})$.

(b) Nonnegativity: $N_i^k \geq 0, \forall t, i, k$.

(c) Partition of unity: $\sum\limits_{i=0}^{n} N_i^k(t) \equiv 1, t \in [t_{k-1}, t_{n+1}]$.

(d) Smoothness: $N_i^k(t)$ is C^{k-m_j-1} at $t = t_j$, where m_j is the multiplicity of t_j in the knot vector.

Definition 2.3 (B-Spline Functions). *A B-spline function is made up of the linear combinations of the products of B-spline basis functions and control points as follows:*

$$F(t,s) = \sum_{i=0}^{m}\sum_{j=0}^{n} d_{ij} N_i^{k_1}(t)\, N_j^{k_2}(s)\,, \tag{2.5}$$

where k_1 and k_2 are the orders of B-spline basis in two directions and the knot vectors are $T = \{t_0, t_1, \cdots, t_{m+k_1+1}\}$, and $S = \{s_0, s_1, \cdots, s_{n+k_2+1}\}$, respectively.

Notice that the control points of B-splines have a close relation with the Bézier ordinates: If one extract Bézier patches from a given B-spline surface, then the obtained coefficients are just the corresponding Bezier ordinates on that patch.

2.4. WEB-Spline FEM

The WEB method was proposed in 2001 by K. Höllig and coworkers [35] to overcome the mesh generation difficulty in standard FEM. The WEB method introduces a new weighted extended finite element space. The basis functions used in the analysis part are weighted extended B-splines (WEB-splines) that work on a tensor product grid enclosing the domain of interest and attain all basic properties required by finite element formulations [40]. Many considerable advantages are presented in WEB method such as the accurate approximations with relatively smaller number of elements, the arbitrary smoothness that is related to the order of B-splines chosen, the simplicity of the uniform grid and most importantly is omitting the time consuming mesh generation. The utilization of a regular grid eliminates the need for a prepossessing step and allows an efficient employment of the related algorithms [25].

To overcome the imposition of the Dirichlet boundary conditions that usually occur when solving problems on regular grids, the WEB method proposes a weight function to model the homogeneous boundaries and to strongly impose the boundary conditions. For complex domains, R-functions are used to model the computational domain. The use of R-functions have proven to be very successful for representing sophisticated geometries for analysis in the framework of WEB method. At the same time, the WEB method revealed the powerful properties of R-functions when representing sophisticated geometries for problems solved by the finite element method.

In this subsection, we briefly recall the related basis functions for analysis in the WEB method [25, 35]. Let us consider the following Poisson's problem with Dirichlet boundary conditions as a model problem:

$$\begin{cases} -\Delta u(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) = 0, & \mathbf{x} \in \partial\Omega, \end{cases} \tag{2.6}$$

where $\Omega \subset \mathbb{R}^d$ is a bounded domain defined by $w(\mathbf{x})$:

$$\Omega := \{\mathbf{x} | w(\mathbf{x}) \geq 0,\ \mathbf{x} \in \mathbb{R}^d\}, \qquad d = 2, 3.$$

The weak form of problem (2.6) is: find $u \in H^1(\Omega), u|_{\partial\Omega} = 0$, such that, for all $v \in H_0^1(\Omega)$,

$$a(u, v) = f(v),$$

where $a(u, v) = \int_\Omega \nabla u \cdot \nabla v d\Omega$ and $f(v) = \int_\Omega f v d\Omega$.

The numerical solution in WEB is $u_h = \sum\limits_i a_i B_i$, where the weighted extended B-spline basis has the form

$$B_i = \frac{w}{w(\mathbf{x_i})} \Big[b_i + \sum_{j \in J(i)} e_{i,j} b_j \Big], \qquad i \in I, \quad j \in J,$$

where I and J are the sets saving the inner and outer basis index, respectively. w is the weight function, b_i and b_j are inner and boundary splines, respectively and $e_{i,j}$ represents the extension coefficients.

Fig. 2 gives a two-dimensional example, where the domain Ω is highlighted and its boundary is defined by $w(\mathbf{x}) = 0$. Higher dimensional cases are direct generalization of the two dimensional case. The B-spline bases are constructed over a rectangular domain enclosing the domain, where uniformly distributed knots are selected in two directions (repeated knots are used at the ends of knots). Within the knots in two directions, the rectangular domain has been partitioned into small grids, as the dashed lines depicts in Fig. 2. The overall grid is called the background grid and the smallest grid is a cell. The grid cells can be partitioned into three types: interior, boundary and exterior cells. The cells fully inside (outside) the domain are interior (exterior) cells, while the cells partly inside the domain are called boundary cells.

In WEB method, only B-spline basis whose support intersects the domain is considered. The relevant B-spline bases will be classified into inter B-splines and outer B-splines. The rule to clarify the type of a B-spline basis, is to count the numbers of interior cells in the support of the basis. If there is at least one interior cell inside the support of the B-spline basis, then it is an inner B-spline basis; or else, it is an outer B-spline basis. As shown in Fig. 2, the relevant B-spline bases are marked with small dots at the lower-left corner of their corresponding supports. Different colors of dot marks indicate the classification of the biquadratic B-spline bases: the inter B-splines in black and the outer B-splines in white. Specially, b_i and b_j are inter and outer B-spline bases,

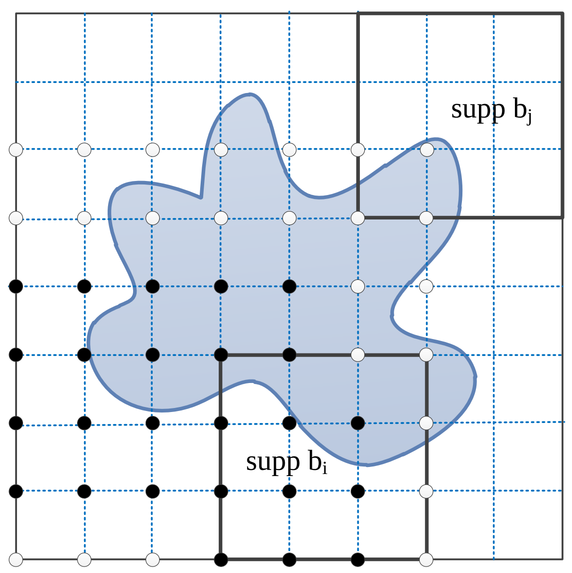


Figure 2: Classification of biquadratic B-splines in WEB method.

respectively. The supports of the two bases are highlighted, from which the types are obtained directly from the numbers of interior cells in each basis's support.

For smooth solutions, the authors have given the theoretical error estimate for WEB spaces and can be found in [25].

For the WEB method, the weight function is a smooth function that depicts the domain. That is, the function value is positive inside the domain, negative outside the domain and vanishes on the boundary $\partial\Omega$. Moreover, it also plays the role of the weight function in the numerical solution to exactly satisfy the boundary conditions. R_0 function has been used to describe complex domains composed by simple base primitives.

In this paper, we use the proposed spline R-function to represent complex domains for partial differential equations solved under the WEB method framework. The spline R-function is consistent with CAD representations and provides a new connection between the geometry and analysis parts in the finite element scheme. In addition to the geometrical advantages,the spline R-function serves as a mediator to an automatic transfer from the CAD geometry to an immediate finite element analysis.

3. Method

In this section, we discuss in detail the construction process for the spline R-function. To start, recall the problem of constructing a shape using the Boolean operations for 2D objects as follows:

Let $u = \{(x,y)|f_1(x,y) \geq 0\}$ and $v = \{(x,y)|f_2(x,y) \geq 0\}$ be two base primitives.

Now our task is to get the representation for the shape after taking the Boolean operation (intersection $\cap$ or union $\cup$) over the two primitives. That is, to derive an implicit function $J(x, y)$ for the shape of $u \cap v$ or $u \cup v$, such that, the set $\{(x, y) | J(x, y) \geq 0\}$ exactly depicts the shape.

In contrast to the classical R_0 system, the real function $J(x, y)$ in our construction is represented in a composed spline form as shown in Section 3.1.1. For convenience, we use SR as the abbreviation for the constructed spline R-function. Note that since both primitives are defined by implicit functions, the difference operation can be obtained easily through the Boolean intersection operation. For example, considering two primitives A and B defined by implicit functions, the object produced by the difference $A - B$ can also be produced by the intersection of A and the complement of B in the domain. Therefore, only intersection and union operations are considered here.

3.1. SR construction process

Before we proceed to consider the specific Boolean operation, let us first go over the main assumption for the construction of SR. Spline R-function for both Boolean intersection and union operations will commit to the same function range selection below.

We assume that the function for both primitives are inside $[-1, 1]$. If this is not the case, a simple scaling process is applied. This does not affect the result since an implicit form is used and thus, $w = 0$ and $cw = 0$, $c > 0 \in R$ lead to the same contour curve. The domain is preserved to be the same after the scaling operation.

3.1.1. Function representation

Let $t := f_1(x, y)$ and $s := f_2(x, y)$, a new two-dimensional Cartesian system is constructed, by setting t as the horizontal axis, s as the vertical axis and the intersection of $s = 0$ and $t = 0$ as the origin O. According to the definition of R-function, it can be found that the sign constraints of $J(x, y)$ are consistent with the correct signs of $F(t, s)$ in the new coordinate system. Hence, we can define $J(x, y)$ with the following formula:

$$J(x, y) := F(t, s) = F(f_1(x, y), f_2(x, y)). \tag{3.1}$$

To clarify the correspondance of sign constraints, we take the Boolean intersection operation as an example. Let the two primitives u and v defined implicitly above be two circles, respectively. As shown in Fig. 3, the circles are intersected and the domain is partitioned into four regions I–IV. Our goal is to get an implicit representation $J(x, y)$ for the intersection of the circles.

By definition, $J(x, y)$ shall be zero along the boundary of the region I, be positive inside and negative outside. Notice that for the region I, we have $f_1(x, y) \geq 0$, $f_2(x, y) \geq 0$ and $J(x, y) \geq 0$. If we consider the above relations in the newly constructed tOs plane, then the region I corresponds to the first quadrant, where $J(x, y) \geq 0$. In other words, in the first quadrant, there is $F(f_1(x, y), f_2(x, y)) = F(t, s) > 0$.

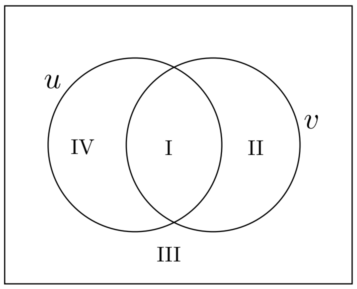


Figure 3: An example for composite view of sign constraints for Boolean operations.

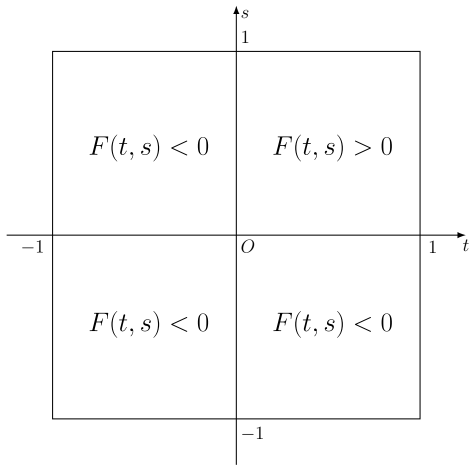


Figure 4: Sign constraints for $\cap$ in composite view.

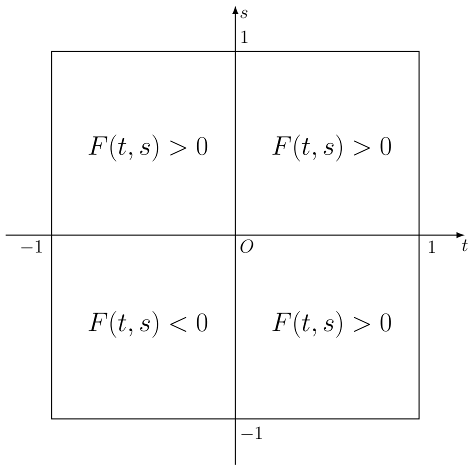


Figure 5: Sign constraints for $\cup$ in composite view.

Similarly, The region II–IV correspond to the quadrants II–IV, separately. The sign constraints for union operation can be set following the same fashion. In this way, the original SR problem can be solved by constructing functions with sign assignments on the composed tOs plane.

Fig. 4 and Fig. 5 highlight the sign constraints for the Boolean intersection and union operations, respectively. The function composition is considered as a special case of a binary relations. Consequently, SR is formulated by spline functions following the corresponding sign constraints. The specific construction procedure is discussed with more details in Section 3.1.2.

Representation of SR in both Bézier form and B-splines will be considered. For SR represented in Bézier form, the B-net method is utilized on tOs plane. Although the SR-function can be applied to any arbitrary bidegree, we restrict our discussion to SR with bidegrees $(n,n), n=1,2,3$. The Bézier ordinates at each quadrant can be determined by applying the convex-hull property and the end-interpolation property of the Bézier function. For SR represented in B-spline form, we use tensor-product B-spline function on tOs plane. Since both the Bézier form and the B-spline form yield the same representation of SR, for B-spline case we just give an overall description. In general, the main difference between the Bézier and B-spline approach is the way the continuity constraints enforced.

3.1.2. SR Criteria setting

Let us first consider the criteria for SR represented in Bézier form. Following the related constraints of classical R-function, a group of criteria for constructing SR are obtained. To facilitate the later descriptions, these criteria are summed up and listed below:

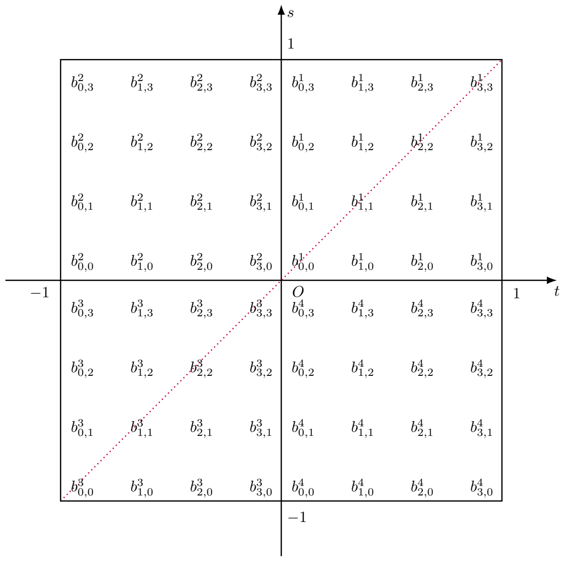
(a) Sign assignments: including both axes and quadrants.

(b) Commutation: $F(t,s) = F(s,t)$.

(c) The continuity between the four Bézier segments.

Criterion (a) is natural following the definition of R-function. As mentioned before, the constructed SR should have correct signs both at different axes and quadrants for both intersection and union operations. Take the constraints for the Boolean intersection operation in Fig. 4 as an example, $F(t,s)$ has positive sign in the first quadrant in the new coordinate system and negative in the other three. To conserve the exact boundary after the Boolean operation, $F(t,s)$ shall be zero at the positive half axes while be strictly negative at the negative half axes.

Criterion (b) is required because SR should not be affected by the input sequence of the two primitives. Take the bicubic SR in Fig. 6 as an example, where there are



s
t
1
-1
1
-1
O
$b^2_{0,3}$ $b^2_{1,3}$ $b^2_{2,3}$ $b^2_{3,3}$ $b^1_{0,3}$ $b^1_{1,3}$ $b^1_{2,3}$ $b^1_{3,3}$
$b^2_{0,2}$ $b^2_{1,2}$ $b^2_{2,2}$ $b^2_{3,2}$ $b^1_{0,2}$ $b^1_{1,2}$ $b^1_{2,2}$ $b^1_{3,2}$
$b^2_{0,1}$ $b^2_{1,1}$ $b^2_{2,1}$ $b^2_{3,1}$ $b^1_{0,1}$ $b^1_{1,1}$ $b^1_{2,1}$ $b^1_{3,1}$
$b^2_{0,0}$ $b^2_{1,0}$ $b^2_{2,0}$ $b^2_{3,0}$ $b^1_{0,0}$ $b^1_{1,0}$ $b^1_{2,0}$ $b^1_{3,0}$
$b^3_{0,3}$ $b^3_{1,3}$ $b^3_{2,3}$ $b^3_{3,3}$ $b^4_{0,3}$ $b^4_{1,3}$ $b^4_{2,3}$ $b^4_{3,3}$
$b^3_{0,2}$ $b^3_{1,2}$ $b^3_{2,2}$ $b^3_{3,2}$ $b^4_{0,2}$ $b^4_{1,2}$ $b^4_{2,2}$ $b^4_{3,2}$
$b^3_{0,1}$ $b^3_{1,1}$ $b^3_{2,1}$ $b^3_{3,1}$ $b^4_{0,1}$ $b^4_{1,1}$ $b^4_{2,1}$ $b^4_{3,1}$
$b^3_{0,0}$ $b^3_{1,0}$ $b^3_{2,0}$ $b^3_{3,0}$ $b^4_{0,0}$ $b^4_{1,0}$ $b^4_{2,0}$ $b^4_{3,0}$

Figure 6: The Bézier ordinates at different quadrants on tOs plane (the bicubic case).

$(3+1)^2 = 16$ Bézier ordinates in each quadrant. As the dashed left diagonal shows, it partitions the Bézier ordinates into two groups according to this criterion. Thus, the unknown ordinates will be mostly reduced by a half.

Criterion (c) originates from the function continuity of SR. For SR with bidegree (m,n), the smoothness along the common axis between every pair of the adjacent quadrants are $C^{m-1,n-1}$. In Fig. 6, the bicubic SR is constructed and a $C^{2,2}$ smoothness is satisfied along the common axes between different quadrants. With Eq. (2.2), the related constraints can be attained to determine the Bézier ordinates in each quadrant.

Given the surface degree, not all the exact values of the coefficients can be determined following the above criteria. Those remaining coefficients are to be determined with some freedom. To describe the details clearly, SR in Bézier form for both the Boolean intersection and union operations are discussed in the remainder of this Sections 3.2–3.3.

In general, the same construction mechanism can be adopted to express the SR-function using B-splines. Criterion (c) will be automatically satisfied by the continuity of the B-spline functions, so only Criteria (a) and (b) need to be considered. Representation using B-splines will lead to a representation with the same numbers of free variables as representation in Bézier form and will represent the same SR. The process of constructing SR in B-spline form is given at Section 3.4.

3.2. SR in Bézier form for Boolean intersection operation

This subsection considers the construction of SR for shapes derived from the Boolean intersection ($w = u \cap v$). To simplify the illustration, we use separated notations to denote the Bézier functions at each quadrant on tOs plane. Following the notation of the B-net method (2.1), the SR is defined as follows:

$$F(t,s) = \begin{cases} I_1(t,s) = \sum_i^m \sum_j^n b_{ij}^1 B_i^m(t) B_j^n(s), & \text{if } (t,s) \in [0,1] \times [0,1], \\ I_2(t,s) = \sum_i^m \sum_j^n b_{ij}^2 B_i^m(t+1) B_j^n(s), & \text{if } (t,s) \in [-1,0) \times [0,1], \\ I_3(t,s) = \sum_i^m \sum_j^n b_{ij}^3 B_i^m(t+1) B_j^n(s+1), & \text{if } (t,s) \in [-1,0) \times [-1,0), \\ I_4(t,s) = \sum_i^m \sum_j^n b_{ij}^4 B_i^m(t) B_j^n(s+1), & \text{if } (t,s) \in [0,1] \times [-1,0). \end{cases} \tag{3.2}$$

Depending on Criterion (a), the function values of $F(t,s)$ for the Boolean intersection operation is positive in the first quadrant and negative in other three quadrants as discussed in Subsection 3.1.1. The corresponding sign constraints in the four quadrants are shown below in Expressions (3.3a) and (3.3b)

$$I_1(t,s) > 0, \quad I_2(t,s) < 0, \tag{3.3a}$$

$$I_3(t,s) < 0, \quad I_4(t,s) < 0. \tag{3.3b}$$

As to the sign constraints on the axes, two cases exist. On one hand, since the boundary of $u \cap v$ should be exactly preserved, the values at the points along the positive half axes should be zero. On the other hand, function values should have strictly negative signs along the negative half axes. Expressions (3.4a)–(3.4d) show the function relations on the four half axes:

$$I(t,0)=0, \quad t\in[0,1], \tag{3.4a}$$
$$I(0,s)=0, \quad s\in[0,1], \tag{3.4b}$$
$$I(t,0)<0, \quad t\in[-1,0), \tag{3.4c}$$
$$I(0,s)<0, \quad s\in[-1,0). \tag{3.4d}$$

Criterion (b) requires that $F(t,s)=F(s,t)$. Depending on this stipulation, Expressions (3.5a)–(3.5c) show the corresponding relations. Notice that when (t,s) is in the second quadrant, the corresponding function of $I_2(t,s)$ is $I_4(s,t)$

$$I_1(t,s)=I_1(s,t), \quad t>0, \quad s>0, \tag{3.5a}$$
$$I_2(t,s)=I_4(s,t), \quad t<0, \quad s>0, \tag{3.5b}$$
$$I_3(t,s)=I_3(s,t), \quad t<0, \quad s<0. \tag{3.5c}$$

Given the surface degree, Criterion (c) is considered in a case by case manner. The specific SR required is then derived after employing all the criteria.

3.2.1. Bilinear case

According to the B-net method, there are four Bézier ordinates at each quadrant for the bilinear SR. Criterion (c) indicates $C^{0,0}$ smoothness along the positive axes. According to Formula (2.1) of the B-net method, relations between the Bézier ordinates in different quadrants are derived. That is,

$$b^2_{1,j}=b^1_{0,j}, \quad j=0,1, \quad b^4_{j,1}=b^1_{j,0}, \quad j=0,1, \tag{3.6a}$$
$$b^3_{j,1}=b^2_{j,0}, \quad j=0,1, \quad b^3_{1,j}=b^4_{0,j}, \quad j=0,1. \tag{3.6b}$$

For Expressions (3.5a)–(3.5c), half of the Bézier ordinates need to be determined. From Expressions (3.4a), (3.4b) and (3.6a), it can be obtained that all the Bézier ordinates appear in (3.6a) are valued zero. Furthermore, concerning (3.6b), only $b^1_{1,1}$, $b^4_{1,0}$, $b^4_{0,0}$ and $b^3_{0,0}$ are left to be assigned. These free variables can be set following the correct signs according to (3.3a), (3.3b), (3.4c), (3.4d). In this way, we get the bilinear SR for the Boolean intersection operation.

3.2.2. Biquadratic case

For biquadratic SR, we have $C^{1,1}$ smoothness along four half axes. Following Formula (2.1), the relations considering the $C^{1,1}$ smoothness are listed below:

$$b^2_{1,j}=b^1_{0,j}, \qquad j=0,1,2, \qquad b^4_{j,1}=b^1_{j,0}, \qquad j=0,1,2, \tag{3.7a}$$

$$b^3_{j,1} = b^2_{j,0}, \qquad j = 0,1,2, \qquad b^3_{1,j} = b^4_{0,j}, \qquad j = 0,1,2, \tag{3.7b}$$
$$2b^3_{j,2} = b^3_{j,1} + b^2_{j,1}, \qquad j = 0,1,2, \qquad 2b^3_{2,j} = b^3_{1,j} + b^4_{1,j}, \qquad j = 0,1,2, \tag{3.7c}$$
$$2b^2_{2,j} = b^2_{1,j} + b^1_{1,j}, \qquad j = 0,1,2, \qquad 2b^4_{j,2} = b^4_{j,1} + b^1_{j,1}, \qquad j = 0,1,2. \tag{3.7d}$$

Combining all the relations of Expressions (3.3a)–(3.5c) and (3.7a)–(3.7d), some Bézier ordinates are determined to have values zero. In addition, it can be found that the Bézier ordinate $b^1_{1,1}$ in the first quadrant has the same value as $b^3_{1,1}$ at the third quadrant. However, it conflicts with the relations obtained from Expressions (3.3a) and (3.3b). Since they should have the opposite signs because of the sign constraints. Thus, $b^1_{1,1} = b^3_{1,1} = 0$. Expressions (3.7a) and (3.7d) indicate that the Bézier ordinates are valued zero. The final free ordinates to be determined are $b^1_{2,2}$, $b^1_{2,1}$, $b^4_{0,0}$, $b^4_{1,0}$, $b^4_{2,0}$ and $b^3_{0,0}$. Correct sign assignments will lead to a sample of biquadratic SR. Moreover, it is guaranteed that the final representation preserves the inner region.

3.2.3. Bicubic case

When concerning the bicubic SR for Boolean intersection operation, the relations for $C^{2,2}$ continuity can be derived analogously. However, we do not list the lengthy relations here.

After applying relations (3.3a)–(3.5c) to the definition of SR in (3.2), several exact values and ranges of some coefficients are derived. During the solving process and similar to the biquadratic case, there exists some difficulty during the process for determining other Bézier ordinates of the bicubic SR. In some cases, there are relations indicating an ordinate with the opposite signs in the same quadrant. In order to overcome this problem, we set that ordinate to be zero. In this way, the numbers of unknown Bézier ordinates can be further reduced.

To get the final representation, additional information about the remained free Bézier ordinates shall be imposed. This can be achieved by simply assigning the correct sign for each parameter. Although no unique values for those Bézier ordinates, the correct signs of coefficients guarantee that the SR is well-defined.

3.3. SR in Bézier form for Boolean union operation

The above subsection illustrates the process of constructing SR with different smoothness for the Boolean intersection operation. Now let us consider the case of SR for Boolean union $u \cup v$. Considering the same group of symbols, the construction process is completely analogous. That is, to find the relations between the Bézier ordinates, according to the criteria (a)–(c).

Considering Criterion (a), we know that $F(t,s)$ should have negative sign if the point (t,s) is in the third quadrant and have positive sign if (t,s) is in other three quadrants. The sign constraints at each quadrant for the union case can be labeled in the following form:

$$I_1(t,s) > 0, \quad I_2(t,s) > 0, \tag{3.8a}$$
$$I_3(t,s) < 0, \quad I_4(t,s) > 0. \tag{3.8b}$$

Notice that the boundary curve of $t \cup s$ should still be zero, the function values on the negative half axes are zero, while the values on the positive half axes shall have positive signs. This leads to the following Expressions:

$$I(t,0) > 0, \quad t \in [0,1], \tag{3.9a}$$
$$I(0,s) > 0, \quad s \in [0,1], \tag{3.9b}$$
$$I(t,0) = 0, \quad t \in [-1,0), \tag{3.9c}$$
$$I(0,s) = 0, \quad s \in [-1,0). \tag{3.9d}$$

The relations following Criterion (b) is the same as the intersection case.

Subsequent to all the relations shown in Expressions (3.5a)–(3.5c) and (3.8a)–(3.9d), several ordinates are determined uniquely under the same token. To determine other Bézier ordinates, the main tool is the continuity of SR. If bilinear SR is considered, $C^{0,0}$ continuity is satisfied along the four half axes. Since in each quadrant there are four Bézier ordinates, applying all the relations above, only $b^1_{1,1}$, $b^1_{1,0}$, $b^4_{1,0}$ and $b^3_{0,0}$ are remained to be assigned. With simple sign assignment, we get the bilinear SR. For biquadratic and bicubic SR, there are relations as in the Boolean intersection case, where there exist ordinates that have signs that do not align with each other. This will facilitate the determination of these free ordinates. As previously mentioned, the corresponding coefficients can be determined by assigning the correct signs.

3.4. SR in B-spline form for Boolean operations

For SR represented in B-spline form, we construct tensor product B-spline function $F(t,s)$ on the composed tOs plane with specific degrees and the corresponding knot vectors, where t and s are the same with that in the Bézier case. Let us assume that in each direction, the order is k (degree $+\,1$) and the associated knot vector is

$$T = \{\underbrace{-1,\cdots,-1}_{k}, 0, \underbrace{1,\cdots,1}_{k}\},$$

then the function can be formulated as follows:

$$F(t,s) = \sum_{i=0}^{k}\sum_{j=0}^{k} d_{ij} N_i^k(t)\, N_j^k(s), \qquad (t,s) \in [-1,1] \times [-1,1]. \tag{3.10}$$

As an example, we construct SR in B-spline form for Boolean intersection operation. Following Criterion (b), we obtain the symmetric relations between coefficients. Since we use the same groups of knot vectors, the symmetry of the coefficients with axis

$t-s=0$ are observed from Expressions (3.5a)–(3.5c). Thus, just half of the coefficients need to be determined. According to Criterion (a), we need to consider the constraints on both axes and quadrants. Only conditions in Expressions (3.3a), (3.3b), (3.4a) and (3.4d) need to be implemented. The corresponding relations are obtained by solving the related equations and inequalities following the above criteria.

Now, let us set $k=2$ and consider representation in the bilinear B-spline form. The knot vectors in both t and s directions are set to be $T=\{-1,-1,0,1,1\}$, so the function becomes:

$$F(t,s)=\sum_{i=0}^{2}\sum_{j=0}^{2} d_{ij}N_i^2\left(t\right)N_j^2\left(s\right), \qquad (t,s)\in[-1,1]\times[-1,1]. \tag{3.11}$$

After applying Criterion (b), Expressions (3.3a), (3.3b), (3.4a) and (3.4d) are considered. The relations between the coefficients are derived by solving the equations and inequalities and the final free coefficients are: $d_{0,0}$, $d_{1,0}$, $d_{2,0}$, $d_{2,2}$, each with an inequality constraint on the function sign.

For the formulation of SR for the Boolean union operation in B-spline form, in view of the choice of knots and degrees in two directions, we get the function $F(t,s)$ (3.10) with symmetry in t and s. It can be observed that on the tOs plane, the required sign constraints for the union case can be fulfilled by mirroring the coefficients in the intersection case around the axis $t+s=0$. The construction for SR with B-splines for other degrees can be constructed in the same fashion.

It has been proved in [18, 19] that the system is complete once the companion Boolean operations are complete. The completeness of the SR-function presented in both Bézier form and B-spline form in this paper can be confirmed in the same way by adopting the steps presented for both intersection and union operations to get the resulting SR representation.

4. Numerical examples

In this section, we present several computational experiments to illustrate the use of SR when solving partial differential equations. For demonstration purpose, we solve the Poisson equation in (2.6) using the WEB method [25]. Considering the equivalence between the two forms of SR, the Bézier form is tested.

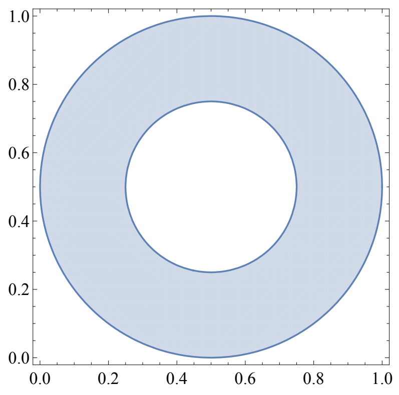
The domain $\Omega\subset\mathbb{R}^d$ in each example is a bounded domain defined by the SR-function $w(\mathbf{x}),\mathbf{x}\in\mathbb{R}^d$ and compared to the case where the weight function is constructed using R_0- function. Following the theory of the WEB method [25], the numerical solution for the Poisson equation is $u_h=\sum\limits_i a_iB_i$, where B_i is the WEB basis.

Four examples are presented on two-dimensional and three-dimensional domains. For two-dimensional problems, SR are constructed for intersection, union and iterated Boolean operations, respectively. To be more specific, a benchmark example of an annular domain is tested in the first example. An interesting union example is then used to test the robustness of the SR, where the primitives have different topological features. Afterwards, a complex shape is constructed to examine the practicability of SR. For the three-dimensional problem, we examine the union operation to observe the behavior of SR. The exact solutions are explicitly given for all examples in order to assess the quality of the numerical solution produced in each example and to simplify the weight function comparisons.

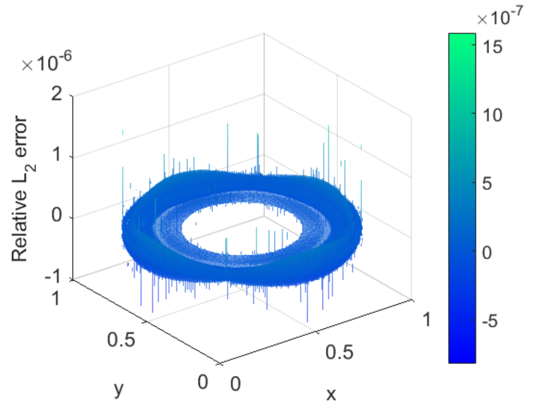
Example 4.1. In this example, we examine the SR for the Boolean intersection operation. We consider an annular domain with circles of the same center but different radii. As illustrated in Fig. 7(a), the positive part of the cubic SR is highlighted to show the computational domain. Here $d=2$, set $\mathbf{x}=(x,y)\in\mathbb{R}^2$. Let $t=0.4^2-(x-0.5)^2-(y-0.5)^2$ and $s=0.1^2-(x-0.5)^2-(y-0.5)^2$ be the implicit representations of the two circles, respectively. Then the domain can be obtained by the Boolean intersection operation of t and $-s$, where the corresponding cubic SR is constructed directly following Section 3. The right-hand side f in the model problem (2.6) is constructed so that the exact solution is

$$u(\mathbf{x})=\sin(t(\mathbf{x})\cdot s(\mathbf{x})).$$

Let 2^l be the grid partition of the background grid in each direction, k be the degree of basis function for each variable, the relative L_2 errors with $l=8, k=3$ for the numerical solution where the weight function uses SR are shown in Fig. 7(b). From the results, we note that the larger errors exist near the boundary, which is consistent with the theoretical predictions for WEB method. Many factors lead to the larger error near the boundaries for approaches solving partial differential equations on structured grids, including the basis functions with small support inside the computational domain and the numerical integration related. For more details, the interested reader is referred to [25,41] for a detailed discussion on this topic.

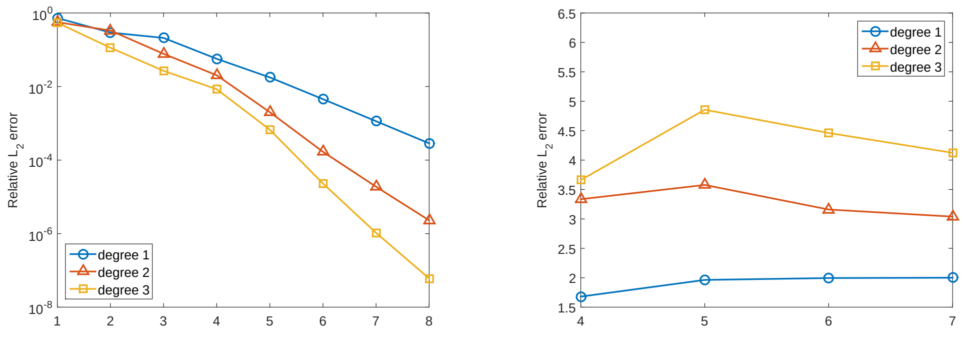


(a) Annular domain



(b) Local error for the numerical solution

Figure 7: Domain and solution local errors using SR in the 2D annular domain in Example 4.1.



(a) Relative L_2 errors of annular domain (b) Convergence rates for annular domain

Figure 8: The relative L_2 errors (left) and convergence rates (right) for two-dimentional annular domain with weight function constructed by SR. Lines in different colors correspond to B-spline degrees $k = 1,2,3$, respectively. The grid width $h = 2^{-l}, l = 1,\cdots,8$.

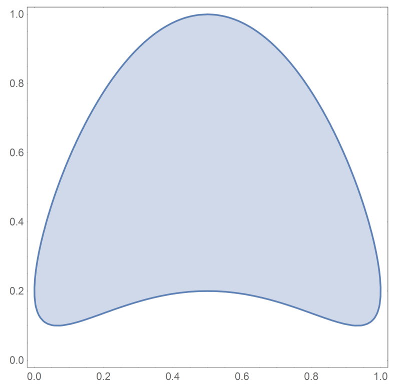
Fig. 8(a) depicts the relative L_2 errors under different degrees and different grid sizes. It can be observed that the decrease of error complies with the theoretical predictions, i.e., the convergence rate is degree $+\ 1$. As illustrated in Fig. 8(b), the convergence rate for each bidegree is consistent with the analysis presented in [25,35] for L_2 norm as the grid width h becomes smaller. Depending on the results, we conclude that the constructed SR satisfies the basic conditions required by R-functions and can be applied successfully to this type of problem.

Example 4.2. The domain Ω for this example is produced by the union of two primitives with different topologies. Fig. 9(a) and 9(b) show the two primitives defined by implicit functions, respectively. The union domain Ω is depicted in Fig. 9(c) and the constructed SR is shown in Fig. 9(d). Let the weight function constructed by SR and R_0 be $w_1(\boldsymbol{x})$ and $w_2(\boldsymbol{x})$, respectively. The right-hand side f in this example is given so that the exact solution is $u = \sin(w_1(\boldsymbol{x}) \cdot w_2(\boldsymbol{x}))$.

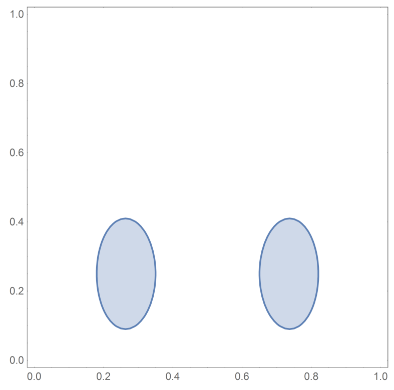
In order to check the validity of the proposed SR in this example, we compare the results with those produced by using R_0-function. The relative L_2 errors and the convergence rates of the numerical solution for a weight function constructed by R_0 and SR are presented in Table 1, where n is the bidegree of the solution space, while l indicates the partitions of the background grid: $h = 1/2^l$, $l = 1,2,\cdots,8$.

Looking at Table 1, we note that both SR and R_0 lead to the correct trends of convergence rates. In particular, the relative L_2 error of using SR is one decimal smaller than R_0 for cases when $k = 2,3$ and $l = 7,8$. The results indicate that SR leads to more accurate results than R_0 for the union shape for higher number of partitions and higher degree of solution space. The reason may be caused by the square root operation in R_0. In fact, from the mathematical expression of R_0 for this union shape, the square root operation is calculated approximately not exactly.

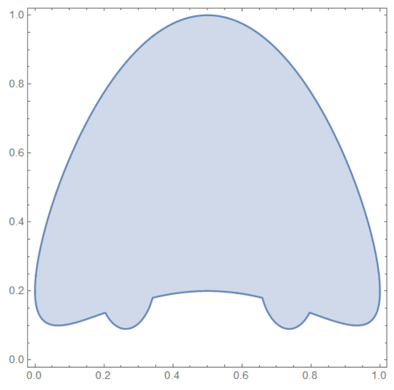
To verify the efficiency of the proposed criteria for SR and observe the behavior of SR with different Bézier ordinates, different groups of free Bézier ordinates need to



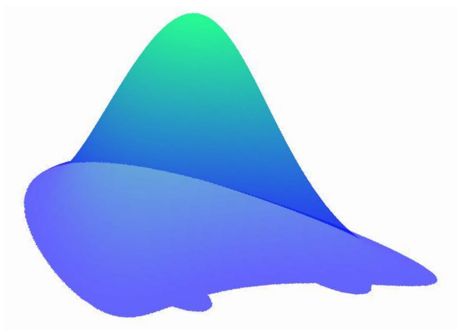
(a) Primitive A



(b) Primitive B



(c) Union domain



(d) SR for union

Figure 9: The domain for the two-dimensional union test in Example 4.2: base primitive A, base primitive B, the corresponding domain obtained by union operation and the constructed SR.

be tested. Table 2 lists six different groups of the Bézier ordinates in the construction for the bicubic SR. These free Bézier ordinates are picked freely with the correct signs following the Criteria (a)–(c).

The solution errors for three groups of Bézier ordinates (No. 1, 5, 6 in Table 2) that used to construct the bicubic SR weight function are compared and plotted. Fig. 10 shows the relative L_2 errors of the numerical solution on the union shape. Particularly, the relative errors for solution space with bidegrees (k,k), $k=1,2,3$ are illustrated in Figs. 10(a)–10(c), respectively. It can be found that different selections of free ordinates have different errors at the beginning, but lead to the same trend of convergence rates.

Example 4.3. We now consider the Poisson equation over a complex domain obtained using both the Boolean intersection and union operations. The computational domain is shown in Fig. 11. Fig. 11(a) and Fig. 11(b) show two base primitives, Fig. 11(c) highlights the shape of A∩C and Fig. 11(d) shows the final domain constructed by SR.

Table 1: The relative L_2 errors and the convergence rates for the solution with weight functions constructed by SR and R_0 in the two dimentions for Example 4.2. R indicates the method (R_0 and SR, respectively), every two columns illustrate the results separately for solution degree $k=1,2,3$. l indicates the grid widths $h=2^{-l}$, $l=1,\cdots,8$.

R	l	$k=1$		$k=2$		$k=3$	
		L_2 error	Rate	L_2 error	Rate	L_2 error	Rate
R_0	1	1.84E-01	—	9.43E-02	—	7.45E-02	—
	2	1.34E-01	0.46	4.35E-02	1.12	1.75E-02	2.09
	3	4.02E-02	1.74	6.87E-03	2.66	1.47E-03	3.57
	4	1.06E-02	1.93	7.61E-04	3.17	1.03E-04	3.84
	5	2.68E-03	1.98	8.93E-05	3.09	6.79E-06	3.92
	6	6.74E-04	1.99	1.09E-05	3.03	4.23E-07	4.00
	7	1.69E-04	2.00	1.36E-06	3.01	2.67E-08	3.99
	8	4.23E-05	2.00	1.69E-07	3.00	1.65E-09	4.01
SR	1	1.38E+00	—	3.54E-01	—	1.64E-01	—
	2	3.12E-01	2.14	1.11E-01	1.67	7.78E-02	1.08
	3	3.27E-02	3.25	7.12E-03	3.96	4.39E-04	7.47
	4	9.56E-03	1.77	1.61E-04	5.47	3.40E-05	3.69
	5	2.75E-03	1.80	1.94E-05	3.05	2.25E-06	3.92
	6	7.43E-04	1.89	2.33E-06	3.06	1.07E-07	4.39
	7	1.93E-04	1.95	2.88E-07	3.02	7.18E-09	3.90
	8	4.91E-05	1.97	3.59E-08	3.00	4.25E-10	4.08

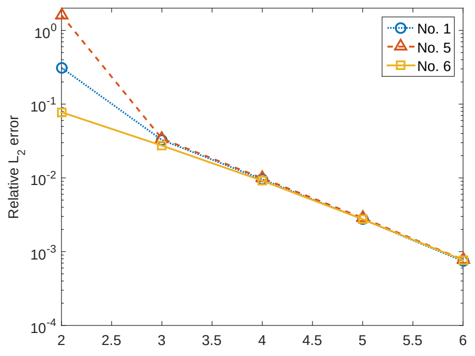
Table 2: Different groups of Bézier ordinates of the bicubic SR for the domain in Example 4.2. No. 1$\cdots$6 illustrate six groups of parameters, with each line showing one group of parameters.

No.	$b^1_{3,3}$	$b^1_{3,2}$	$b^1_{3,1}$	$b^1_{3,0}$	$b^4_{1,0}$	$b^4_{2,0}$	$b^4_{3,0}$	$b^3_{0,0}$
1	1	$\frac{1}{10}$	$\frac{1}{10}$	1	1	$\frac{1}{10}$	1	-1
2	1	$\frac{1}{10}$	$\frac{2}{5}$	1	1	$\frac{1}{10}$	1	-1
3	1	$\frac{3}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$-\frac{1}{10}$
4	1	$\frac{4}{5}$	$\frac{1}{2}$	$\frac{1}{2}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$-\frac{1}{10}$
5	5	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	$\frac{1}{10}$	-1
6	10	2	1	1	$\frac{1}{20}$	$\frac{1}{20}$	$\frac{1}{20}$	$-\frac{1}{10}$

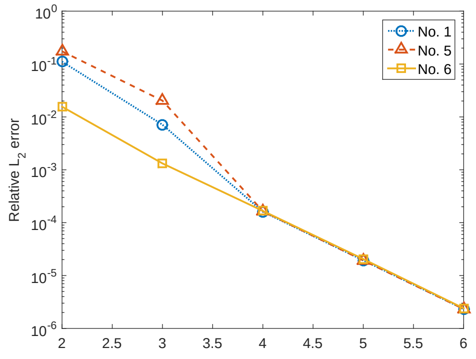
The exact solution is set as $u(x,y)=\sin(w(x,y))$.

Fig. 12 shows the results using SR for solution space with $k=1,2$ and 3, respectively. The relative L_2 errors converge to machine precision as the grids become smaller, especially for $k=2,3$. It can be concluded that weight function using SR lives up to the theoretical predictions and thus SR is an acceptable choice to represent complex domains under Boolean operations.

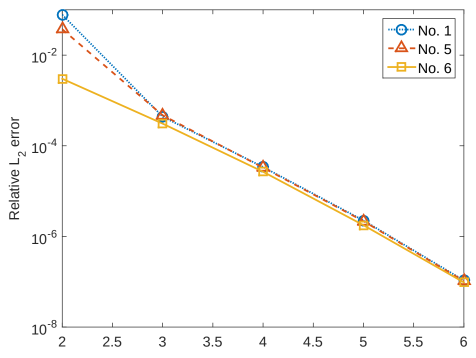
Example 4.4. In this example, we consider the Poisson problem (2.6) over a domain produced by the union of two three-dimensional base primitives, as shown in Fig. 13. Let $sr(x,y,z)$ and $r_0(x,y,z)$ be the SR and R_0, respectively. To compare the results of weight function using SR with that using R_0, the exact solution is set to be $u=$



(a) $k=1$



(b) $k=2$



(c) $k=3$

Figure 10: Results for weight function using SR with different parameters for Example 4.2. The relative L_2 errors for the solution space with $k=1,2,3$ are shown in Figs. 10(a), 10(b) and 10(c), respectively. Each subfigure depicts errors using three different parameters (No. 1, 5 and 6) with specific k and the grid widths $h=2^{-l}$, $l=1,\cdots,7$.

$\sin(sr(x,y,z)\times r_0(x,y,z))$, where we get the same accurate solution for both weight functions.

Table 3 presents the error results for the numerical solution of the Poisson equation where the weight functions uses SR and R_0 function are adopted. The relative L_2 errors and convergence rates under different degrees of the solution space and the varied grid widths are listed. It can be observed that both methods lead to convergence rates that are consistent with the theory in [25]. The results indicate that SR has good potential when representing complex three-dimensional domains for problems governed by partial differential equations.

The results of the numerical examples presented in this section indicate that the Boolean union and intersection operations are successfully implemented using SR. The convergence rates are compatible with those presented in the literature and provides a good indication for the reliability of the theoretical approach.

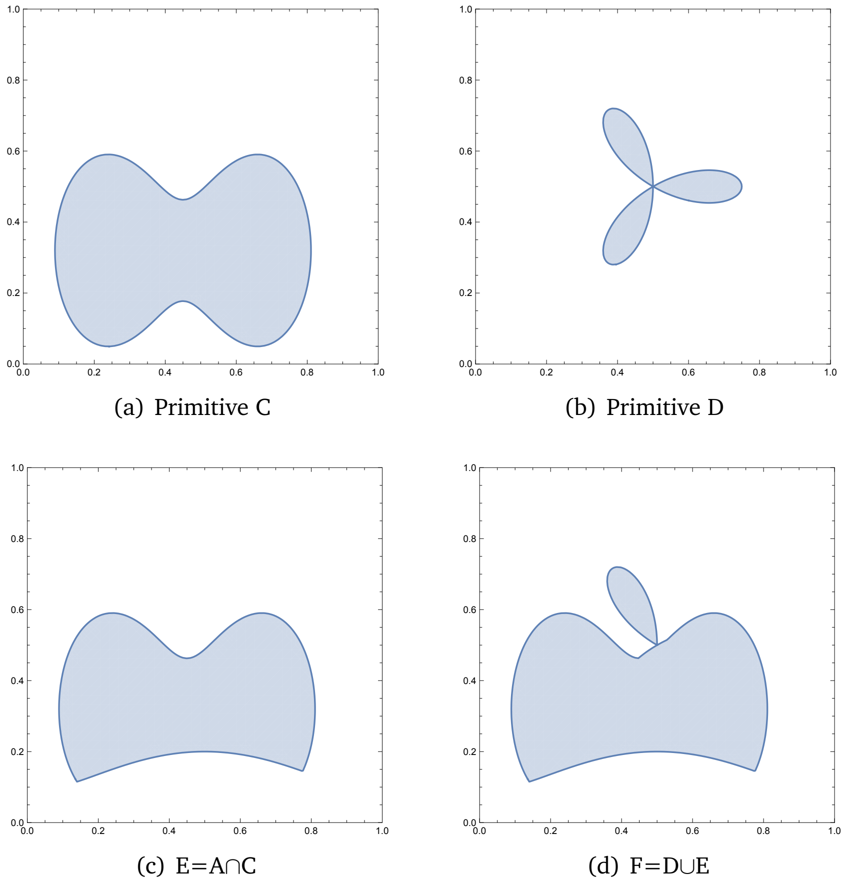


Figure 11: Domain composed of 3 shapes using SR in two dimensions for Example 4.3: two primitives C, D, the intersection of A and C by the Boolean intersection operation and the final domain after another Boolean union operation.

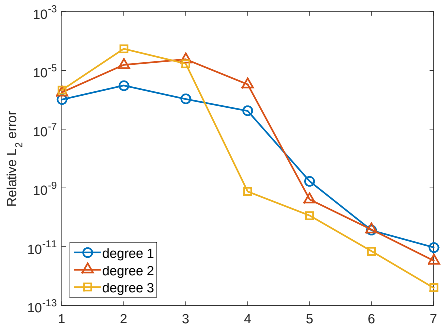


Figure 12: The relative L_2 errors of the numerical solution in Example 4.3 with weight function using SR are shown for $k = 1, 2, 3$ and $h = 2^{-l}$, $l = 1, \cdots, 7$.

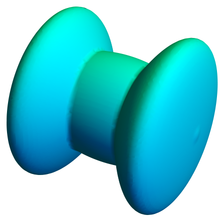


Figure 13: Domain of the 3D union test.

Table 3: The relative L_2 errors and the convergence rates for the numerical solution of Example 4.4 with weight functions using SR and R_0 in three dimensions. R indicates the method (R_0 and SR, respectively), l indicates the grid widths $h = 2^{-l}$, $l = 1, \cdots, 7$ and every two columns tabulate the results for solution degree $k = 1, 2, 3$, respectively.

R	l	$k=1$		$k=2$		$k=3$	
		L_2 error	Rate	L_2 error	Rate	L_2 error	Rate
R_0	1	8.59E-01	—	3.61E-01	—	8.72E+01	—
	2	2.12E-01	2.02	2.03E-01	0.83	2.01E-01	8.76
	3	1.08E-01	0.97	1.51E-01	0.43	7.72E-02	1.38
	4	6.23E-02	0.79	3.01E-02	2.33	8.01E-03	3.27
	5	1.97E-02	1.66	2.65E-03	3.51	5.14E-04	3.96
	6	5.13E-03	1.94	2.68E-04	3.31	3.07E-05	4.07
	7	1.30E-03	1.98	3.14E-05	3.09	2.01E-06	3.93
SR	1	7.92E+01	—	1.67E+01	—	8.08E-01	—
	2	1.01E+01	2.97	3.60E+00	2.22	1.62E-01	2.32
	3	5.89E-01	4.10	1.21E+00	1.57	1.18E-02	3.79
	4	3.67E-02	4.00	2.68E-03	8.82	4.08E-04	4.85
	5	1.33E-02	1.47	2.68E-04	3.33	3.14E-05	3.70
	6	4.18E-03	1.67	3.39E-05	2.98	2.62E-06	3.58
	7	1.20E-03	1.80	4.28E-06	2.99	1.90E-07	3.79

5. Conclusions

A new formulation for Boolean operations over implicitly defined primitives is considered. Spline R-function (SR) is constructed in both Bézier form and B-spline form on a new coordinate system following the criteria indicated by the definition of R-function. For SR in Bézier form, the B-net method is adopted, where the requirements for a specific boolean operation have been converted into a series of constraints between the Bézier ordinates. SR is then obtained by assigning these ordinates following the corresponding constraints. In the implementation part, it is shown that the proposed SR-function is well-defined. Numerical results indicate that SR is robust and leads to accurate results and thus is a good choice for complex domains composed by Boolean operations when solving partial differential equations on such domains. In addition, compared with R_0 function, SR is easier to be integrated into CAD systems due to the direct relation with CAD representations. In fact, since only algebraic operations are used in SR, a repeated de Boor algorithm can be used to achieve the final SR, which facilitates the practical usage.

Several possibilities exist for further improving the usage of spline R-function. Applying SR into different areas served by traditional R_0 function with appropriate modifications is a straightforward process. Since there are still Bézier ordinates in SR that have not been fully understood yet, more properties of SR can be explored to get better formulation for more practical purposes. Besides, the integration of SR into CAD seems to be promising.

Acknowledgements We would like to thank the anonymous reviewers and our laboratory group for helpful discussions and comments. The work is supported by the NSF of China (No. 11771420).

References

[1] T. Dokken, V. Skytt and O. Barrowclough, *Trivariate spline representations for computer aided design and additive manufacturing,* arXiv preprint arXiv:1803.05756.

[2] L. Cheng, J. Liu and A. C. To, *Concurrent lattice infill with feature evolution optimization for additive manufactured heat conduction design,* Struct. Multidisciplinary Optim., (2018), pp. 1–25.

[3] Q. Li, Q. Hong, Q. Qi, X. Ma, X. Han and J. Tian, *Towards additive manufacturing oriented geometric modeling using implicit functions,* Vis. Comput. Industry Biomedicine Art, 1(1) (2018), pp. 9.

[4] A. Qarariyah, F. Deng, T. Yang, Y. Liu and J. Deng, *Isogeometric analysis on implicit domains using weighted extended pht-splines,* J. Comput. Appl. Math., 350 (2019), pp. 353–371.

[5] B. Zhao, Z. Lin, J. Fu and Y. Sun, *Generation of truss-structure objects with implicit representation for 3d-printing,* Int. J. Comput. Integrated Manufact., 30(8) (2017), pp. 871–879.

[6] D. Li, N. Dai, X. Jiang, Z. Shen and X. Chen, *Density aware internal supporting structure modeling of 3d printed objects,* in: Virtual Reality and Visualization (ICVRV), 2015 International Conference on, IEEE, (2015), pp. 209–215.

[7] L. Liu, A. Shamir, C. C. Wang and E. Whiting, *3d printing oriented design: geometry and optimization,* in: SIGGRAPH ASIA Courses, (2014), pp. 1–1.

[8] J. Bloomenthal and C. Bajaj, Introduction to Implicit Surfaces, Morgan Kaufmann, 1997.

[9] A. Gomes, I. Voiculescu, J. Jorge, B. Wyvill and C. Galbraith, Implicit Curves and Surfaces: Mathematics, Data Structures and Algorithms, Springer Science & Business Media, 2009.

[10] A. Raffo, O. J. Barrowclough and G. Muntingh, *Reverse engineering of cad models via clustering and approximate implicitization,* arXiv preprint arXiv:1810.07451.

[11] Y. Song, Y. Luo, Y. Liu, J. Deng and Z. Yang, *Compression algorithm for implicit 3d b-spline solids,* Commun. Math. Stat., 6(2) (2018), pp. 119–140.

[12] B.-Q. Zuo, Z.-D. Huang, Y.-W. Wang and Z.-J. Wu, *Isogeometric analysis for csg models,* Comput. Methods Appl. Mech. Eng., 285 (2015), pp. 102–124.

[13] A. Pasko, V. Adzhiev, A. Sourin and V. Savchenko, *Function representation in geometric modeling: concepts, implementation and applications,* Vis. Comput., 11(8) (1995), pp. 429–446.

[14] K. Upreti and G. Subbarayan, *Signed algebraic level sets on nurbs surfaces and implicit boolean compositions for isogeometric cad–cae integration,* Comput. Aided Design, 82 (2017), pp. 112–126.

[15] A. Ricci, *A constructive geometry for computer graphics,* Comput. J., 16(2) (1973), pp. 157–160.

[16] Y. D. Fougerolle, A. Gribok, S. Foufou, F. Truchetet and M. A. Abidi, *Boolean operations with implicit and parametric representation of primitives using r-functions,* IEEE Trans. Vis. Comput. Graphics, 11(5) (2005), pp. 529–539.

[17] V. Shapiro, Theory of R-Functions and Applications: A Primer, Tech. Rep., Cornell University, (1991).

[18] V. L. Rvachev and T. I. Sheiko, *R-functions in boundary value problems in mechanics,* Appl. Mech. Rev., 48(4) (1995), pp. 151–188.

[19] V. Shapiro, *Semi-analytic geometry with r-functions,* ACTA Numer., 16 (2007), pp. 239.

[20] V. Shapiro and I. Tsukanov, *Implicit functions with guaranteed differential properties,* in: Proceedings of the Fifth ACM Symposium on Solid Modeling and Applications, ACM, (1999), pp. 258–269.

[21] S. Lamtyugova and M. Sidorov, *Numerical analysis of the external slow flows of a viscous fluid using the r-function method,* J. Eng. Math., 91(1) (2015), pp. 59–79.

[22] S. N. Lamtyugova and M. V. Sidorov, *Numerical analysis of the problem of flow past a cylindrical body applying the r-functions method and the galerkin method,* ECONTECHMOD : An International Quarterly Journal on Economics of Technology and Modelling Processes, 3(3) (2014), pp. 43–50.

[23] V. Shapiro, *Real functions for representation of rigid solids,* Comput. Aided Geometric Design, 11(2) (1994), pp. 153–175.

[24] V. L. Rvachev, T. I. Sheiko, V. Shapiro and I. Tsukanov, *Transfinite interpolation over implicitly defined sets,* Comput. Aided Geometric Design, 18(3) (2001), pp. 195–220.

[25] K. Höllig, Finite Element Methods with B-Splines, Vol. 26, SIAM, 2003.

[26] G. Kreisselmeier and R. Steinhauser, *Systematic control design by optimizing a vector performance index,* in: Computer Aided Design of Control Systems, Elsevier, 1980, pp. 113–117.

[27] M. Hursin, S. Xiao and T. Jevremovic, *Synergism of the method of characteristic, r-functions and diffusion solution for accurate representation of 3d neutron interactions in research reactors using the agent code system,* Ann. Nuclear Energy, 33(13) (2006), pp. 1116–1133.

[28] V. F. Kravchenko, *Atomic and r-functions in radiophysical applications*.

[29] G. E. Farin, Curves and Surfaces for CAGD: A Practical Guide, Morgan Kaufmann, 2002.

[30] L. Piegl and W. Tiller, The NURBS Book, Springer Science & Business Media, 2012.

[31] J. Deng, F. Chen, X. Li, C. Hu, W. Tong, Z. Yang and Y. Feng, *Polynomial splines over hierarchical t-meshes,* Graphical Models, 70(4) (2008), pp. 76–86.

[32] T. W. Sederberg, J. Zheng, A. Bakenov and A. Nasri, *T-splines and t-nurccs,* ACM Transactions on Graphics (TOG), 22(3) (2003), pp. 477–484.

[33] C. Li, P. Wang, *The instability in the dimensions of spline spaces over t-meshes with nested t-cycles,* Numer. Math. Theory Methods Appl., 12(1) (2019), pp. 187–211.

[34] J. Hoschek, D. Lasser and L. L. Schumaker, Fundamentals of Computer Aided Geometric Design, AK Peters, Ltd., 1993.

[35] K. Höllig, U. Reif and J. Wipper, *Weighted extended b-spline approximation of dirichlet problems,* SIAM J. Numer. Anal., 39(2) (2001), pp. 442–462.

[36] Y. D. Fougerolle, A. Gribok, S. Foufou, F. Truchetet and M. A. Abidi, *Implicit surface modeling using supershapes and r-functions,* Proc. Pacific Graphics, 05 (2005), pp. 169–172.

[37] V. Rvachev, Theory of R-Functions and Some Applications, (1982).

[38] R.-H. Wang, Multivariate Spline Functions and Their Applications, Vol. 529, Springer Science & Business Media, 2013.

[39] J. Deng, F. Chen and Y. Feng, *Dimensions of spline spaces over t-meshes,* J. Comput. Appl. Math., 194(2) (2006), pp. 267–283.

[40] K. Höllig, C. Apprich and A. Streit,*Introduction to the web-method and its applications,* Adv. Comput. Math., 23(1-2) (2005), pp. 215–237.

[41] T. Yang, A. Qarariyah, H. Kang and J. Deng, *Numerical integration over implicitly defined domains with topological guarantee,* arXiv preprint arXiv:1901.03498.