Mathematics

https://www.aimspress.com/journal/Math

AIMS Mathematics, 10(11): 26077–26105.
DOI: 10.3934/math.20251148
Received: 17 August 2025
Revised: 06 October 2025
Accepted: 17 October 2025
Published: 11 November 2025

Research article

Role of s-convexity in the generation of fractals as Julia and Mandelbrot sets via three-step fixed point iteration

Anita Tomar[1], **Swati Antal**[2], **Mohammad Sajid**[3,*] **and Darshana J. Prajapati**[4]

[1] Pt.L.M.S. Campus, Sridev Suman Uttarakhand University, Rishikesh, Uttarakhand, India

[2] Army Cadet College Wing, IMA, Dehradun, Uttarakhand, India

[3] Department of Mechanical Engineering, College of Engineering, Qassim University, Saudi Arabia

[4] Madhuben and Bhanubhai Patel Institute of Technology (MBIT), The CVM University, New V.V. Nagar, Gujarat, India

* **Correspondence:** Email: msajd@qu.edu.sa.

Abstract: In this manuscript, we investigate the dynamics of higher-order polynomials with complex coefficients by employing the Jungck–Noor iteration scheme (one of the iterative methods) in conjunction with s-convexity, which controls the weighting of previous iterates versus current polynomial evaluations, tuning convergence speed, escape dynamics, and fractal density from compact, high-brightness patterns to intricate, detailed structures. This framework enables us to establish new escape criteria and to visualize nonclassical deviations of the celebrated Mandelbrot and Julia sets. The resulting fractal structures display intricate geometries that not only enrich the theoretical study of complex dynamics but also resemble patterns observed in natural systems. To highlight the novelty of our work, we provide both graphical and numerical illustrations that demonstrate how variations in polynomial parameters and iteration settings influence shape transformations, symmetry, color distributions, and computational complexity. A key observation is that each point in the Mandelbrot set encodes detailed information about the corresponding Julia fractal, reinforcing the deep interplay between the two families. Moreover, when real-valued parameters are considered in the polynomial map and iteration process, some fractals exhibit striking motifs suggestive of potential applications, for example, in pattern design within the textile industry. Our study also opens pathways for extending this framework to noise-perturbed systems and physical models, which will be explored in future work.

Keywords: escape time algorithms; iterative methods; Jungck-Noor iteration; s-convexity; fractals; Julia set; Mandelbrot set; computational complexity
Mathematics Subject Classification: 28A10, 31E05, 37F10, 37F46, 47H10

1. Introduction

Fractal geometry captivates researchers through its artistic appeal and the complexity of its structures [1, 2]. Unlike traditional Euclidean geometry, fractal geometry enables the modeling of problems that would otherwise be analytically intractable [3]. The concept of a "fractal" was introduced in 1975 by Benoit Mandelbrot, whose most famous contribution, the Mandelbrot set [4], is celebrated for its intricate boundary, aesthetic richness, and wide-ranging applications. Mandelbrot also established that the boundary of the Mandelbrot set corresponds to Julia sets, a concept originally studied by Gaston Julia [5]. Together, the Mandelbrot and Julia sets have become central to the visualization and analysis of complex dynamical systems (see [6–8], and references therein). A simple quadratic polynomial of the form $f(z) = z^2 + a_3$, where $a_3 \in C$, can generate highly intricate fractals. Early constructions of Mandelbrot and Julia sets relied on the Picard iteration [9] to compute escape times. Over the years, researchers have introduced advanced iterative schemes such as the Krasnosel'skii–Mann method [10], Jungck–Ishikawa with s-convexity [11], Ishikawa [12], Jungck–CR with s-convexity [13], Mann [14], Noor [15], Noor iteration with s-convexity [16], Picard–Mann with s-convexity [17], Jungck–Mann with s-convexity [18], four-step iteration with s-convexity [19], SP-iteration with s-convexity [20, 21], and Fibonacci-Mann [22], each revealing new aspects of fractal dynamics for various polynomial families [23–25]. More recently, studies have extended fractal dynamics to fractional systems, diffusion-limited aggregation (DLA), highlighting their significance in physical and applied contexts [26–28]. Building upon this literature, Jolaoso et al. [29] investigated biomorphs using higher-order polynomials of the form $f(z) = z^n + a_1z^2 + a_2z + a_3$.

In the present paper, we refine this polynomial model and employ the Jungck–Noor iteration with s-convexity to establish a novel escape-time criterion. The s-convexity is used to ensure clarity and precision, as it plays a crucial role in modifying convergence behavior. The Jungck–Noor and Jungck–Ishikawa orbits are also formally defined, with explicit expressions for the points Tu_k and Ty_k, to eliminate ambiguity. While the Julia and Mandelbrot sets are mathematically invariant under different iterative schemes, the choice of algorithm influences the numerical approximation, visual representation, and computational cost. Our approach highlights how polynomial degree n, convexity parameter s, and iteration parameters α_1, α_2, and α_3 shape the resulting fractals in terms of geometry, symmetry, and color. To ensure transparency, Section 4.1 provides the rationale for parameter choices in both Julia and Mandelbrot cases, showing how they produce nonclassical deviations and "mutant" variants. Our MATLAB-based algorithms generate fractals in under four seconds, supported by a color-mapping technique for enhanced visualization. Numerical experiments reveal a nonlinear dependence of generation time on iteration parameters, with s exerting the strongest influence. The zoomed boundaries of the Mandelbrot set reveal embedded Julia sets, reaffirming their deep structural interdependence and self-similarity. Interestingly, several generated patterns bear resemblance to Kachhi thread work, a traditional embroidery art from Gujarat, India, indicating potential applications in textile design. In addition to aesthetic and computational insights, our theorems and corollaries provide generalizations and refinements of existing results, addressing gaps in earlier works such as Nazeer et al. [30]. We also acknowledge the relevance of recent studies on noise-perturbed Julia and Mandelbrot sets (Andreadis & Karakasidis [31]; Sun et al. [32]), which provide a natural direction for extending this framework to perturbed and physically inspired systems.

The remainder of the paper is structured as follows: Section 2 introduces preliminary concepts, including a formal definition of s-convexity. Section 3 develops the Jungck–Noor iteration with s-convexity and derives escape radii. Section 4 presents algorithms and visualizations of Julia and Mandelbrot mutants. Section 5 analyzes numerical results, including the generation time versus parameter dependence. Section 6 concludes with key findings, while a concise discussion of future work highlights possible extensions to noise-perturbed maps, higher-dimensional systems, and applications in physical modeling.

2. Preliminaries

The filled Julia set [1,5] of a complex polynomial $f : \mathbb{C} \to \mathbb{C}$ of degree ≥ 2 is described as

$$F_f = \{z \in \mathbb{C} : \{|f^k(z)|\}_{k=0}^{\infty} \text{ is bounded}\},$$

where $f^k(z)$ is the k-fold composition of f with itself (that is, the k^{th} iteration of the function). The set F_f consists of all complex numbers whose orbits remain bounded and do not reach infinity. The boundary of F_f is the Julia set of f, that is, $J_f = \partial F_f$.

The Mandelbrot set M [2,4] is defined as

$$M = \{a_3 \in \mathbb{C} : F^k_{f_{a_3}} \text{ is connected}\}.$$

For the complex-valued polynomial $f_{a_3}(z) = z^2 + a_3$, the connected Julia set $F_{f_{a_3}}$ represents the set of all complex-valued parameters. Equivalently, the Mandelbrot set can be expressed as

$$M = \{a_3 \in \mathbb{C} : \{|f^k_{a_3}(\tau)|\}_{k=0}^{\infty} \nrightarrow \infty \text{ as } k \to \infty\},$$

where τ is a critical point of $f(z)$.

The definition of the Mandelbrot set provides a computational method for visualizing it. We consider a square in the Argand plane and select a mesh of equidistant points within this square as values of a_3. Using a computer, we determine whether the orbit of 0 for each a_3-value remains bounded or escapes to infinity. If the orbit of 0 diverges, we color the corresponding pixel white; otherwise, we shade it dark. The set of dark-colored points represents the Mandelbrot set. Some points near the boundary of the Mandelbrot set may exhibit orbits that take a large number of iterations before escaping. In such cases, we use a finite number of iterations to approximate whether a given a_3-value belongs to the Mandelbrot set.

For Julia sets, z represents a coordinate in the image and acts as a variable, while a_3 is a fixed complex constant. The value of a_3 remains unchanged during the visualization of a specific Julia set, but different values of a_3 yield different Julia set images.

Li et al. [33] used the Jungck-Mann iteration, incorporating s-convexity, to generate filled Julia sets and their boundaries (see also [34]). Nazeer et al. [30] applied Jungck-Mann and Jungck-Ishikawa fixed-point iterations, also with s-convexity, to generate both Julia and Mandelbrot fractals. If two maps $S, T : \mathbb{C} \to \mathbb{C}$ are given such that T has degree ≥ 2, S is one-to-one and we consider sequences generated by Jungck-type iterations, where the usual convex weights are replaced by s-weighted weights $(1-\alpha)^s$ and α^s with $s \in (0, 1]$, and then for an initial point $z_0 \in \mathbb{C}$, parameters $\alpha_1, \alpha_2, \alpha_3 \in (0, 1)$, and $k = 1, 2, \ldots$, the sequence $\{z_k\}$ follows one of the iterative schemes below:

(i) Jungck-Mann orbit equipped with s-convexity (JMOs) if

$$Sz_k : Sz_k = (1-\alpha_1)^s Sz_{k-1} + {\alpha_1}^s Tz_{k-1}.$$

This is a one-step feedback system defined by four parameters: (T, z_0, α_1, s).

(ii) Jungck-Ishikawa orbit equipped with s-convexity (JIOs) if

$$\begin{aligned} Sz_k : Sz_k &= (1-\alpha_1)^s Sz_{k-1} + {\alpha_1}^s Ty_{k-1}, \\ Sy_{k-1} &= (1-\alpha_2)^s Sz_{k-1} + {\alpha_2}^s Tz_{k-1}. \end{aligned}$$

This is a two-step feedback system defined by five parameters: $(T, z_0, \alpha_1, \alpha_2, s)$. Here, Ty_k denotes the image of the point y_k under T. The intermediate point y_k is the (pre-image) point satisfying $S(y_k) = (1-\alpha_2)^s S(z_k) + \alpha_2^s T(z_k)$. When S is invertible (or at least has a suitable right/left inverse on the relevant range) we recover $y_k = S^{-1}((1-\alpha_2)^s Sz_k + \alpha_2^s Tz_k)$. If S is not invertible, we work directly with the S-image sequences Sz_k and Sy_k (this is a standard Jungck approach).

(iii) Jungck-Noor orbit equipped with s-convexity (JNOs) [35] if

$$\begin{aligned} Sz_k : Sz_k &= (1-\alpha_1)^s Sz_{k-1} + \alpha_1^s Ty_{k-1}, \\ Sy_{k-1} &= (1-\alpha_2)^s Sz_{k-1} + \alpha_2^s Tu_{k-1}, \\ Su_{k-1} &= (1-\alpha_3)^s Sz_{k-1} + \alpha_3^s Tz_{k-1}. \end{aligned} \tag{2.1}$$

This is a three-step feedback system defined by six parameters: $(T, z_0, \alpha_1, \alpha_2, \alpha_3, s)$. Tu_k means T applied to the point u_k . The point u_k is determined (uniquely when S is invertible on the relevant set) by the first equation: $u_k = S^{-1}((1-\alpha_3)^s Sz_k + \alpha_3^s Tz_k)$. Similarly, y_k is determined from the second equation once u_k is known. If S is not invertible, one may equivalently treat the iteration on the S-image level (i.e., iterate the sequence Sz_k using the formulas above); this is the usual Jungck formalism.

Remark 2.1. The Jungck-Noor orbit with s-convexity diminishes to the Jungck-Ishikawa orbit [11] when $\alpha_3 = 0$, $s = 1$; Jungck-Maan Orbit [18] when $\alpha_2 = \alpha_3 = 0$, $s = 1$; Picard orbit [9] when $S(z) = z, \alpha_1 = 1, \alpha_2 = \alpha_3 = 0$, $s = 1$; Mann orbit [14] when $S(z) = z, \alpha_2 = \alpha_3 = 0$, $s = 1$; Ishikawa orbit [12] when $S(z) = z$, $\alpha_3 = 0$, $s = 1$; Jungck-Mann orbit with s-convexity when $\alpha_2 = \alpha_3 = 0$; and Jungck-Ishikawa orbit with s-convexity [30] when $\alpha_3 = 0$.

3. Escape time algorithm in Jungck-Noor orbit equipped with s-convexity

In this study, we employ the three-step Jungck–Noor fixed point iteration enhanced with s-convexity, refining the classical approach for n^{th} order complex polynomials. Given an initial approximation z_0 and a complex polynomial $f(z)$, the iteration generates each next point as a weighted combination of previous iterates and polynomial evaluations: $Sz_k = (1-\alpha_1)^s Sz_{k-1} + \alpha_1^s Tz_{k-1}, k \geq 1$, where S and T are operators defined by $f(z), \alpha_1$ is the iteration weight, and $s \in (0, 1]$ is the s-convexity parameter. This parameter regulates the contribution of the previous iterate relative to the current polynomial evaluation and intermediate Jungck step, introducing a nonlinear weighting effect. It directly influences convergence speed, escape dynamics, fractal geometry, density, brightness, and boundary detail. Smaller values of s slow convergence but produce finer structural details, while larger values accelerate convergence with simpler visual patterns. Thus, s-convexity serves as a tuning mechanism, enabling systematic generation of fractals ranging from compact, high-brightness patterns ($s \leq 0.5$) to intricate, highly detailed structures ($s \to 1$).

Equipping the iteration with s-convexity modifies the weighting of successive steps. Instead of linear convex combinations, the parameter s introduces a nonlinear weighting effect. This adjustment directly influences the convergence rate (escape or boundedness), the geometry of the fractal structure (shape, symmetry, and density), and the computational complexity (time per iteration). In particular, smaller values of s tend to slow convergence but produce finer structural details, whereas larger values accelerate convergence at the expense of reduced complexity in visual patterns. Thus, s-convexity serves as a tuning mechanism, allowing the iterative scheme to balance computational efficiency with the richness of fractal features. We prove the escape criteria and determine threshold escape radii to explore some mutants of celebrated fractals via Jungck-Noor iteration equipped with s-convexity for a polynomial $f(z) = z^n + a_1z^2 - a_2z + a_3$, where $n \geq 3$ and a_1, a_2, and a_3 are complex numbers.

Theorem 3.1. Suppose $f(z) = z^n + a_1z^2 - a_2z + a_3$ is a polynomial of degree n, where a_1, a_2, a_3 are complex parameters. Let $|z_0| \geq |a_3| > \left(\frac{2(1+|a_2|)}{s\alpha_i|(a_3^{n-2}+a_1)|}\right)$ for $i = 1, 2, 3$ and real constants $\alpha_1, \alpha_2, \alpha_3, s$ in the interval $(0, 1)$. Let $\{z_i\}_{i\in\mathbb{N}}$ be the Jungck Noor-iteration equipped with s-convexity as defined in (iii), and let $Sz = a_2z$ and $Tz = z^n + a_1z^2 + a_3$. Then, as k tends to ∞, the orbit of z_0 tends to ∞, i.e., $|z_k| \to \infty$ as $k \to \infty$.

Proof. If $k = 1$

$$Su_{k-1} = (1-\alpha_3)^sSz_{k-1} + \alpha_3^sTz_{k-1}$$

implies

$$\begin{aligned}
|Su_0| &= |(1-\alpha_3)^sSz_0 + \alpha_3^sTz_0| \\
&= |(1-\alpha_3)^sa_2z_0 + \alpha_3^s(z_0^n + a_1z_0^2 + a_3)| \text{ using } S(z) = a_2z \text{ and } T(z) = z^n + a_1z^2 + a_3 \\
&\geq |(1-\alpha_3)^sa_2z_0 + s\alpha_3(z_0^n + a_1z_0^2 + a_3)|, \quad \alpha_3^s \geq s\alpha_3, \text{ since } \alpha_3, s \in (0,1] \\
&\geq |s\alpha_3(z_0^n + a_1z_0^2) + (1-\alpha_3)^sa_2z_0| - s\alpha_3|a_3| \\
&\geq |s\alpha_3(z_0^n + a_1z_0^2) + (1-\alpha_3)^sa_2z_0| - s\alpha_3|z_0|, \quad |z_0| \geq |a_3| \\
&\geq |s\alpha_3(z_0^n + a_1z_0^2)| - |(1-\alpha_3)^sa_2z_0| - s\alpha_3|z_0|.
\end{aligned}$$

Expanding $(1-\alpha_3)^s$ up to linear terms of α_3, we get

$$\begin{aligned}
|Su_0| &\geq |s\alpha_3(z_0^n + a_1z_0^2)| - |(1-s\alpha_3)a_2z_0| - s\alpha_3|z_0|, \text{ (For } s \in (0,1] \text{ and } \alpha_3 \in (0,1), (1-\alpha_3)^s \leq 1 - s\alpha_3) \\
&= s\alpha_3|(z_0^n + a_1z_0^2)| - |a_2z_0| + |s\alpha_3a_2z_0| - s\alpha_3|z_0| \\
&\geq s\alpha_3|(z_0^n + a_1z_0^2)| - |a_2z_0| - s\alpha_3|z_0|, \quad |s\alpha_3a_2z_0| \geq 0,
\end{aligned}$$

which gives us

$$\begin{aligned}
|a_2u_0| &\geq s\alpha_3|(z_0^n + a_1z_0^2)| - |z_0| - |a_2z_0|, \quad s\alpha_3 < 1 \\
&= s\alpha_3|z_0^2||(z_0^{n-2} + a_1)| - |z_0|(1 + |a_2|) \\
&\geq |z_0|\Big(s\alpha_3|z_0||(a_3^{n-2} + a_1)| - (1 + |a_2|)\Big), \text{ using the assumption that } |z_0| \geq |a_3| \\
&= |z_0|(1 + |a_2|)\Big(\frac{s\alpha_3|z_0||(a_3^{n-2} + a_1)|}{(1 + |a_2|)} - 1\Big).
\end{aligned}$$

Thus

$$|u_0| \geq \frac{|a_2 u_0|}{(1+|a_2|)} = |z_0|\Big(\frac{s\alpha_3|z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1\Big).$$

Now

$$|z_0| \geq |a_3| > \tfrac{2(1+|a_2|)}{s\alpha_3|(a_3^{n-2}+a_1)|} \text{ implies that } \tfrac{s\alpha_3|z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1 > 1.$$

Hence $|u_0| > |z_0|$.

Now,

$$Sy_{k-1} = (1-\alpha_2)^s Sz_{k-1} + \alpha_2^s Tu_{k-1},$$

which implies

$$\begin{aligned}
|Sy_0| &= |(1-\alpha_2)^s Sz_0 + \alpha_2^s Tu| \\
&= |(1-\alpha_2)^s a_2 z_0 + \alpha_2^s(u_0^n + a_1 u_0^2 + a_3)| \\
&\geq |(1-\alpha_2)^s a_2 z_0 + s\alpha_2(u_0^n + a_1 u_0^2 + a_3)|, \quad \alpha_2^s \geq s\alpha_2,\ \alpha_2, s \in (0,1] \\
&\geq |s\alpha_2(u_0^n + a_1 u_0^2) + (1-\alpha_2)^s a_2 z_0| - s\alpha_2|a_3| \\
&\geq |s\alpha_2(u_0^n + a_1 u_0^2) + (1-\alpha_2)^s a_2 z_0| - s\alpha_2|z_0|, \quad |z_0| \geq |a_3| \\
&\geq |s\alpha_2(u_0^n + a_1 u_0^2)| - |(1-\alpha_2)^s a_2 z_0| - s\alpha_2|z_0|.
\end{aligned}$$

Expanding $(1-\alpha_2)^s$ up to linear terms of α_2, we have

$$\begin{aligned}
|Sy_0| &\geq |s\alpha_2(u_0^n + a_1 u_0^2)| - |(1-s\alpha_2)a_2 z_0| - s\alpha_2|z_0| \ \big(\text{For } s \in (0,1] \text{ and } \alpha_2 \in (0,1),\ (1-\alpha_2)^s \leq 1 - s\alpha_2\big) \\
&= s\alpha_2|(u_0^n + a_1 u_0^2)| - |a_2 z_0| + |s\alpha_2 a_2 z_0| - s\alpha_2|z_0| \\
&\geq s\alpha_2|(u_0^n + a_1 u_0^2)| - |a_2 z_0| - s\alpha_2|z_0|, \quad |s\alpha_2 a_2 z_0| \geq 0
\end{aligned}$$

which gives us

$$\begin{aligned}
|a_2 y_0| &\geq s\alpha_2|(u_0^n + a_1 u_0^2)| - |z_0| - |a_2 z_0|, \quad s\alpha_2 < 1 \\
&= s\alpha_2|u_0^2||(u_0^{n-2} + a_1)| - |z_0|(1+|a_2|) \\
&> s\alpha_2|z_0^2||(z_0^{n-2} + a_1)| - |z_0|(1+|a_2|), \ |u_0| > |z_0| \\
&\geq |z_0|\Big(s\alpha_2|z_0||(a_3^{n-2} + a_1)| - (1+|a_2|)\Big) \\
&= |z_0|(1+|a_2|)\Big(\frac{s\alpha_2|z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1\Big).
\end{aligned}$$

Thus

$$|y_0| \geq \frac{|a_2 y_0|}{(1+|a_2|)} = |z_0|\Big(\frac{s\alpha_2|z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1\Big).$$

Now

$$|z_0| \geq |a_3| > \tfrac{2(1+|a_2|)}{s\alpha_2|(a_3^{n-2}+a_1)|} \text{ implies that } \tfrac{s\alpha_2|z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1 > 1.$$

Hence $|y_0| > |z_0|$.

Again,

$$Sz_k = (1-\alpha_1)^s Sz_{k-1} + \alpha_1^s Ty_{k-1},$$

which implies

$$\begin{aligned}
|Sz_1| &= |(1-\alpha_1)^s Sz_0 + \alpha_1^s Ty_0| \\
&= |(1-\alpha_1)^s a_2 z_0 + \alpha_1^s (y_0^n + a_1 y_0^2 + a_3)| \\
&\geq |(1-\alpha_1)^s a_2 z_0 + s\alpha_1 (y_0^n + a_1 y_0^2 + a_3)|, \quad \alpha_1^s \geq s\alpha_1, \text{ since } \alpha_1, s \in (0,1] \\
&\geq |s\alpha_1(y_0^n + a_1 y_0^2) + (1-\alpha_1)^s a_2 z_0| - s\alpha_1 |a_3| \\
&\geq |s\alpha_1(y_0^n + a_1 y_0^2) + (1-\alpha_1)^s a_2 z_0| - s\alpha_1 |z_0|, \quad |z_0| \geq |a_3| \\
&\geq |s\alpha_1(y_0^n + a_1 y_0^2)| - |(1-\alpha_1)^s a_2 z_0| - s\alpha_1 |z_0|.
\end{aligned}$$

Expanding $(1-\alpha_1)^s$ up to linear terms of α_1, we get

$$\begin{aligned}
|Sz_1| &\geq |s\alpha_1(y_0^n + a_1 y_0^2)| - |(1-s\alpha_1) a_2 z_0| - s\alpha_1 |z_0| \ \ \big(\text{For } s \in (0,1] \text{ and } \alpha_1 \in (0,1),\ (1-\alpha_1)^s \leq 1 - s\alpha_1\big) \\
&= s\alpha_1 |(y_0^n + a_1 y_0^2)| - |a_2 z_0| + |s\alpha_1 a_2 z_0| - s\alpha_1 |z_0| \\
&\geq s\alpha_1 |(y_0^n + a_1 y_0^2)| - |a_2 z_0| - s\alpha_1 |z_0|, \quad |s\alpha_1 a_2 z_0| \geq 0
\end{aligned}$$

which gives us

$$\begin{aligned}
|a_2 z_1| &\geq s\alpha_1 |(y_0^n + a_1 y_0^2)| - |z_0| - |a_2 z_0|, \quad s\alpha_1 < 1 \\
&= s\alpha_1 |y_0^2||(y_0^{n-2} + a_1)| - |z_0|(1 + |a_2|) \\
&> s\alpha_1 |z_0^2||(z_0^{n-2} + a_1)| - |z_0|(1 + |a_2|), \ |y_0| > |z_0| \\
&\geq |z_0|\Big(s\alpha_1 |z_0||(a_3^{n-2} + a_1)| - (1 + |a_2|)\Big) \\
&= |z_0|(1 + |a_2|)\Big(\frac{s\alpha_1 |z_0||(a_3^{n-2} + a_1)|}{(1 + |a_2|)} - 1\Big).
\end{aligned}$$

Thus

$$|z_1| \geq \frac{|a_2 z_1|}{(1 + |a_2|)} = |z_0|\Big(\frac{s\alpha_1 |z_0||(a_3^{n-2} + a_1)|}{(1 + |a_2|)} - 1\Big).$$

Since $|z_0| \geq |a_3| > \frac{2(1+|a_2|)}{s\alpha_1 |(a_3^{n-2}+a_1)|}$, then $\frac{s\alpha_1 |z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1 > 1$. Hence, there exists a $\lambda > 0$ so that $\frac{s\alpha_1 |z_0||(a_3^{n-2}+a_1)|}{(1+|a_2|)} - 1 > 1 + \lambda > 1$. As a result,

$$|z_1| \geq (1 + \lambda)|z_0|.$$

So, $|z_1| > |a_3|$ ($|z_1| > (1 + \lambda)|a_3| > |a_3|$, since $|z_0| > |a_3|$).

Hence, $|z_1| > |a_3| > \frac{2(1+|a_2|)}{(s\alpha_1 |a_3^{(}n-2)+a_1|)}$. Also $|z_1| \geq \frac{2(1+|a_2|)}{(s\alpha_2 |a_3^{(}n-2)+a_1|)}$ and $|z_1| \geq \frac{2(1+|a_2|)}{(s\alpha_3 |a_3^{(}n-2)+a_1|)}$. Hence, we can apply the above arguments for $k = 2$, and obtain

$$|z_2| \geq (1 + \lambda)|z_1| \geq (1 + \lambda)^2 |z_0|.$$

In the same way, for all k , we obtain $|z_k| > (1 + \lambda)^k |z_0|$ (following the above procedure repeatedly).

Hence, $|z_k| \to \infty$ as $k \to \infty$. □

Corollary 3.1. Let $|z_m| > \max\left\{|a_3|, \frac{2(1+|a_2|)}{s\alpha_1|(a_3^{n-2}+a_1)|}, \frac{2(1+|a_2|)}{s\alpha_2|(a_3^{n-2}+a_1)|}, \frac{2(1+|a_2|)}{s\alpha_3|(a_3^{n-2}+a_1)|}\right\}$, where $m \geq 0$. Then $|z_{m+k}| > (1+\lambda)^k|z_k|$ and the Jungck-Noor orbit equipped with s-convexity for the sequence $\{z_k\}$ tends ∞ as $k \to \infty$ for an initial point $z_0 \in \mathbb{C}$.

Remark 3.1. (i) Jolaoso et al. [29] assumed $a_1 \in \{0, 1\}$ while considering a_2 and a_3 to be complex numbers. However, in our approach, we treat a_1, a_2, and a_3 as complex numbers by replacing a_2 with $-a_2$. This choice of parameters has not been previously studied in the context of the Jungck-Noor orbit equipped with s-convexity (JNOs).

(ii) Based on Remark 2.1, our conclusions enable the derivation of the escape time algorithm for the iterative methods such as the Jungck-Mann fixed point iteration [34] and the Jungck-Ishikawa iteration [36], both endowed with s-convexity, for polynomial $f(z) = z^n + a_1z^2 - a_2z + a_3$, where $n \geq 3$, and $a_1, a_2, a_3 \in \mathbb{C}$. For $a_1 = 0$, the polynomial reduces to $f(z) = z^n - a_2z + a_3$, which was previously studied by Nazeer et al. [30]. However, the inclusion of the term a_1z^2, where a_1 is any complex number, adds a new fascinating dimension to our conclusions, making them particularly insightful within the three-step Jungck-Noor iterative procedure equipped with s-convexity.

(iii) Nazeer et al. [30] proved the escape criterion using Jungck-Mann and Jungck-Ishikawa iterations equipped with s-convexity for the function $f(z) = z^n - a_2z + a_3$, but they did not explore the corresponding fractals. The authors claimed to have applied the principle of mathematical induction to prove Theorem 4.9 (respectively, Theorem 3.9) [30]. First, they used Theorems 4.1 and 4.5 (respectively, Theorems 3.1 and 3.5) to prove the initial step of the theorem. However, their inductive assumption was not applied in their proof for $n+1$, meaning that they did not fully utilize the principle of mathematical induction.

(iv) Other results of Nazeer et al. [30] contain errors due to their use of binomial expansion truncated at linear terms, which is not valid for the assumed parameters. One may easily check that neither $(1-\alpha)^s \geq 1 - s\alpha$ nor $(1-(1-\alpha))^s \geq 1 - s(1-\alpha)$ holds in $(0, 1]$.

(v) Mistakes of the same type can also be observed in papers where the escape criterion is derived for complex polynomials using different iterations equipped with s-convexity, such as Cho et al. [16], Gdawiec et al. [37], Kang et al. [35], Kumari et al. [21], Kwun et al. [13], Li et al. [33], Nazeer et al. [30], Zhang et al. [38], and others. As a result, we have developed a method for obtaining the escape criterion for more general polynomials using a three-step Jungck-Noor iterative procedure equipped with s-convexity (see Tomar et al. [34] for Jungck-Mann iteration equipped with s-convexity and Antal et al. [36] for Jungck-Ishikawa iteration equipped with s-convexity).

(vi) Our theorems and corollaries demonstrate the significance of s-convexity in the Jungck-Noor iteration for exploring the geometry of complex fractals. Furthermore, if we substitute the convexity parameter $s = 1$, we get the comparable results present in the literature.

(vii) Corollary 3.1 provides an algorithm for generating fractals of T_{a_3}. If $|z| \leq |a_3|$, we get JNOs of z. If $|z_k|$ lies outside the circle of radius $\max\left\{|a_3|, \frac{2(1+|a_2|)}{s\alpha_1|(a_3^{n-2}+a_1)|}, \frac{2(1+|a_2|)}{s\alpha_2|(a_3^{n-2}+a_1)|}, \frac{2(1+|a_2|)}{s\alpha_3|(a_3^{n-2}+a_1)|}\right\}$ for $n \geq 3$, then the orbit JNOs escapes. This implies that z does not lie in the interior of the Mandelbrot or Julia fractals. On the other hand, if $|z_k|$ does not go beyond this bound, then by the definitions of Julia and Mandelbrot fractals, we apply these techniques to explore Mandelbrot and Julia sets in Section 5.

(viii) It is worth mentioning here that Banach [39] used Picard iteration [9] to approximate a fixed point of the involved contraction mapping, which does not necessarily converge for slightly weaker mapping. As a result, one-step Mann iteration [14], two-step Ishikawa iteration [12], and three-step Noor iteration [15], among others, have been initiated by distinct researchers to resolve this issue for various contractions.

4. Julia and Mandelbrot fractals

To generate fractals using the escape time algorithm for the higher-order complex polynomial via the Jungck-Noor iteration scheme equipped with s-convexity, we first develop algorithms (see Algorithms 1 and 2). We then implement these algorithms using MATLAB 8.5.0 (R2015a) software, and apply a color map (see Figure 1) to visualize the results.

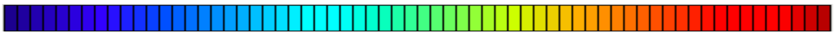


Figure 1. Colormap.

We set the maximum number of iterations to $K = 50$. The resolution of the obtained fractals is adjusted to 400×400 pixels. All simulations are executed on a computer with the following specifications: Intel(R) Core(TM) i5-8250U CPU (@1.60GHz) processor, 8GB RAM, and Microsoft Windows 10(64-bit).

Algorithm 1 : Generation of Julia fractals

Input: $T(z) = z^n + a_1 z^2 + a_3$, $Sz = a_2 z$, where $a_1, a_2, a_3 \in \mathbb{C}$, $a_2 \neq 0$; $\alpha_1, \alpha_2, \alpha_3, s \in (0, 1]$-parameters of the Jungck-Noor fixed point procedure equipped with s-convexity; K-a maximum number of iterations; $A \subset \mathbb{C}$-area; *colormap*$[0..C-1]$-color map with C colors.

Output: Julia set for area A.

```
for $z_0 \in A$ do
    $R_1 = \left(\dfrac{2(1+|a_2|)}{s\alpha_1|a_3^{n-2}+a_1|}\right)$
    $R_2 = \left(\dfrac{2(1+|a_2|)}{s\alpha_2|a_3^{n-2}+a_1|}\right)$
    $R_3 = \left(\dfrac{2(1+|a_2|)}{s\alpha_3|a_3^{n-2}+a_1|}\right)$
    $R = max\,[|a_3|, R_1, R_2, R_3]$
    $n = 0$
    while $n \leq K$ do
        $u_n = \dfrac{(1-\alpha_1)^s S z_n + \alpha_1^s T(z_n)}{a_2}$
        $y_n = \dfrac{(1-\alpha_2)^s S z_n + \alpha_2^s T(u_n)}{a_2}$
        $z_{n+1} = \dfrac{(1-\alpha_3)^s S z_n + \alpha_1^s T(y_n)}{a_2}$
        if $|z_{n+1}| > R$ then
            break
        end if
        $n = n + 1$
    end while
    $i = \lfloor (C-1)\frac{n}{K} \rfloor$
    color $z_0$ with $colormap[i]$
end for
```

Algorithm 2 : Generation of Mandelbrot fractals

Input: $T(z) = z^n + a_1 z^2 + a_3$, $Sz = a_2 z$, where $a_1, a_2, a_3 \in \mathbb{C}$, $a_2 \neq 0$; $\alpha_1, \alpha_2, \alpha_3, s \in (0, 1]$-parameters of the Jungck-Noor fixed point procedure equipped with s-convexity; K-a maximum number of iterations; $A \subset \mathbb{C}$-area; $colormap[0..C-1]$-color map with C colors.
Output: Mandelbrot set for area A.

```
for $a_3 \in A$ do
    $R_1 = \left( \frac{2(1+|a_2|)}{t\alpha_1 |a_3^{n-2} + a_1|} \right)$
    $R_2 = \left( \frac{2(1+|a_2|)}{t\alpha_2 |a_3^{n-2} + a_1|} \right)$
    $R_3 = \left( \frac{2(1+|a_2|)}{t\alpha_3 |a_3^{n-2} + a_1|} \right)$
    $R = max\,[|a_3|, R_1, R_2, R_3]$
    $n = 0$
    $z_0 = 0$
    while $n \leq K$ do
        $u_n = \frac{(1-\alpha_1)^s S z_n + \alpha_1^s T(z_n)}{a_2}$
        $y_n = \frac{(1-\alpha_2)^s S z_n + \alpha_2^s T(u_n)}{a_2}$
        $z_{n+1} = \frac{(1-\alpha_3)^s S z_n + \alpha_1^s T(y_n)}{a_2}$
        if $|z_{n+1}| > R$ then
            break
        end if
        $n = n + 1$
    end while
    $i = \lfloor (C-1)\frac{n}{K} \rfloor$
    color $a_3$ with $colormap[i]$
end for
```

When presenting Julia and Mandelbrot sets, we make the parameter choices deliberately to highlight specific dynamical behaviors rather than arbitrarily. We vary parameters such as polynomial degree, constant coefficients, iteration weights, and convexity systematically to capture effects like symmetry, boundary transitions, fractal area shrinkage, and computational efficiency. A summary of representative parameter choices and their rationale is given in Table 1.

Table 1. Parameter selection and rationale for fractal visualization.

Fractal type	Parameter(s)	Value(s) chosen	Rationale
Julia Sets	Degree n	3–20, irrational e	To study area shrinkage, circularity, and petal structures
Julia Sets	Constants a_1, a_2, a_3	Positive/negative, conjugates, 1–30	To capture symmetry, reflections, and partition effects
Julia Sets	Iteration parameters $\alpha_1, \alpha_2, \alpha_3$	$0.9 \to 0.01$, 0.2–0.05	To test sensitivity, shape transitions, and efficiency
Julia & Mandelbrot Sets	Convexity s	0.1–1	To explore brightness, density, and computational trade-offs
Mandelbrot Sets	Constants a_1, a_2	Varying ± values	To demonstrate symmetry, rotational effects, and shrinkage
Both	Conjugation	$a_1 \mapsto \bar{a}_1$	To show mirror symmetry
Both	Iteration Depth, Escape Radius	200–500 iters, radius = 2	To balance fine details with computational cost

These choices are tailored to the Jungck–Noor iteration framework to study complex dynamics and fractal visualization.

4.1. *Julia fractals*

In Figure 2(a), a beautiful fractal is visible for $n = 3$. It is akin to the double urn, having a decorative boundary with limb-like structures, and is symmetrical about the X-axis. Figure 2 represents fractals for the same parameter values, differing only in the degree of the polynomial n. We observe that the area occupied by the fractal decreases with the increase in the value of n, and the time taken to generate each Julia set decreases from 2.101533 to 1.468428 seconds.

The input parameters (Figure 2) are $a_1 = 0.8 - 0.8i, a_2 = 10, a_3 = -2.275 - 0.99i, \alpha_1 = 0.022165, \alpha_2 = 0.022679, \alpha_3 = 0.022165$, and $s = 0.5$.

30
25
20
15
10
5
0
30
25
20
15
10
5
0

(a) $n = 3, A = [-12, 12]^2$

(b) $n = 4, A = [-5.5, 5.5]^2$

Figure 2. Effect of the degree of the complex-valued polynomial n on the Julia sets.

Figure 3 represents a fractal for $n = 20$ with the parameter values $a_1 = 0, a_2 = 200, a_3 = -2.275 - 0.99i, \alpha_1 = 0.022165, \alpha_2 = 0.022679, \alpha_3 = 0.022165$, and $s = 0.7$, within the domain $A = [-2, 2]^2$. We observe that as parameter n increases, the fractal shape becomes more circular, which is akin to a circular saw. The time taken to generate this fractal is 1.939024 seconds.

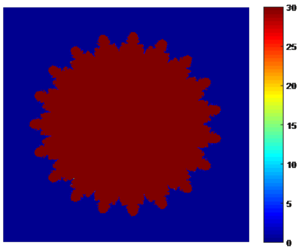


Figure 3. Julia set for a higher-order polynomial for $n = 20$.

Remark 4.1. Figure 4 represents a fractal generated with the parameters $a_1 = 1, a_2 = 40, a_3 = -2.275-0.99i, \alpha_1 = 0.022165$, $\alpha_2 = 0.022679, \alpha_3 = 0.022165$, and $s = 0.51$. This is a special kind of Julia fractal, generated when n is an irrational number that occupies a significantly huge area ($[-38, 38]^2$), compared to all other explored fractals. Almost all the fractals lie in the plane having an area less than or equal to $[23 \times 23]^2$. As a result, we may point out that not only do we get fractals for a complex-valued polynomial of a higher degree, but also for the complex-valued functions of the type whose degree is an irrational number. The time taken to generate this fractal is 0.637488 seconds.

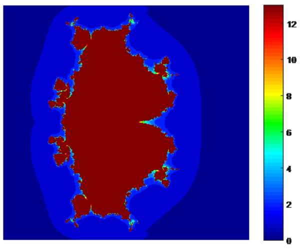
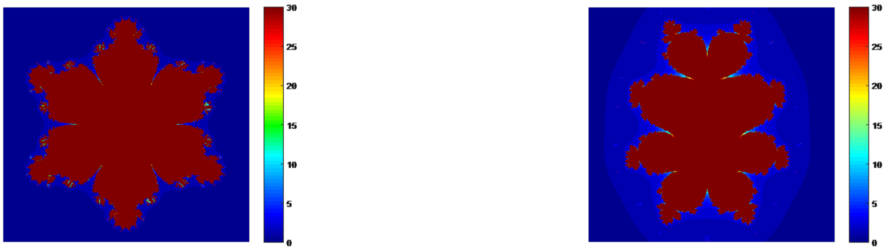


Figure 4. Julia set for a function whose degree is an irrational number ($e \approx 2.718$).

The input parameters for Figure 5 are as follows:

(1) $a_1 = 0, a_2 = 150, a_3 = -2.275 - 0.99i, \alpha_1 = 0.022165, \alpha_2 = 0.022679, \alpha_3 = 0.022165, n = 7, s = 0.351$.
(2) $a_1 = 1, a_2 = 2, a_3 = -0.45 - 45i, \alpha_1 = 0.001498, \alpha_2 = 0.0798, \alpha_3 = 0.001498, n = 3, s = 0.95$.

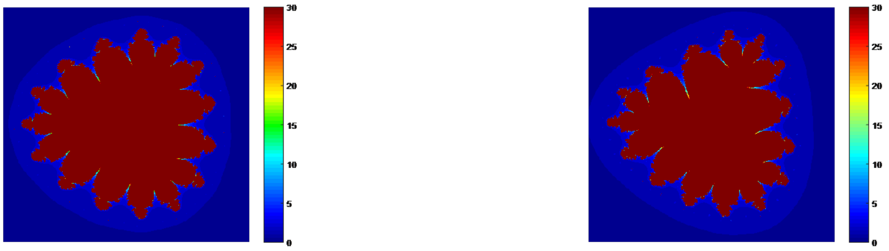
Clearly, the septic Julia set generated in 1.717499 seconds exhibits symmetry along both axes, while the cubic Julia set generated in 1.235802 seconds is symmetrical only about the Y-axis.



(a) $n = 7, s = 0.351, A = [-3.5, 3.5]^2$ **(b)** $n = 3, s = 0.99, A = [-11, 11]^2$

Figure 5. Some random Julia sets.

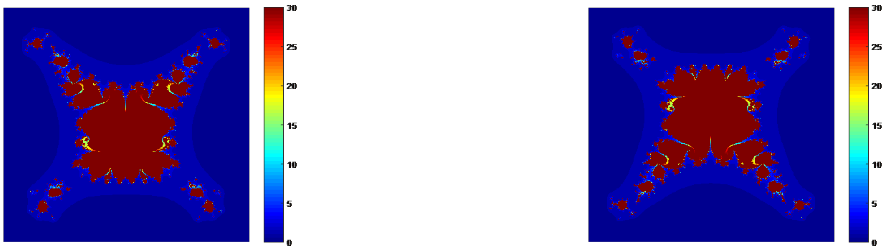
The parameters used in Figure 6 are $a_1 = 1, a_2 = 2, a_3 = 0.0045 - 70i, \alpha_1 = 0.00001, \alpha_2 = 0.0098, \alpha_3 = 0.00001$, and $n = 4$. It is interesting to note that in Figures 6(a) and 6(b), as the value of s decreases from 0.99 to 0.7, the area in which fractals lie decreases from $[-11, 11]^2$ to $[-7, 7]^2$, while the time taken to generate them increases from 1.237418 and 1.390322 seconds.



(a) $s = 0.99, A = [-11, 11]^2$ **(b)** $s = 0.7, A = [-7, 7]^2$

Figure 6. Effect of convexity parameter s on the quartic Julia sets.

By taking the conjugate of the complex parameter a_1, we obtain a mirror image about the initial line (see Figure 7). The time taken to generate these quintic Julia sets is 1.493986 and 1.532822 seconds. The parameters used in Figure 7 are as follows: $a_2 = 4, a_3 = 0.7, \alpha_1 = 0.00001, \alpha_2 = 0.0098, \alpha_3 = 0.00001, n = 5$, and $s = 0.5$.

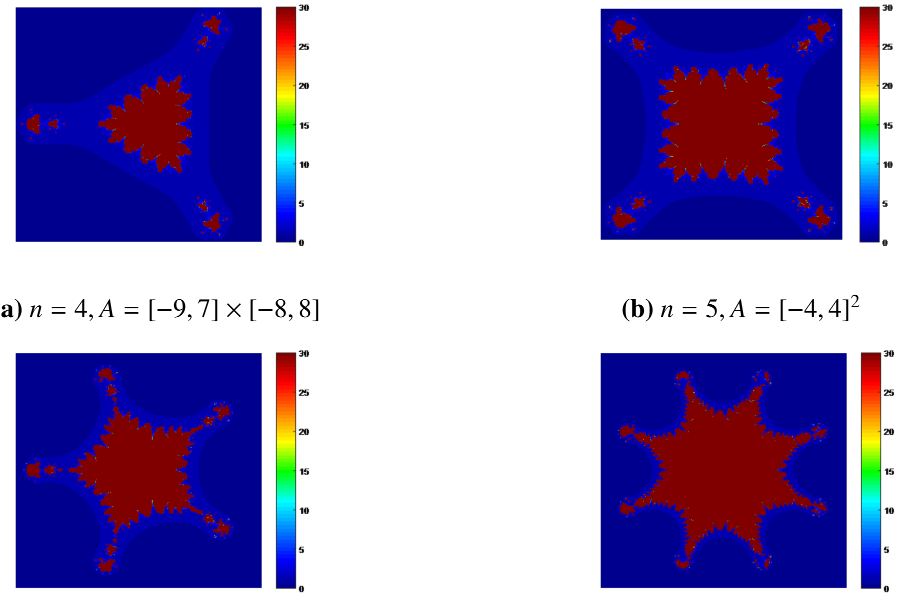


(a) $a_1 = 19i, A = [-6, 6]^2$ **(b)** $a_1 = -19i, A = [-6, 6]^2$

Figure 7. Effect of the conjugate of parameter a_1 on the quintic Julia sets.

Figure 8 represents Julia fractals of the polynomial for real parameter values: $a_1 = 1$, $a_2 = 1$, $a_3 =$

-1, $s = 0.95$, $\alpha_1 = 0.001498$, $\alpha_2 = 0.0798$, $\alpha_3 = 0.001498$ for $n = 4$ to 6 and $n = 9$, respectively. These fractals are very interesting as they appear like Kachhi thread works (Kutch embroidery), a traditional art form famous in Gujarat (India) and in other symmetries. For $n = 4$, the amazing triangular form emerges. Similarly, for $n = 5$, the pattern takes on a square-like structure, following a general trend where the fractal shape aligns with one less than the polynomial's degree.



(a) $n = 4, A = [-9, 7] \times [-8, 8]$

(b) $n = 5, A = [-4, 4]^2$

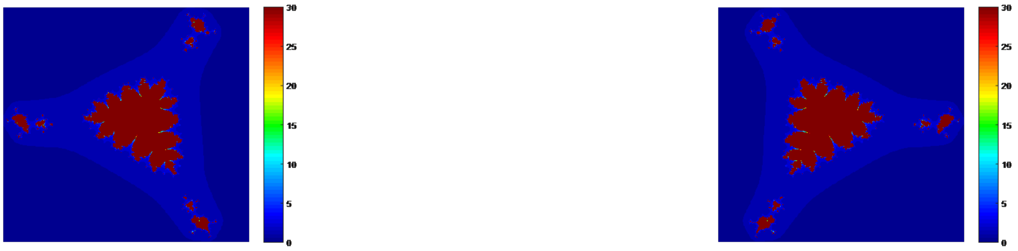
(c) $n = 6, A = [-4, 4]^2$

(d) $n = 9, A = [-2.5, 2.5]^2$

Figure 8. Effect of the degree of the polynomial n on the Julia sets.

Further, we notice that the area acquired by the Julia set and the time taken to generate it decrease (2.250810 to 1.390145 seconds) with the increase in the degree of the polynomial n.

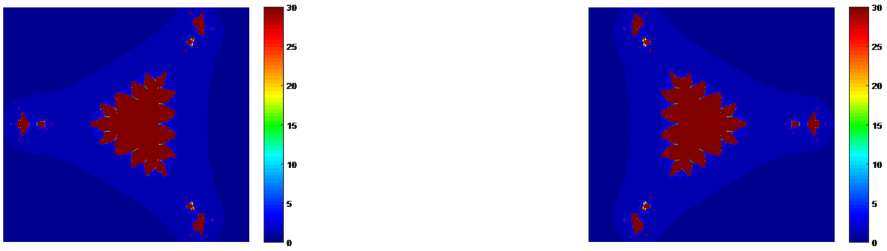
From Figures 9 and 10, it can be seen that as the parameters a_1 and a_2 change their sign, the size of the fractal remains unchanged. However, this results in mirror images in the fractal pattern. The time taken to generate these images is 1.281162 to 1.306099 seconds. The parameters used in Figures 9 and 10 are as follows: $a_3 = 0.7, \alpha_1 = 0.00001, \alpha_2 = 0.0098, \alpha_3 = 0.00001, n = 4$, and $s = 0.55$.



(a) $a_1 = -9i, a_2 = 2.9, A = [-13.5, 10.5] \times [-12, 12]$

(b) $a_1 = 9i, a_2 = -2.9, A = [-10.5, 13.5] \times [-12, 12]$

Figure 9. Effect of the conjugate of the parameter a_1 on the quartic Julia sets.

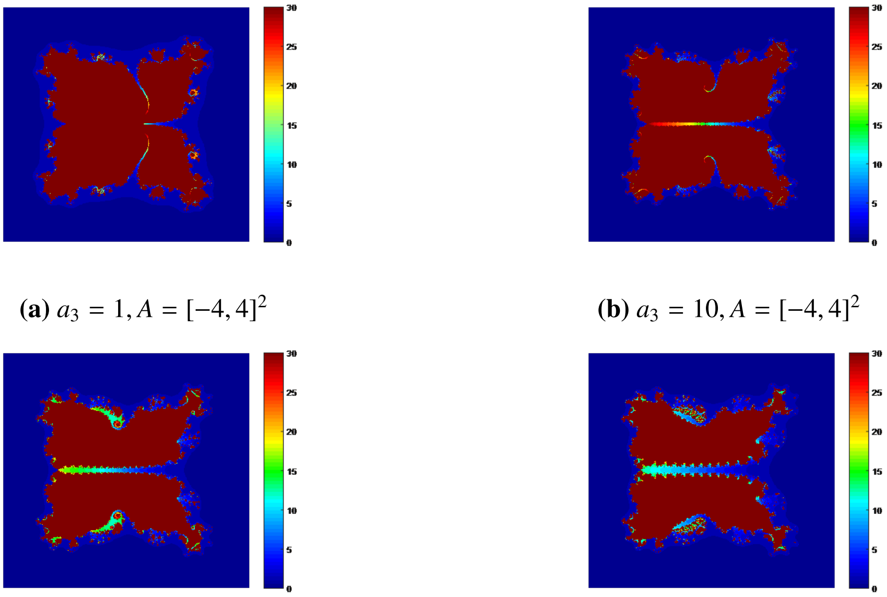


(a) $a_1 = 9, a_2 = 2.9, A = [-13.5, 10.5] \times [-12, 12]$ **(b)** $a_1 = 9, a_2 = -2.9, A = [-10.5, 13.5] \times [-12, 12]$

Figure 10. Effect of the change in sign of the parameter a_2 on the quartic Julia sets.

The parameters used in Figures 11(a) and 11(d) are as follows: $a_1 = 6, a_2 = 20, \alpha_1 = 0.01565, \alpha_2 = 0.0679, \alpha_3 = 0.0165, n = 5$, and $s = 0.41$. We observe that as the value of the constant parameter a_3 changes from 1 to 30, a beautiful aura appears between the quintic Julia sets, which divides them into two parts. The gap between these sections widens with the increase in the constant term a_3, while the time taken to generate these decreases from 1.548287 to 1.480358 seconds. However, the area remains the same.

The parameters used in Figure 12 are as follows: $a_1 = 6, a_2 = 20, a_3 = 10, \alpha_1 = 0.1565, \alpha_3 = 0.0165, n = 5$, and $s = 0.41$. A significant change in the shape of the quintic Julia set is observed as the value of α_2 used in the Noor-iteration decreases from 0.9 to 0.01. Noticeably, the shape similar to a kite turns into a beautiful flower in the same area. Also, the time taken to generate these increases from 1.462533 to 1.476392 seconds.



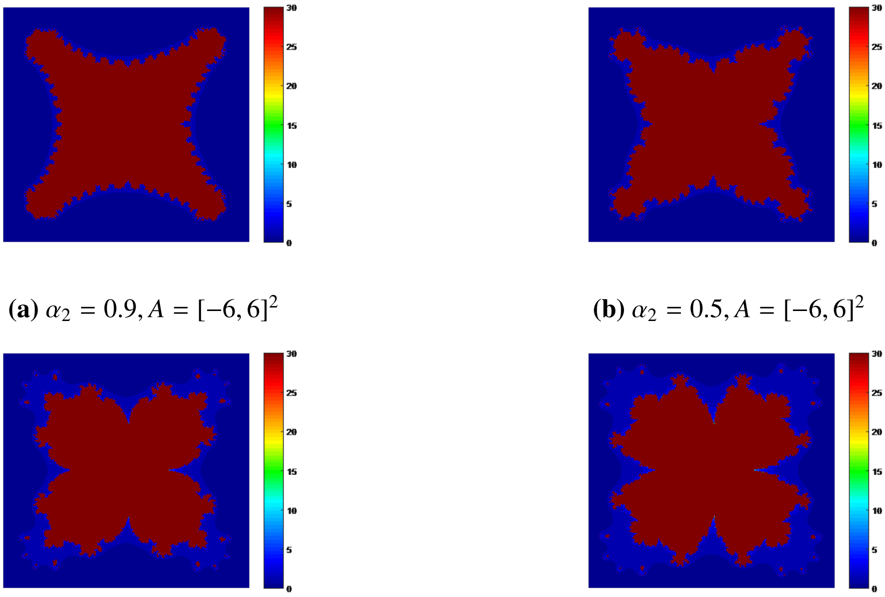
(a) $a_3 = 1, A = [-4, 4]^2$

(b) $a_3 = 10, A = [-4, 4]^2$

(c) $a_3 = 20, A = [-4, 4]^2$

(d) $a_3 = 30, A = [-4, 4]^2$

Figure 11. Effect of the parameter a_3 on the quintic Julia sets.



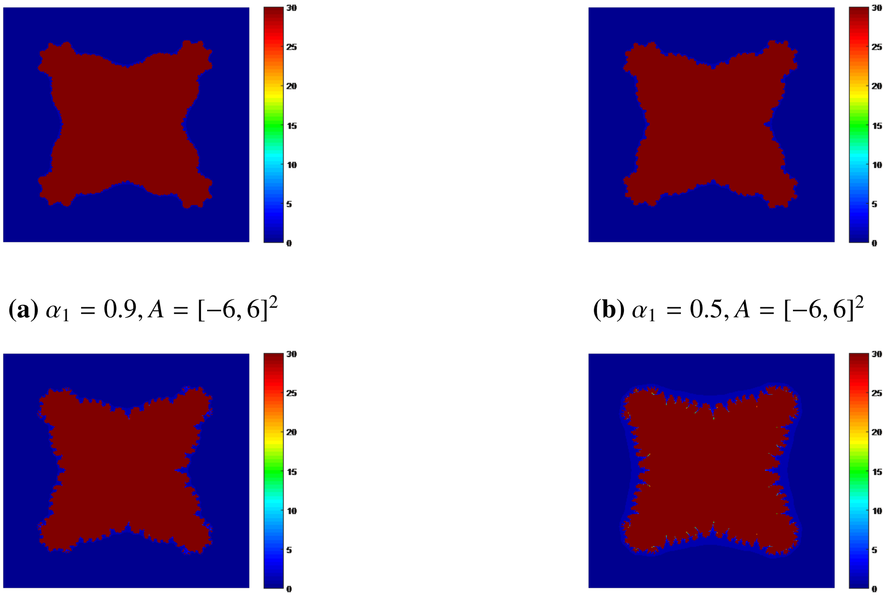
(a) $\alpha_2 = 0.9, A = [-6, 6]^2$

(b) $\alpha_2 = 0.5, A = [-6, 6]^2$

(c) $\alpha_2 = 0.1, A = [-6, 6]^2$

(d) $\alpha_2 = 0.01, A = [-6, 6]^2$

Figure 12. Effect of parameter α_2 on the quintic Julia sets.



(a) $\alpha_1 = 0.9, A = [-6, 6]^2$

(b) $\alpha_1 = 0.5, A = [-6, 6]^2$

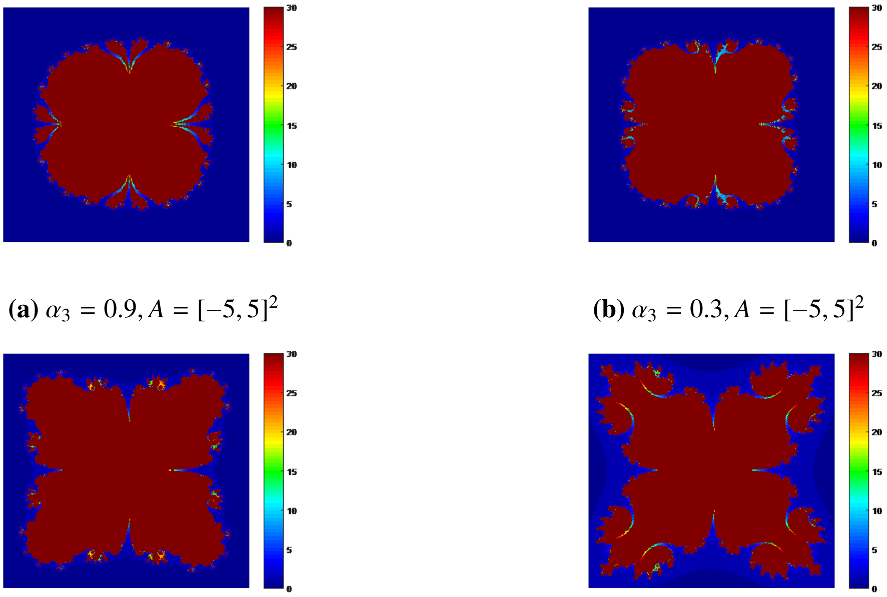
(c) $\alpha_1 = 0.1, A = [-6, 6]^2$

(d) $\alpha_1 = 0.01, A = [-6, 6]^2$

Figure 13. Effect of a decrease in parameter α_1 on the quintic Julia sets.

The parameters used in Figure 13 are as follows: $a_1 = 6, a_2 = 20, a_3 = 10, \alpha_2 = 0.0679, \alpha_3 = 0.0165, n = 5$, and $s = 0.41$. Not much change in the shape and size of the quintic Julia sets is seen as the value of α_1 used in the Noor-iteration decreases from 0.9 to 0.01. However, the number of kinks along the boundary increases. However, the time taken to generate these fractals decreases from 1.529644 to 1.491905 seconds with a decrease in the value of α_1.

The parameters used in Figure 14 are as follows: $a_1 = 6, a_2 = 20, a_3 = 10, \alpha_1 = 0.01565, \alpha_2 = 0.0679, n = 5$, and $s = 0.41$. A decrease in α_3 used in the Noor-iteration adds beauty to the quintic Julia set, maintaining the same area. However, a decrease in the generation time of the quintic Julia sets, from 1.525437 to 1.355064 seconds, is observed.



(a) $\alpha_3 = 0.9, A = [-5, 5]^2$ **(b)** $\alpha_3 = 0.3, A = [-5, 5]^2$

(c) $\alpha_3 = 0.1, A = [-5, 5]^2$ **(d)** $\alpha_3 = 0.01, A = [-5, 5]^2$

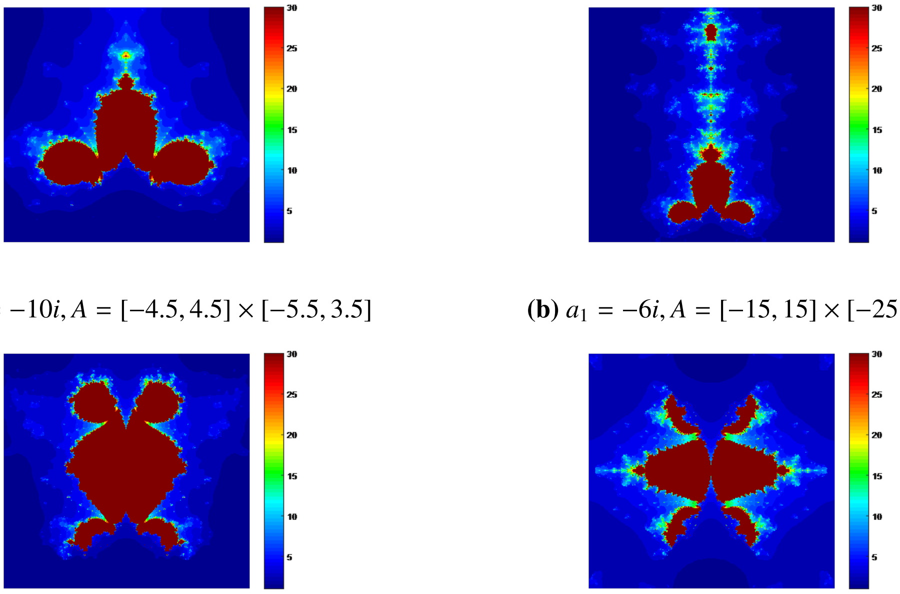
Figure 14. Effect of a decrease in parameter α_3 on the quintic Julia sets.

4.2. *Mandelbrot fractals*

A significant change in shape and a decrease in the area in which the cubic Mandelbrot set lies, are observed as the absolute value of parameter a_1 used in the complex-valued cubic polynomial decreases. The time taken to generate these lies between 1.618447 and 2.845240 seconds.

The real parameters used in Figure 15 are as follows: $a_2 = -1, \alpha_1 = 0.022165, \alpha_2 = 0.022679, \alpha_3 = 0.022165, s = 0.5$, and $n = 3$.

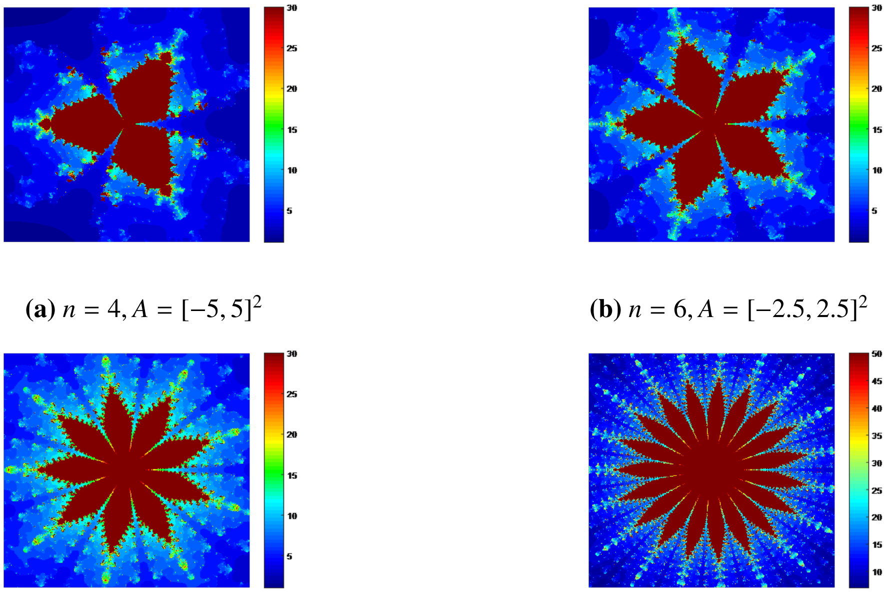
The parameters used in Figure 16 are as follow: $a_1 = 0, a_2 = -1, \alpha_1 = 0.022165, \alpha_2 = 0.022679, \alpha_3 = 0.022165$, and $s = 0.5$. Fractals resembling Rangoli, a traditional Indian art form created during festive seasons, are observed. The number of petals in each fractal is $n - 1$, where n is the degree of the complex-valued polynomial. It is fascinating to see that the quartic Mandelbrot set bears a resemblance to the Mitsubishi logo. Also, there is a decrease in the occupied area with the increase in n and the time taken to generate these increases from 2.011438 to 3.371230 seconds.



(a) $a_1 = -10i, A = [-4.5, 4.5] \times [-5.5, 3.5]$ **(b)** $a_1 = -6i, A = [-15, 15] \times [-25, 5]$

(c) $a_1 = -4i, A = [-15, 15] \times [-20, 10]$ **(d)** $a_1 = 0, A = [-10, 10]^2$

Figure 15. Effect of parameter a_1 on the cubic Mandelbrot sets.

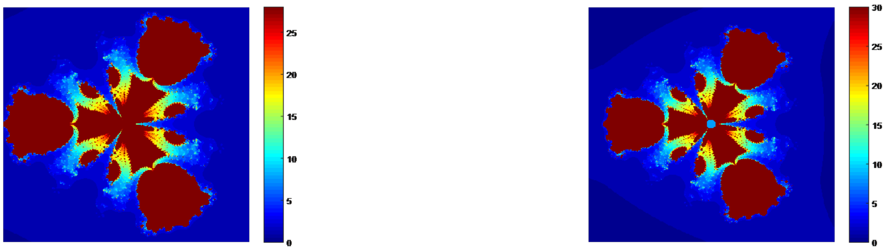


(a) $n = 4, A = [-5, 5]^2$ **(b)** $n = 6, A = [-2.5, 2.5]^2$

(c) $n = 10, A = [-1.5, 1.5]^2$ **(d)** $n = 20, A = [-0.8, 0.8]^2$

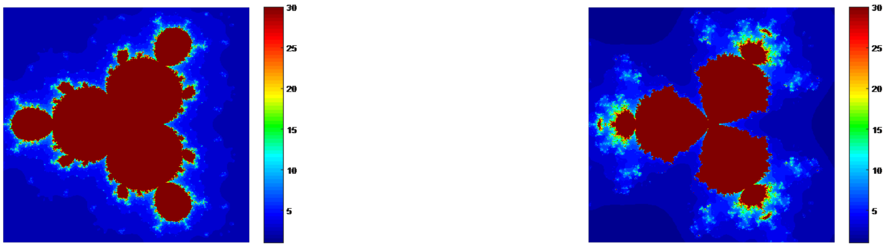
Figure 16. Effect of the degree of polynomial n on the Mandelbrot sets.

The parameters used in Figure 17 are as follow: $a_1 = 0, \alpha_1 = 0.03, \alpha_2 = 0.25, \alpha_3 = 0.05, s = 0.6$, and $n = 4$. There is not much difference in the shapes of the quartic Mandelbrot sets except for the appearance of a ring at the center and a decrease in the area in which these lie when the value of the coefficient of z (parameter a_2) decreases from 3 to 1. The time taken to generate these increases from 1.021400 to 2.140588 seconds.



(a) $a_2 = 3, A = [-23, 23]^2$ **(b)** $a_2 = 1, A = [-6, 6]^2$

Figure 17. Effect of parameter a_2 on the quartic Mandelbrot sets.

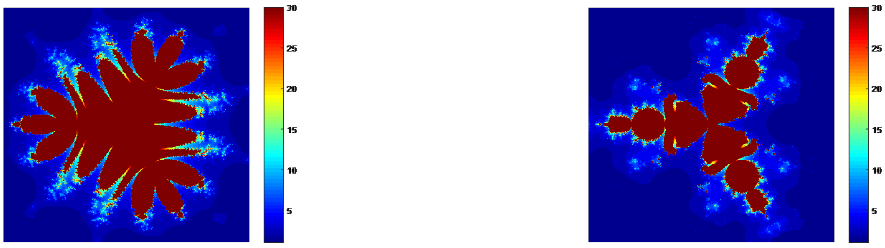


(a) $\alpha_1 = 0.9, A = [-1.5, 1.5]^2$ **(b)** $\alpha_1 = 0.1, A = [-16, 16]^2$

Figure 18. Effect of parameter α_1 on the quartic Mandelbrot sets.

The parameters used in Figure 18 are as follow: $a_1 = 0, a_2 = -1, \alpha_2 = 0.022679, \alpha_3 = 0.022165, s = 0.9$, and $n = 4$. A decrease in the value of parameter α_1 adds color and volume to the quartic Mandelbrot set and the time taken to generate these are 2.021277 and 1.777622 seconds, respectively.

The parameters used in Figure 19 are as follow: $a_1 = 0, a_2 = -1, \alpha_1 = 0.1, \alpha_3 = 0.022165, s = 0.9$, and $n = 4$. A decrease in parameter α_2 adds beauty and volume to the quartic Mandelbrot set and the time taken to generate these are 1.774143 and 1.157254 seconds, respectively.

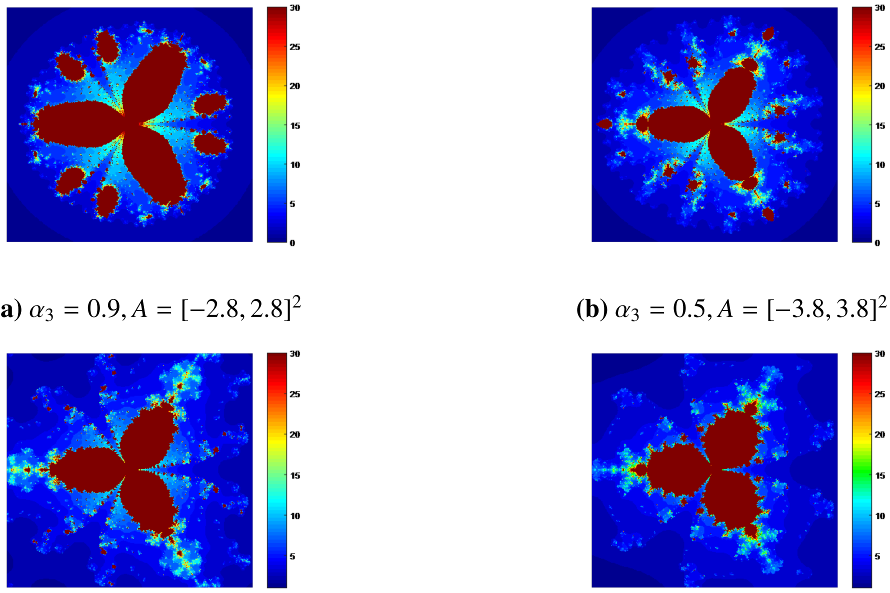


(a) $\alpha_2 = 0.9, A = [-2.2, 2.2]^2$ **(b)** $\alpha_2 = 0.1, A = [-18, 18]^2$

Figure 19. Effect of parameter α_2 on the quartic Mandelbrot sets.

The parameters used in Figure 20 are as follow: $a_1 = 0, a_2 = -1, \alpha_1 = 0.05, \alpha_2 = 0.01, s = 0.6$, and $n = 4$. A decrease in the value of parameter α_3 adds beauty and volume to the quartic Mandelbrot set.

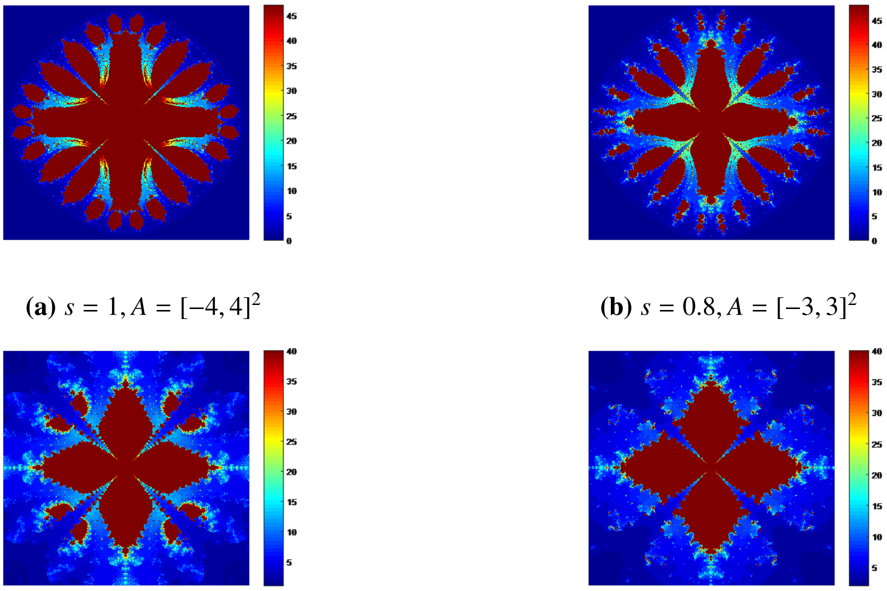
The time taken to generate these fractals decreases from 2.071507 to 1.944949 seconds.



(a) $\alpha_3 = 0.9, A = [-2.8, 2.8]^2$ **(b)** $\alpha_3 = 0.5, A = [-3.8, 3.8]^2$

(c) $\alpha_3 = 0.2, A = [-4, 4]^2$ **(d)** $\alpha_3 = 0.05, A = [-6, 6]^2$

Figure 20. Effect of parameter α_3 on the quartic Mandelbrot sets.



(a) $s = 1, A = [-4, 4]^2$ **(b)** $s = 0.8, A = [-3, 3]^2$

(c) $s = 0.5, A = [-1.3, 1.3]^2$ **(d)** $s = 0.1, A = [-0.4, 0.4]^2$

Figure 21. Effect of convexity parameter s on the quintic Mandelbrot sets.

The parameters used in Figure 21 are as follow: $a_1 = 0, a_2 = -1, \alpha_1 = 0.05, \alpha_2 = 0.1, \alpha_3 = 0.5$, and $n = 5$. The time taken to generate these quintic Julia sets lies between 2.647744 and 3.023838 seconds. The area in which the fractal lies decreases with the decrease in the convexity parameter s. Also, a significant change in shape is observed.

Remark 4.2. The comparative study reveals that the proposed Jungck-Noor iteration with s-convexity [35] surpasses the classical Picard iteration [9] in terms of generality, convergence speed, computational efficiency, and fractal visualization. This approach opens new avenues for exploring higher-order polynomials and complex fractal structures, making it essential for modern fractal dynamics and polynomial escape criteria analysis. Notably, Jungck-type iterations exhibit a faster convergence rate than the Picard iteration. Studies have shown that multi-step iterative schemes in fractal generation reduce computational time by over 30% compared to Picard iterations, further highlighting the efficiency of the proposed method.

However, fully analyzing the results is challenging due to the complexity of the three-step Jungck-Noor iteration, which involves multiple parameters $(\alpha_1, \alpha_2, \alpha_3, s)$ and additional parameters from the underlying higher-order polynomial $f(z)$ (such as a_1, a_2, a_3). Consequently, studying the impact of individual parameter variations remains an ongoing research topic. Even a single parameter change (e.g., in Figure 7, where only s is modified) results in distinct fractal structures. Thus, in Corollary 3.1, we have confined our analysis to selected parameter combinations, covering a majority of possible variations to establish escape radii. Mandelbrot and Julia fractals have been explored for various complex-valued polynomials using different parameter sets. However, this study presents only a subset of fractals to highlight the distinct impact of each parameter. Key observations are:

(i) As n increases, the area occupied by the fractal diminishes (see Figures 8 and 16).
(ii) The fractal structure becomes circular (see Figure 3 for $n = 20$, where the Julia fractal resembles a circular saw or teething ring).
(iii) Fractals of a polynomial of degree n typically contain $n - 1$ petals.
(iv) Parameters such as a_1, a_2, and a_3 and the iteration parameters α_1, α_2, and α_3 play a crucial role in defining fractal shapes.
(v) Changes in the signs of a_1 and a_2 result in reflexive and rotational symmetry (see Figures 7, 9, and 10).
(vi) The area occupied by the Mandelbrot set decreases as α_1, α_2, and α_3 increase.
(vii) Beautiful fractal images emerge for both rational and irrational values of n.
(viii) The generation time for all fractals ranges between 1.02 and 3.37 seconds.
(ix) Graphical illustrations emphasize the significance of the three-step Jungck-Noor fixed point iteration with s-convexity in studying complex fractal structures.
(x) Adjusting parameters (see Remark 2.1) allows for generalizations and improvements over previous fixed point iteration results in the literature.
(xi) The value of s determines the volume of the fractal (see Figures 5 and 21).
(xii) When $s \leq 0.5$, increased brightness appears around the boundary.
(xiii) As $s \to 1$, brightness decreases, revealing more intricate details in the fractal structure.

5. Dependence between visualization time and parameters of the Jungck-Noor iteration with s-convexity

In Section 4, we observed significant variations in the shapes, sizes, and generation times of the Mandelbrot and Julia fractals generated using the Jungck-Noor procedure with s-convexity. Analyzing the dependency of fractal shape, size, and generation time on the parameters involved in the iterative process is non-trivial. To explore this non-trivial dependency, we conducted numerical experiments to investigate the computational efficiency of the Jungck-Noor iteration in generating Mandelbrot and Julia fractals.

For each experiment, we:

(1) Varied the parameters $\alpha_1, \alpha_2, \alpha_3$, and s.
(2) Measured the time required to generate each fractal.
(3) Observed that non-escaping points in escape-time algorithms significantly impact the generation time. A higher number of non-escaping points within the computed area leads to a longer generation time.

We studied the complex-valued polynomial $f(z) = z^4 + a_1z^2 - a_2z + a_3$. The complex plane $[-4, 4] \times [-4, 4]$ was divided into a 400×400 grid. The Jungck-Noor iteration with s-convexity was applied at each grid point. We computed the generation time for parameter pairs: $(\alpha_1,\ s)$, $(\alpha_2,\ s)$, $(\alpha_3,\ s)$ $\in \{0.1, 0.2, 0.3, \dots, 0.9\} \times \{0.1, 0.2, 0.3, \dots, 0.9\}$ using MATLAB 8.5.0 (R2015a). The resolution of the generated fractals was 400×400 pixels, following Algorithms 1 and 2.

Figures 22–27 present the analytical expressions for generation time as a function of (α_1, s), (α_2, s), and (α_3, s). These expressions were generated using the TableCurve 3D v4.0 software from SYSTAT Software Inc. The approximations follow a linear model:

$$
\begin{aligned}
w &= a + b\alpha_1 + cs,\\
w &= a + b\alpha_2 + cs,\\
w &= a + b\alpha_3 + cs,
\end{aligned}
$$

where a, b, and c are constants determined via the least squares method for surface fitting. The coefficient of determination, $r^2 = 1$, confirms a perfect fit for the data. Key observations are:

Minimum generation time of the Mandelbrot set:

(i) $\alpha_1 = 0.1$, $s = 0.7$, and $t = 1.5799$ seconds (Figure 22);
(ii) $\alpha_2 = 0.2$, $s = 0.1$, and $t = 1.62908$ seconds (Figure 23); $\alpha_3 = 0.1$, $s = 1$, and $t = 1.62134$ seconds (Figure 24).

Maximum generation time of the Mandelbrot set:

(i) $\alpha_1 = 0.9$, $s = 0.6$, and $t = 2.93227$ seconds;
(ii) $\alpha_2 = 0.1$, $s = 0.1$, and $t = 3.4774$ seconds;
(iii) $\alpha_3 = 0.9$, $s = 0.9$, and $t = 2.50632$ seconds.

Minimum generation time of the Julia set:

(i) $\alpha_1 = 0.1$, $s = 1$, and $t = 1.50644$ seconds (Figure 25);
(ii) $\alpha_2 = 0.1$, $s = 0.7$, and $t = 1.08346$ seconds (Figure 26);
(iii) $\alpha_3 = 0.2$, $s = 1$, and $t = 1.26573$ seconds (Figure 27).

Maximum generation time of the Julia set:

(i) $\alpha_1 = 0.8$, $s = 0.4$, and $t = 1.87705$ seconds;

(ii) $\alpha_2 = 0.7$, $s = 0.2$, and $t = 2.205$ seconds;

(iii) $\alpha_3 = 1$, $s = 0.8$, and $t = 2.1386$ seconds.

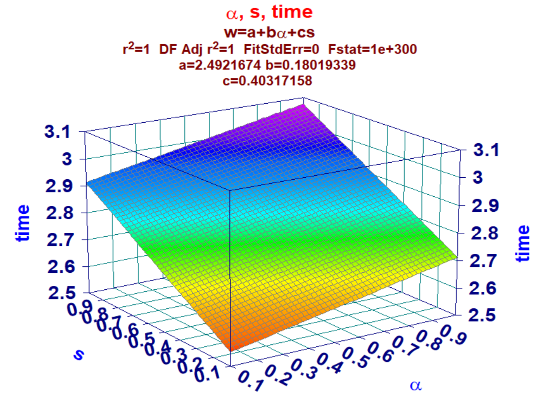


Figure 22. Time-space for the Mandelbrot set depending on ($\alpha = \alpha_1, s$).

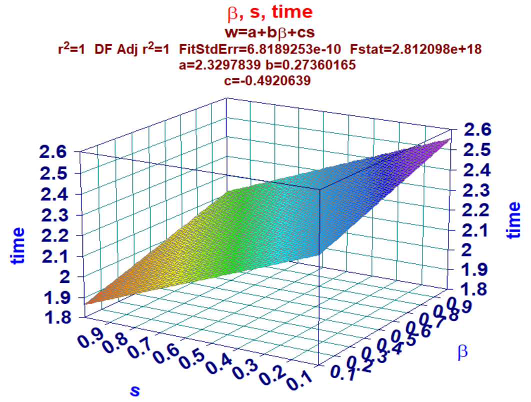


Figure 23. Time-space for the Mandelbrot set depending on ($\beta = \alpha_2, s$).

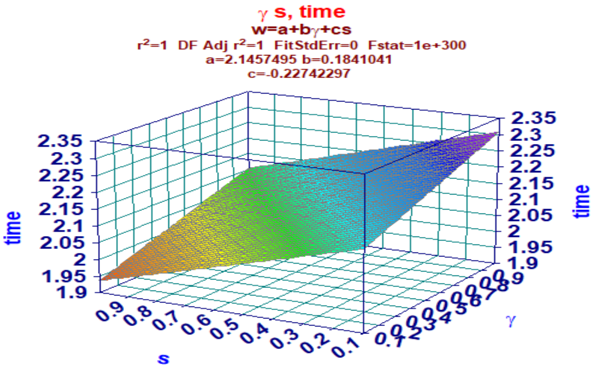


Figure 24. Time-space for the Mandelbrot set depending on ($\gamma = \alpha_3, s$).

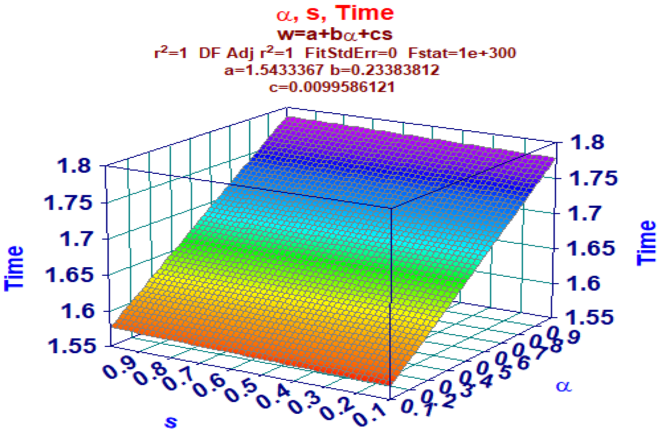


Figure 25. Time-space for the Julia set depending on ($\alpha = \alpha_1, s$).

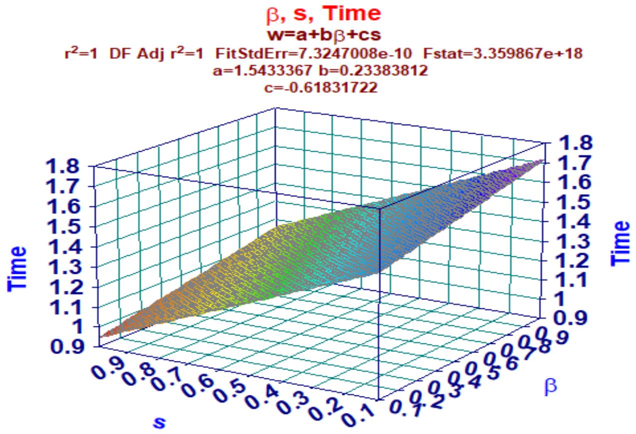


Figure 26. Time-space for the Julia set depending on ($\beta = \alpha_2, s$).

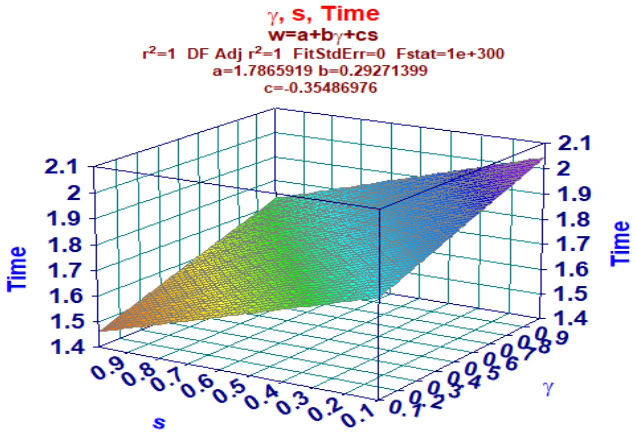


Figure 27. Time-space for the Julia set depending on ($\gamma = \alpha_3, s$).

Comparison with Picard iteration

(a) Jungck-Noor may indeed converge in fewer steps than Picard. Because each Jungck-Noor step is computationally heavier, the total wall-clock time can still be greater.

(b) The generation time for the standard Picard iteration is 1.05 seconds.

(c) Although the Jungck–Noor iteration requires fewer steps to reach convergence compared to the classical Picard method, each Jungck–Noor step involves more computational operations (due to additional averaging terms and function evaluations). Consequently, the overall wall-clock time for fractal visualization may be higher, even if the iteration count is lower. To make this distinction precise, we introduce a cost model that accounts for both the number of iterations and the per-iteration arithmetic complexity.

We can justify it by the cost model. Define cost as: Total $Cost = N \times C_{iter}$, where N = number of iterations until convergence, C_{iter} = computational cost of a single iteration (measured in floating-point operations or execution time per iteration). Then:

(i) Picard iteration: lower C_{iter}, higher N.

(ii) Jungck–Noor iteration: higher C_{iter}, lower N. So the trade-off depends on the product $N \times C_{iter}$, not just the iteration count.

Remark 5.1. The fractal patterns generated in this study resemble traditional designs like "Kachhi thread work", establishing a link between mathematical theory and real-world applications in fashion and textile design. These intricate fractal structures also have potential applications in digital watermarking and AI-generated art, demonstrating the practical significance of the theoretical concepts explored in this paper.

The escape criteria for polynomials and the visualization of Mandelbrot and Julia sets presented in this study can be applied in various fields, including: (a) Image processing, (b) Textile design, (c) Cryptography, (d) AI-generated art, and (e) Pattern recognition. These applications highlight the broader impact of fractal theory beyond pure mathematics, extending into technology, security, and creative industries.

6. Conclusions

In this study, we utilize the three-step Jungck-Noor fixed point iteration with s-convexity for nth-order complex-valued polynomials, refining the approach of Jolaoso et al. [29]. Our method introduces a more natural escape criterion for higher-order polynomials, which was absent in prior studies [13, 16, 21, 30, 35, 37], etc. Key findings include:

(1) For irrational values of n, the generated Julia fractals occupy a larger area compared to conventional fractals (see Figure 4).

(2) The fractal generation process takes between 1 to 4 seconds, with variations influenced by:
(a) Polynomial degree (n)
(b) Convexity parameter (s)
(c) Iteration parameters (α_1, α_2, α_3)
(d) Shape and symmetry parameters (a_1, a_2, a_3).

(3) Numerical experiments reveal that s-convexity plays a dominant role in determining generation time, surpassing the influence of α_1, α_2, and α_3.

(4) Certain fractals resemble Kachhi threadwork designs, suggesting potential applications in textile design.
(5) The findings hold significance for fixed point theorists, physicists, and textile engineers, bridging theoretical insights with practical applications in science and industry.

Future research may extend this framework by examining higher-order polynomials ($a_4z^4, a_5z^5, \dots$) and their impact on symmetry, escape dynamics, and fractal dimension. Such work could reveal new geometries, self-similar structures, and conditions for symmetry breaking, akin to phase transitions. Rising computational demands for higher powers also call for efficient escape-time algorithms and possible complexity classifications of fractals. Another open problem is a construction of parameter bifurcation diagrams with analytically determined thresholds, which could clarify how coefficients shape fractal structures. Finally, comparative studies across iterative schemes, as suggested in the literature, may help assess relative efficiency and graphical fidelity.

7. Open questions

While this paper establishes escape criteria for the polynomial $f(z) = z^n + a_1z^2 - a_2z + a_3$ using the Jungck-Noor iteration and s-convexity, several important open questions remain:

(1) Can more generalized escape criteria be established for polynomials of the form $f(z) = z^n + \sum_{i=1}^{m} a_iz^i$, where $m > 3$?
(2) Can the impact of higher-order terms on the nature of fractal structures be fully characterized? How do these terms affect the complexity, symmetry, and self-similarity of fractals?

In addition, some of the existing literature permits comparative analyses of different graphs generated under distinct iterative schemes. Extending our study to include such cross-method comparisons for producing identical fractals would provide deeper insights into the relative efficiency and graphical fidelity of the methods. This direction will be pursued in future work.

Author contributions

Anita Tomar: Conceptualization, methodology, investigation, writing−review and editing; Swati Antal: Investigation, formal analysis, software, writing−original draft; Mohammad Sajid: Conceptualization, methodology, funding acquisition, validation, writing−review and editing; Darshana J. Prajapati: Conceptualization, methodology, validation, writing−original draft. All authors have read and approved the final version of the manuscript for publication.

Use of Generative-AI tools declaration

The authors declare they have not used artificial intelligence (AI) tools in the creation of this article.

Acknowledgments

The researchers would like to thank the Deanship of Graduate Studies and Scientific Research at Qassim University for financial support (QU-APC-2025).

Conflict of interest

The authors declare that they have no conflicts of interest.

References

1. M. F. Barnsley, *Fractals everywhere*, 2 Eds., San Diego: Academic Press, 1993. https://doi.org/10.1016/c2013-0-10335-2
2. R. L. Devaney, *A first course in chaotic dynamical systems: Theory and experiment*, 2 Eds., New York: Chapman and Hall/CRC, 2020. https://doi.org/10.1201/9780429280665
3. J. Hu, H. Shen, X. Liu, J. Wang, RDMA transports in datacenter networks: Survey, *IEEE Netw.*, **38** (2024), 380–387. https://doi.org/10.1109/mnet.2024.3397781
4. B. B. Mandelbrot, *The fractal geometry of nature*, New York: W. H. Freeman and Company, 1982.
5. G. M. Julia, Mémoire sur l'itération des fonctions rationnelles, *J. Math. Pures Appl.*, **8** (1918), 47–245.
6. S. Antal, A. Tomar, D. J. Prajapati, M. Sajid, Fractals as Julia sets of complex sine function via fixed point iterations, *Fractal Fract.*, **5** (2021), 272. https://doi.org/10.3390/fractalfract5040272
7. D. J. Prajapati, S. Rawat, A. Tomar, M. Sajid, R. C. Dimri, A brief study of dynamics of Julia sets for entire transcendental function using Mann iterative scheme, *Fractal Fract.*, **6** (2022), 397. https://doi.org/10.3390/fractalfract6070397
8. A. Tomar, V. Kumar, U. S. Rana, M. Sajid, Fractals as Julia and Mandelbrot sets of complex cosine functions via fixed point iterations, *Symmetry*, **15** (2023), 478. https://doi.org/10.3390/sym15020478
9. E. Picard, Mémoire sur la théorie des équations aux dérivées partielles et la méthode des approximations successives, *J. Math. Pures Appl.*, **6** (1890), 145–210.
10. H. Afshari, H. Aydi, Some results about Krosnoselski–Mann iteration process, *J. Nonlinear Sci. Appl.*, **9** (2016), 4852–4859. https://doi.org/10.22436/jnsa.009.06.120
11. M. O. Olatinwo, Some stability and strong convergence results for the Jungck–Ishikawa iteration process, *Creat. Math. Inform.*, **17** (2008), 33–42.
12. S. Ishikawa, Fixed points by a new iteration method, *Proc. Amer. Math. Soc.*, **44** (1974), 147–150. https://doi.org/10.2307/2039245
13. Y. C. Kwun, A. A. Shahid, W. Nazeer, M. Abbas, S. M. Kang, Fractal generation via CR-iteration scheme with s-convexity, *IEEE Access*, **7** (2019), 69986–69997. https://doi.org/10.1109/access.2019.2919520
14. W. R. Mann, Mean value methods in iteration, *Proc. Amer. Math. Soc.*, **4** (1953), 506–510. https://doi.org/10.2307/2032162
15. M. A. Noor, New approximation schemes for general variational inequalities, *J. Math. Anal. Appl.*, **251** (2000), 217–229. https://doi.org/10.1006/jmaa.2000.7042
16. S. Y. Cho, A. A. Shahid, W. Nazeer, S. M. Kang, Fixed point results for fractal generation in Noor orbit and s-convexity, *SpringerPlus*, **5** (2016), 1843. https://doi.org/10.1186/s40064-016-3530-5

17. A. A. Shahid, W. Nazeer, K. Gdawiec, The Picard–Mann iteration with s-convexity in the generation of Mandelbrot and Julia sets, *Monatsh. Math.*, **195** (2021), 565–584. https://doi.org/10.1007/s00605-021-01591-z

18. M. Tanveer, S. Kang, W. Nazeer, Y. Kwun, New tricorns and multicorns antifractals in Jungck–Mann orbit, *Int. J. Pure Appl. Math.*, **111** (2016), 287–302. https://doi.org/10.12732/ijpam.v111i2.13

19. N. Adhikari, W. Sintunavarat, Exploring the Julia and Mandelbrot sets of $z^p + \log c^t$ using a four-step iteration scheme extended with s-convexity, *Math. Comput. Simul.*, **220** (2024), 357–381. https://doi.org/10.1016/j.matcom.2024.01.010

20. S. Rawat, D. Prajapati, A. Tomar, K. Gdawiec, Generation of Mandelbrot and Julia sets as fractals for generalized rational maps using SP-iteration process equipped with s-convexity, *Math. Comput. Simul.*, **220** (2024), 148–169. https://doi.org/10.1016/j.matcom.2023.12.040

21. S. Kumari, M. Kumari, R. Chugh, Generation of new fractals via SP-orbit with s-convexity, *Int. J. Eng. Technol.*, **9** (2017), 2491–2504. https://doi.org/10.21817/ijet/2017/v9i3/1709030282

22. N. Özgür, S. Antal, A. Tomar, Julia and Mandelbrot sets of transcendental functions via Fibonacci–Mann iteration, *J. Funct. Spaces*, **2022** (2022), 2592573. https://doi.org/10.1155/2022/2592573

23. M. Ashish, M. Rani, R. Chugh, Study of cubic Julia sets in NO, *J. Comput. Sci. Technol.*, **1** (2013), 13–17.

24. K. Gdawiec, W. Kotarski, A. Lisowska, Biomorphs via modified iterations, *J. Nonlinear Sci. Appl.*, **9** (2016), 2305–2315. https://doi.org/10.22436/jnsa.009.05.33

25. M. Rani, Cubic superior Julia sets, In: *Proceedings of the 5th European conference on European computing conference*, 2011, 80–84.

26. V. V. Strotov, S. A. Smirnov, S. E. Korepanov, A. V. Cherpalkin, Object distance estimation algorithm for real-time FPGA-based stereoscopic vision system, In: *Proceedings Volume 10792, High-performance computing in Geoscience and remote sensing VIII*, 2018. https://doi.org/10.1117/12.2324851

27. J. S. Pan, S. Q. Zhang, S. C. Chu, C. C. Hu, J. Wu, Efficient FPGA implementation of sine cosine algorithm using high-level synthesis, *J. Internet Technol.*, **25** (2024), 865–876. https://doi.org/10.70003/160792642024112506007

28. Y. Wang, S. Liu, H. Li, D. Wang, On the spatial Julia set generated by fractional Lotka–Volterra system with noise, *Chaos Soliton Fract.*, **128** (2019), 129–138. https://doi.org/10.1016/j.chaos.2019.07.044

29. L. O. Jolaoso, S. H. Khan, Some escape time results for general complex polynomials and biomorph generation by a new iteration process, *Mathematics*, **8** (2020), 2172. https://doi.org/10.3390/math8122172

30. W. Nazeer, S. M. Kang, M. Tanveer, A. A. Shahid, Fixed point results in the generation of Julia and Mandelbrot sets, *J. Inequal. Appl.*, **2015** (2015), 298. https://doi.org/10.1186/s13660-015-0820-3

31. I. Andreadis, T. E. Karakasidis, On a topological closeness of perturbed Julia sets, *Appl. Math. Comput.*, **217** (2010), 2883–2890. https://doi.org/10.1016/j.amc.2010.08.024

32. Y. Sun, P. Li, Z. Lu, Generalized quaternion M sets and Julia sets perturbed by dynamical noises, *Nonlinear Dynam.*, **82** (2015), 143–156. https://doi.org/10.1007/s11071-015-2145-7

33. D. Li, M. Tanveer, W. Nazeer, X. Guo, Boundaries of filled Julia sets in generalized Jungck–Mann orbit, *IEEE Access*, **7** (2019), 76859–76867. https://doi.org/10.1109/access.2019.2920026

34. A. Tomar, D. J. Prajapati, S. Antal, S. Rawat, Variants of Mandelbrot and Julia fractals for higher-order complex polynomials, *Math. Methods Appl. Sci.*, 2022, 1–13. https://doi.org/10.1002/mma.8262

35. S. M. Kang, W. Nazeer, M. Tanveer, A. A. Shahid, New fixed point results for fractal generation in Jungck–Noor orbit with s-convexity, *J. Funct. Spaces*, **2015** (2015), 963016. https://doi.org/10.1155/2015/963016

36. S. Antal, A. Tomar, D. J. Prajapati, M. Sajid, Variants of Julia and Mandelbrot sets as fractals via Jungck–Ishikawa fixed point iteration system with s-convexity, *AIMS Mathematics*, **7** (2022), 10939–10957. https://doi.org/10.3934/math.2022611

37. K. Gdawiec, A. A. Shahid, Fixed point results for the complex fractal generation in the S-iteration orbit with s-convexity, *Open J. Math. Sci.*, **2** (2018), 56–72. http://dx.doi.org/10.30538/oms2018.0017

38. H. Zhang, M. Tanveer, Y. X. Li, Q. Peng, N. A. Shah, Fixed point results of an implicit iterative scheme for fractal generations, *AIMS Mathematics*, **6** (2021), 13170–13186. https://doi.org/10.3934/math.2021761

39. S. Banach, Sur les opérations dans les ensembles abstraits et leur application aux équations intégrales, *Fund. Math.*, **3** (1922), 133–181. https://doi.org/10.4064/fm-3-1-133-181