Title: One more look on visualization of operation of a root-finding algorithm

Author: Ireneusz Gościniak, Krzysztof Gdawiec

Citation style: Gościniak Ireneusz, Gdawiec Krzysztof. (2020). One more look on visualization of operation of a root-finding algorithm. "Soft Computing" 2020 (article in press), doi 10.1007/s00500-020-04784-0

Soft Computing
https://doi.org/10.1007/s00500-020-04784-0

METHODOLOGIES AND APPLICATION

One more look on visualization of operation of a root-finding algorithm

Ireneusz Gościniak[1] · **Krzysztof Gdawiec**[1]

Abstract
Many algorithms that iteratively find solution of an equation require tuning. Due to the complex dependence of many algorithm's elements, it is difficult to know their impact on the work of the algorithm. The article presents a simple root-finding algorithm with self-adaptation that requires tuning, similarly to evolutionary algorithms. Moreover, the use of various iteration processes instead of the standard Picard iteration is presented. In the algorithm's analysis, visualizations of the dynamics were used. The conducted experiments and the discussion regarding their results allow to understand the influence of tuning on the proposed algorithm. The understanding of the tuning mechanisms can be helpful in using other evolutionary algorithms. Moreover, the presented visualizations show intriguing patterns of potential artistic applications.

Keywords Self-adaptation · Root finding · Dynamics · Iterations · Visualization

1 Introduction

Methods for determining function's local minimum are based on the value of the function or its gradient (mostly numerically determined). The gradient method determines the direction of particle movement based on the knowledge of the gradient of the objective function (at the point reached in the previous step), and on this basis, it calculates the next position of the particle (Polak 1997)—a typical example of the gradient method is described by the expression (the gradient descent):

$$x_i' = x_i - \gamma \nabla f(x_i), \tag{1}$$

where $-\nabla f(x_i)$ is the negative gradient of f, γ—step size, x'_i—the current position of the ith particle in a D-dimensional environment and x_i—the previous position of the ith particle. Many modifications of this method are described in the literature, and their operation mechanisms are well known (Klein et al. 2009; Konečný and Richtárik 2017; Senov and Granichin 2017).

A group of algorithms solving such problems includes evolutionary algorithms. The analysis of evolutionary algorithm is a very complex task (Gosciniak 2008, 2017). Some optimization algorithms can have similar behavioural characteristics as evolutionary algorithms (Weise 2009). A particular attention should be paid to the particle swarm optimization (PSO) algorithms (Zhang et al. 2014). The complex nature of particles movement in these algorithms does not allow a precise definition of their effect on the algorithm.

Particle's movement in the PSO algorithm is described by the following equation (Yang and Li 2010):

$$x_i' = x_i + v_i', \tag{2}$$

where x_i' is the current position of the ith particle in a D-dimensional environment, x_i—the previous position of the ith particle and v_i'—the current velocity of the ith particle in a D-dimensional environment that is given by the following formula:

$$v_i' = \omega v_i + \eta_1 r_1 (x_{pb\,i} - x_i) + \eta_2 r_2 (x_{gb} - x_i), \tag{3}$$

where v_i is the previous velocity of the ith particle, ω—the inertia weight ($\omega \in [0, 1]$), η_1, η_2—acceleration constants ($\eta_1, \eta_2 \in (0, 1]$), r_1, r_2—random numbers generated uni-

Communicated by V. Loia.

✉ Ireneusz Gościniak
ireneusz.gosciniak@us.edu.pl

Krzysztof Gdawiec
kgdawiec@ux2.math.us.edu.pl

[1] Institute of Computer Science, University of Silesia, Będzińska 39, 41-200 Sosnowiec, Poland

Published online: 04 March 2020

formly in the [0, 1] interval, $x_{pb\,i}$—the best position of the ith particle and x_{gb}—the global best position of the particles.

The inertia weight (ω) and acceleration constants (η_1, η_2) play a very important role in the algorithm—they are responsible for the particle dynamics. The acceleration constants direct the particle towards its own best position or the global best position (Sengupta and Mishra 2014). The particle can be trapped in a local optima for too high values of the acceleration constants or cannot reach the solution for too low values of these constants. Balance of the exploration and exploitation is also controlled by the inertia weight (Bansal et al. 2011). The control rules of inertia weight can be classified as follows: constant (Shi and Eberhart 1998); random (Eberhart and Shi 2001); time varying (linear decreasing Shi and Eberhart 1999, sigmoid increasing/decreasing Malik et al. 2007, simulated annealing Al-Hassan et al. 2006, Sugeno function Lei et al. 2006, exponential decreasing law Chen et al. 2006; Li and Gao 2009, logarithmic decreasing law Gao et al. 2008); adaptive control using feedbacks of the optimization process (best fitness Saber et al. 2006; Shi and Eberhart 2001, fitness of the current and previous iterations Yang et al. 2007, global best and average local best fitness Arumugam and Rao 2008, particle rank Panigrahi et al. 2008, distance to particle and global best positions Qin et al. 2006 and distance to global best position Suresh et al. 2008). The value of inertia weight mostly presented in the literature is within the range of [0.4, 0.9] (see Jordehi and Jasni 2013). Both the inertia weight and the acceleration constants allow to control the particle behaviour within a wider range. The test function (problem being solved) and the selected method of iteration also influence particle's behaviour. The particle dynamics is determined by the adaptation mechanics resulting mostly from particles' cooperation. The method of algorithm parameters selecting is not deterministic—these values are selected by a tuning process depending on a solved problem and the iteration method that is used (similarly to evolutionary algorithms Bansal et al. 2011; Sengupta and Mishra 2014).

Due to the particle motion mechanism and particle's behaviour control parameters, we can use the wording—a particle dynamics. The tuning—selection of the proper values of the parameters—is a very important problem for the algorithm. In most cases, the algorithm tuning process is intuitive and it is based on user experience.

Finding of roots of a given function f, i.e. solving the equation $f(x) = 0$, is a very important problem in practical applications, e.g. physics (Franklin 2013), electronics (Chun and Kwasinski 2011) or computer graphics (Chen and Ma 2015). In many of these applications, the function is a polynomial one. So, the following question arises: can we solve the polynomial equation by its radicals (i.e. giving the formula for the roots in terms of the polynomial's coefficients, the four algebraic operations: addition, subtraction, multiplication, division and the extraction of roots, i.e. square roots, cube roots, etc.)? The answer to this question is that for polynomials of degree greater than four we cannot find such formulas (Grant and Kleiner 2015). The formulas for quadratic polynomial are known since the Babylonians. For the cubic polynomial the formulas were found by G. Cardano, whereas for the quartics by L. Ferrari both in the XVI century. In 1779 P. Ruffini and in 1826 N.-H. Abel proved the unsolvability by radicals of the polynomial equation of degree greater than four—the Abel–Ruffini theorem (Grant and Kleiner 2015). Thus, to solve a general polynomial equation we need to use some numerical methods, which are iterative methods and define discrete dynamical systems. In the literature, we can find many such methods, e.g. Newton's method (Kalantari 2009) and Halley's method (Kalantari 2009), or even the gradient descent method can be used for root finding (Kotarski and Lisowska 2018).

The analysis of dynamics of dynamical systems is a very important problem (Sayama 2015), because it gives us an understanding of the changes that occur in the system. To analyse the dynamics, we can use many different methods (Broer and Takens 2011). One of such methods is a graphical presentation of the dynamics. The graphical representation provides a lot of intuitive, geometrical insight into the system's dynamics, which would be hard to infer if we had just looked at algebraic equations (Sayama 2015). In the analysis of the complex polynomial root-finding process, the graphical presentation of dynamics is a very popular method and it even got its own name, namely polynomiography (Kalantari 2009). In polynomiography we visualize the root-finding process by using the number of iterations needed to obtain the root of a given polynomial and some colour map. The obtained images show the dynamics of the process. For the visualization and the analysis of algorithm's behaviour, we can use methods similar to these used in polynomiography (Kalantari 2004). The polynomiography visualizes the complex nature of the relationships between parameters.

The images generated using polynomiography very often reveal aesthetic patterns with a very complex structure (Gdawiec 2017). Because these complex patterns are generated with a simple method polynomiography, besides its analytic purposes, it also found application in computer-aided design and in the arts (Kalantari 2004). Polynomiography assists the designer in the creation of aesthetic patterns, because the designer must select only some parameters used in the method and the computer generates the pattern, whereas without polynomiography to create the same pattern the designer must spend a lot of time and put a lot of work in the creation process.

The PSO-based gradient-like method is presented in the paper. In this simple root-finding algorithm, tuning parameters such as inertia weight and acceleration constant, and an adaptive mechanism depending on the location of the particle are proposed. This algorithm is used to show the influence

of the tuning parameters on particle's behaviour. Moreover, we propose the use of different iteration processes instead of the standard Picard iteration that is used in the root-finding methods and optimization algorithms.

The aim of the article is to show the dynamics of particle motion and the influence of the parameters on particle's behaviour. For this purpose, we use a visualization method and analysis similar to the polynomiography. The used visualization method—presented in the article—can also have an artistic meaning. According to the authors, the understanding of the dynamics of particle's motion is a very important for its use in the group of evolutionary algorithms. The proper selection of particle dynamics may not cause the influence of the environment on the algorithm—as it is realized in the algorithm presented in Gosciniak (2017).

Let's try to summarize. What cannot be found in this article? The article is not a recipe how to effectively adjust the parameters of the algorithm. (Tuning an algorithm is a complex problem to be solved by other optimization algorithms—for instance by a genetic algorithm.) So what is this article about? Its main task is to visualize and analyse the complex nature of particle dynamics resulting from the selection of algorithm's coefficients and its interaction with the environment. However, if we analyse the behaviour of particles in evolutionary algorithms working in multidimensional spaces, a fractal analysis method can be proposed (Gosciniak 2017). What are the benefits of the article? Besides the mentioned visualization and the analysis of the complex nature of particle dynamics, the results of the paper can find application in the generative art. (Another example of this kind of algorithm can be the Newton's method, which is at the forefront in this field.)

The rest of the paper is organized as follows. Section 2 introduces a root-finding algorithm that is based on the gradient method. This method will be used to illustrate the influence of the parameters on its behaviour. Next, Sect. 3 introduces the iteration processes known in fixed-point theory for finding the fixed points of different types of functions. Section 4 presents a method of visualization of algorithm's operation. And next, in Sect. 5 based on the obtained polynomiographs we discuss the research results. Finally, in Sect. 6 we give some concluding remarks.

2 The root-finding algorithm

Let $f : \mathbb{R}^D \to \mathbb{R}$ be some function. We want to find zeroes of f, i.e. to solve the following equation

$$f(x) = 0. \tag{4}$$

To solve (4), we use the following algorithm:

$$x_{n+1} = x_n + v_{n+1}, \tag{5}$$

where $x_0 \in \mathbb{R}^D$ is a starting position, $v_0 = [0, 0, \ldots, 0]$ is a starting velocity, v_{n+1} is the current velocity of the particle ($v_{n+1} = [v^1_{n+1}, v^2_{n+1}, \ldots, v^D_{n+1}]$) and x_n is the previous position of the particle ($x_n = [x^1_n, x^2_n, \ldots, x^D_n]$). The algorithm uses a similar methodology as the PSO algorithm, i.e. it sums the position of the particle x_n with its current velocity v_{n+1}.

The current velocity of the particle is determined by the component of velocity of the previous iteration (inertia) and the component resulting from its current position (acceleration):

$$v^k_{n+1} = \omega v^k_n + \eta f(x_n) \boxed{\frac{f(x_n) - f(x_n + \varepsilon_k)}{\sum_{j=1}^{D} |f(x_n) - f(x_n + \varepsilon_j)|}}, \tag{6}$$

where $k \in \{1, 2, \ldots, D\}$, v^k_{n+1}—the current velocity of the particle in the direction k, v^k_n—the previous velocity of the particle in the direction k, $\omega \in [0, 1)$—inertia weight, $\eta \in (0, 1]$—acceleration constant, ε_k—the step in the direction k ($\varepsilon_1 = [\tau, 0, \ldots, 0]$, $\varepsilon_2 = [0, \tau, 0, \ldots, 0]$, …, $\varepsilon_D = [0, \ldots, 0, \tau]$, where $\tau > 0$).

Determining the particle motion towards the root is similar to the gradient method—the marked part of Eq. (6). The acceleration is influenced by the adaptation mechanism resulting from the value of the function in the previous position of the particle—$f(x_n)$. (It works effectively at determining a root of the function.) For this reason (when $\omega = 0$) for x_n fulfilling the condition $f(x_n) = 0$, the next particle position is $x_{n+1} = x_n$. The algorithm's tuning involves the selecting inertia weight (ω) and acceleration constant (η)—similarly as in the PSO algorithm. The effect of changes in these parameters, i.e. particle dynamics, will be visualized using the method described in Sect. 4.

3 Iteration processes

Let $T : \mathbb{R}^D \to \mathbb{R}^D$ be given by the following formula:

$$T(x_n) = x_n + v_{n+1}. \tag{7}$$

Thus, algorithm (5) can be written in the following form

$$x_{n+1} = T(x_n). \tag{8}$$

In the literature, this type of iteration is called the Picard iteration and it is widely used in many optimization algorithms and in finding fixed points of a given function.

In fixed-point theory, some other iteration processes were introduced to find the fixed points of different types of functions:

1. The Mann iteration (Mann 1953):

$$x_{n+1} = (1-\alpha_n)x_n + \alpha_n T(x_n),\ n = 0, 1, 2, \ldots, \quad (9)$$

where $\alpha_n \in (0, 1]$ for all $n \in \mathbb{N}$.
2. The Ishikawa iteration (Ishikawa 1974):

$$\begin{aligned} x_{n+1} &= (1-\alpha_n)x_n + \alpha_n T(y_n), \\ y_n &= (1-\beta_n)x_n + \beta_n T(x_n),\ n = 0, 1, 2, \ldots, \end{aligned} \quad (10)$$

where $\alpha_n \in (0, 1]$ and $\beta_n \in [0, 1]$ for all $n \in \mathbb{N}$.
3. The Agarwal iteration (Agarwal et al. 2007) (S-iteration):

$$\begin{aligned} x_{n+1} &= (1-\alpha_n)T(x_n) + \alpha_n T(y_n), \\ y_n &= (1-\beta_n)x_n + \beta_n T(x_n),\ n = 0, 1, 2, \ldots, \end{aligned} \quad (11)$$

where $\alpha_n \in [0, 1]$ and $\beta_n \in [0, 1]$ for all $n \in \mathbb{N}$.

Let us notice that the Mann iteration for $\alpha_n = 1$ reduces to the Picard iteration, the Ishikawa iteration reduces to the Mann iteration when $\beta_n = 0$ and to the Picard iteration when $\alpha_n = 1$, $\beta_n = 0$, and the S-iteration reduces to the Picard iteration when $\alpha_n = 0$, or $\alpha_n = 1$ and $\beta_n = 0$. A review of various iteration processes and their dependencies can be found in Gdawiec and Kotarski (2017).

All the presented iterations used only one mapping, but in the fixed-point theory exist iterations that use several mappings and are used to find common fixed points of the mappings. Let us recall some of them.

1. The Das–Debata iteration (Das and Debata 1986):

$$\begin{aligned} x_{n+1} &= (1-\alpha_n)x_n + \alpha_n T_2(y_n), \\ y_n &= (1-\beta_n)x_n + \beta_n T_1(x_n),\ n = 0, 1, 2, \ldots, \end{aligned} \quad (12)$$

where $\alpha_n \in (0, 1]$ and $\beta_n \in [0, 1]$ for all $n \in \mathbb{N}$.
2. The Khan–Cho–Abbas iteration (Khan et al. 2011):

$$\begin{aligned} x_{n+1} &= (1-\alpha_n)T_1(x_n) + \alpha_n T_2(y_n), \\ y_n &= (1-\beta_n)x_n + \beta_n T_1(x_n),\ n = 0, 1, 2, \ldots, \end{aligned} \quad (13)$$

where $\alpha_n \in (0, 1]$ and $\beta_n \in [0, 1]$ for all $n \in \mathbb{N}$.
3. The generalized Agarwal iteration (Khan et al. 2011):

$$\begin{aligned} x_{n+1} &= (1-\alpha_n)T_3(x_n) + \alpha_n T_2(y_n), \\ y_n &= (1-\beta_n)x_n + \beta_n T_1(x_n),\ n = 0, 1, 2, \ldots, \end{aligned} \quad (14)$$

where $\alpha_n \in (0, 1]$ and $\beta_n \in [0, 1]$ for all $n \in \mathbb{N}$.

Let us notice that the Das–Debata iteration for $T_1 = T_2$ reduces to the Ishikawa iteration, the Khan–Cho–Abbas iteration reduces to the Agarwal iteration when $T_1 = T_2$, and the generalized Agarwal iteration reduces to the Khan–Cho–Abbas iteration when $T_1 = T_3$ and to the Agarwal iteration when $T_1 = T_2 = T_3$.

Having such a variety of different iteration processes, we can use them in our algorithm. In the iterations with a single mapping we use (7) as the mapping, and in the case of iteration with several mappings we use also (7), but with different values of ω and η parameters for each of the mappings.

4 Visualization of the dynamics

To visualize the dynamics of the proposed algorithm, we will use method that is very similar to the method used in polynomiography (Gdawiec et al. 2015). Because of this similarity, we will call the obtained images polynomiographs, although our algorithm finds roots of arbitrary functions and not only of polynomials as in polynomiography.

In the algorithm, we choose iteration method, e.g. one of the methods presented in Sect. 3, parameters ω, η for a single mapping T or $\omega_1, \omega_2, \omega_3$ and η_1, η_2, η_3 for T_1, T_2, T_3 (depending on the chosen iteration). Moreover, we choose the maximum number of iterations m which the algorithm should realize, the accuracy of the computations r and a colouring function $C : \mathbb{N} \to \{0, 1, \ldots, 255\}^3$. Then, for each x_0 in the solution space $\mathbf{A}$ we use our algorithm. The iterations of the algorithm proceed till the convergence criterion:

$$|x_{n+1} - x_n| < r \quad (15)$$

is satisfied or the maximum number of iterations is reached. When the algorithm ends, we assign a colour to x_0 using colouring function C and the number of performed iterations.

The pseudocode of the visualization algorithm is presented in Algorithm 1. In the algorithm, the selected iteration method is denoted as I_q, where q is a vector of parameters of the iteration.

The solution space $\mathbf{A}$ is contained in a D-dimensional space, so Algorithm 1 will return visualization in this space. If $D = 2$, then we obtain a single image that can be easily presented on a screen. When $D > 2$, then it is hard to visualize the result on a 2D surface of the screen. In this case, we make cross section of $\mathbf{A}$ with a two-dimensional plane and visualize this cross section.

5 Discussion on the research results

In this section, we present and discuss the results of visualizing the dynamics of the method introduced in Sect. 2 together with the various iteration methods presented in Sect. 3. In our study, we used functions $f_i : \mathbb{R}^2 \to \mathbb{R}$ for $i \in \{1, 2, 3, 4\}$ given by the following formulas:

Algorithm 1: Visualization of the dynamics

Input: f – function, $\mathbf{A} \subset \mathbb{R}^D$ – solution space, m – the maximum number of iterations, I_q – iteration method, $q \in [0, 1]^N$ – parameters of the iteration I_q, ω, ω_1, ω_2, ω_3, η, η_1, η_1, η_2, η_3 – parameters defining functions T, T_1, T_2, T_3, C – colouring function, r – accuracy

Output: visualization of the dynamics

```
foreach x_0 ∈ A do
    i = 0
    v_0 = [0, 0, ..., 0]
    while i ≤ m do
        x_{n+1} = I_q(x_n)
        if |x_{n+1} − x_n| < r then
            break
        i = i + 1
    colour x_0 with C(i)
```

$$f_1(x, y) = \sqrt{(-x^3 + 3xy^2 + 1)^2 + (y^3 - 3x^2y)^2}, \tag{16}$$

the function f_1 has three roots, namely $[1, 0]$, $[-0.5, -0.866025]$, $[-0.5, 0.866025]$;

$$f_2(x, y) = \sqrt{\begin{array}{r}(x^4 - 6x^2y^2 + y^4 - 10x^2 + 10y^2 + 9)^2 + \\ + (4x^3y - 4xy^3 - 20xy)^2\end{array}}, \tag{17}$$

the function f_2 has four roots, namely $[-3, 0]$, $[-1, 0]$, $[1, 0]$, $[3, 0]$;

$$f_3(x, y) = \sqrt{(x^5 - 10x^3y^2 + 5xy^4 - x)^2 + (5x^4y - 10x^2y^3 + y^5 - y)^2}, \tag{18}$$

the function f_3 has five roots, namely $[-1, 0]$, $[0, -1]$, $[0, 0]$, $[0, 1]$, $[1, 0]$;

$$f_4(x, y) = \sqrt{\begin{array}{r}(x^6 - 15x^4y^2 + 15x^2y^4 - y^6 + 10x^3 - 30xy^2 - 8)^2 + \\ + (6x^5y - 20x^3y^3 + 6xy^5 + 30x^2y - 10y^3)^2\end{array}}, \tag{19}$$

the function f_4 has six roots, namely $[-2.207, 0]$, $[-0.453, -0.785]$, $[-0.453, 0.785]$, $[0.906, 0]$, $[1.103, -1.911]$, $[1.103, 1.911]$.

Because $\mathbb{R}^2$ is isomorphic with $\mathbb{C}$ and for each $[x, y] \in \mathbb{R}^2$ we have $x + iy = z \in \mathbb{C}$, then the functions f_i for $i \in \{1, 2, 3, 4\}$ can be written in the following form:

$$f_1(z) = |z^3 - 1|, \tag{20}$$

$$f_2(z) = |z^4 - 10z^2 + 9|, \tag{21}$$

$$f_3(z) = |z^5 - z|, \tag{22}$$

$$f_4(z) = |z^6 + 10z^3 - 8|. \tag{23}$$

Fig. 1 Colour map used in the experiments

The functions (20)–(23) have the same roots as the following functions:

$$g_1(z) = z^3 - 1, \tag{24}$$

$$g_2(z) = z^4 - 10z^2 + 9, \tag{25}$$

$$g_3(z) = z^5 - z, \tag{26}$$

$$g_4(z) = z^6 + 10z^3 - 8. \tag{27}$$

Similar approach, to the one presented in this paper, for function $z^3 - 1$ for which visualization of the dynamics using Mann and Ishikawa iteration in the Newton's method can be found in Kotarski et al. (2012).

In all experiments, the same colour map was used to colour the obtained images. The colour map is presented in Fig. 1. The colour map has 256 colours (ordered from left to right). The colour number represents the number of iterations needed to reach the solution. Moreover, in every experiment the following parameters were used: $\tau = 1.0e{-}3$, $m = 256$, $r = 1.0e{-}2$, image resolution 800×800 pixels. The solution spaces depend on the function and are the following: $\mathbf{A}_1 = \mathbf{A}_3 = [-2.0, 2.0]^2$, $\mathbf{A}_2 = [-4.0, 4.0] \times [-2.0, 2.0]$, $\mathbf{A}_4 = [-2.3, 1.7] \times [-2.0, 2.0]$.

The software used in the research was implemented in the C++ programming language. The experiments were realized on a computer with the Intel Core i5-2520M processor, 4 GB RAM, and Linux 3.16.7-42-desktop openSUSE 13.2 (Harlequin 64-bits, KDE Platform version 4.14.9).

In all experiments, the polynomiographs are generated using Algorithm 1 in 2D space. Thus, looking at the generated images we can read two important characteristics. The first one is the speed of convergence of the algorithm, i.e. the colour of each point gives us information on how many iterations were performed by the algorithm to reach the root. The second characteristic is the dynamics of the algorithm. Low dynamics is in areas where the variation of colours is small, whereas in areas with a large variation of colours the dynamics is high.

5.1 The Picard iteration

The behaviour of particles is affected by the speed and inertia. They are controlled by two parameters: the acceleration constant (η) and the inertia weight (ω) for the described above benchmark functions. The visualization of method's dynamics allows to analyse their impact on the algorithm's work.

We start the experiments with the f_1 function given by (16). This function has three roots that, thanks to the symme-

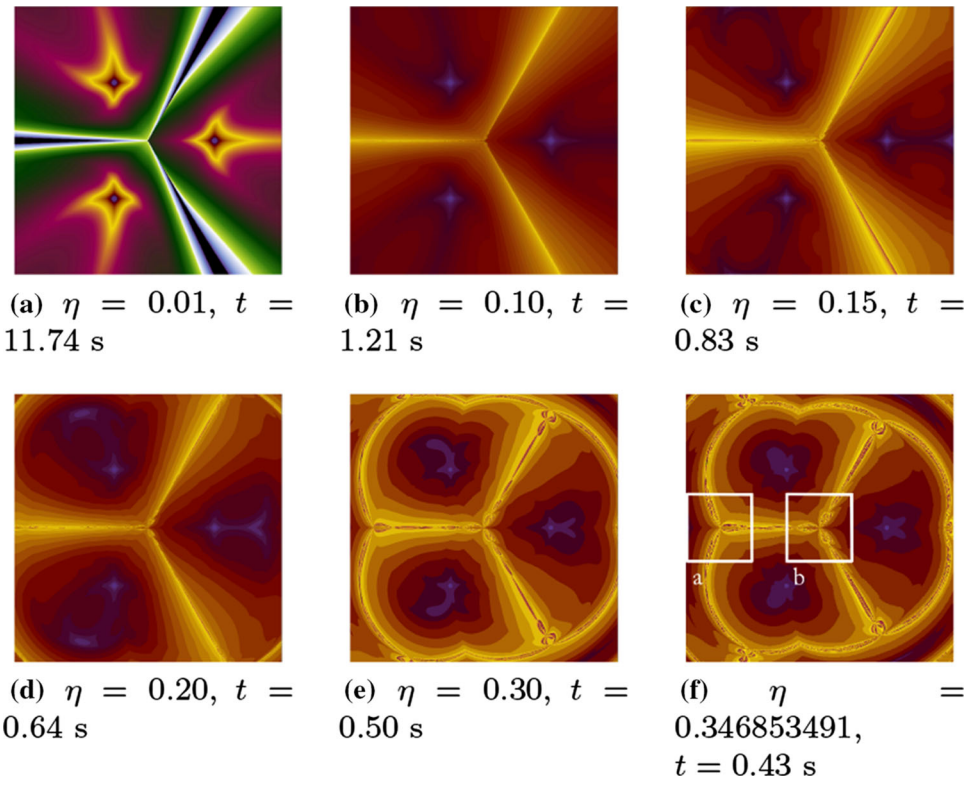


Fig. 2 Polynomiographs of f_1 and the Picard iteration for $\omega = 0$ and varying η

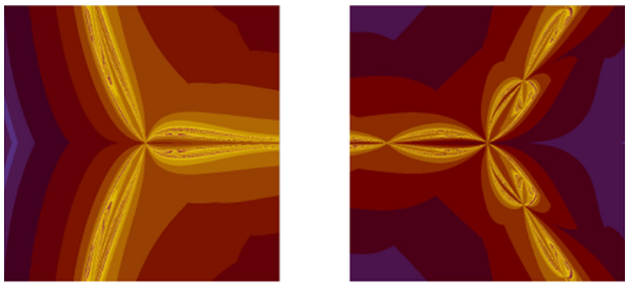
(a) $\eta = 0.01$, $t = 11.74$ s **(b)** $\eta = 0.10$, $t = 1.21$ s **(c)** $\eta = 0.15$, $t = 0.83$ s

(d) $\eta = 0.20$, $t = 0.64$ s **(e)** $\eta = 0.30$, $t = 0.50$ s **(f)** $\eta = 0.346853491$, $t = 0.43$ s

try similar areas, create the corresponding dynamics of the particle. As was already mentioned, the parameters of the examples have been chosen so that in addition to the visualization of particle's dynamics, the polynomiograph also represents (according to the authors) an aesthetic image.

In Fig. 2 visualizations of the dynamics for the f_1 and the Picard iteration using $\omega = 0$ and varying η are presented. The acceleration constant is the only factor influencing a particle motion—it affects its speed. (The larger the value of η, the greater the speed v_{n+1}.) Moreover, the generation time of each image is shown in Fig. 2. Looking at the results, we see that the increase in particle's velocity can shorten the creation time of the image. The black colour shows places which have not reached a solution in the given number of 256 iterations—Fig. 2a. The areas of the same colour indicate the same number of iterations required to determine the $f(x_n) = 0$—they look similar to the contour lines on the map (Fig. 2f). The magnifications of the marked areas in Fig. 2f are presented in Fig. 3.

Inertia helps the particle to escape from a not promising area of solution space. Figure 4 presents examples of visualizations obtained for f_1 and the Picard iteration using rather high inertia, i.e. $\omega = 0.7$ and varying η, and also their generation times. The shortest time of the creation of polynomiographs in Fig. 4 is obtained by the polynomiograph from Fig. 4c. Both too low and too high velocities may cause the increase in the time of polynomiograph creation.



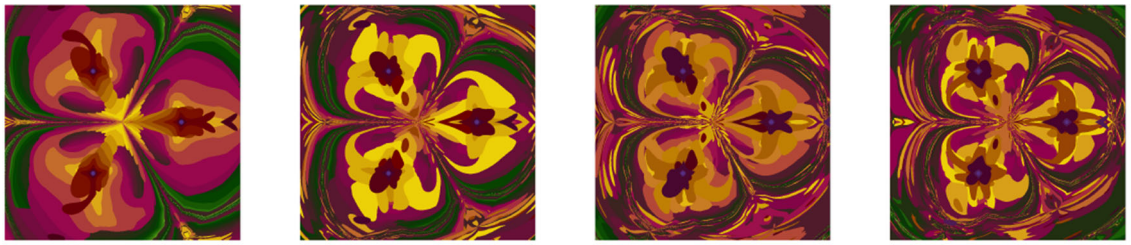
(a) **(b)**

Fig. 3 Magnification of the marked areas in Fig. 2f

Figure 5 presents examples of visualizations obtained for f_1 and the Picard iteration using $\eta = 0.025$ and varying ω, and also their generation times. From the results, it is possible to observe that the higher the value of ω, the lower the generation time—the minimal value of time is obtained by polynomiograph in Fig. 5c. In this case, both too low and too high inertia may cause the extension of the time.

Figures 6 and 7 show examples of visualizations obtained for $\eta = 0.1$ and $\eta = 0.3$ (respectively) and various values of ω for function f_1. Looking at the figures, we see that for the higher velocity of particle (acceleration constant increases in the following figures) the high value of inertia weight causes the increase in the creation time of the image.

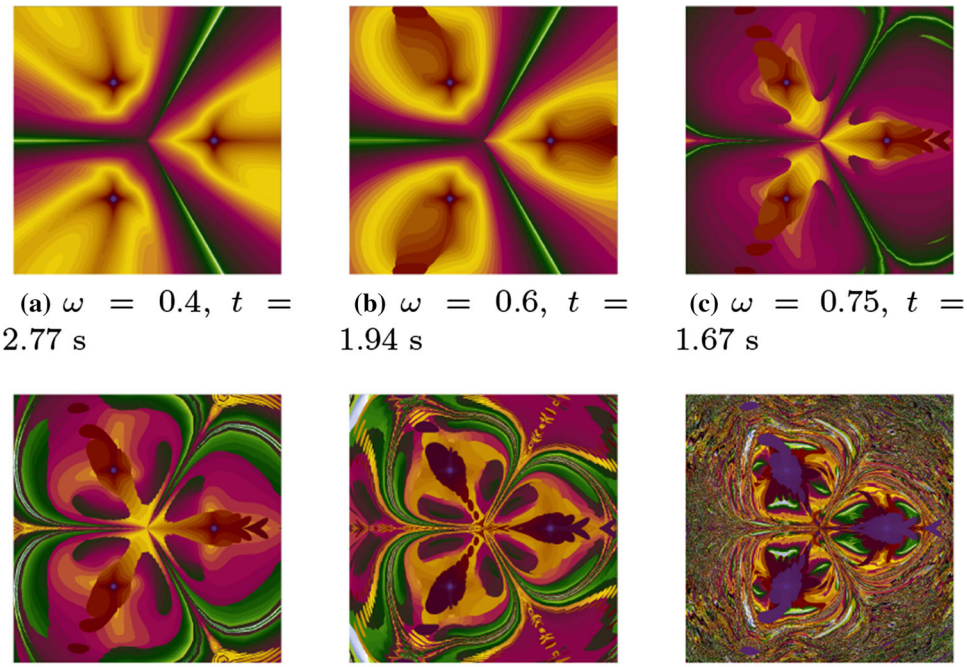
From all the images obtained for f_1 and the Picard iteration, we see that the dynamics of the method changes in



(a) $\eta = 0.05$, $t =$ 1.33 s (b) $\eta = 0.1$, $t =$ 1.18 s (c) $\eta = 0.15$, $t =$ 1.15 s (d) $\eta = 0.2$, $t =$ 1.20 s

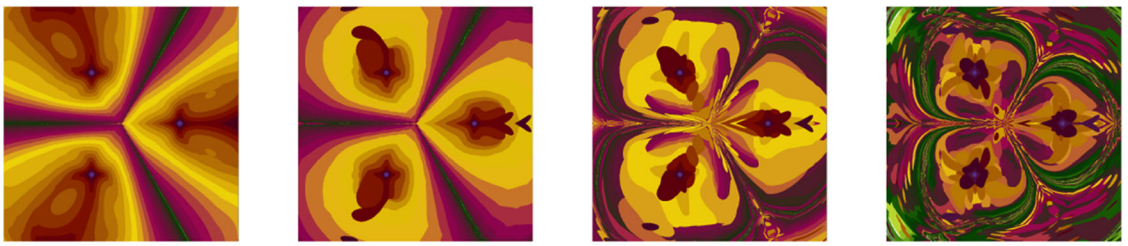
Fig. 4 Polynomiographs of f_1 and the Picard iteration for $\omega = 0.7$ and varying η

Fig. 5 Polynomiographs of f_1 and the Picard iteration for $\eta = 0.025$ and varying ω



(a) $\omega = 0.4$, $t =$ 2.77 s (b) $\omega = 0.6$, $t =$ 1.94 s (c) $\omega = 0.75$, $t =$ 1.67 s

(d) $\omega = 0.8$, $t =$ 1.95 s (e) $\omega = 0.9$, $t =$ 3.09 s (f) $\omega = 0.98$, $t =$ 9.92 s

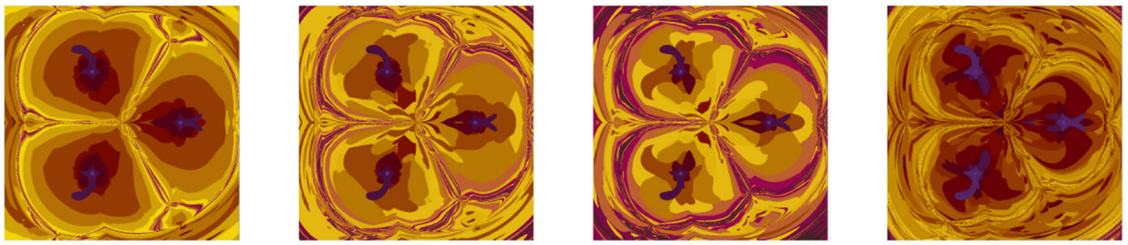


(a) $\omega = 0.2$, $t =$ 0.97 s (b) $\omega = 0.4$, $t =$ 0.85 s (c) $\omega = 0.6$, $t =$ 1.07 s (d) $\omega = 0.8$, $t =$ 1.67 s

Fig. 6 Polynomiographs of f_1 and the Picard iteration for $\eta = 0.1$ and varying ω

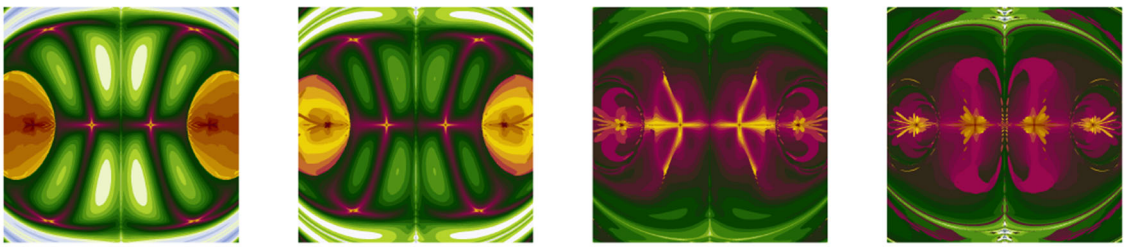
a significant way when we change two parameters of the method (ω, η). A smooth image represents low dynamics of the particles. Moreover, we see that using the proposed method we obtain interesting patterns and more diverse patterns are observed for particles with greater dynamics.

Polynomiographs of the Picard iteration for the f_2 function are shown in Fig. 8. Two groups of areas forming a different dynamics are clearly visible in Fig. 8a. The proper value of ω is responsible for creating a particle dynamics balance—it is evident in Fig. 8c. In this case, we observe a



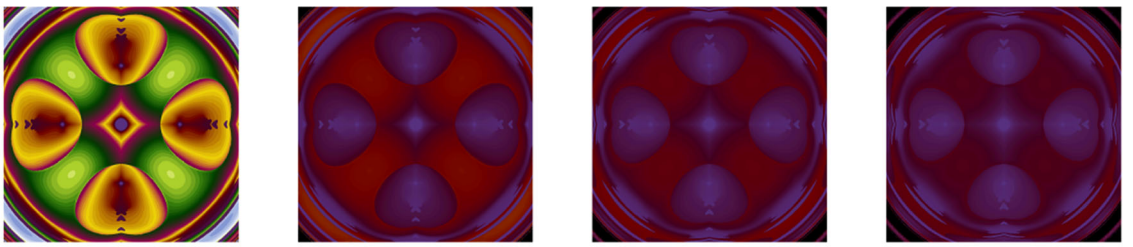
(a) $\omega = 0.2$, $t = 0.48$ s **(b)** $\omega = 0.3$, $t = 0.60$ s **(c)** $\omega = 0.4$, $t = 0.68$ s **(d)** $\omega = 0.5$, $t = 0.81$ s

Fig. 7 Polynomiographs of f_1 and the Picard iteration for $\eta = 0.3$ and varying ω



(a) $\omega = 0.2$, $t = 2.66$ s **(b)** $\omega = 0.3$, $t = 2.29$ s **(c)** $\omega = 0.45$, $t = 1.80$ s **(d)** $\omega = 0.55$, $t = 2.13$ s

Fig. 8 Polynomiographs of f_2 and the Picard iteration for $\eta = 0.01$ and varying ω



(a) $\eta = 0.3$, $t = 2.94$ s **(b)** $\eta = 0.4$, $t = 2.50$ s **(c)** $\eta = 0.5$, $t = 2.49$ s **(d)** $\eta = 0.6$, $t = 2.75$ s

Fig. 9 Polynomiographs of f_3 and the Picard iteration for $\omega = 0.5$ and varying η

reduction in the time of the polynomiograph creation. However, an excessive increase in the inertia coefficient extends this time.

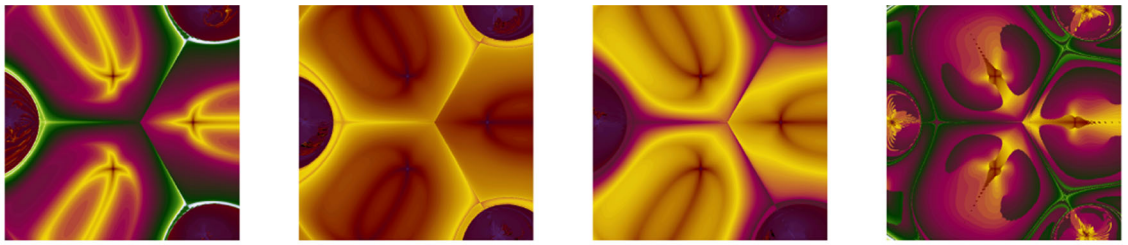
The next figure—Fig. 9—shows polynomiographs of the Picard iteration for the f_3 function. The function f_3, thanks to the symmetry of roots placement, has four areas that create similar particle dynamics. The increase in the value of the acceleration constant (the particle moves faster—it increases in its dynamics) can shorten the time of creation of the polynomiograph. The dynamics of the particle is limited in the whole area of the polynomiograph—it is shown in Fig. 8b–d. However, areas from which the particle does not reach the solution (black colour) arise.

Two groups of areas with different dynamics are visible in Fig. 10. The images in this figure were created using the f_4 function and the Picard iteration. Increasing the value of the inertia weight increases the dynamics of the particle—it shortens the time of creation of the polynomiograph (Fig. 10a, b). The excessive value of inertia weight can be compensated by a decrease in the value of acceleration constant—consequently, similar particle dynamics is obtained in the whole area (Fig. 10c, d).

5.2 The Mann iteration

The Mann iteration is enhanced by the α parameter in relation to the Picard iteration. It gives the possibility to adjust the dynamic of the particle behaviour independently of the ω and η.

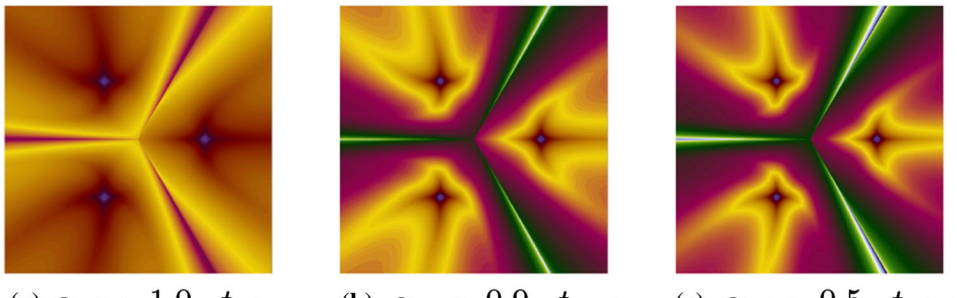
Figure 11 presents the dynamics of the method for $\omega = 0$, $\eta = 0.025$ using the Mann iteration and f_1. For $\alpha = 1$, we obtain the Picard iteration (Fig. 11a). From the images, we see that the decrease in the value of the parameter α, which is responsible for a linear combination in the Mann itera-



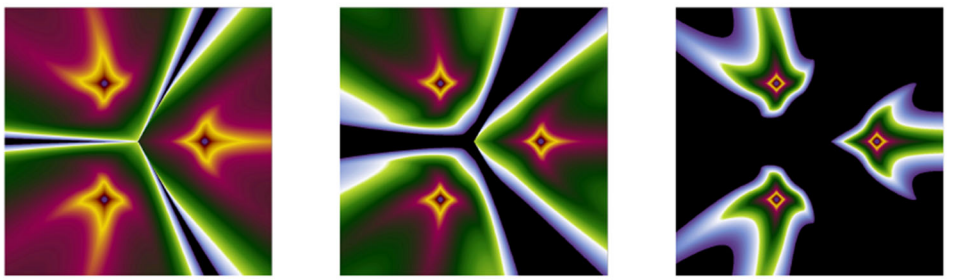
(a) $\omega = 0.2$, $\eta = 0.3$, $t = 9.23$ s **(b)** $\omega = 0.4$, $\eta = 0.3$, $t = 7.12$ s **(c)** $\omega = 0.4$, $\eta = 0.2$, $t = 10.29$ s **(d)** $\omega = 0.8$, $\eta = 0.1$, $t = 7.69$ s

Fig. 10 Polynomiographs of f_4 and the Picard iteration for varying ω and η

Fig. 11 Polynomiographs of f_1 and the Mann iteration for $\omega = 0$, $\eta = 0.025$ and varying α



(a) $\alpha = 1.0$, $t = 4.83$ s **(b)** $\alpha = 0.9$, $t = 5.32$ s **(c)** $\alpha = 0.5$, $t = 9.48$ s



(d) $\alpha = 0.4$, $t = 11.91$ s **(e)** $\alpha = 0.3$, $t = 15.49$ s **(f)** $\alpha = 0.2$, $t = 19.16$ s

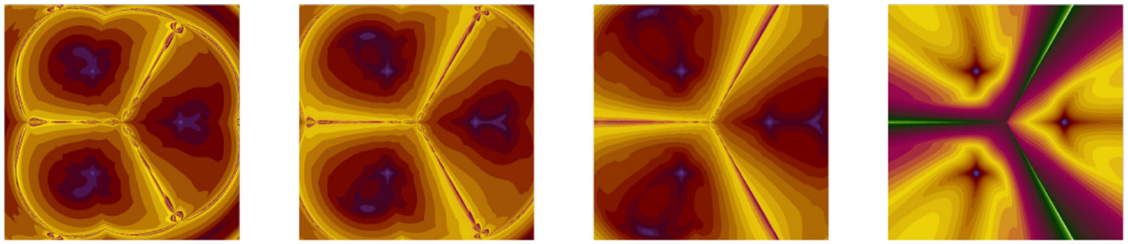
tion, reduces the dynamics of particles. We can also observe that reducing the dynamics of particles can cause the lack of finding a solution (black colour), even for large areas of the solution space. Moreover, we see that the lower the value of α, the greater the generation time.

The acceleration constant (η) affects particle dynamics. Using a wide range of changes in $\alpha = 0.2, 0.5, 0.7, 0.9$ (which limits the particle dynamics) in Fig. 12, there are no areas where the algorithm does not reach the solution. It is a result of increased particle dynamics by increasing the acceleration constant ($\eta = 0.35$).

Relatively large dynamics of particles for f_1 is presented in Figs. 13 and 14 obtained for different values of parameters ω and η defining T and the α parameter in the Mann iteration. From the images, we can observe that the decrease in the value of the parameter α, which reduces the proportion of the part specified by T in the Mann iteration, reduces the dynamic of the particle. The high particle dynamics allows to obtain interesting patterns.

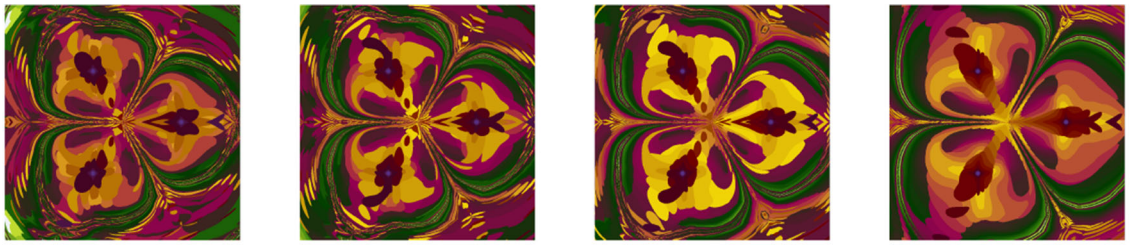
Significantly lower particles dynamics is presented in Fig. 15. The polynomiographs in this figure were obtained for f_1, $\omega = 0.3$, $\eta = 0.3$ and varying α in the Mann iteration. These images are similar to the images obtained for the Picard iteration. The decrease in particle's dynamics smooths out the images—it is also visible in the remaining images. Figure 15a, b and Fig. 15c, d—although different in particle's dynamics—are similar in the colouring.

Tuning the algorithm is a multidimensional problem. It means that an increase in the number of parameters complicates this task. The polynomiographs from Fig. 16 (generated for the f_2 function) show the particle dynamics constituting the intersection of the algorithm tuning space for the α. The increase in the α parameter value increases the dynamics of the particles. Properly selected value of this parameter



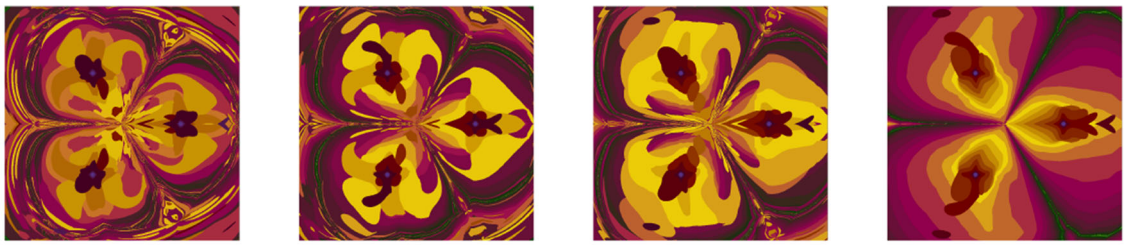
(a) $\alpha = 0.9$, $t = 0.48$ s (b) $\alpha = 0.7$, $t = 0.58$ s (c) $\alpha = 0.5$, $t = 0.71$ s (d) $\alpha = 0.2$, $t = 1.80$ s

Fig. 12 Polynomiographs of f_1 and the Mann iteration for $\omega = 0$, $\eta = 0.35$ and varying α



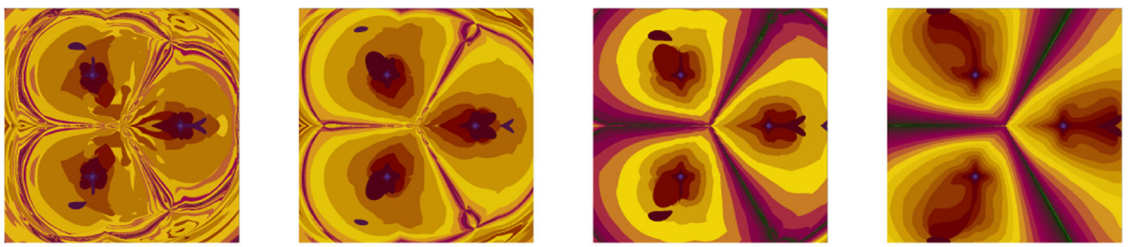
(a) $\alpha = 0.9$, $t = 1.70$ s (b) $\alpha = 0.7$, $t = 1.71$ s (c) $\alpha = 0.5$, $t = 1.76$ s (d) $\alpha = 0.3$, $t = 1.92$ s

Fig. 13 Polynomiographs of f_1 and the Mann iteration for $\omega = 0.8$, $\eta = 0.1$ and varying α



(a) $\alpha = 0.9$, $t = 0.91$ s (b) $\alpha = 0.7$, $t = 0.95$ s (c) $\alpha = 0.5$, $t = 0.99$ s (d) $\alpha = 0.3$, $t = 1.12$ s

Fig. 14 Polynomiographs of f_1 and the Mann iteration for $\omega = 0.6$, $\eta = 0.2$ and varying α

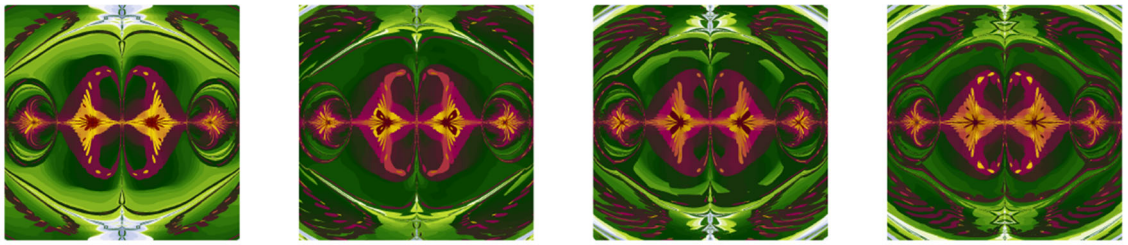


(a) $\alpha = 0.9$, $t = 0.58$ s (b) $\alpha = 0.7$, $t = 0.64$ s (c) $\alpha = 0.5$, $t = 0.68$ s (d) $\alpha = 0.3$, $t = 0.91$ s

Fig. 15 Polynomiographs of f_1 and the Mann iteration for $\omega = 0.3$, $\eta = 0.3$ and varying α

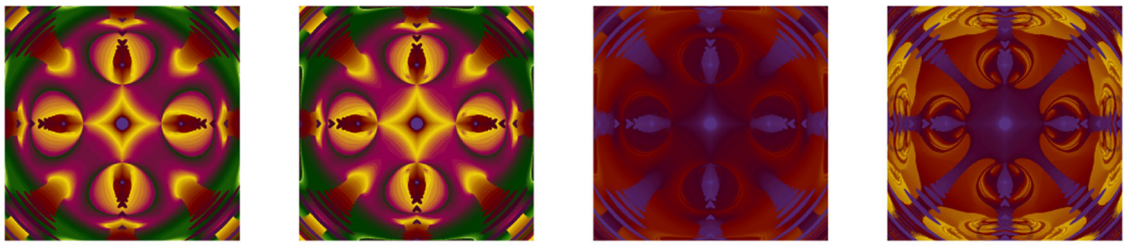
(Fig. 16c) minimizes the time of the polynomiograph creation.

As in the previous example, in Fig. 17 polynomiographs generated using a varying α parameter are presented, but this time the f_3 function was used. A limitation of the dynamic range observed in Fig. 17c occurs for the shortest time of a polynomiograph creation. Moreover, we can observe that the change in the α parameter by 0.1 affects the dynamics in different way. In Fig. 17a, b the dynamics is almost the



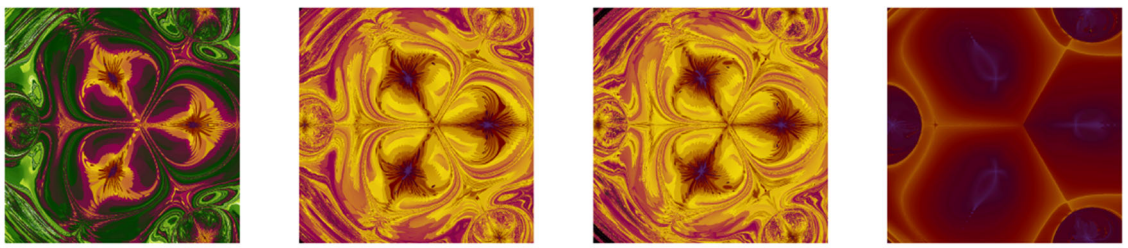
(a) $\alpha = 0.4$, $t = 4.93$ s **(b)** $\alpha = 0.5$, $t = 4.74$ s **(c)** $\alpha = 0.6$, $t = 4.69$ s **(d)** $\alpha = 0.7$, $t = 4.85$ s

Fig. 16 Polynomiographs of f_2 and the Mann iteration for $\omega = 0.8$, $\eta = 0.01$ and varying α



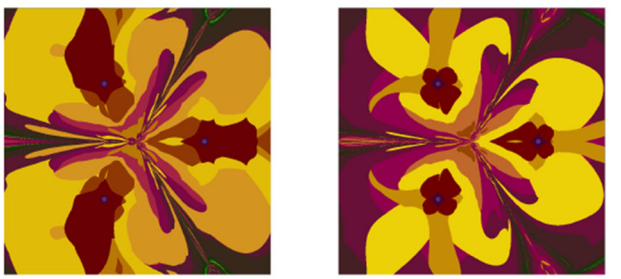
(a) $\alpha = 0.6$, $t = 3.01$ s **(b)** $\alpha = 0.7$, $t = 2.81$ s **(c)** $\alpha = 0.8$, $t = 2.66$ s **(d)** $\alpha = 0.9$, $t = 3.66$ s

Fig. 17 Polynomiographs of f_3 and the Mann iteration for $\omega = 0.8$, $\eta = 0.025$ and varying α



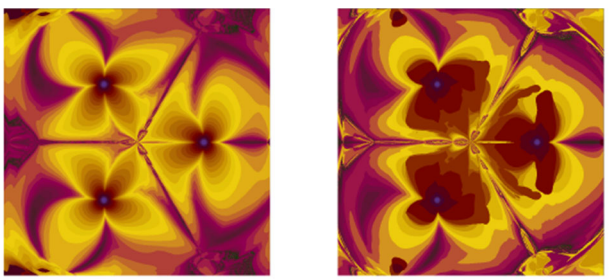
(a) $\omega = 0.9$, $\eta = 0.02$, $t = 13.06$ s **(b)** $\omega = 0.9$, $\eta = 0.03$, $t = 12.17$ s **(c)** $\omega = 0.9$, $\eta = 0.04$, $t = 12.31$ s **(d)** $\omega = 0.5$, $\eta = 0.04$, $t = 4.43$ s

Fig. 18 Polynomiographs of f_4 and the Mann iteration for $\alpha = 0.1$ and varying η and ω



(a) $\alpha = 0.6$, $t = 0.95$ s **(b)** $\alpha = 0.8$, $t = 1.03$ s

Fig. 19 Polynomiographs of f_1 and the Ishikawa iteration for $\beta = 0.4$, $\omega = 0.5$, $\eta = 0.3$ and varying α

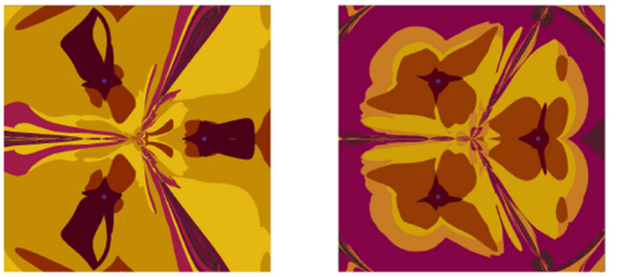


(a) $\alpha = 0.2$, $t = 2.86$ s **(b)** $\alpha = 0.4$, $t = 1.62$ s

Fig. 20 Polynomiographs of f_1 and the Ishikawa iteration for $\beta = 0.8$, $\omega = 0.5$, $\eta = 0.3$ and varying α

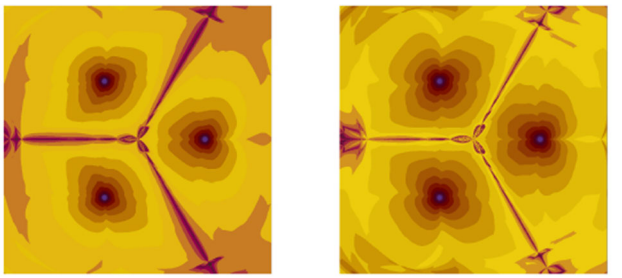
same, whereas in Fig. 17b, c and in Fig. 17c, d the change is significant.

Examples of polynomiographs for the Mann iteration and function f_4 are presented in Fig. 18. The value of the α param-



(a) $\beta = 0.2$, $t = 0.57$ s (b) $\beta = 0.4$, $t = 0.67$ s

Fig. 21 Polynomiographs of f_1 and the Ishikawa iteration for $\alpha = 0.7$, $\omega = 0.2$, $\eta = 0.6$ and varying β



(a) $\beta = 0.7$, $t = 1.50$ s (b) $\beta = 0.8$, $t = 1.52$ s

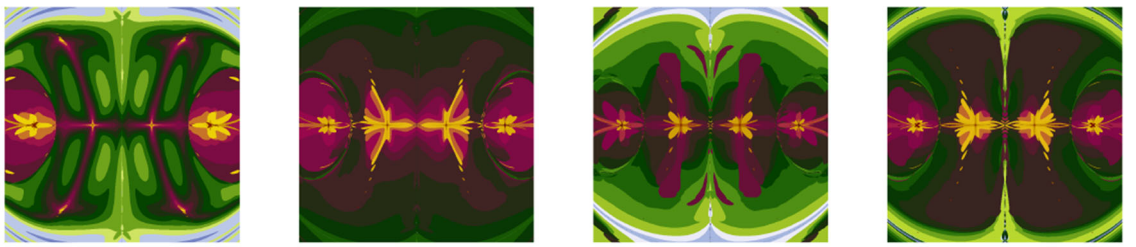
Fig. 22 Polynomiographs of f_1 and the Ishikawa iteration for $\alpha = 0.3$, $\omega = 0.2$, $\eta = 0.4$ and varying β

eter is small, which significantly limits the dynamics of the particle. The increase in the value of the acceleration constant slightly reduces the time of the polynomiograph creation—Fig. 18b. A significant reduction in the inertia weight causes a reduction in particle dynamics and reduces the time needed to the polynomiograph creation—it is visible in Fig. 18d.

5.3 The Ishikawa and the Das–Debata iterations

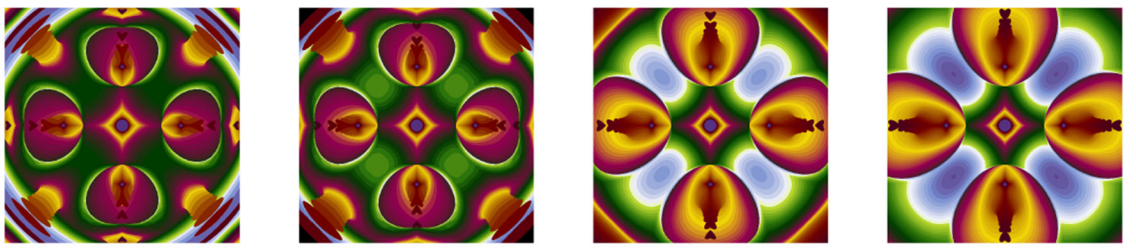
In the Ishikawa iteration, an additional parameter β is introduced in relation to the Mann iteration. This parameter, like the parameter α, has an impact on the dynamics of particles. It is also involved in the creation of an additional reference point (the sample in the solution space). In this way, particle motion is a linear combination of its position and a movement of reference point. This gives the possibility to define two different operators (T_1 and T_2) for transforming the current position of the particle and the position of the reference point—the Das–Debata iteration. For the differentiation of these operators, the parameters ω_1, ω_2 and η_1 and η_2 are responsible.

Two types of iterations—Ishikawa and Das–Debata—give a wide range of possibilities for the regulation of the algorithm's work. The use of these iterations causes twisting contour lines. The α parameter, as in the case of the Mann iteration, limits the dynamics of a particle. A similar function as the parameter α is realized by the parameter β in relation to the reference point.



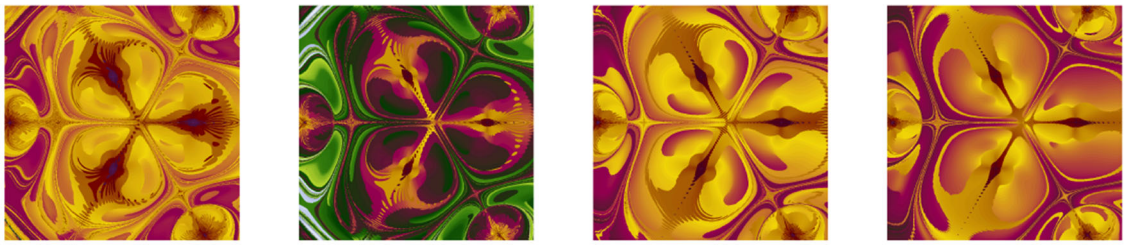
(a) $\omega = 0.3$, $\alpha = 0.9$, $t = 2.43$ s (b) $\omega = 0.4$, $\alpha = 0.9$, $t = 2.31$ s (c) $\omega = 0.5$, $\alpha = 0.9$, $t = 2.87$ s (d) $\omega = 0.5$, $\alpha = 0.7$, $t = 2.83$ s

Fig. 23 Polynomiographs of f_2 and the Ishikawa iteration for $\beta = 0.9$, $\eta = 0.01$ and varying α, ω



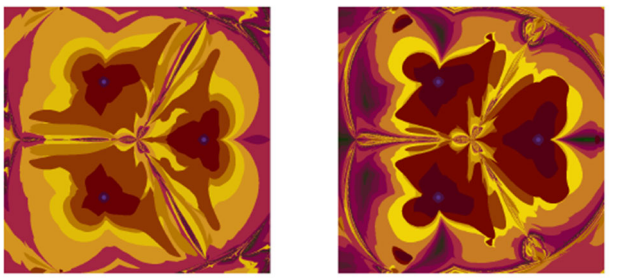
(a) $\alpha = 0.9$, $\beta = 0.9$, $t = 3.55$ s (b) $\alpha = 0.9$, $\beta = 0.4$, $t = 4.01$ s (c) $\alpha = 0.4$, $\beta = 0.9$, $t = 6.43$ s (d) $\alpha = 0.4$, $\beta = 0.4$, $t = 9.66$ s

Fig. 24 Polynomiographs of f_3 and the Ishikawa iteration for $\omega = 0.7$, $\eta = 0.02$ and varying α, β



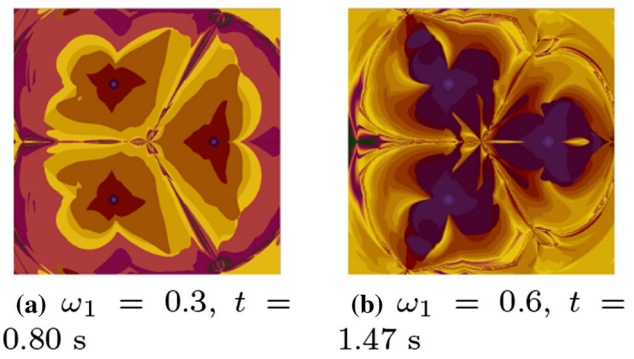
(a) $\alpha = 0.9$, $\beta = 0.9$, $t = 21.12$ s **(b)** $\alpha = 0.9$, $\beta = 0.4$, $t = 22.17$ s **(c)** $\alpha = 0.4$, $\beta = 0.9$, $t = 22.28$ s **(d)** $\alpha = 0.4$, $\beta = 0.4$, $t = 24.67$ s

Fig. 25 Polynomiographs of f_4 and the Ishikawa iteration for $\omega = 0.9$, $\eta = 0.0007$ and varying α, β



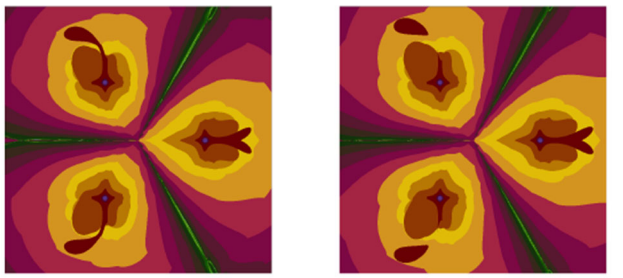
(a) $\omega_1 = 0.2$, $t = 0.85$ s **(b)** $\omega_1 = 0.4$, $t = 1.01$ s

Fig. 26 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.6$, $\beta = 0.7$, $\eta_1 = 0.4$, $\omega_2 = 0.4$, $\eta_2 = 0.4$ and varying ω_1



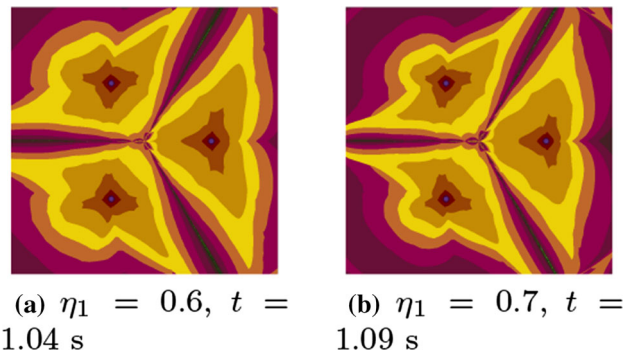
(a) $\omega_1 = 0.3$, $t = 0.80$ s **(b)** $\omega_1 = 0.6$, $t = 1.47$ s

Fig. 27 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.6$, $\beta = 0.5$, $\eta_1 = 0.5$, $\omega_2 = 0.3$, $\eta_2 = 0.4$ and varying ω_1



(a) $\eta_1 = 0.2$, $t = 1.22$ s **(b)** $\eta_1 = 0.3$, $t = 1.17$ s

Fig. 28 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.4$, $\beta = 0.1$, $\omega_1 = 0.3$, $\omega_2 = 0.3$, $\eta_2 = 0.4$ and varying η_1



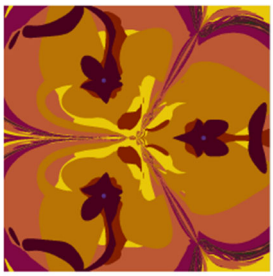
(a) $\eta_1 = 0.6$, $t = 1.04$ s **(b)** $\eta_1 = 0.7$, $t = 1.09$ s

Fig. 29 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.4$, $\beta = 0.3$, $\omega_1 = 0.2$, $\omega_2 = 0.4$, $\eta_2 = 0.4$ and varying η_1

Figures 19 and 20 show the reduction of the dynamics of the particle for f_1 by the α parameter in the Ishikawa iteration. A small value of β parameter causes that the reference point is created close to the active point. The same value of the α parameter is the cause of creation of similar patterns.

The reduction of the dynamics of the reference point creation for f_1 by the α parameter in the Ishikawa iteration is shown in Figs. 21 and 22. A significant decrease in the dynamics of the reference point creation weakens the effect of twisting lines—it is also visible in the direction to the centre of the pattern. For large values of the β parameter, major changes occur on the periphery of the image—Fig. 22.

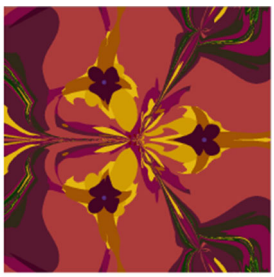
Polynomiographs presented in Fig. 23 for the f_2 function and the Ishikawa iteration show particle dynamics for changing ω and α parameters. Comparing Fig. 23a, b, an increase in dynamics due to an increase in the ω value is observed, which shortens the time of the polynomiograph creation. Then a further increase in dynamics due to an increase in the ω value results in a longer time of a polynomiograph creation—Fig. 23c. Reducing the dynamics of the particle through the appropriate selection of the α parameter reduces the time of the polynomiograph creation. This method of particle dynamics reduction is ineffective and may cause the particle not to reach a solution.



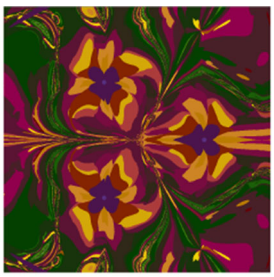
(a) $\omega_2 = 0.2$, $t = 0.66$ s

(b) $\omega_2 = 0.8$, $t = 1.70$ s

Fig. 30 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.9$, $\beta = 0.7$, $\omega_1 = 0.4$, $\eta_1 = 0.2$, $\eta_2 = 0.5$ and varying ω_2

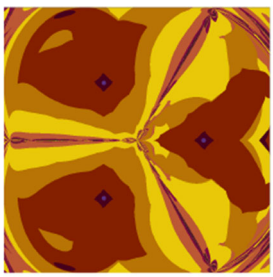


(a) $\omega_2 = 0.5$, $t = 1.05$ s



(b) $\omega_2 = 0.9$, $t = 1.92$ s

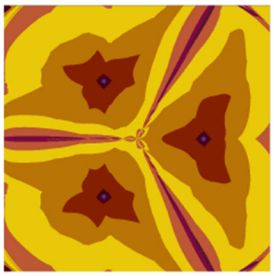
Fig. 31 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.7$, $\beta = 0.5$, $\omega_1 = 0.4$, $\eta_1 = 0.2$, $\eta_2 = 0.7$ and varying ω_2



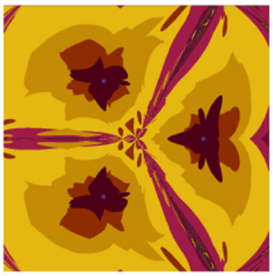
(a) $\eta_2 = 0.3$, $t = 0.56$ s

(b) $\eta_2 = 0.8$, $t = 0.65$ s

Fig. 32 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.8$, $\beta = 0.4$, $\omega_1 = 0.1$, $\eta_1 = 0.6$, $\omega_2 = 0.3$ and varying η_2

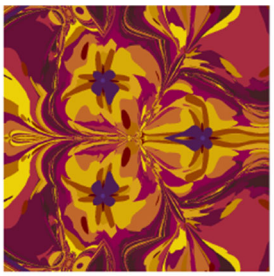


(a) $\eta_2 = 0.5$, $t = 0.64$ s

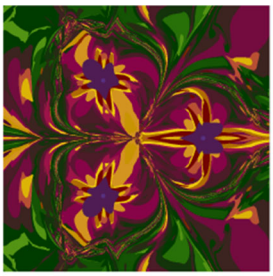


(b) $\eta_2 = 0.9$, $t = 0.62$ s

Fig. 33 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.6$, $\beta = 0.7$, $\omega_1 = 0.1$, $\eta_1 = 0.3$, $\omega_2 = 0.2$ and varying η_2

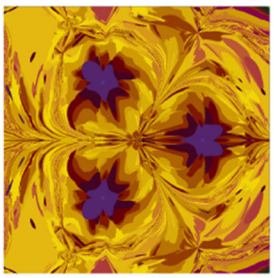


(a) $\alpha = 0.7$, $\beta = 0.5$, $\eta_2 = 0.8$, $t = 1.91$ s

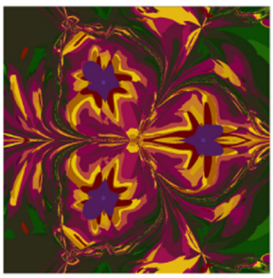


(b) $\alpha = 0.5$, $\beta = 0.4$, $\eta_2 = 0.9$, $t = 2.51$ s

Fig. 34 Polynomiographs of f_1 and the Das–Debata iteration for $\omega_1 = 0.4$, $\eta_1 = 0.2$, $\omega_2 = 0.9$



(a) $\omega_1 = 0.6$, $t = 2.47$ s



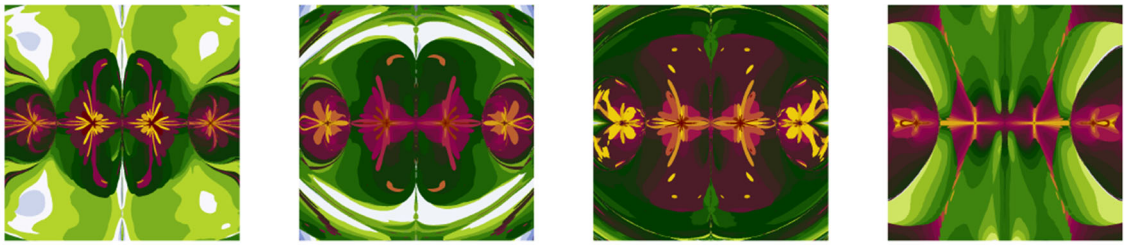
(b) $\omega_1 = 0.7$, $t = 2.43$ s

Fig. 35 Polynomiographs of f_1 and the Das–Debata iteration for $\alpha = 0.5$, $\beta = 0.4$, $\eta_1 = 0.2$, $\omega_2 = 0.9$, $\eta_2 = 0.9$ and varying ω_1

Figure 24 shows the changes in particle dynamics for the function f_3 under the influence of changes in the α and β parameters for the Ishikawa iteration. Comparing the times of polynomiographs creation in Fig. 24b, c with Fig. 24a or d, the effect of change in the α parameter on particle's dynamics is noticeably greater than that of β.

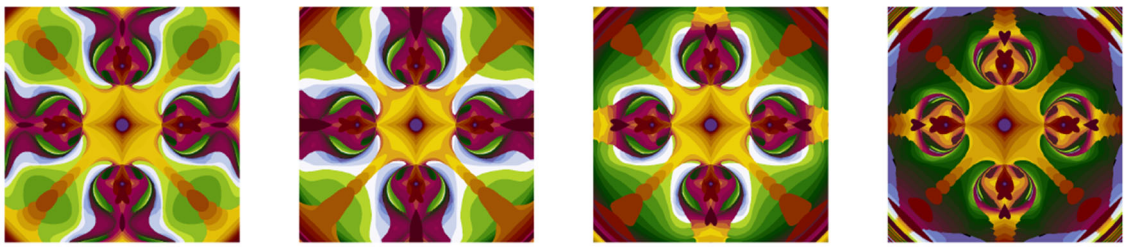
Polynomiographs obtained for the f_4 function using the Ishikawa iteration are presented in Fig. 25. The obtained results confirm the observations from the previous example (Fig. 24), i.e. the change in the α parameter has a much greater effect on the behaviour of the particle than the change in the β parameter.

Figures 26, 27, 28, 29, 30, 31, 32, 33, 34 and 35 present polynomiographs created using the Das–Debata iteration for the f_1 function. In this iteration, the ω_1 inertia weight is responsible for the dynamics of the reference point creation. Small differences of the dynamics of the reference point creation can cause the creation of similar patterns—Figs. 26 and 27. A similar effect is given by the change in the reference point dynamics by determining the η_1 parameter—see



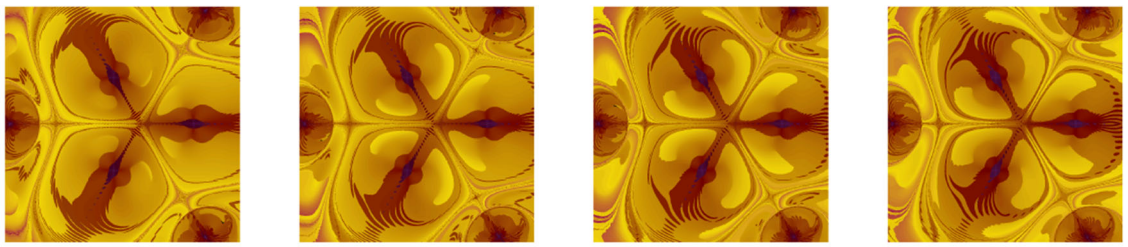
(a) $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.3$, $\eta_1 = 0.01$, $\omega_2 = 0.7$, $\eta_2 = 0.03$, $t = 4.73$ s **(b)** $\alpha = 0.8$, $\beta = 0.3$, $\omega_1 = 0.7$, $\eta_1 = 0.03$, $\omega_2 = 0.5$, $\eta_2 = 0.01$, $t = 3.71$ s **(c)** $\alpha = 0.8$, $\beta = 0.6$, $\omega_1 = 0.5$, $\eta_1 = 0.02$, $\omega_2 = 0.7$, $\eta_2 = 0.01$, $t = 3.47$ s **(d)** $\alpha = 0.3$, $\beta = 0.3$, $\omega_1 = 0.5$, $\eta_1 = 0.02$, $\omega_2 = 0.7$, $\eta_2 = 0.01$, $t = 4.85$ s

Fig. 36 Polynomiographs of f_2 and the Das–Debata iteration for varying α, β, ω_1, η_1, ω_2, η_2



(a) $\omega_1 = 0.5$, $t = 4.36$ s **(b)** $\omega_1 = 0.6$, $t = 3.99$ s **(c)** $\omega_1 = 0.7$, $t = 3.88$ s **(d)** $\omega_1 = 0.8$, $t = 4.16$ s

Fig. 37 Polynomiographs of f_3 and the Das–Debata iteration for $\alpha = 0.5$, $\beta = 0.5$, $\eta_1 = 0.05$, $\omega_2 = 0.9$, $\eta_2 = 0.05$ and varying ω_1



(a) $\eta_2 = 0.0004$, $t = 16.94$ s **(b)** $\eta_2 = 0.0005$, $t = 16.78$ s **(c)** $\eta_2 = 0.0006$, $t = 16.88$ s **(d)** $\eta_2 = 0.0007$, $t = 16.97$ s

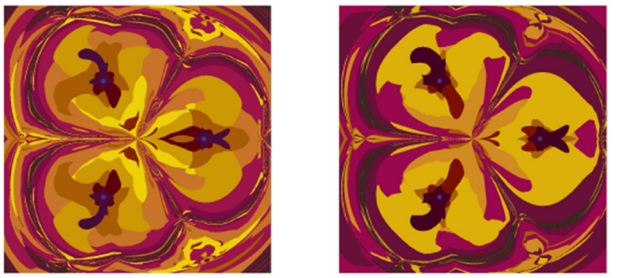
Fig. 38 Polynomiographs of f_4 and the Das–Debata iteration for $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.8$, $\eta_1 = 0.05$, $\omega_2 = 0.9$ and varying η_2

Figs. 28 and 29. Small differences of the η_1 parameter are the cause of creation of similar patterns.

Dynamics of particle's movement has a significant impact on the appearance of the polynomiograph. The parameters ω_2 and η_2 are responsible for this dynamics. As it was already mentioned, both the dynamics of the reference point and the dynamics of the particle are responsible for the creation of intriguing images. Figures 30 and 31 show the effect of the change in inertia weight ω_2 on the particle dynamics. High particle dynamics allows to obtain interesting images. Figures 32 and 33 show the effect of the change in the acceleration constant η_2 on the creation of a polynomiograph. In these images, a smaller dynamics is observed.

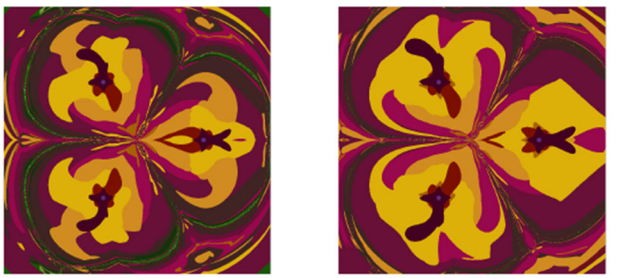
The images in Figs. 34 and 35 visualize a high dynamics of particle's movement. Due to varying of a large number of parameters, the detailed analysis of their dependence is difficult. The general rule that proper selection of the dynamics of particles reduces the operating time of the algorithm is confirmed. Moreover, from the images we see that the use of Das–Debata iteration allows to create interesting patterns.

Polynomiographs of the f_2 function and the Das–Debata iteration for varying all parameters are presented in Fig. 36. The obtained polynomiographs present a high dynamics of the particle. The shortest time of polynomiograph's creation is obtained for Fig. 36c. In this case, the narrowed range of the particle's dynamic for the entire polynomiograph area is observed. As it was already mentioned, tuning the algorithm cannot be covered by the deterministic rule due to the very complex nature of the relationship between the algorithm's coefficients and the environmental impact.



(a) $\beta = 0.1$, $t = 1.60$ s (b) $\beta = 0.9$, $t = 1.27$ s

Fig. 39 Polynomiographs of f_1 and the Agarwal iteration for $\alpha = 0.5$, $\omega_1 = \omega_2 = \omega_3 = 0.4$, $\eta_1 = \eta_2 = \eta_3 = 0.3$ and varying β



(a) $\alpha = 0.3$, $t = 1.42$ s (b) $\alpha = 0.7$, $t = 1.28$ s

Fig. 40 Polynomiographs of f_1 and the Agarwal iteration for $\beta = 0.5$, $\omega_1 = \omega_2 = \omega_3 = 0.4$, $\eta_1 = \eta_2 = \eta_3 = 0.3$ and varying α

Figure 37 presents polynomiographs obtained for the f_3 function using the Das–Debata iteration and varying ω_1 parameter. In subsequent images, it is possible to observe the effect of increasing in the value of ω_1, which is responsible for the dynamics of creating the reference point. Changes in the values of inertia weight of the reference point transformation have a small impact on the dynamics of the particle, as well as changes in the β parameter. Appropriate selection of the ω_1 parameter can shorten the time of the polynomiograph's generation—it is presented in Fig. 37c.

The results of using the Das–Debata iteration for f_4 and a varying η_2 parameter are presented in Fig. 38. Even for low values of the acceleration constant parameter, but with high dynamics of the reference point creation and a high inertia weight of the particle, it is possible to obtain a sufficient particle dynamics for an effective movement in the solution space. The obtained polynomiographs present very similar particle dynamics. The shortest time of polynomiograph's creation is obtained for Fig. 38b.

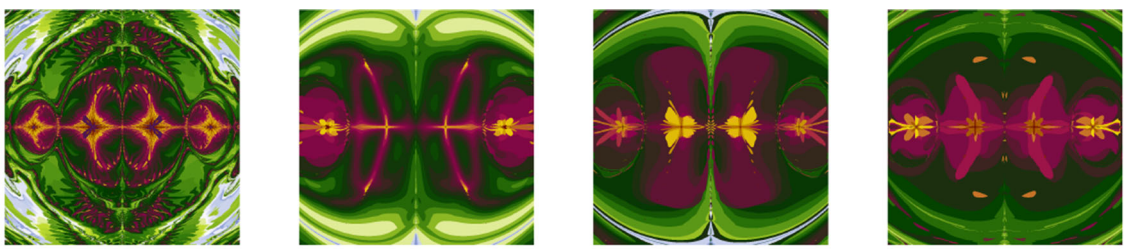
5.4 The Agarwal and the Khan–Cho–Abbas iterations

The generalized Agarwal iteration is more complex than the previously presented iterations. It introduces an additional transformation T_3 (with parameters ω_3, η_3) of particle in relation to the Das–Debata iteration.

Figures 39 and 40 present images obtained with the Agarwal iteration ($T_1 = T_2 = T_3$) for f_1 and varying α and β parameters. When we look at these images, we see that they have features of a twisting line, similarly to the Ishikawa iteration. Depending on the dynamics of the particles, it is possible to obtain images similar to the ones obtained with the Ishikawa iteration.

Operator T_3 causes the introduction of an additional dynamics of particles, the contribution of which produces the unique visual effects—the additional twisting lines are visible. The increase in dynamics of particles allows creating images of an artistic meaning. Moreover, we observe that the time of generation of the images (the working time of the algorithm) is longer that in the case of the previous iterations. This is due to the complexity of the iteration and the particle's dynamics.

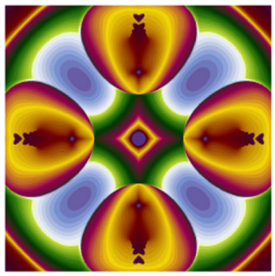
Polynomiographs of the f_2 function and the Agarwal iteration with varying α and β parameters are presented in Fig. 41. In the presented images, the shortest time of polynomiograph's creation is obtained for the parameters responsible for the highest dynamics of the particle (Fig. 41a). The introduction of the additional transformations causes that even extreme changes in the α and β parameters do not introduce drastic changes in particle's dynamics (Fig. 41b, c). Signif-



(a) $\alpha = 0.9$, $\beta = 0.9$, $t = 12.18$ s (b) $\alpha = 0.9$, $\beta = 0.1$, $t = 16.85$ s (c) $\alpha = 0.1$, $\beta = 0.9$, $t = 16.74$ s (d) $\alpha = 0.1$, $\beta = 0.1$, $t = 18.23$ s

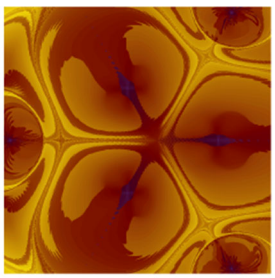
Fig. 41 Polynomiographs of f_2 and the Agarwal iteration for $\omega_1 = \omega_2 = \omega_3 = 0.9$, $\eta_1 = \eta_2 = \eta_3 = 0.01$ and varying α and β

(a) $\omega = 0.9$, $t = 9.44$ s

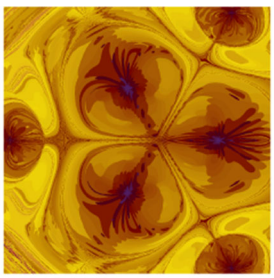


(b) $\omega = 0.6$, $t = 10.33$ s

Fig. 42 Polynomiographs of f_3 and the Agarwal iteration for $\alpha = 0.9$, $\beta = 0.9$, $\eta_1 = \eta_2 = \eta_3 = 0.01$ and varying $\omega = \omega_1 = \omega_2 = \omega_3$

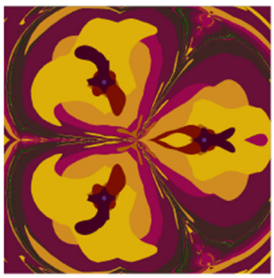


(a) $\eta = 0.001$, $t = 18.41$ s

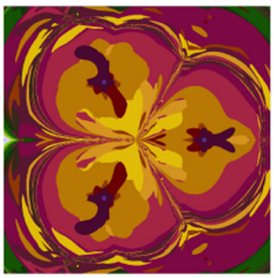


(b) $\eta = 0.004$, $t = 14.61$ s

Fig. 43 Polynomiographs of f_4 and the Agarwal iteration for $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = \omega_2 = \omega_3 = 0.8$ and varying $\eta = \eta_1 = \eta_2 = \eta_3$



(a) $\omega_1 = 0.1$, $t = 1.29$ s



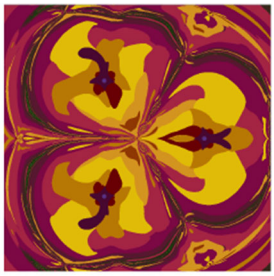
(b) $\omega_1 = 0.7$, $t = 1.48$ s

Fig. 44 Polynomiographs of f_1 and the generalized Agarwal iteration for $\alpha = 0.5$, $\beta = 0.5$, $\eta_1 = 0.4$, $\omega_2 = \omega_3 = 0.4$, $\eta_2 = \eta_3 = 0.3$ ($T_2 = T_3$) and varying ω_1

icant changes in dynamics are noticeable for simultaneous change in both parameters (Fig. 41d).

Polynomiographs of the Agarwal iteration for the f_3 function are shown in Fig. 42. They present the limitation of particle dynamics by reducing the values of the inertia weights ω_1, ω_2 and ω_3—limiting the dynamics results in longer polynomiograph creation time.

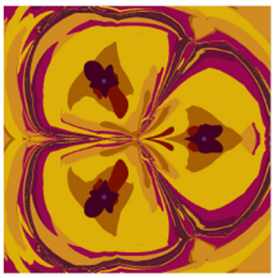
The last example for the Agarwal iteration—Fig. 43—presents images obtained for the f_4 function. The obtained images show the particle dynamics increase by the increasing



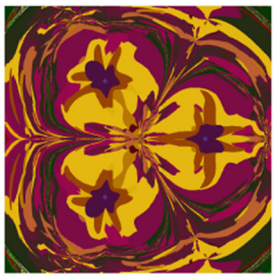
(a) $\eta_1 = 0.1$, $t = 1.51$ s

(b) $\eta_1 = 0.7$, $t = 1.01$ s

Fig. 45 Polynomiographs of f_1 and the generalized Agarwal iteration for $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.1$, $\omega_2 = \omega_3 = 0.4$, $\eta_2 = \eta_3 = 0.3$ ($T_2 = T_3$) and varying η_1



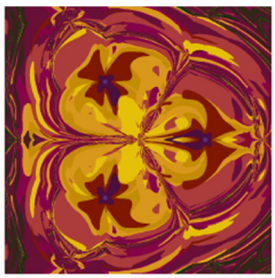
(a) $\omega_2 = 0.1$, $t = 1.09$ s



(b) $\omega_2 = 0.9$, $t = 1.99$ s

Fig. 46 Polynomiographs of f_1 and the Khan–Cho–Abbas iteration for $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.5$, $\eta_1 = 0.2$, $\eta_2 = 0.5$, $\omega_3 = 0.4$, $\eta_3 = 0.3$ and varying ω_2

(a) $\eta_2 = 0.1$, $t = 1.67$ s

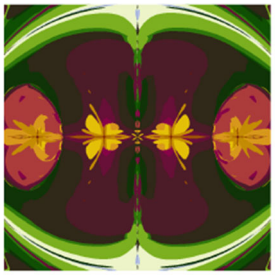


(b) $\eta_2 = 0.9$, $t = 1.71$ s

Fig. 47 Polynomiographs of f_1 and the Khan–Cho–Abbas iteration for $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.5$, $\eta_1 = 0.2$, $\omega_2 = 0.6$, $\omega_3 = 0.4$, $\eta_3 = 0.3$ and varying η_2

in the acceleration constant. In this case, this results in a shortening of the polynomiograph's creation time.

The generalized Agarwal iteration fulfils the condition $T_1 \neq T_3$. Operators T_2 and T_3 are the same for Figs. 44 and 45, which were obtained for the f_1 function. The dynamics of the particle is influenced by three transformations. Even large change in the transformation T_1 (by using ω_1 or η_1), which

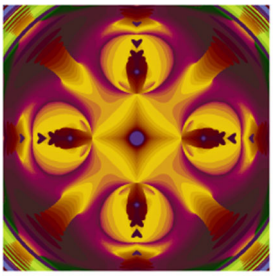


(a) $\omega_1 = 0.9$, $\eta_1 = 0.01$, $\omega_2 = 0.7$, $\eta_2 = 0.03$, $\omega_3 = 0.9$, $\eta_3 = 0.01$, $t = 6.29$ s

(b) $\omega_1 = 0.7$, $\eta_1 = 0.03$, $\omega_2 = 0.9$, $\eta_2 = 0.01$, $\omega_3 = 0.7$, $\eta_3 = 0.03$, $t = 6.09$ s

Fig. 48 Polynomiographs of f_2 and the Khan–Cho–Abbas iteration for $\alpha = 0.5$, $\beta = 0.8$, and varying ω_1, η_1, ω_2, η_2, ω_3, η_3

(a) $\omega_1 = 0.9$, $\eta_1 = 0.01$, $\omega_2 = 0.7$, $\eta_2 = 0.03$, $\omega_3 = 0.9$, $\eta_3 = 0.01$, $t = 7.16$ s



(b) $\omega_1 = 0.7$, $\eta_1 = 0.03$, $\omega_2 = 0.9$, $\eta_2 = 0.01$, $\omega_3 = 0.7$, $\eta_3 = 0.03$, $t = 5.35$ s

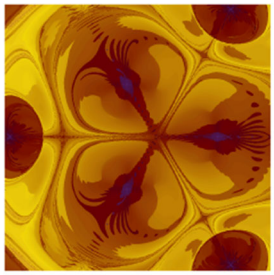
Fig. 49 Polynomiographs of f_3 and the Khan–Cho–Abbas iteration for $\alpha = 0.5$, $\beta = 0.8$ and varying ω_1, η_1, ω_2, η_2, ω_3, η_3

is responsible for the creation of a reference point, may have smaller effect on the appearance of the polynomiographs.

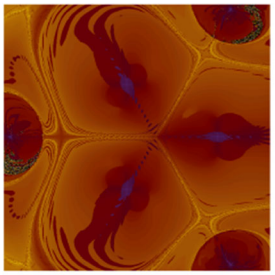
Figures 46 and 47 present polynomiographs for the Khan–Cho–Abbas iteration and the f_1 function. The excessive growth of the dynamics of the particles determined by the change in the parameters of the transformation T_2 is clearly visible in these images. It confirms only the observation that the excessive growth of dynamics of particles extends the time of image creation.

Polynomiographs of the Khan–Cho–Abbas iteration for fixed α, β and varying ω_1, η_1, ω_2, η_2, ω_3, η_3 for the f_2 function are presented in Fig. 48. The ω and η parameters change in the opposite directions, i.e. when ω increases/decreases, then η decreases/increases. It results in a characteristic change in dynamics presented in the polynomiographs by changing the colours. The shortest time of polynomiograph's creation is obtained for Fig. 48b.

A similar change in the parameters is introduced in the examples of using the Khan–Cho–Abbas iteration that are shown in Figs. 49 and 50. The images in Fig. 49 are obtained

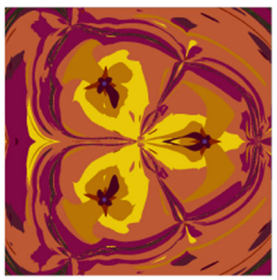


(a) $\beta = 0.9$, $\omega_1 = 0.9$, $\eta_1 = 0.001$, $\omega_2 = 0.7$, $\eta_2 = 0.002$, $\omega_3 = 0.9$, $\eta_3 = 0.001$, $t = 20.38$ s

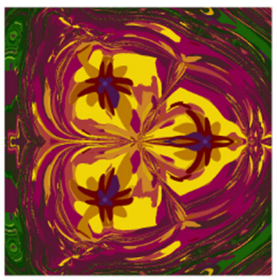


(b) $\beta = 0.8$, $\omega_1 = 0.7$, $\eta_1 = 0.002$, $\omega_2 = 0.9$, $\eta_2 = 0.001$, $\omega_3 = 0.7$, $\eta_3 = 0.002$, $t = 16.45$ s

Fig. 50 Polynomiographs of f_4 and the Khan–Cho–Abbas iteration for $\alpha = 0.5$ and varying β, ω_1, η_1, ω_2, η_2, ω_3, η_3

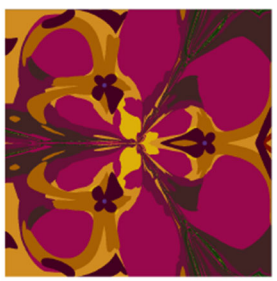


(a) $\omega_3 = 0.1$, $t = 1.16$ s

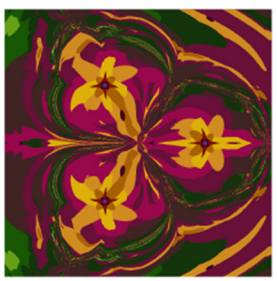


(b) $\omega_3 = 0.8$, $t = 1.71$ s

Fig. 51 Polynomiographs of f_1 and the generalized Agarwal iteration for $\alpha = 0.5$, $\beta = 0.3$, $\omega_1 = 0.3$, $\eta_1 = 0.4$, $\omega_2 = 0.6$, $\eta_2 = 0.7$, $\eta_3 = 0.5$ and varying ω_3



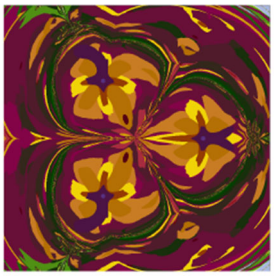
(a) $\eta_3 = 0.1$, $t = 1.30$ s



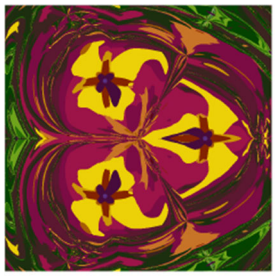
(b) $\eta_3 = 0.9$, $t = 1.59$ s

Fig. 52 Polynomiographs of f_1 and the generalized Agarwal iteration for $\alpha = 0.8$, $\beta = 0.3$, $\omega_1 = 0.2$, $\eta_1 = 0.5$, $\omega_2 = 0.6$, $\eta_2 = 0.7$, $\omega_3 = 0.5$ and varying η_3

for the f_3 function, whereas the ones in Fig. 50 for the f_4 function. Shortening the time of the polynomiograph, creation for Figs. 49b and 50b is obtained—these images visualize the reduction of the particle dynamic range. In all

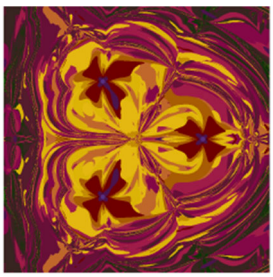


(a) $\omega_1 = 0.8$, $\eta_2 = 0.3$, $\eta_3 = 0.3$, $t = 2.26$ s

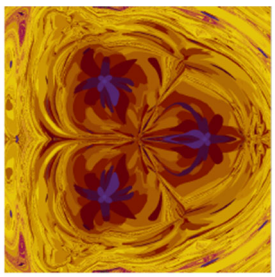


(b) $\omega_1 = 0.3$, $\eta_2 = 0.7$, $\eta_3 = 0.5$, $t = 2.07$ s

Fig. 53 Polynomiographs of f_1 and the generalized Agarwal iteration for $\alpha = 0.5$, $\beta = 0.5$, $\eta_1 = 0.4$, $\omega_2 = 0.6$, $\omega_3 = 0.9$



(a) $\alpha = 0.8$, $\beta = 0.2$, $t = 2.23$ s

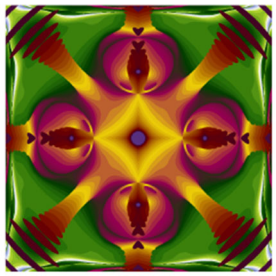


(b) $\alpha = 0.5$, $\beta = 0.3$, $t = 2.63$ s

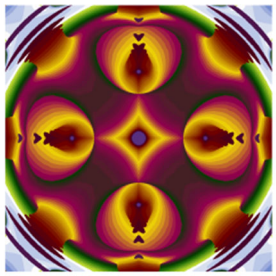
Fig. 54 Polynomiographs of f_1 and the Agarwal iteration for $\omega_1 = 0.3$, $\eta_1 = 0.4$, $\omega_2 = 0.6$, $\eta_2 = 0.7$, $\omega_3 = 0.9$, $\eta_3 = 0.5$

cases, the influence of the reference point on the movement of the particle is limited.

The generalized Agarwal iteration introduces T_3 transformation with ω_3 and η_3 parameters. Figures 51 and 52 show the effect of changing in these parameters on the creation of polynomiographs for the f_1 function. Due to the adaptation mechanism on which the particle acceleration depends, the



(a) $\alpha = 0.5$, $\beta = 0.5$, $\eta_1 = 0.03$, $\omega_2 = 0.7$, $\eta_2 = 0.02$, $\omega_3 = 0.8$, $\eta_3 = 0.002$, $t = 8.03$ s



(b) $\alpha = 0.7$, $\beta = 0.2$, $\eta_1 = 0.01$, $\omega_2 = 0.6$, $\eta_2 = 0.04$, $\omega_3 = 0.9$, $\eta_3 = 0.003$, $t = 6.15$ s

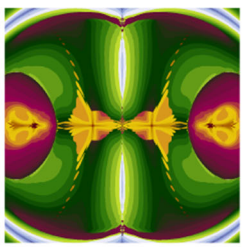
Fig. 56 Polynomiographs of f_3 and the generalized Agarwal iteration for $\omega_1 = 0.9$ and varying α, β, η_1, ω_2, η_2, ω_3, η_3

inertia of the particle has a significantly greater effect on the visual effects. This can be also observed in other presented images.

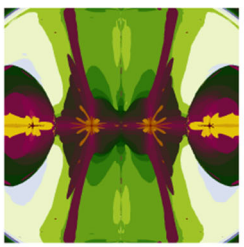
Moreover, the images in Figs. 53 and 54 were obtained for the f_1 function using the generalized Agarwal iteration and varying different parameters. The shortest times are obtained for images in Figs. 53b and 54a. These images illustrate the high particle dynamics, and they look like brush painting images.

Polynomiographs of the generalized Agarwal iteration for the f_2 function and varying all parameters are presented in Fig. 55. Shortening the time of polynomiographs creation occurs in subsequent images. (The shortest time is obtained for Fig. 55d.) The generalized Agarwal iteration gives the greatest possibility of the particle dynamics control.

The last two figures present polynomiographs generated by the generalized Agarwal iteration for the f_3 (Fig. 56) and f_4 (Fig. 57) functions. The generation algorithm is tuned just like in the previous example. We observe a limitation of the



(a) $\alpha = 0.4$, $\beta = 0.7$, $\omega_1 = 0.2$, $\eta_1 = 0.01$, $\omega_2 = 0.8$, $\eta_2 = 0.003$, $\omega_3 = 0.3$, $\eta_3 = 0.01$, $t = 6.33$ s

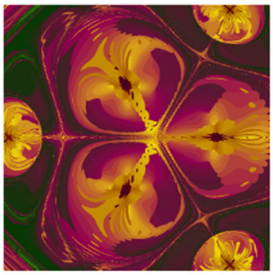


(b) $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.3$, $\eta_1 = 0.02$, $\omega_2 = 0.7$, $\eta_2 = 0.01$, $\omega_3 = 0.8$, $\eta_3 = 0.002$, $t = 6.14$ s

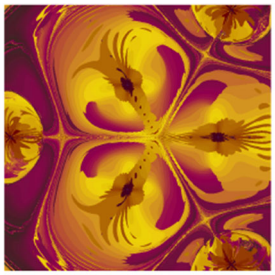
(c) $\alpha = 0.5$, $\beta = 0.8$, $\omega_1 = 0.7$, $\eta_1 = 0.01$, $\omega_2 = 0.6$, $\eta_2 = 0.01$, $\omega_3 = 0.5$, $\eta_3 = 0.002$, $t = 5.74$ s

(d) $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.3$, $\eta_1 = 0.02$, $\omega_2 = 0.7$, $\eta_2 = 0.01$, $\omega_3 = 0.3$, $\eta_3 = 0.002$, $t = 4.95$ s

Fig. 55 Polynomiographs of the generalized Agarwal iteration for varying α, β, ω_1, η_1, ω_2, η_2, ω_3, η_3



(a) $\alpha = 0.5$, $\beta = 0.5$, $\omega_1 = 0.9$, $\eta_1 = 0.003$, $\omega_2 = 0.7$, $\eta_2 = 0.002$, $\omega_3 = 0.8$, $\eta_3 = 0.002$, $t = 16.73$ s



(b) $\alpha = 0.4$, $\beta = 0.8$, $\omega_1 = 0.7$, $\eta_1 = 0.004$, $\omega_2 = 0.8$, $\eta_2 = 0.003$, $\omega_3 = 0.9$, $\eta_3 = 0.001$, $t = 14.79$ s

Fig. 57 Polynomiographs of f_4 and the generalized Agarwal iteration for varying α, β, ω_1, η_1, ω_2, η_2, ω_3, η_3

dynamic range for the images having the shortest time of polynomiograph creation (Figs. 56b, 57b).

6 Conclusions

The inertia weight and the acceleration constants are presented in many optimization algorithms which require parameters tuning—it is exemplified by the PSO algorithm. The discussion presented in the article allows to show their impact on the work of the proposed root-finding algorithm. Both too large and too small dynamics of the particles will have an adverse effect on the work of the algorithm. Moreover, the presented experiments show that the dependence of particle's dynamics on the parameters used in the root-finding method (inertia weight and acceleration constant) and in the various iteration methods is a non-trivial function.

The visualization of the dynamics is a tool that not only illustrates the behaviour of particles, but also creates attractive patterns of artistic value. Thus, these patterns and the method of their creation can be used in graphics design. Some of the presented images look like brush paintings. Particle dynamics reflects the dynamics of the brush movement. This allows to obtain images that can be determined as art.

The genetic algorithm can be used to tune the algorithm to obtain optimal behaviour due to the large number of parameters. This approach can be realized as a future work.

Compliance with ethical standards

Conflict of interest The authors declare that they have no conflicts of interest.

Human participants or animals This article does not contain any studies with human participants or animals performed by the authors.

References

Agarwal R, O'Regan D, Sahu D (2007) Iterative construction of fixed points of nearly asymptotically nonexpansive mappings. J Nonlinear Convex Anal 8(1):61–79

Al-Hassan W, Fayek M, Shaheen S (2006) Psosa: an optimized particle swarm technique for solving the urban planning problem. In: 2006 international conference on computer engineering and systems, pp 401–405. https://doi.org/10.1109/ICCES.2006.320481

Arumugam M, Rao M (2008) On the improved performances of the particle swarm optimization algorithms with adaptive parameters, cross-over operators and root mean square (RMS) variants for computing optimal control of a class of hybrid systems. Appl Soft Comput 8(1):324–336. https://doi.org/10.1016/j.asoc.2007.01.010

Bansal J, Singh P, Saraswat M, Verma A, Jadon S, Abraham A (2011) Inertia weight strategies in particle swarm optimization. In: 2011 third world congress on nature and biologically inspired computing, pp 633–640. https://doi.org/10.1109/NaBIC.2011.6089659

Broer H, Takens F (2011) Dynamical systems and chaos. Springer, New York

Chen XD, Ma W (2015) A planar quadratic clipping method for computing a root of a polynomial in an interval. Comput Graph 46:89–98. https://doi.org/10.1016/j.cag.2014.09.014

Chen G, Huang X, Jia J, Min Z (2006) Natural exponential inertia weight strategy in particle swarm optimization. In: 2006 6th world congress on intelligent control and automation, vol 1, pp 3672–3675. https://doi.org/10.1109/WCICA.2006.1713055

Chun S, Kwasinski A (2011) Analysis of classical root-finding methods applied to digital maximum power point tracking for sustainable photovoltaic energy generation. IEEE Trans Power Electron 26(12):3730–3743. https://doi.org/10.1109/TPEL.2011.2157707

Das G, Debata J (1986) Fixed points of quasinonexpansive mappings. Indian J Pure Appl Math 17(11):1263–1269

Eberhart R, Shi Y (2001) Tracking and optimizing dynamic systems with particle swarms. In: Proceedings of the 2001 congress on evolutionary computation (IEEE Cat. No.01TH8546), vol 1, pp 94–100. https://doi.org/10.1109/CEC.2001.934376

Franklin J (2013) Computational methods for physics. Cambridge University Press, Cambridge. https://doi.org/10.1017/CBO9781139525398

Gao Y, An X, Liu J (2008) A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation. In: 2008 international conference on computational intelligence and security, vol 1, pp 61–65. https://doi.org/10.1109/CIS.2008.183

Gdawiec K (2017) Fractal patterns from the dynamics of combined polynomial root finding methods. Nonlinear Dyn 90(4):2457–2479. https://doi.org/10.1007/s11071-017-3813-6

Gdawiec K, Kotarski W (2017) Polynomiography for the polynomial infinity norm via Kalantari's formula and nonstandard iterations. Appl Math Comput 307:17–30. https://doi.org/10.1016/j.amc.2017.02.038

Gdawiec K, Kotarski W, Lisowska A (2015) Polynomiography based on the non-standard Newton-like root finding methods. Abstr Appl Anal 2015, Article ID 797594. https://doi.org/10.1155/2015/797594

Gosciniak I (2008) Immune algorithm in non-stationary optimization task. In: 2008 international conference on computational intelligence for modelling control automation, pp 750–755. https://doi.org/10.1109/CIMCA.2008.181

Gosciniak I (2017) Discussion on semi-immune algorithm behaviour based on fractal analysis. Soft Comput 21(14):3945–3956. https://doi.org/10.1007/s00500-016-2044-y

Grant H, Kleiner I (2015) Turning points in the history of mathematics. Birkhäuser, Basel. https://doi.org/10.1007/978-1-4939-3264-1

Ishikawa S (1974) Fixed points by a new iteration method. Proc Am Math Soc 44(1):147–150. https://doi.org/10.1090/S0002-9939-1974-0336469-5

Jordehi A, Jasni J (2013) Parameter selection in particle swarm optimisation: a survey. J Exp Theor Artif Intell 25(4):527–542. https://doi.org/10.1080/0952813X.2013.782348

Kalantari B (2004) Polynomiography and applications in art, education and science. Comput Graph 28(3):417–430. https://doi.org/10.1016/j.cag.2004.03.009

Kalantari B (2009) Polynomial root-finding and polynomiography. World Scientific, Singapore. https://doi.org/10.1142/9789812811837

Khan S, Cho Y, Abbas M (2011) Convergence to common fixed points by a modified iteration process. J Appl Math Comput 35(1):607–616. https://doi.org/10.1007/s12190-010-0381-z

Klein S, Pluim J, Staring M, Viergever M (2009) Adaptive stochastic gradient descent optimisation for image registration. Int J Comput Vis 81(3):227–239. https://doi.org/10.1007/s11263-008-0168-y

Konečný J, Richtárik P (2017) Semi-stochastic gradient descent methods. Front Appl Math Stat 3:9. https://doi.org/10.3389/fams.2017.00009

Kotarski W, Lisowska A (2018) Polynomiography via the hybrids of gradient descent and Newton methods with Mann and Ishikawa iterations. In: Rocha A, Adeli H, Reis L, Costanzo S (eds) Trends and advances in information systems and technologies, advances in intelligent systems and computing, vol 746. Springer, Cham, pp 455–464. https://doi.org/10.1007/978-3-319-77712-2_43

Kotarski W, Gdawiec K, Lisowska A (2012) Polynomiography via Ishikawa and Mann iterations. In: Bebis G, Boyle R, Parvin B, Koracin D, Fowlkes C, Wang S, Choi MH, Mantler S, Schulze J, Acevedo D, Mueller K, Papka M (eds) Advances in visual computing, lecture notes in computer science, vol 7431. Springer, Berlin, pp 305–313. https://doi.org/10.1007/978-3-642-33179-4_30

Lei K, Qiu Y, He Y (2006) A new adaptive well-chosen inertia weight strategy to automatically harmonize global and local search ability in particle swarm optimization. In: 2006 1st international symposium on systems and control in aerospace and astronautics, pp 977–980. https://doi.org/10.1109/ISSCAA.2006.1627487

Li H, Gao Y (2009) Particle swarm optimization algorithm with exponent decreasing inertia weight and stochastic mutation. In: 2009 second international conference on information and computing science, vol 1, pp 66–69. https://doi.org/10.1109/ICIC.2009.24

Malik R, Rahman T, Hashim S, Ngah R (2007) New particle swarm optimizer with sigmoid increasing inertia weight. Int J Comput Sci Secur 1:35–44

Mann W (1953) Mean value methods in iteration. Proc Am Math Soc 4(3):506–510. https://doi.org/10.1090/S0002-9939-1953-0054846-3

Panigrahi B, Pandi V, Das S (2008) Adaptive particle swarm optimization approach for static and dynamic economic load dispatch. Energy Convers Manag 49(6):1407–1415. https://doi.org/10.1016/j.enconman.2007.12.023

Polak E (1997) Optimization algorithms and consistent approximations. Springer, New York. https://doi.org/10.1007/978-1-4612-0663-7

Qin Z, Yu F, Shi Z, Wang Y (2006) Adaptive inertia weight particle swarm optimization. In: Rutkowski L, Tadeusiewicz R, Zadeh L, Żurada J (eds) Artificial intelligence and soft computing—ICAISC 2006. Springer, Berlin, pp 450–459

Saber A, Senjyu T, Urasaki N, Funabashi T (2006) Unit commitment computation—a novel fuzzy adaptive particle swarm optimization approach. In: 2006 IEEE PES power systems conference and exposition, pp 1820–1828. https://doi.org/10.1109/PSCE.2006.296189

Sayama H (2015) Introduction to the modeling and analysis of complex systems. Open SUNY Textbooks, Geneseo

Sengupta A, Mishra V (2014) Time varying vs fixed acceleration coefficient PSO driven exploration during high level synthesis: Performance and quality assessment. In: 2014 international conference on information technology, pp 281–286. https://doi.org/10.1109/ICIT.2014.16

Senov A, Granichin O (2017) Projective approximation based gradient descent modification. IFAC PapersOnLine 50(1):3899–3904. https://doi.org/10.1016/j.ifacol.2017.08.362

Shi Y, Eberhart R (1998) A modified particle swarm optimizer. In: Proceedings of IEEE international conference on evolutionary computation, pp 69–73. IEEE Computer Society, Washington, DC, USA. https://doi.org/10.1109/ICEC.1998.699146

Shi Y, Eberhart R (1999) Empirical study of particle swarm optimization. In: Proceedings of the 1999 congress on evolutionary computation-CEC99 (cat. no. 99TH8406), vol 3, pp 1945–1950. https://doi.org/10.1109/CEC.1999.785511

Shi Y, Eberhart R (2001) Fuzzy adaptive particle swarm optimization. In: Proceedings of the 2001 congress on evolutionary computation (IEEE cat. no. 01TH8546), vol 1, pp 101–106. https://doi.org/10.1109/CEC.2001.934377

Suresh K, Ghosh S, Kundu D, Sen A, Das S, Abraham A (2008) Inertia-adaptive particle swarm optimizer for improved global search. In: 2008 eighth international conference on intelligent systems design and applications, vol 2, pp 253–258. https://doi.org/10.1109/ISDA.2008.199

Weise T (2009) Global optimization algorithms—theory and application, 2nd edn. http://www.it-weise.de/projects/book.pdf

Yang S, Li C (2010) A clustering particle swarm optimizer for locating and tracking multiple optima in dynamic environments. IEEE Trans Evolut Comput 14:959–974. https://doi.org/10.1109/TEVC.2010.2046667

Yang X, Yuan J, Yuan J, Mao H (2007) A modified particle swarm optimizer with dynamic adaptation. Appl Math Comput 189(2):1205–1213. https://doi.org/10.1016/j.amc.2006.12.045

Zhang W, Ma D, Wei J, Liang H (2014) A parameter selection strategy for particle swarm optimization based on particle positions. Expert Syst Appl 41(7):3576–3584. https://doi.org/10.1016/j.eswa.2013.10.061