Contemporary Mathematics
https://ojs.wiserpub.com/index.php/CM/

Research Article

A Ternary Shape-Preserving Refinement Algorithm with Fourth Order Accuracy

Fahad Sameer Alshammari[1*], Pakeeza Ashraf[2†], Javeria Ashraf[2], Amna Kalsoom[3], Ali Akgul[4,5,6,7,8]

[1] Department of Mathematics, College of Science and Humanities in Alkharj, Prince Sattam bin Abdulaiz University, Al-Kharj 11942, Saudi Arabia
[2] Department of Mathematics, Government Sadiq College Women University, Bahawalpur, Pakistan
[3] Department of Mathematics & Statistics, International Islamic University, Islamabad, Pakistan
[4] Department of Electronics and Communication Engineering, Saveetha School of Engineering, SIMATS, Chennai, Tamilnadu, India
[5] Department of Computer Engineering, Biruni University, Istanbul 34010, Turkey
[6] Department of Mathematics, Mathematics Research Center, Near East University, Nicosia/Mersin 10, 99138, Turkey
[7] Applied Science Research Center, Applied Science Private University, Amman, Jordan
[8] Department of Mathematics, Art and Science Faculty, Siirt University, Siirt 56100, Turkey
E-mail: f.alshammari@psau.edu.sa (F.S.A.); pakeeza@gscwu.edu.pk (P.A.)

Received: 14 October 2025; **Revised:** 28 November 2025; **Accepted:** 1 December 2025

Abstract: This paper introduces a new class of stationary refinement algorithms that combine the advantages of the four-point interpolatory algorithm and the cubic B-spline. This algorithm includes a tension parameter. This analysis identifies the range of this parameter and specifies conditions under which the proposed algorithm maintains shape-preserving properties such as monotonicity and convexity. The algorithm achieves improved smoothness, reaching C^3 and fourth-order accuracy, while maintaining the same support length as both parent algorithms. Unlike many high-order algorithms that are nonlinear and complex, this method remains simple and efficient. Numerical examples are provided to demonstrate its practical performance, and the proposed algorithm is particularly well-suited for a discontinuous type of data set.

***Keywords*:** refinement algorithm, monotonicity, convexity, accuracy, iterative optimization, update rule, learning rate, gradient-based update, local computation, convergence analysis

MSC: 65D17, 65D10, 65D07, 65D05

1. Introduction

A refinement algorithm uses a set of refinement rules to create smooth curves at level i from a set of initial control points. Interpolating and approximating algorithms are the most popular refinement algorithms for creating smooth curves. It is referred to as an interpolating algorithm if it keeps the point of level i as a subset of the point of level $i+1$; otherwise, it is called an approximating algorithm [1]. Let R_A, be a single operator operates on the provided sequence g^i via the discrete convolution of the form

DOI: https://doi.org/10.37256/cm.7320268896

$$R_A: g^i \mapsto R_A g^i \qquad \text{with} \qquad (R_A g^i): = \sum_{n\in\mathbb{Z}} a_{j-3n} g_n^i, \;\; i = 0, 1, 2, \cdots .$$

where $R_A g^i = g^{i+1}$ and R_A is a bounded operator. The mask of the refinement algorithm R_A is the sequence of the coefficients represented as $m = \{m_n: n \in \mathbb{Z}\}$ and g^0 is the initial data generates g^i after i levels of refinements. For g^0, the limit function is called the basic limit function. Assuming that m possesses finite support indicates that the initial data value influences just the immediate area in the resultant limit function. So that modifications to a control point only impact a small number of patches. A uniform convergence of the data sequence g^i to a continuous function is possible as i goes to ∞ for a properly selected refinement rule.

The development of curve refinement methods started with de Rham [2], who created a basic technique with C^0 continuity. Later on, a corner-cutting piecewise linear approximation technique with C^1 continuity was presented by Chaikin [3]. Dubuc [4] introduced a 4-point interpolating method using Lagrange polynomials, later improved by Dyn et al. [5] with a shape parameter. After that, by using the same techniques, Deslauriers and Dubuc (DD) [6] generalized the 4-point binary algorithm to an n-ary 2N-point algorithm. This generalization offered more flexibility and control in curve refinement. To study smoothness, researchers used tools like the joint spectral radius [7] and Laurent polynomials. Ashraf et al. [8] created a new class of approximate binary refinement algorithms by presenting a generalized method. A 6-point binary interpolating algorithm with C^2 continuity was described by Weissman [9]. Tang et al. [10] used the Laurent polynomial to examine the smoothness and convergence of the 4-point algorithm, finding that it was C^1. Several monotonicity-preserving local non-linear interpolating techniques were introduced by Kuijt and Damme [11].

As researchers explored ways to create smoother curves more efficiently, they discovered that switching from binary to ternary refinement results in better continuity without increasing the support length. Beccari et al. [12] showed how using more refinements could enhance smoothness while keeping things compact. Inspired by this, Hassan et al. [13] developed a four-point ternary algorithm with a tension parameter that allowed the curve to reach C^2 smoothness. However, to keep the curve from losing its shape, Cai [14] introduced limits on that parameter to preserve convexity. Later, Pitoli [15] examined how bell-shaped masks in the ternary algorithm helped maintain the overall form of the data.

Generally, the DD four-point refinement algorithm (R_D-algorithm) and the cubic B-spline (R_B-algorithm) are the most practical methods. The R_B-algorithm combines optimal smoothness with minimal support length of its fundamental limit function, as established in [16]. It effectively retains the convexity and monotonicity of the input data, and its ability to reproduce linear polynomials indicates that it achieves a second-order accuracy. On the other hand, only C^1 smoothness is provided by the R_D-algorithm, which achieves fourth-order accuracy. To combine the benefits of both, researchers introduced J-splines [17], a blended family of algorithms, which later evolved into a two-parameter version studied by Rossignac [18] for smoothness properties. Kaklis et al. [19] provided a C^2 shape-preserving interpolant for a given data set on a developable surface. More recently, Yang et al. [20] proposed a family of four-point binary refinement algorithms with C^2 continuity, reproducing linear polynomials but attaining fourth-order accuracy. Building on this, Razaq et al. [21] developed an innovative hybrid family of non-uniform ternary algorithms with mixed symmetry for better shape approximation. Kim et al. [22] presented a novel shape preserving subdivision scheme with third-order accuracy. They proved that limit functions preserve both monotonicity and convexity of the given data, even in cases where the data are non-strictly monotone or convex.

Motivated by the above framework, a new class of stationary refinement algorithms (R_A-algorithm) is proposed, which combines the benefits of both R_D-algorithm and R_B-algorithm. This algorithm has a single tension parameter that fills in the gap between R_B-algorithm and R_D-algorithm. For a variety of applications, this offers a broad range of refinement algorithms that balance accuracy, smoothness, and shape-preserving properties. The following are the particular contributions. First, R_A-algorithm achieves a higher level of smoothness that is C^3 by sacrificing the interpolating characteristic, while keeping the support length the same as R_B-algorithm and R_D-algorithm, that is, $[-2.5, 2.5]$. Second, the proposed algorithm can achieve fourth-order accuracy while reproducing up to linear polynomials. Analyzing the order of approximation of the proposed refinement algorithm, especially for the function in the Sobolev space $\mathbb{W}^{m,\infty}(\mathbb{R})$ with

$m \in \mathbb{N}_0$. Assuming the provided data is derived from some underlying generating function $g \in \mathbb{W}^{m,\infty}$, we confirm that our technique achieves the order of approximation 'four'. The suggested refinement algorithm maintains the convexity and monotonicity of the originally provided data, even though it is linear and stationary, in contrast to the majority of them, which are non-linear and non-stationary [11]. The performance of our refinement algorithm is demonstrated with some numerical examples. A ternary shape-preserving refinement algorithm with fourth-order accuracy provides a high-quality iterative framework for generating smooth curves while rigorously maintaining monotonicity and convexity. By drawing conceptual parallels to stochastic gradient descent—such as iterative update rules, stability-driven weighting, and constraint-preserving behavior—we can better understand how refinement masks function as optimization-like operators that progressively reduce geometric error.

The following sections comprise this paper: Start with some fundamental concepts and definitions in Section 2. A new class of stationary refinement algorithms is constructed in Section 3. In Section 4, the proposed algorithm is analyzed in terms of its accuracy and smoothness. Section 5 focuses on its ability to preserve monotonicity and convexity. Lastly, Section 6 presents some numerical examples that illustrate the performance of the algorithm.

2. Fundamental concepts and definitions

First, some background information and notation related to the univariate stationary refinement algorithm are presented, i.e., $\mathbb{N}_0^+$ is the set of positive integers including zero. $\mathbb{R}^+$ represents the set of positive real numbers. The set of all algebraic polynomials of degree at most n is represented by the symbol $\prod_n$. For a given sequence $g \in l^\infty(\mathbb{Z})$, we make use of the notations:

$$\begin{cases} \nabla g_n = g_n - g_{n-1}, \\ \\ \Delta g_n = g_{n-1} - 2g_n + g_{n+1}. \end{cases} \tag{1}$$

At level $i \in \mathbb{N}_0^+$ a sequence $g^i := \{g_n^i : n \in \mathbb{Z}\}$ is supposed to be connected to grid points $3^{-i}\mathbb{Z}$. Furthermore, we utilize the following notations for a given sequence:

$$\begin{cases} (D_k g^i)_n := 3^i (g_n^i - g_{n-1}^i), \\ \\ (D_k^2 g^i)_n := 3^{2i} (g_{n-1}^i - 2g_n^i + g_{n+1}^i). \end{cases} \tag{2}$$

At this time and subsequently, we do not include the subscript i unless it would be confusing. That is, $Dg_n^i := D_i g_n^i$.

Definition 1 A technique for ternary refinement algorithm when a limit function $R_A^\infty f^0 \in C(\mathbb{R})$ exists and $R_A^\infty g^0$ displays complex behavior for some initial non-zero values then R_A-algorithm applied to $g^0 \in l^\infty(\mathbb{Z})$ is said to be uniformly convergent if the sequence of linear functions G^i interpolating g^i at 3^{-i} converges to $R_A^\infty g^0$ as explained below:

$$\lim_{i \to \infty} \| G^i - R_A^\infty g^0 \|_{L^\infty(I)} = 0. \tag{3}$$

Here I is a closed and bounded set in $\mathbb{R}$.

Definition 2 ([20]) If the limit function possesses continuous derivatives up to order $m \in \mathbb{N}_0^+$, the R_A-algorithm is said to be C^m-convergent.

Specifically, for the initial sequence $\delta := \{\delta_{0,n} : n \in \mathbb{Z}\}$, where $\delta_{n,0}$ denotes the Kronecker delta centered at zero. The basic limit function of R_A-algorithm is described by

$$\xi := R_A^{\infty}\delta.$$

Since the support for the mask m is finite, the fundamental limit function ξ is also compactly supported. It is evident from the linearity of the refining algorithm that

$$R_A^{\infty} g^0 = \sum_{n\in\mathbb{Z}} g_n^0 \xi(. - n).$$

It is known (for example, see [16]) for a refinement algorithm R_A with mask $m = \{m_n : n \in \mathbb{Z}\}$ to be convergent a necessary condition is that both the even-indexed and odd-indexed coefficients must separately satisfy the partition of unity. In other words,

$$\sum_{n\in\mathbb{Z}} m_{l-3n} = 1, \qquad l = 0, 1, 2.$$

Indeed, for a particular refinement algorithm with mask m, the a-transform (also known as the symbol of R_A-algorithm) incorporates its basic characteristics, including convergence, smoothness, and the polynomial reproducing (or generation) property:

$$L(a) := \sum_{n\in\mathbb{Z}} m_n a^n, \qquad a \in \mathbb{C} \setminus \{0\}.$$

Since the mask m has finite support, it is clear that $L(a)$ is a Laurent polynomial. Furthermore, R_A-algorithm corresponding to the mask m has $\|.\|_{\infty}$-norm is defined as follows:

$$\|R_A\|_{\infty} := max\{\sum_{n\in\mathbb{Z}} |m_{3n}|, \sum_{n\in\mathbb{Z}} |m_{1+3n}|, \sum_{n\in\mathbb{Z}} |m_{2+3n}|\}.$$

Along with convergence and smoothness, another crucial criterion for a refinement algorithm is its order of accuracy is directly linked to its ability to reproduce (or generate) polynomials.

Definition 3 [20] Suppose that $Q^0 = \{q(n) : n \in \mathbb{Z}\}$ where $q \in \prod_c$ with $c \in \mathbb{N}_0^+$. R_A is said to reproduce polynomials in $\prod_c$ if $q = R_A^{\infty} q^0$ and R_A-algorithm is convergent. In addition, R_A-algorithm is said to be $\prod_c$-generating if $R_A^{\infty} q^0 \in \prod_c$.

Since a convergent refinement algorithm R_A with a finitely supported mask m guarantees the order of approximation $c+1$ when it reproduces polynomials of degree c, it is a desired property [23]. Contrary to it, the order of approximation $c+1$ is not normally implied by a refinement algorithm that produces polynomials of degree c. For example, R_B-algorithm of degree $c \in \mathbb{N}$ is known to generate $\prod_c$, but only reproduce $\prod_1$, resulting in order of approximation 2. Levin [24] demonstrated that applying a suitable preprocessing operator can enhance reproduction, though the resulting accuracy may still fall short of the generation degree. However, in the next part, we introduced a new refinement algorithm including a tension parameter. Although it exactly reproduces linear polynomials, it is capable of generating polynomials up to degree $\prod_3$), and still achieves fourth-order accuracy with improved smoothness, i.e., C^3.

3. New refinement algorithm

In this section, a new refinement algorithm is introduced as well as the polynomial generation property is also discussed.

3.1 *Refinement algorithm*

To construct the proposed refinement algorithm, we begin by observing that the classical four-point R_D-algorithm achieves fourth-order accuracy but only C^1 smoothness, while the R_B-algorithm is C^2 smooth and shape-preserving but offers only second-order accuracy. Our goal is to design a refinement procedure that simultaneously inherits the higher accuracy of R_D-algorithm and the enhanced smoothness and shape-preserving behavior of R_B-algorithm, without increasing the support size. To achieve this, we modify the R_D-algorithm refinement rule by introducing a controlled perturbation based on the second-order finite difference of the data sequence. Since the second-difference operator directly influences curvature and local shape, it provides a natural mechanism for adjusting smoothness while maintaining the compact structure of the mask. Motivated by this idea, we introduce a tension parameter ψ_0 that regulates the contribution of the perturbation term. A new R_A-algorithm with a tension parameter ψ_0 and a finite difference approach is presented that combines the benefits of both parent algorithms, R_D-algorithm and R_B-algorithm. Given a data sequence $g^i := \{g_n^i : n \in \mathbb{Z}\}$ at level i, the improved sequence $g^{i+1} := \{g_n^{i+1} : n \in \mathbb{Z}\}$ is described by a linear combination of the current values of g^i within the three refinement rules as follows:

$$\begin{cases} g_{3n}^{i+1} = \dfrac{1}{81}((21\psi_0)g_{n-1}^i + (81-42\psi_0)g_n^i + (21\psi_0)g_{n+1}^i), \\ g_{3n+1}^{i+1} = \dfrac{1}{81}((-5+14\psi_0)g_{n-1}^i + (60-21\psi_0)g_n^i + 30g_{n+1}^i + (-4+7\psi_0)g_{n+2}^i), \\ g_{3n+2}^{i+1} = \dfrac{1}{81}((-4+7\psi_0)g_{n-1}^i + 30g_n^i + (60-21\psi_0)g_{n+1}^i + (-5+14\psi_0)g_{n+2}^i). \end{cases} \tag{4}$$

The parameter ψ_0 is the essential component used to create our refinement mask. Its selection significantly affects the accuracy, regularity, and shape-preservation properties of the related algorithm. In this research, supposing that the initially given sequence is associated with $3^{-i_0}\mathbb{Z}$, we advise choosing ψ_0 as

$$\psi_0 = \tau 3^{-2i_0},\ \tau > 0. \tag{5}$$

Our construction in (4) has the specific characteristics. First letting R_D-algorithm with the mask $\{\frac{-4}{81}, \frac{-5}{81}, 0, \frac{30}{81}, \frac{60}{81}, 1, \frac{60}{81}, \frac{30}{81}, 0, \frac{-5}{81}, \frac{-4}{81}\}$, The proposed refinement algorithm modifies the refinement rule of the four-point R_D-algorithm through a linear perturbation, which can be described as follows:

$$R_A g^i = R_D g^i + 3^{-2i}\frac{\psi_0}{81}\mathbb{D}_2(g^i), \tag{6}$$

where $\mathbb{D}_2$ is linear operator, described as

$$\begin{cases} \mathbb{D}_2(g^i)_{3n} = 21(D^2 g^i)_n, \\ \mathbb{D}_2(g^i)_{3n+1} = 14(D^2 g^i)_n + 7(D^2 g^i)_{n+1}, \\ \mathbb{D}_2(g^i)_{3n+2} = 7(D^2 g^i)_n + 14(D^2 g^i)_{n+1}. \end{cases} \tag{7}$$

The construction of the proposed refinement algorithm involves a modification of the classical four-point R_D-algorithm, achieved by incorporating the second-order finite difference of g^i. Moreover, the parameter ψ_0 is adaptively chosen based on the density of the initial data, as detailed in (5). This allows the proposed refinement algorithm to achieve fourth-order accuracy and improved smoothness without extending its support length. However, The generation of cubic polynomials by the proposed algorithm implies the inclusion of the factor $(1+a+a^2)^4$ in its symbol. For clarity in subsequent discussions, we present the corresponding symbol, defined in (4):

$$L(a) = \frac{(1+a+a^2)^4}{3^4}\left(\frac{1}{a^3}(7\psi_0 - 4) + \frac{1}{a^4}(11 - 14\psi_0) + \frac{1}{a^5}(7\psi_0 - 4)\right). \tag{8}$$

Remark 1 From Table 1, it is simple to see that R_A-algorithm generalizes both R_D-algorithm and R_B-algorithm. If $\psi_0 = 0$, the algorithm is R_D-algorithm. Also, when $\psi_0 = 1$ it becomes the R_B-algorithm with the mask $\{\frac{1}{27}, \frac{1}{9}, \frac{7}{27}, \frac{10}{27}, \frac{13}{27}, \frac{13}{27}, \frac{13}{27}, \frac{10}{27}, \frac{7}{27}, \frac{1}{9}, \frac{1}{27}\}$. R_B-algorithm can therefore be used to represent the refinement algorithm in (4) as follows:

$$R_A g^i := R_B g^i - 3^{-2i}\frac{(1-\psi_0)}{81}\mathbb{D}_2(g^i), \tag{9}$$

with the operator $\mathbb{D}_2$ in (7). Later on, we shall observe that the representation will be helpful in analyzing shape-preservation properties of R_A algorithm.

Table 1. Generalization of proposed algorithm

Parameter	Refinement Mask	Algorithm
ψ_0	$\{\frac{-4+7\psi_0}{81}, \frac{-5+14\psi_0}{81}, \frac{21\psi_0}{81}, \frac{30}{81}, \frac{60-21\psi_0}{81}, \frac{81-42\psi_0}{81}, \frac{60-21\psi_0}{81}, \frac{30}{81}, \frac{21\psi_0}{81}, \frac{-5+14\psi_0}{81}, \frac{-4+7\psi_0}{81}\}$	R_A-Algorithm
0	$\{\frac{-4}{81}, \frac{-5}{81}, 0, \frac{30}{81}, \frac{60}{81}, 1, \frac{60}{81}, \frac{30}{81}, 0, \frac{-5}{81}, \frac{-4}{81}\}$	R_D-Algorithm
1	$\{\frac{1}{27}, \frac{1}{9}, \frac{7}{27}, \frac{10}{27}, \frac{13}{27}, \frac{13}{27}, \frac{13}{27}, \frac{10}{27}, \frac{7}{27}, \frac{1}{9}, \frac{1}{27}\}$	R_B-Algorithm

3.2 *Polynomial generating property*

The presence of smoothing factor $(1+a+a^2)^\delta$, $\delta \in \mathbb{N}$, in the symbol of a refinement algorithm is equal to ξ of R_A-algorithm satisfying the Strang-Fix condition of order δ. This condition ensures that the proposed refinement algorithm is capable of generating all polynomials of degree at most δ. But, later on we shall discover that even though R_A-algorithm in (6) reproduce polynomial in $\prod_1$, can achieves an approximation order of four.

We use the following notation: for $q \in \prod_3$ and $i \in \mathbb{N}_0$,

$$Q_n^i := q(n3^{-i}) \text{ and } q^{(2),\,i+1} := \mathbb{D}_2(q^2)$$

Theorem 1 Suppose that $\mathbb{R}_A$ be the refinement algorithm associated with the parameter ψ_0 as defined in equation (5). For each $q \in \prod_3$, the algorithm satisfies the following generation property:

$$R_A^{\infty} q^0 = q + \frac{\psi_0}{81} q''.$$

Proof. We proceed by mathematical induction to show that for any $i \in \mathbb{N}_0^+$,

$$R_A^{i+1} q^0 = q^{i+1} + \frac{\psi_0}{81} q^{(2),\,i+1} \sum_{n=0}^{i} 3^{-2n}. \tag{10}$$

The case $i = 0$ follows directly from (6) given that the four-point algorithm reproduces a polynomial in $\prod_3$. Then, suppose (10) is correct for $i \in \mathbb{N}_0^+$. Since $q \in \prod_3$, clearly, $(D^2 q^{(2),\,i})_n = 0$ for $n \in \mathbb{Z}$, indicating that $\mathbb{D}_2(q^{(2),\,i}) = 0$. Consequently, using (6), the induction hypothesis leads to the following result:

$$\begin{aligned} R_A^{i+1} q^0 &= R_A(q^i + \frac{\psi_0}{81} q^{(2),\,i} \sum_{n=0}^{i-1} 3^{-2n}) \\ &= (R_D + 3^{-2i} \frac{\psi_0}{81} \mathbb{D}_2)(q^i + \frac{\psi_0}{81} q^{(2),\,i} \sum_{n=0}^{i-1} 3^{-2n}) \\ &= q^{i+1} + \frac{\psi_0}{81} q^{(2),\,i+1} \sum_{n=0}^{i} 3^{-2n}. \end{aligned}$$

Here, the mathematical induction is done, so we can easily find that

$$q_{3n+v}^{(2),\,i+1} = q''(n3^{-i}) + O(3^{-i}) \text{ for } v = 0,\, 1,\, 2.$$

□

4. Study of convergence, smoothness and accuracy

In this section, the main properties of refinement algorithm are discussed.

4.1 *Convergence and smoothness*

The most basic factors for choosing a refinement algorithm are smoothness and convergence as well as the length of the refinement mask. To examine these properties of R_A-algorithm in (4), we present a few basic symbols. For given refinement rule R_A-algorithm, let $R_{A,1}$ be the refinement rule for the divided difference of the original control points so that

$$Dg^{i+1} = R_{A,1} Dg^i,$$

where $g^i = R_A^i g^0$. Laurent polynomial $L_1(a)$ for $R_{A,1}$ is defined by

$$L_1(a) = \frac{3a^2}{1+a+a^2} L(a),$$

with $L(a)$ the symbol connected with R_A-algorithm. In general, the symbol of the β^{th}-order difference algorithm (presented as $R_{A,\beta}$) of R_A-algorithm is described as

$$L_\beta(a) = \sum_{n\in\mathbb{Z}} m_n^{[\beta]}. \tag{11}$$

Here, $L_0(a) = L(a)$ and $m_n^{[0]} = m_n$. The following results offers a popular method for determining if R_A-algorithm is convergent.

Theorem 2 ([16]) Suppose that R_A and $R_{A,1}$ are the stationary refinement algorithm with symbol $L(a)$ and $L_1(a)$ respectively. Then, the algorithm is said to be uniformly convergent (that is, it produces C^0 limit function) if and only if there exist a positive integer s such that $\|(\frac{1}{3}R_{A,1})^s\|_\infty < 1$ where the algorithm $\frac{1}{3}R_{A,1}$ is connected with $\frac{1}{3}L_1(a)$.

Additionally, an adequate prerequisite for R_A to be C^a, $a \geq 1$, is given below:

Theorem 3 ([16]) Suppose R_A be a stationary refinement algorithm with the symbol

$$L(a) = \frac{1}{3a^2}(1+a+a^2)e(a),$$

with the Laurent polynomial $e(a)$. If the refinement algorithm R_e associated with $e(a)$ is C^a, then the algorithm R_A is $C^{a+1}(\mathbb{R})$.

We are now ready to show that (4) provides C^3-smoothness over a specified range of ψ_0. If $\psi_0 = 0$, the mask given in (4) corresponds to the four-point algorithm which is C^1-continuous. Therefore, in the following discussion, we focus on the case $\psi_0 > 0$.

Theorem 4 Consider the refinement algorithm R_A in (4), and let the parameter ψ_0 be specified as in the same equation. Then, the R_A-algorithm produces limit functions that are C^3-smooth for $\frac{4}{7} < \psi_0 < 1$.

Proof. We begin by analyzing the uniform convergence of R_A-algorithm. To proceed, recall that the Laurent polynomial $L(a)$ in (8) corresponding to refinement mask m in (4). According to equation (11), Laurent polynomial of the corresponding discrete first difference algorithm $R_{A,1}$ given by

$$L_1(a) = \frac{(1+a+a^2)^4}{3^4}\left(\frac{1}{a^3}(7\psi_0 - 4) + \frac{1}{a^4}(11 - 14\psi_0) + \frac{1}{a^5}(7\psi_0 - 4)\right).$$

The mask $m^{[1]}:=(m_n^{[1]}: n\in\mathbb{Z})$ corresponding to $R_{A,1}$ algorithm takes the form:

$$m^{[1]}:=\left(\frac{-4+7\psi_0}{27},\frac{-1+7\psi_0}{27},\frac{5+7\psi_0}{27},\frac{26-14\psi_0}{27},\frac{29-14\psi_0}{27},\frac{26-14\psi_0}{27},\frac{5+7\psi_0}{27},\frac{-1+7\psi_0}{27},\frac{-4+7\psi_0}{27}\right).$$

It is clear that, $\sum_{n\in\mathbb{Z}} m_{l+3n}^{[1]}=1$ for $l=0,1,2$. Moreover, $\|\frac{1}{3}R_{A,1}\|_\infty<1$, Therefore, R_A-algorithm is uniformly convergent for the given range $-\frac{25}{14}<\psi_0<\frac{53}{14}$. Next, to check the C^1 smoothness of R_A-algorithm, consider the second order difference $R_{A,2}$. In view of (8) and (11), the symbol associated with $R_{A,2}$ is given by

$$L_2(a):=\frac{(1+a+a^2)^3}{3^3}\left(\frac{1}{a^3}(7\psi_0-4)+\frac{1}{a^4}(11-14\psi_0)+\frac{1}{a^5}(7\psi_0-4)\right),$$

such that the mask connected with $R_{A,2}$ is given as

$$m^{[2]}:=(\frac{-4+7\psi_0}{9},\frac{1}{3},\frac{2}{3},\frac{17-14\psi_0}{9},\frac{2}{3},\frac{1}{3},\frac{-4+7\psi_0}{9}).$$

Thus, it is simple to induce that $\|\frac{1}{3}R_{A,2}\|_\infty<1$, indicating that R_A-algorithm is C^1 for $-\frac{1}{14}<\psi_0<\frac{13}{7}$. To investigate the C^2 smoothness of R_A-algorithm, we analyze the third-order difference $R_{A,3}$ defined by the following symbol:

$$L_3(a):=\frac{(1+a+a^2)^2}{3^2}\left(\frac{1}{a^3}(7\psi_0-4)+\frac{1}{a^4}(11-14\psi_0)+\frac{1}{a^5}(7\psi_0-4)\right).$$

The mask connected with $L_3(a)$ is given as

$$m^{[3]}:=\left(\frac{-4+7\psi_0}{3},\frac{7-7\psi_0}{3},1,\frac{7-7\psi_0}{3},\frac{-4+7\psi_0}{3}\right),$$

leading to the bound $\|\frac{1}{3}R_{A,3}\|_\infty<1$ implying that R_A-algorithm is C^2 for $\frac{1}{7}<\psi_0<\frac{10}{7}$. Finally, to check the C^3 smoothness of R_A-algorithm, Let us assume that the third order difference $R_{A,4}$ with the symbol

$$L_4(a):=\frac{(1+a+a^2)}{3}\left(\frac{1}{a^3}(7\psi_0-4)+\frac{1}{a^4}(11-14\psi_0)+\frac{1}{a^5}(7\psi_0-4)\right).$$

The mask attached to $L_4(a)$ is provided as

$$m^{[4]}:=(-4+7\psi_0,\,11-14\psi_0,\,-4+7\psi_0),$$

also $\|\frac{1}{3}R_{A,3}\|_\infty<1$ for $\frac{4}{7}<\psi_0<1$. It demonstrates that R_A-algorithm is C^3. □

Basic limit function: The sequence $\delta=\delta_{0,n}: n\in\mathbb{Z}$ where the kronecker delta $\delta_{n,0}$ is used at level 0. Basic limit function for the refinement algorithm R_A is expressed as $\xi=R_A^\infty\delta$. It is important to note that the support of this function

is confined to the interval $[-2.5, 2.5]$, similar to both interpolating and approximating algorithm (i.e., R_D-algorithm and R_B-algorithm). Figure 1 discuss the smoothness of algorithm by letting the initial data g^0 be the Kronecker delta sequence, i.e., $g^0 = \{g^0 := \delta_{l,0}:\ l \in \mathbb{Z}\}$ and $\delta_{l,0}$ equals to zero if $j \neq 0$ otherwise equals to one.

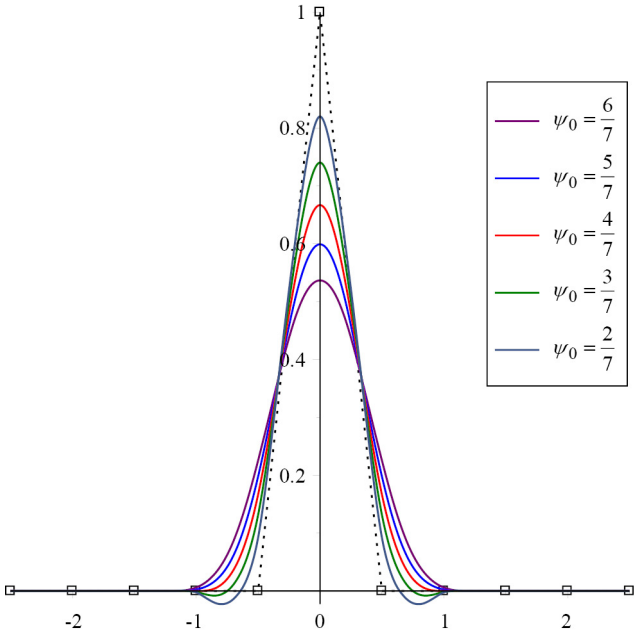

Figure 1. Basic limit function of R_A-algorithm with ψ_0 = 2/7, 3/7, 4/7, 5/7, 6/7

4.2 *Evaluation of order of accuracy*

The mask in (4), investigate the order of accuracy of R_A-algorithm with parameter ψ_0 and the smoothness of the related refinement algorithm. The paper aim to demonstrate that, given the appropriate choice of parameter ψ_0, the proposed algorithm achieves fourth-order accuracy. That is, applying R_A-algorithm to the initial data set $g^0 = \{g(n3^{-i_0}): n \in \mathbb{Z}\}$ involving a smooth function g generates a limit function $R_A^\infty g$ such that $\|g - R_A^\infty g^0\| \leq c_g 3^{-4i_0}$. The notation $\hat{g}^i$, $i \in \mathbb{Z}_+$ is introduced, the sequence formed by sampling a smooth function g at the grid points $3^{-i}\mathbb{Z}$ corresponding to the i^{th} level, i.e.,

$$\hat{g}^i := \{g(n3^{-i}): n \in \mathbb{Z}\}, \qquad i \in \mathbb{Z}_+, \tag{12}$$

Lemma 1 Suppose R_A be the proposed refinement algorithm with parameter $\psi_0 = \tau 3^{-i_0}$. Let $\hat{g}^i$, $i \geq i_0$, be a sequence of data sampled from a function $g \in W_\infty^4(\mathbb{R})$ density 3^{-i}, as given (4). Then we have

$$\|R_A \hat{g}^i - \hat{g}^{i+1}\|_\infty \leq c_g 3^{-2(i+i_0)},$$

for some constant $c_g > 0$ depending on g.

Proof. Let us first estimate the case of odd rule, i.e., $|R_A \hat{g}^i_{3n+1} - \hat{g}^{i+1}_{3n+1}|$. For this, we represent our proposed algorithm in term of R_D-algorithm:

$$(R_A\hat{g}^i)_{3n+1} = (R_D\hat{g}^i)_{3n+1} + \frac{\tau}{81}3^{-2i_0}\mathbb{D}_2(\hat{g}^i)_{3n+1}. \tag{13}$$

Clearly $\mathbb{D}_2(\hat{g}^i)_{3n+1} = O(3^{-2i})$ for $n \in \mathbb{Z}$ because g is C^3 and g'' is bounded. It shows that

$$|(R_A\hat{g}^i)_{3n+1} - \hat{g}^{i+1}_{3n+1}| \le |(R_D\hat{g}^i)_{3n+1} - \hat{g}^{i+1}_{3n+1}| + O(3^{-2(i+i_0)}). \tag{14}$$

Since R_D-algorithm provide the fourth-order accuracy and $g \in W^4(\mathbb{R})$. It readily holds that $|(R_D\hat{g}^i)_{3n+1} - \hat{g}^{i+1}_{3n+1}| = O(3^{-4i})$. Combining this with (6) verifies the claim of this lemma. similarly done for odd rules. □

Theorem 5 Suppose the initial sequence of data $g^0 := g(3^{-i_0}n): n \in \mathbb{Z}$ is sampled from a function $g \in W^4_\infty(\mathbb{R})$, i.e., $g^0 := \hat{g}^{i_0}$. Let R_A be the proposed refinement algorithm with the refinement rule in (4), and $0 \le \psi_0 \le \tau 3^{-i_0}$ with constant $\tau > 0$. Then we have

$$\|R^\infty_A g^0 - g\|_\infty \le c_{g,\tau}3^{-4i_0},$$

where $c_{g,\tau} > 0$ is a constant depending on g and τ but independent of i_0.

Proof. For any point, $R := \{n3^{-i}: n \in \mathbb{Z}, i \in \mathbb{Z}_+\}$. Given that the initial sequence g^0 is supported on $3^{-i_0}\mathbb{Z}$, we consider $g^i := R_A^{i-i_0}\hat{g}^{i_0}$. The following telescoping sum can be obtained by direct computation:

$$g^i = R_A\hat{g}^{i-1} + \sum_{l=i_0}^{i-2} R_A^{i-l-1}(R_A\hat{g}^l - \hat{g}^{l+1}).$$

The stability of R_A-algorithm is ensured by $\|R^i_A\|_\infty \le c$ for any $i \in \mathbb{N}$. As a result, for any $i > i_0$ and $n \in \mathbb{Z}$, the relation below holds:

$$|g^i_n - g(n3^{-i})| \le \|R_A\hat{g}^{i-1} - \hat{g}^i\|_\infty + \sum_{l=i_0}^{i-2}\|R_A^{i-l-1}\|\|R_A\hat{g}^l - \hat{g}^{l+1}\|_\infty \le c\sum_{l=i_0}^{i-1}\|R_A\hat{g}^l - \hat{g}^{l+1}\|_\infty.$$

To estimate $\|R_A\hat{g}^l - \hat{g}^{l+1}\|_\infty$, by assumption $0 \le \psi_0 \le 3^{-2i_0}$ for some $\tau > 0$. By the previous lemma, it is immediate that $\|R_A\hat{g}^l - \hat{g}^{l+1}\|_\infty \le c_{g,\tau}3^{-2(i_0+l)}$, for $l \ge i_0$ with $c_{g,\tau} > 0$ depending on g and τ. It leads to the bound

$$\sum_{l=i_0}^{i-1}\|R_A\hat{g}^l - \hat{g}^{l+1}\|_\infty \le c_g\sum_{l=i_0}^{i-1}3^{-2(i_0+l)} \le c_g 3^{-4i_0}.$$

□

5. Evaluation of shape preservation properties

Smoothness, order of accuracy and shape-preservation properties like convexity and monotonicity are among the characteristics of a refinement algorithm that are crucial for practical and commercial applications. In this section, we examine the shape-preservation properties of our refinement rule. The refining rules are rewritten as follows:

$$R_A g^i := R_B g^i - 3^{-2i-4}(1-\psi_0)\mathbb{D}_2(g^i), \tag{15}$$

where $\mathbb{D}_2$ is given in (7) and R_B denotes the cubic B-spline associated with the mask $B = \{\frac{1}{27}, \frac{3}{27}, \frac{7}{27}, \frac{10}{27}, \frac{13}{27}, \frac{13}{27}, \frac{13}{27}, \frac{10}{27}, \frac{7}{27}, \frac{3}{27}, \frac{1}{27}\}$. If $\psi_0 = 0$, the proposed algorithm becomes the four-point refinement algorithm. The interpolatory nature of the four-point algorithm is a well-known intrinsic disadvantage. The resultant curves may have artifacts (such as self-intersection) if the initial points are irregular. Consequently, it makes sense to think that the parameter ψ_0 contributes to balancing the trade-off between the R_D-algorithm and the R_B-algorithm. After doing numerical experiments with different ψ_0, we have determined that a reasonable range for ψ_0 is

$$\frac{4}{7} < \psi_0 < 1 \tag{16}$$

provide the best smoothness as given in theorem 4.3.

5.1 *Monotonicity-preservation property*

We first demonstrate that our refinement algorithm R_A in (15) satisfies the monotonicity preservation property under a mild assumption. More precisely, we prove that if the discrete forward differences satisfy $(Dg^i)_n \geq 0$ for $n \in \mathbb{Z}$ and $i \in \mathbb{N}_0$. then this inequality is preserved under refinement, i.e., $(Dg^{i+1})_n \geq 0$. Before proceeding with the proof, we recall that R_B-algorithm satisfies the following relations:

$$\begin{cases} (DR_B g^i)_{3n} = \dfrac{2}{9}(Dg^i)_{n-1} + \dfrac{5}{9}(Dg^i)_n + \dfrac{2}{9}(Dg^i)_{n+1}, \\ (DR_B g^i)_{3n+1} = \dfrac{1}{9}(Dg^i)_{n-1} + \dfrac{4}{9}(Dg^i)_n + \dfrac{4}{9}(Dg^i)_{n+1}, \\ (DR_B g^i)_{3n+2} = \dfrac{4}{9}(Dg^i)_n + \dfrac{4}{9}(Dg^i)_{n+1} + \dfrac{1}{9}(Dg^i)_{n+2}. \end{cases} \tag{17}$$

This observation confirms that R_B-algorithm preserves monotonicity and using (15) and (17), we can now find the following identities

$$\begin{cases} (Dg^{i+1})_{3n} = \dfrac{2}{9}(Dg^i)_{n-1} + \dfrac{5}{9}(Dg^i)_n + \dfrac{2}{9}(Dg^i)_{n+1} \\ \qquad\qquad - 3^{-2i}\dfrac{(1-\psi_0)}{81}[21(D^3g^i)_n], \\ (Dg^{i+1})_{3n+1} = \dfrac{1}{9}(Dg^i)_{n-1} + \dfrac{4}{9}(Dg^i)_n + \dfrac{4}{9}(Dg^i)_{n+1} \\ \qquad\qquad - 3^{-2i}\dfrac{(1-\psi_0)}{81}[21(D^3g^i)_n], \\ (Dg^{i+1})_{3n+2} = \dfrac{4}{9}(Dg^i)_n + \dfrac{4}{9}(Dg^i)_{n+1} + \dfrac{1}{9}(Dg^i)_{n+2} \\ \qquad\qquad - 3^{-2i}\dfrac{(1-\psi_0)}{81}[21(D^3g^i)_{n+1}]. \end{cases} \tag{18}$$

Definition 4 (*Condition-M*). Assume that ψ_0 be provided in (5), and let g^{i_0} be connected to $3^{-i_0}\mathbb{Z}$. The sequence g^{i_0} is said to satisfy *condition-M* if

$$(Dg^{i_0})_n \geq 3^{-2i_0}\frac{(1-\psi_0)}{c}|(D^3g^{i_0})_n|, \ \forall n \in \mathbb{Z}, \tag{19}$$

where $c > 0$.

Lemma 2 Assuming that the collection of grid points $3^{-i_0}\mathbb{Z}$ has the initial sequence g^{i_0} associated to it and let $g^{i+1} = R_A g^i$, $i \geq i_0$. If g^{i_0} satisfies the *condition-M*, then for each $i \geq i_0$,

$$(Dg^i)_n \geq 3^{-2i}\frac{(1-\psi_0)}{c}|(D^3g^i)_n|, \forall n \in \mathbb{Z}. \tag{20}$$

Proof. Mathematical induction will be used to validate this theory. As the original sequence, by hypothesis, meets the *condition-M*, showing that (20) is accurate for i_0. Under the assumption that (20) is satisfied for $i > i_0$, we proceed to prove that:

$$(Dg^{i+1})_n \geq 3^{-2(i+1)}\frac{1-\psi_0}{c}|(D^3g^{i+1})_n|.$$

We break down our proof into three conditions: (1) $n = 3j$, (2) $n = 3j+1$ and (3) $n = 3j+2$. First, we address the condition $n = 3j$. In order to do this, applying (18), the expression is simply derived.

$$(D^3g^{i+1})_{3j} = 7.3^{2(i+1)}((Dg^{i+1})_{3j-1} - 2(Dg^{i+1})_{3j} + (Dg^{i+1})_{3j+1})$$

$$= 7(D^3g^i)_j - \frac{1-\psi_0}{9}.7(21(D^3g^i)_j - 42(D^3g^i)_j + 21(D^3g^i)_j).$$

Thus, it follows that

$$|(D^3g^{i+1})_{3j}| \leq 7|(D^3g^i)_j|.$$

When combined with (18), this creates the relation

$$\begin{cases} (Dg^{i+1})_{3j} - 3^{-2(i+1)}\dfrac{1-\psi_0}{c}|(D^3g^{i+1}))_{3j}| \geq \dfrac{2}{9}((Dg^i)_{j-1} - 3^{-2i}\dfrac{1-\psi_0}{c}|(D^3g^i)_{j-1}|) \\ \qquad\qquad + \dfrac{5}{9}((Dg^i)_j - 3^{-2i}\dfrac{1-\psi_0}{c}|(D^3g^i)_j|) \\ \qquad\qquad + \dfrac{2}{9}((Dg^i)_{j+1} - 3^{-2i}\dfrac{1-\psi_0}{c}|(D^3g^i)_{j+1}|). \end{cases} \tag{21}$$

The induction hypothesis for i ensures that the last term does not take negative values. We now consider the case $n = 3j+1$. From the discussion above and the identities in (18), it follows that:

$$(D^3g^{i+1})_{3j+1} = (D^3g^i)_j - \frac{1-\psi_0}{9}.7(21(D^3g^i)_j - 42(D^3g^i)_j + 21(D^3g^i)_j)$$

$$|(D^3g^{i+1})|_{3j+1} \leq 7|(D^3g^i)_j|.$$

Hence, applying the same argument in (21)

$$\begin{aligned}(Dg^{i+1})_{3j+1} - 3^{-2(i+1)}\frac{1-\psi_0}{c}|(D^3g^{i+1}))_{3j+1}| \geq &\frac{1}{9}((Dg^i)_{j-1} - 3^{-2i}\frac{1-\psi_0}{c}|(D^3g^i)_{j-1}|) \\ &+ \frac{4}{9}((Dg^i)_j - 3^{-2i}\frac{1-\psi_0}{c}|(D^3g^i)_j|) \\ &+ \frac{4}{9}((Dg^i)_{j+1} - 3^{-2i}\frac{1-\psi_0}{c}|(D^3g^i)_{j+1}|).\end{aligned}$$

Now, we turn to the case $n = 3j+2$. Using the identities in (18), we can obtain

$$|(D^3g^{i+1})|_{3j+2} \leq \frac{49}{3}|(D^3g^i)_j| + \frac{-28}{3}|(D^3g^i)_{j+1}|.$$

A direct computation using (18) yields the outcome.

$$(Dg^{i+1})_{3j+2} - 3^{-2(i+1)}\frac{1-\psi_0}{c}|(D^3 g^{i+1}))_{3j+1}| \geq 0.$$

□

Theorem 6 Let the initial sequence attached to g^{i_0} is connected to $3^{-i_0}\mathbb{Z}$ and $g^{i+1} = R_A g^i$, $i \geq i_0$, with R_A-algorithm in (4). If g^{i_0} assure the *condition-M*, then we have $(Dg^i)_n \geq 0$ with $i \geq i_0$ and $n \in \mathbb{Z}$. Thus R_A-algorithm maintains monotonicity.

Proof. In view of Lemma 2, we can establish the bound for any $i \geq i_0$, $(Dg^{i+1})_n \geq 3^{-2(i+1)}\frac{1-\psi_0}{c}|(D^3 g^{i+1})_n|$. As a result, the conclusion of this theorem is established without delay. □

5.2 *Convexity-preservation property*

The proposed refinement rule R_A possesses convexity preservation property under a moderate restriction. We use the notation

$$\nabla g_n^i := g_n^i - g_{n-1}^i \text{ and } \Delta g_n^i := g_{n-1}^i - 2g_n^i + g_{n+1}^i.$$

Lemma 3 Suppose the cubic B-spline algorithm is denoted as R_B. Then, for any $n \in \mathbb{Z}$,

$$\begin{cases} (D^2 R_B g^{i+1})_{3n} = \frac{1}{3}(D^2 g^i)_{n-1} + \frac{1}{3}(D^2 g^i)_n + \frac{1}{3}(D^2 g^i)_{n+1} \\ (D^2 R_B g^{i+1})_{3n+1} = \frac{2}{3}(D^2 g^i)_n + \frac{1}{3}(D^2 g^i)_{n+1}, \\ (D^2 R_B g^{i+1})_{3n+2} = \frac{1}{3}(D^2 g^i)_n + \frac{2}{3}(D^2 g^i)_{n+1}. \end{cases} \tag{22}$$

Lemma 4 Let R_A-algorithm be defined in (4), and ψ_0 as provided in (5). If $g^{i+1} = R_A g^i$ for $i \geq i_0$, then we obtain

$$\begin{cases} (D^2 g^{i+1})_{3n} = \frac{1}{3}(D^2 g^i)_{n-1} + \frac{1}{3}(D^2 g^i)_n + \frac{1}{3}(D^2 g^i)_{n+1} \\ \qquad\qquad - 3^{-i}\frac{1-\psi_0}{27}(21\nabla(D^3 g^i)_{n+1}), \\ (D^2 g^{i+1})_{3n+1} = \frac{2}{3}(D^2 g^i)_n + \frac{1}{3}(D^2 g^i)_{n+1}, \\ (D^2 g^{i+1})_{3n+2} = \frac{1}{3}(D^2 g^i)_n + \frac{2}{3}(D^2 g^i)_{n+1}. \end{cases} \tag{23}$$

Lemma 5 Suppose R_A-algorithm denote the refinement algorithm given in (4) when used on initial data sequence g^{i_0} supported on the grid points $3^{-i_0}\mathbb{Z}$. For each $i \geq i_0$, the refinement process is given recursively by $g^{i+1} = R_A g^i$. The third-order differences of the refined data satisfy the following relations:

$$\begin{cases} \nabla(D^3 g^{i+1})_{3n} = -\dfrac{161}{27}\nabla(D^3 g^i)_{n+1}, \\ \nabla(D^3 g^{i+1})_{3n+1} = 0, \\ \nabla(D^3 g^{i+1})_{3n+2} = 0. \end{cases} \tag{24}$$

Proof. Applying Lemma 4, direct evaluation results in

$$\begin{aligned} 3^{-(i+1)\nabla(D^3 g^{i+1})_{3n}} &= \frac{7}{3}((D^2 g^{i+1})_{3n-1} - 2(D^2 g^{i+1})_{3n} + (D^2 g^{i+1})_{3n+1}) \\ &= \frac{-7}{9}3^{-i}\nabla(D^3 g^i)_{n+1} - 3^{-i}\frac{(1-\psi_0)}{81}(98\nabla(D^3 g^i)_{n+1}) \\ &= -\frac{161}{27}\nabla(D^3 g^i)_{n+1}, \end{aligned}$$

This completes the verification of the first equation. The remaining two can be derived analogously.

$$3^{-(i+1)}\nabla(D^3 g^{i+1})_{3n+1} = 0 \text{ and } 3^{-(i+1)}\nabla(D^3 g^{i+1})_{3n+2} = 0.$$

□

Definition 5 (Condition-A). Let ψ_0 be a constant specified in (5), and suppose the initial sequence g^{i_0} be associated with the grid $3^{-i_0}\mathbb{Z}$. The sequence g^{i_0} is said to satisfy the *Condition-A* if

$$(D^2 g^{i_0})_n \geq 3^{-2i_0}\frac{1-\psi_0}{c}|\nabla(D^3 g^{i_0})_{n+1}|, \ \forall n \in \mathbb{Z}, \tag{25}$$

where $c > 0$.

Theorem 7 Let R_A be the refinement rule in (4) with ψ_0 in (16). Assume that the initial sequence, denoted by g^{i_0} be attached to $3^{-i_0}\mathbb{Z}$, and satisfies the given *Condition-A*. Then, for any $i > i_0$, $(\Delta g^i)_n \geq 0$ for $n \in \mathbb{Z}$, which indicates convexity preservation property of R_A-algorithm.

Proof. Applying the induction of hypothesis, beginning with a verification for all $i \geq i_0$,

$$(D^2 g^i)_n \geq \frac{1-\psi_0}{c}3^{-i}|\nabla(D^3 g^i)_{n+1}|, \ \forall n \in \mathbb{Z}. \tag{26}$$

As assumed, the initial sequence g^{i_0} clearly meets (26). Hence, supposing the inequality (26) holds for the case $i > i_0$, we will check it for the case $i+1$. To begin with, we consider the case of $3i-1$. Implementing Lemma 5, we obtain

$$(D^2 g^{i+1})_{3n-1} - \frac{1-\psi_0}{c} 3^{-(i+1)} |\nabla (D^3 g^{i+1})_{3n}| \geq \frac{1}{3}(D^2 g^i)_{n-1} + \frac{2}{3}(D^2 f^k)_n$$

$$- 3^{-(i+1)} \frac{1-\psi_0}{c} \frac{161}{27} |\nabla (D^3 g^i)_{n+1}|.$$

$$(D^2 g^{i+1})_{3n-1} - \frac{1-\psi_0}{c} 3^{-(i+1)} |\nabla (D^3 g^{i+1})_{3n}| \geq \frac{1}{3}((D^2 g^i)_{n-1} - 3^{-i} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_n|)$$

$$+ \frac{2}{3}((D^2 g^i)_n - 3^{-i} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_{n+1}|).$$

According to the induction hypothesis the final term is non-negative for i, Next consider the case $3i$. Due to Lemma 5, we get

$$(D^2 g^{i+1})_{3n} - \frac{1-\psi_0}{c} 3^{-(i+1)} |\nabla (D^3 g^{i+1})_{3n+1}| \geq \frac{1}{3}(D^2 g^i)_{n-1} + \frac{1}{3}(D^2 g^i)_n + \frac{1}{3}(D^2 g^i)_{n+1}$$

$$- 3^{-i} \frac{1-\psi_0}{27} |21 \nabla (D^3 g^i)_n|$$

$$- 3^{-(i+1)} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_n - 2\nabla (D^3 g^i)_{n+1}$$

$$+ \nabla (D^3 g^i)_{n+2}|.$$

$$(D^2 g^{i+1})_{3n} - \frac{1-\psi_0}{c} |\nabla (D^3 g^{i+1})_{3n+1}| \geq \frac{1}{3}((D^2 g^i)_{n-1} - 3^{-i} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_n|)$$

$$+ \frac{1}{3}((D^2 g^i)_n - 3^{-i} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_{n+1}|)$$

$$+ \frac{1}{3}((D^2 g^i)_{n+1} - 3^{-i} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_{n+2}|).$$

The induction hypothesis ensures that the last term is non-negative for i. Next consider the case $3i+1$.

$$(D^2 g^{i+1})_{3n+1} - \frac{1-\psi_0}{c} |\nabla (D^3 g^{i+1})_{3n+2}| \geq \frac{2}{3}(D^2 g^i)_n + \frac{1}{3}(D^2 g^i)_{n+1}$$

$$- 3^{-(i+1)} \frac{1-\psi_0}{c} |\nabla (D^3 g^i)_{n+2} - \nabla (D^3 g^i)_{n+1}|.$$

$$(D^2g^{i+1})_{3n+1} - \frac{1-\psi_0}{c}|\nabla(D^3g^{i+1})_{3n+2}| \geq \frac{2}{3}((D^2g^i)_n - 3^{-i}\frac{1-\psi_0}{c}|\nabla(D^3g^i)_{n+1}|)$$

$$+ \frac{1}{3}((D^2g^i)_{n+1} - 3^{-i}\frac{1-\psi_0}{c}|\nabla(D^3g^i)_{n+2}|).$$

The final term is non-negative, as established using the induction hypothesis for i. □

6. Numerical examples

Numerical examples are presented in this section to evaluate the R_A-algorithm. In these examples, we present the initial curve with dotted lines, control points with $'\square'$ and the refined results with solid lines.

Example 1 (**Order of accuracy**). The objective of this example is to numerically verify the accuracy of the refinement algorithm defined in equation (4). To this end, we evaluate the approximation orders predicted by Theorem 4.4 using the following two functions:

$$g_1(x) := x^4,$$

$$g_2(x) := x^4\cos(x).$$

Input data sequences are obtained by sampling these functions over the interval $[0, \frac{1}{10}]$ with density 3^{-i_0} for $i = 0, 1, 2, ..., 7$. The parameter ψ_0 is set as $\psi_0 = 3^{-i_0}$ which correspond to $\tau = 1$ in (5) for each level of sampling.

Table 2 shows the maximum error and order of accuracy attained by the proposed algorithm. The functions g_1 and g_2 belong to the Sobolev space achieve the optimal approximation order of four. These numerical results are consistent with the theoretical results in Theorem 5.

Table 2. The maximum error and accuracy order of R_A-algorithm

Density (h_0)	Max. error g_1	Order g_1	Max. error g_2	Order g_2
3^{-1}	9.5×10^{-5}	–	9.4×10^{-5}	–
3^{-2}	1.17×10^{-6}	4.0	1.2×10^{-6}	4.0
3^{-3}	1.4×10^{-8}	4.0	1.4×10^{-8}	4.0
3^{-4}	1.79×10^{-10}	4.0	1.8×10^{-10}	4.0
3^{-5}	2.21×10^{-12}	4.0	2.2×10^{-12}	4.0
3^{-6}	2.72×10^{-14}	4.0	2.7×10^{-14}	4.0
3^{-7}	3.36×10^{-16}	4.0	3.34×10^{-16}	4.0

Example 2 (**Comparison with parents algorithms**). This example illustrates the numerical results of our refinement algorithm R_A with their parents algorithms (i.e., R_D and R_B). Figure 2(a) and 2(b) show the limit curve when ψ_0 is set to 0, it corresponds the R_D-algorithm though this results in an undesirable artifact and when ψ_0 increases toward 1 it becomes R_B-algorithm the curves deviate more significantly from the control polygon. Since the R_A-algorithm has the same support size as both parent algorithms, its computational complexity remains unchanged.

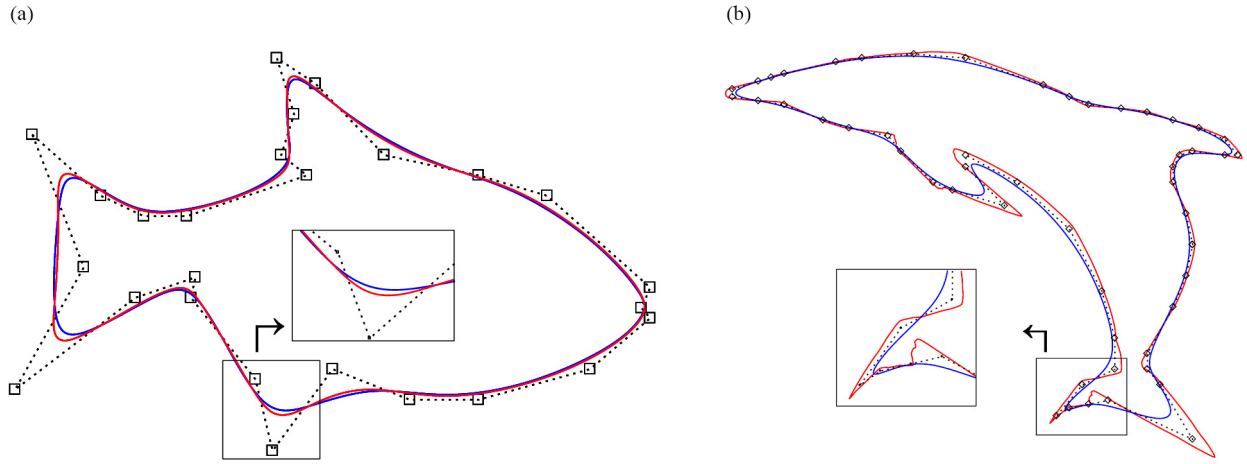

Figure 2. Comparison of R_A-algorithm (blue) with parent algorithms (a) shows comparison with R_B-algorithm (red) and (b) shows comparison with R_D-algorithm (red), all at the refinement step size = 3

Example 3 (**Discontinuous function**). In this example, we show and analyze the performance of the R_A-algorithm on a discontinuous function defined as:

$$g(x) = \begin{cases} x, & \text{if } 0 \leq x < 1, \\ x+1 & \text{if } 1 \leq x \leq 2. \end{cases}$$

This step function is sampled uniformly at nine points from $x = 0$ to $x = 2$. the algorithm is applied using fixed parameter value $\psi_0 = \frac{5}{7}$. Figure 3 shows the limit curve of the discontinuous function and the refinement algorithm. The results show that R_A-algorithm effectively handles discontinuities without introducing unwanted oscillations, while preserving local features near the jumps. This highlights the robustness of the algorithm when applied to non-smooth or irregular data.

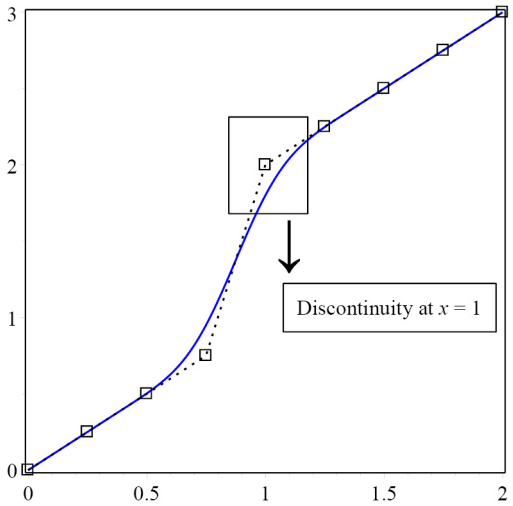

Figure 3. Discontinuous function with R_A-algorithm at $\psi_0 = 6/7$

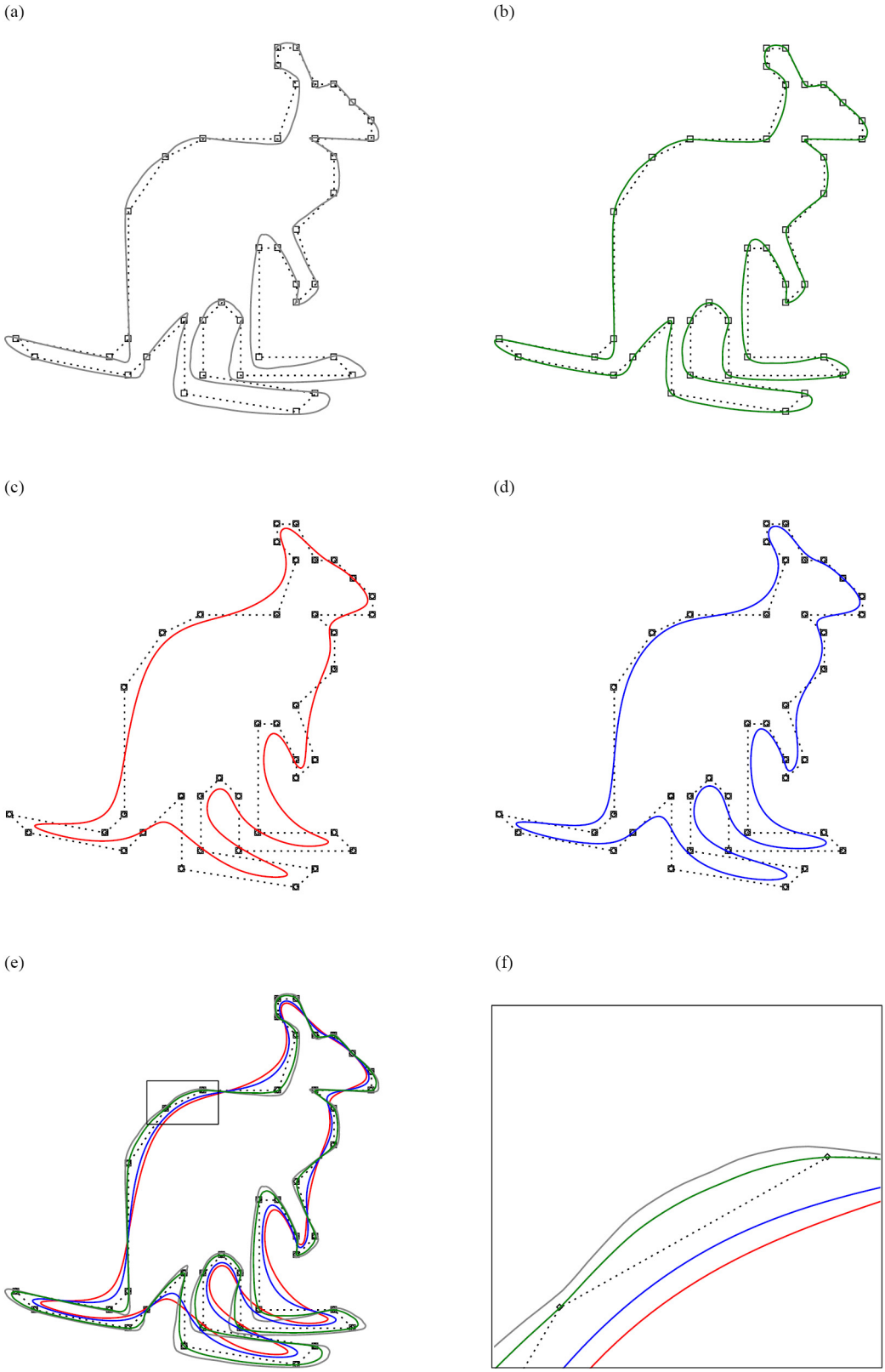

Figure 4. Comparison of curves for different smoothness levels C^0, C^1, C^2, C^3 corresponding to $\psi_0 = -\frac{1}{5}, 0, 1, \frac{5}{7}$, including zoomed-in area to highlight the differences. (a) C^0 continuity; (b) C^1 continuity; (c) C^2 continuity; (d) C^3 continuity; (e) Comparison; (f) zoom-area

Example 4 (**Smoothness levels**). In this example, we demonstrate the effect of all smoothness levels on the initial limit curve. By Theorem 4, the limiting curve is three times continuously differentiable as it has C^3 smoothness. Figure 4 shows the resulting limit curves generated by our refinement algorithm for the tension parameter from different smoothness levels. When ψ_0 is set from C^0 continuity this results a non-smooth deformation. As ψ_0 increases towards C^3 continuity, the curve becomes smoother and moves further from the control polygon, showing that the value of the tension parameter ψ_0 has a strong impact on the geometric form of the curve.

Example 5 (**Monotonicity-preservation**). This example presents the monotonicity preservation of our refinement algorithm R_A in (4). Let the initial data is taken from the monotone data set presented in Table 3 and their first derivatives in Table 4. Based on the monotone data set given in Table 3, the initial data is strictly monotone, i.e., $g_n^{i_0} = \{0.0067, 0.018, 0.048, 0.12, 0.27, 0.50, 0.71, 0.91, 1.0, 1.0, 1.0\}$ corresponds to the sample points $x = -5, -4, ..., 5$. The first order differences of the data are, $Dg_n^{i_0} = \{0.0113, 0.030, 0.072, 0.15, 0.23, 0.21, 0.20, 0.09, 0, 0, 0\}$, all of which are non-negative. This confirms that the data set satisfies the monotonicity condition $Dg_n^{i_0} \geq 0$, and therefore fulfills Condition M for the refinement process. Figure 5 shows the limit curve of the refinement algorithm and their first derivative. The tension parameter $\psi_0 = \frac{3}{7}, \frac{4}{7}, \frac{5}{7}, \frac{6}{7}$ then, these results confirm the monotone preservation of our R_A-algorithm, ensuring that the generated limit curves maintain the same monotonic trend as the underlying data.

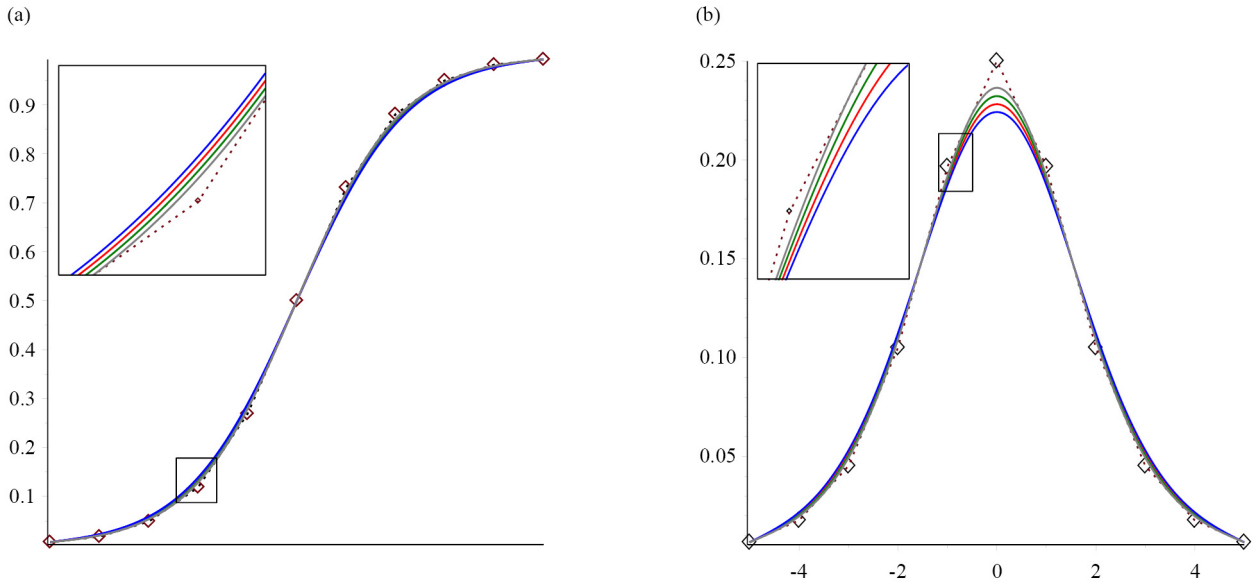

Figure 5. Monotonicity preservation: Limit functions for $\psi_0 = \frac{6}{7}$ (blue), $\psi_0 = \frac{5}{7}$ (red), $\psi_0 = \frac{4}{7}$ (green), $\psi_0 = \frac{3}{7}$ (gray) in (a) and their first derivative in (b)

Table 3. Monotonicity preservation

x	-5	-4	-3	-2	-1	0	1	2	3	4	5
y	0.0067	0.018	0.048	0.12	0.27	0.50	0.71	0.91	1.0	1.0	1.0

Table 4. First derivative of given monotonic function

x	-5	-4	-3	-2	-1	0	1	2	3	4	5
y	0.007	0.018	0.045	0.105	0.197	0.250	0.197	0.105	0.045	0.018	0.007

Example 6 (**Convexity-preservation**). In this example, we apply the R_A-algorithm to analyze its performance in maintaining convexity using the values presented in Table 5. The data $g_n^{i_0} = \{54.60, 9.49, 2.72, 1.28, 1, 1.28, 2.72, 9.49, 54.60\}$ is strictly convex. The second order differences of the data are $D^2 g_n^{i_0} = \{29.34, 5.33, 1.44, 0.72, 0, 0.72, 1.44, 5.33, 29.34\}$, all of which are non-negative. This confirms that the data set satisfies the convexity condition $(D^2 g^{i_0})_n \geq 0$, and therefore fulfills Condition-A for the refinement process. The sequence is symmetric and strictly positive except at the central point where it reaches its minimum. Figure 6 shows the limit curve of R_A-algorithm and their second derivative with the choice of tension parameter as $\psi_0 = \frac{3}{7}, \frac{4}{7}, \frac{5}{7}, \frac{6}{7}$, then This result shows that the proposed refinement algorithm R_A maintain the convexity of initial data.

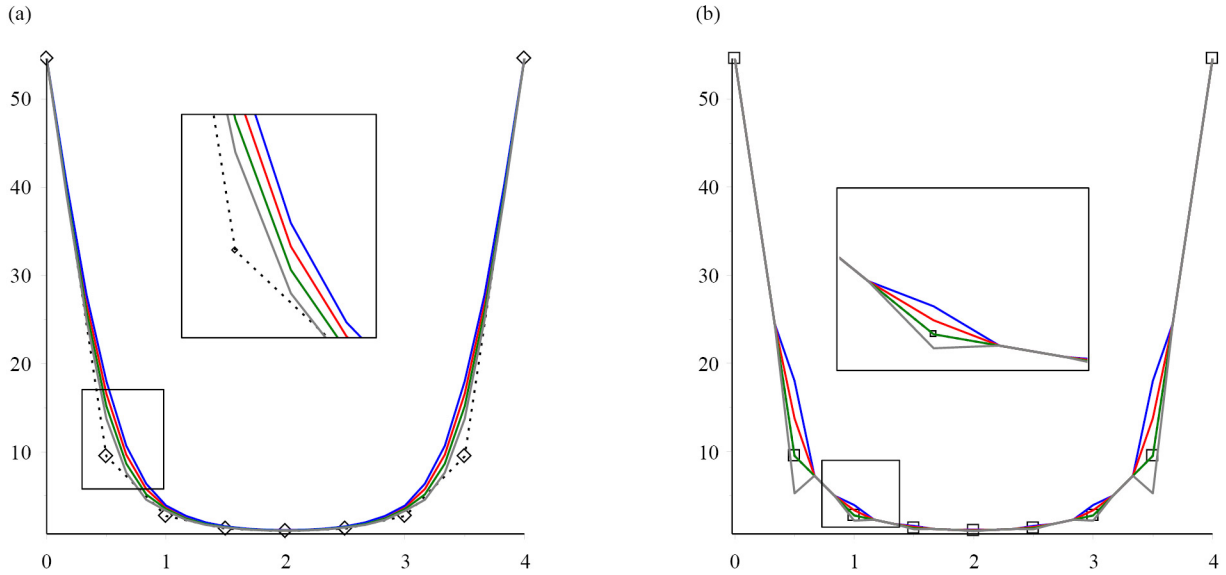

Figure 6. Convexity preservation: Limit functions for $\psi_0 = \frac{6}{7}, \frac{5}{7}, \frac{4}{7}, \frac{3}{7}, \frac{2}{7}$ (a) and their 2nd derivative (b)

Table 5. Convexity preservation

x	0	0.5	1	1.5	2	2.5	3	3.5	4
y	54.60	9.49	2.72	1.28	1	1.28	2.72	9.49	54.60

7. Conclusion

This study propose a new refinement algorithm that successfully achieves fourth-order accuracy, while enhancing smoothness to the level of C^3 continuity. Through a detailed analytical approach, constraints on the tension parameter were carefully defined to uphold the structural stability and functional flexibility of proposed algorithm. Furthermore, we have derived and validated the mathematical conditions required for the preservation of monotonicity and convexity properties in many geometric modeling and data approximation tasks. By applying the method to multiple numerical scenarios, we verified its capability to provide reliable results while preserving vital shape properties. These results demonstrate that the developed algorithm is a promising tool for applications requiring smooth and shape-preservation refinement, such as computer graphics, scientific visualization, and curve design.

Authors contributions

Pakeeza Ashraf: Writing—review editing, Writing—original draft, Methodology, Formal analysis, Data curation, Conceptualization, Supervision. Amna Kalsoom: Writing—original draft, Methodology, Formal analysis, Conceptualization. Javeria Ashraf: Writing—original draft, Formal analysis, Software. Ali Akgul: Writing—review editing, Formal analysis, Data curation, Conceptualization. Fahad Sameer Alshammari: Writing—review editing, Investigation.

Availability of data and materials

The data used to support the findings of the study are available within this paper.

Conflict of interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Ashraf P, Mustafa G. A generalized non-stationary 4-point *b*-ary approximating scheme. *British Journal of Mathematics and Computer Science*. 2014; 4(1): 104–119. Available from: https://doi.org/10.9734/BJMCS/2014/4120.

[2] de Rham G. Sur une courbe plane [On a plane curve]. *Journal de Mathématiques Pures et Appliquées [Journal of Pure and Applied Mathematics]*. 1956; 35: 25–42.

[3] Chaikin GM. An algorithm for high-speed curve generation. *Computer Graphics and Image Processing*. 1974; 3(4): 346–349. Available from: https://doi.org/10.1016/0146-664X(74)90028-8.

[4] Dubuc S. Interpolation through an iterative scheme. *Journal of Mathematical Analysis and Applications*. 1986; 114(1): 185–204. Available from: https://doi.org/10.1016/0022-247X(86)90077-6.

[5] Dyn N, Levin D, Gregory JA. A 4-point interpolatory subdivision scheme for curve design. *Computer Aided Geometric Design*. 1987; 4(4): 257–268. Available from: https://doi.org/10.1016/0167-8396(87)90001-X.

[6] Deslauriers G, Dubuc S. Symmetric iterative interpolation processes. *Constructive Approximation*. 1989; 5: 49–68. Available from: https://doi.org/10.1007/BF01889598.

[7] Mejstrik T. *Joint Spectral Radius and Subdivision Schemes*. Vienna, Austria; 2019.

[8] Ashraf P, Mustafa G, Ghaffar A, Zahra R, Nisar KS, Mahmoud EE, et al. Unified framework of approximating and interpolatory subdivision schemes for construction of class of binary subdivision schemes. *Journal of Function Spaces*. 2020; 2020(1): 6677778. Available from: https://doi.org/10.1155/2020/6677778.

[9] Weissman A. *A 6-Point Interpolatory Subdivision Scheme for Curve Design*. University of Tel-Aviv; 1989.

[10] Tang Y, Ko KP, Lee BG. A new proof of the smoothness of 4-point Deslauriers-Dubuc scheme. *Journal of Applied Mathematics and Computing*. 2005; 8: 553–562.

[11] Kuijt F, van Damme R. Monotonicity preserving interpolatory subdivision schemes. *Journal of Computational and Applied Mathematics*. 1999; 101(1-2): 203–229. Available from: https://doi.org/10.1016/S0377-0427(98)00220-9.

[12] Beccari C, Casciola G, Romani L. Shape controlled interpolatory ternary subdivision. *Applied Mathematics and Computation*. 2009; 215(3): 916–927. Available from: https://doi.org/10.1016/j.amc.2009.06.014.

[13] Hassan MF, Ivrissimitzis IP, Dodgson NA, Sabin MA. An interpolating 4-point C^2 ternary stationary subdivision scheme. *Computer Aided Geometric Design*. 2002; 19(1): 1–18. Available from: https://doi.org/10.1016/S0167-8396(01)00084-X.

[14] Cai Z. Convexity preservation of the interpolating four-point C^2 ternary stationary subdivision scheme. *Computer Aided Geometric Design*. 2009; 26(5): 560–565. Available from: https://doi.org/10.1016/j.cagd.2009.02.004.

[15] Pitolli F. Ternary shape-preserving subdivision schemes. *Mathematics and Computers in Simulation*. 2014; 106: 185–194. Available from: https://doi.org/10.1016/j.matcom.2013.04.003.

[16] Dyn N. Subdivision schemes in CAGD. *Advances in Numerical Analysis*. 1992; 2: 36–104.

[17] Maillot J, Stam J. A unified subdivision scheme for polygonal modeling. *Computer Graphics Forum*. 2001; 20(3): 471–479.

[18] Rossignac J, Schaefer S. J-splines. *Computer-Aided Design*. 2008; 40(10-11): 1024–1032. Available from: https://doi.org/10.1016/j.cad.2008.09.002.

[19] Kaklis PD, Stamatelopoulos S, Ginnis AA. Shape-preserving interpolation on surfaces via variable-degree splines. *Computer Aided Geometric Design*. 2024; 109: 102276. Available from: https://doi.org/10.1016/j.cagd.2024.102276.

[20] Yang H, Kim K, Yoon J. A family of C^2 four-point stationary subdivision schemes with fourth-order accuracy and shape-preserving properties. *Journal of Computational and Applied Mathematics*. 2024; 446: 115843. Available from: https://doi.org/10.1016/j.cam.2024.115843.

[21] Razaq S, Mustafa R, Mustafa G. A hybrid family of non-uniform ternary schemes with mixed symmetry for approximating shapes. *Journal of Applied Mathematics and Computing*. 2025; 71: 2821–2857. Available from: https://doi.org/10.1007/s12190-024-02342-7.

[22] Kim Y, Yang H, Yoon J. Shape-preserving subdivision algorithm with the third-order accuracy and C^2 smoothness. *Mathematics and Computers in Simulation*. 2025; 235: 160–174. Available from: https://doi.org/10.1016/j.matcom.2025.03.030.

[23] Dyn N. Interpolatory subdivision schemes. In: *Tutorials on Multiresolution in Geometric Modelling: Summer School Lecture Notes*. Springer Berlin, Heidelberg; 2002. p.25–50. Available from: https://doi.org/10.1007/978-3-662-04388-2.

[24] Levin A. Polynomial generation and quasi-interpolation in stationary non-uniform subdivision. *Computer Aided Geometric Design*. 2003; 20(1): 41–60. Available from: https://doi.org/10.1016/S0167-8396(02)00115-3.