Control of Dynamics of the Modified Newton-Raphson Algorithm

Ireneusz Gościniak[a,*], Krzysztof Gdawiec[a]

[a] *Institute of Computer Science, University of Silesia, Będzińska 39, 41–200 Sosnowiec, Poland*

Abstract

Many algorithms that iteratively find solution of an equation are described in the literature. In this article we propose an algorithm that is based on the Newton-Raphson root finding method and which uses an adaptation mechanics. The adaptation mechanics is based on a linear combination of some membership functions and allows a better control of algorithm's dynamics. The proposed approach allows to visualize the adaptation mechanics impact on the operation of the algorithm. Moreover, various iteration processes and their operation mechanics are discussed in this research. The understanding of the impact of the proposed modifications on the algorithm's operation can be helpful at using other algorithms. The obtained visualizations have also an artistic potential and can be used for instance in creating mosaics, wallpapers etc.

Keywords: root finding, dynamics, iterations, visualization

1. Introduction

Determining the local minimum of a function $\mathbf{F}$ is a task implemented by algorithms which are based on the function's value or gradient. The movement direction of a particle is determined by the gradient. A step is calculated from the position of the particle reached in the previous iteration and the next position is determined by using the gradient [1]. The gradient descent method is a typical example of this kind of methods and it can be written in the following form:

$$\mathbf{z}'_i = \mathbf{z}_i - \gamma \nabla \mathbf{F}(\mathbf{z}_i), \tag{1}$$

where $-\nabla\mathbf{F}$ – negative gradient, γ – step size, $\mathbf{z'}_i$ – the current position of the ith particle in a D dimensional environment, $\mathbf{z}_i$ – the previous position of the ith particle. This approach is used by many different algorithms and their properties are well described in the literature.

*Corresponding author

Email addresses: ireneusz.gosciniak@us.edu.pl (Ireneusz Gościniak), kgdawiec@ux2.math.us.edu.pl (Krzysztof Gdawiec)

Another approach in solving the minimization problem is the use of evolutionary algorithms. The analysis of particle's motion in this group of algorithms is not a trivial problem [2, 3]. A similar behaviour to the evolutionary algorithms show the Particle Swarm Optimization (PSO) algorithms [4]. The interaction of particles in the swarm causes a very complex behaviour. The motion of a particle in the PSO algorithm is described using the following formula:

$$\mathbf{z}'_i = \mathbf{z}_i + \mathbf{v}'_i, \tag{2}$$

where $\mathbf{z}'_i$ – the current position of the ith particle in a D dimensional environment, $\mathbf{z}_i$ – the previous position of the ith particle, $\mathbf{v}'_i$ – the current velocity of the ith particle in a D dimensional environment that is described by the following equation:

$$\mathbf{v}'_i = \omega \mathbf{v}_i + \eta_1 r_1 (\mathbf{z}_{pbest\ i} - \mathbf{z}_i) + \eta_2 r_2 (\mathbf{z}_{gbest} - \mathbf{z}_i), \tag{3}$$

where $\mathbf{v}_i$ – the previous velocity of the ith particle, ω – 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 uniformly in the $[0, 1]$ interval, $\mathbf{z}_{pbest\ i}$ – the best position of the ith particle, $\mathbf{z}_{gbest}$ – the global best position of the particles.

Inertia weight (ω) and acceleration constants (η_1, η_2) are responsible for the particle motion. The inertia weight is responsible for the balance of the exploration and exploitation. Numerous approaches to setting the inertia weight have been proposed in the literature. These control rules of inertia weight can be classified as follows:

1. constant [5],
2. random [6],
3. time varying, e.g., linear decreasing [7], sigmoid increasing/decreasing [8], simulated annealing [9], Sugeno function [10], exponential decreasing law [11] and [12], logarithmic decreasing law [13],
4. adaptive control (using feedbacks of the optimization process), e.g., best fitness [14] and [15], fitness of the current and previous iterations [16], global best and average local best fitness [17], particle rank [18], distance to particle and global best positions [19], distance to global best position [20].

The inertia weight value greater than 1.2 performs exploration process. Whereas, if the inertia weight value is smaller than 0.8 it is responsible for the exploitation. The value of inertia weight mostly presented in literature is within the range of $[0.4, 0.9]$ (see [21]).

Acceleration constants η_1, η_2 affect the pulling of particles towards the best position. The η_1 pulls the particle towards its own best position $\mathbf{z}_{pbest\ i}$ and the η_2 attracts all the particles towards the global best position $\mathbf{z}_{gbest}$ [22]. The particles can be trapped in false optima for too high values of η_1, η_2. If η_1, η_2 take too low values the algorithm cannot reach solution.

The parameters tuning is a very important problem for these groups of algorithms [23, 22] and the particle dynamics is determined by the adaptation mechanics resulting from particles co-operation.

Graphical visualization is one of many different methods to analyse the dynamics [24]. In the root finding problem the methods of visualizing the dynamics have even their own name, namely polynomiography [25]. In these methods the algorithm counts the number of iterations required to obtain the root of a given polynomial. Next, the number of iterations is visualised using some colour map. This method can be used not only for polynomials but also for other functions. Despite the fact that the name polynomiography is related to polynomials we will use the same name for the methods of visualizing the dynamics for arbitrary functions.

In this paper, we propose a method for solving non-linear system of equations that is based on the Newton-Raphson method and the idea of the PSO algorithms. Like in the PSO algorithm the proposed method introduces the inertia weight and the acceleration constant, but we replace the constant values of these parameters with the adaptation mechanics. Visualization method is used to show the influence of the adaptation functions on the particle's behaviour. Graphical results presented in this paper allow to provide not only the discussion on the behaviour of a particle, but they have also aesthetic character and artistic meaning.

The rest of the paper is organized as follows. In Sec. 2 we introduce the root finding algorithm that is based on the Newton-Raphson method and the PSO algorithm idea. Then, in Sec. 3 we define the adaptation mechanics that is used in the proposed algorithm. In the next section – Sec. 4 – we present some iteration processes known in the literature. The algorithm for creating polynomiographs is presented in Sec. 5. Then, in Sec. 6 we discuss the research results illustrated by the obtained polynomiographs. Finally, in Sec. 7 we give some short concluding remarks.

2. The algorithm

The problem of solving a system of D non-linear equations with D variables is widely known and studied problem in the literature. There are many different methods for solving such systems and one of the most commonly used methods is the Newton-Raphson method [26].

Let $f_1, f_2, \ldots, f_D : \mathbb{R}^D \to \mathbb{R}$ and let

$$\mathbf{F}(z^1, z^2, \ldots, z^D) = \begin{bmatrix} f_1(z^1, z^2, \ldots, z^D) \\ f_2(z^1, z^2, \ldots, z^D) \\ \vdots \\ f_D(z^1, z^2, \ldots, z^D) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \mathbf{0}. \tag{4}$$

To solve the equation $\mathbf{F}(\mathbf{z}) = \mathbf{0}$, where $\mathbf{F} : \mathbb{R}^D \to \mathbb{R}^D$ is a continuous function with continuous first partial derivatives and $\mathbf{z} = [z^1, z^2, \ldots, z^D]$ using the Newton-Raphson first we select a starting point $\mathbf{z}_0 = [z_0^1, z_0^2, \ldots, z_0^D]$. Then, we use the following iterative formula:

$$\mathbf{z}_{n+1} = \mathbf{z}_n - \mathbf{J}^{-1}(\mathbf{z}_n)\mathbf{F}(\mathbf{z_n}) \quad n = 1, 2, \ldots, \tag{5}$$

where

$$\mathbf{J}(\mathbf{z}) = \begin{bmatrix} \frac{\partial f_1}{\partial z_1}(\mathbf{z}) & \frac{\partial f_1}{\partial z_2}(\mathbf{z}) & \dots & \frac{\partial f_1}{\partial z_D}(\mathbf{z}) \\ \frac{\partial f_2}{\partial z_1}(\mathbf{z}) & \frac{\partial f_2}{\partial z_2}(\mathbf{z}) & \dots & \frac{\partial f_2}{\partial z_D}(\mathbf{z}) \\ \vdots & \vdots & \vdots & \vdots \\ \frac{\partial f_D}{\partial z_1}(\mathbf{z}) & \frac{\partial f_D}{\partial z_2}(\mathbf{z}) & \dots & \frac{\partial f_D}{\partial z_D}(\mathbf{z}) \end{bmatrix} \tag{6}$$

is the Jacobian matrix of $\mathbf{F}$ and $\mathbf{J}^{-1}$ is its inverse.

Let $\mathbf{N}(\mathbf{z}) = -\mathbf{J}^{-1}(\mathbf{z})\mathbf{F}(\mathbf{z})$. Using $\mathbf{N}$ the Newton-Raphson method can be written in the following form:

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \mathbf{N}(\mathbf{z}_n), \quad n = 0, 1, 2, \dots. \tag{7}$$

Basing on the idea of the PSO algorithm we can propose another method for solving (4). Let $\mathbf{z}_0 \in \mathbb{R}^D$ be a starting position and $\mathbf{v}_0 = [0, 0, \dots, 0]$ be a starting velocity. To find the roots of (4) we use the following iterative algorithm:

$$\mathbf{z}_{n+1} = \mathbf{z}_n + \mathbf{v}_{n+1}, \tag{8}$$

where $\mathbf{v}_{n+1}$ is the current velocity of the particle ($\mathbf{v}_{n+1} = [v^1_{n+1}, v^2_{n+1}, \dots, v^D_{n+1}]$), $\mathbf{z}_n$ is the previous position of the particle ($\mathbf{z}_n = [z^1_n, z^2_n, \dots, z^D_n]$).

The algorithm determines the current position of the particle summing the previous position of particle $\mathbf{z}_n$ with its current velocity $\mathbf{v}_{n+1}$. Combining the Newton-Raphson method with the PSO approach the current velocity of the particle is determined by the inertia weight and the acceleration constant as follows:

$$\mathbf{v}_{n+1} = \omega\mathbf{v}_n + \eta\mathbf{N}(\mathbf{z}_n), \tag{9}$$

where $\mathbf{v}_{n+1}$ – the current velocity of the particle, $\mathbf{v}_n$ – the previous velocity of the particle, $\omega \in [0, 1)$ – inertia weight, $\eta \in (0, 1]$ – acceleration constant.

The inertia weight (ω) and the acceleration constant (η) influence the particle's behaviour. They can be changed depending on the algorithm's progress. The effect of changes of these parameters determines particle dynamics, and it will be visualized using the algorithm described in Sec. 5.

3. Applying mechanics of adaptation

The particle's dynamics is a function of inertia weight (ω) and the acceleration constant (η). These parameters can be changed during the progress of the iteration process. The change can be made depending on the value of $\|\mathbf{F}(\mathbf{z}_n)\|$, because the lower the value of $\|\mathbf{F}(\mathbf{z}_n)\|$ the better the position of particle for the problem being analysed. It is possible to propose a modification of (9) by applying the adaptation functions $\omega, \eta : \mathbb{R} \to \mathbb{R}$. In order to simplify the definition of the adaptation function, a linear combination of membership functions μ_1, μ_2, μ_3 and corresponding weights $\omega_1, \omega_2, \omega_3$, η_1, η_2, η_3 is proposed.

As membership functions we can use different functions that are used in fuzzy numbers theory. In this paper we propose the use of the following three

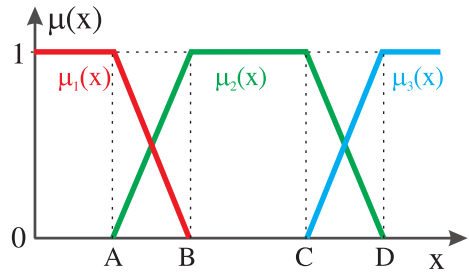


Figure 1: Membership functions for modelling the adaptation functions

membership functions:

$$\mu_1(x) = \begin{cases} 1 & \text{if } x < A, \\ \frac{B-x}{B-A} & \text{if } A \leq x \leq B, \\ 0 & \text{if } B < x, \end{cases} \tag{10}$$

$$\mu_2(x) = \begin{cases} \frac{x-A}{B-A} & \text{if } A \leq x < B, \\ 1 & \text{if } B \leq x < C, \\ \frac{D-x}{D-C} & \text{if } C \leq x \leq D, \\ 0 & \text{if } x < A \vee D < x, \end{cases} \tag{11}$$

$$\mu_3(x) = \begin{cases} 0 & \text{if } x < C, \\ \frac{x-C}{D-C} & \text{if } C \leq x < D, \\ 1 & \text{if } D \leq x, \end{cases} \tag{12}$$

where $x, A, B, C, D \in \mathbb{R}$ and $A < B \leq C < D$. The graphs of μ_1, μ_2, μ_3 are presented in Fig. 1.

Having membership functions μ_1, μ_2, μ_3 we define the adaptation functions in the following way:

- the inertia adaptation function

$$\omega(x) = \omega_1 \cdot \mu_1(x) + \omega_2 \cdot \mu_2(x) + \omega_3 \cdot \mu_3(x), \tag{13}$$

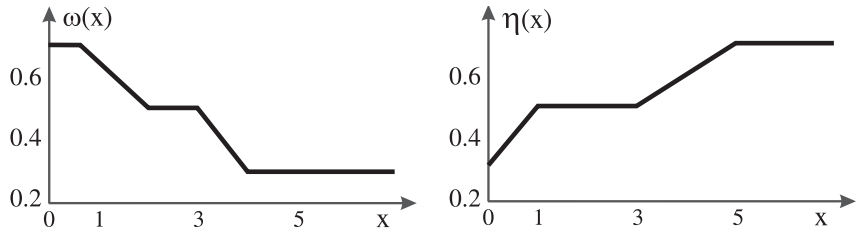
  where $x \in \mathbb{R}$ and $\omega_1, \omega_2, \omega_3 \in [0, 1)$,

- the acceleration adaptation function

$$\eta(x) = \eta_1 \cdot \mu_1(x) + \eta_2 \cdot \mu_2(x) + \eta_3 \cdot \mu_3(x), \tag{14}$$

  where $x \in \mathbb{R}$ and $\mu_1, \mu_2, \mu_3 \in (0, 1]$.

Each of the adaptation functions is defined by seven parameters: ω_1, ω_2, ω_2, A, B, C, D for the ω function and η_1, η_2, η_3, A, B, C, D for the η function. Fig. 2 presents examples of adaptation functions. The parameters defining these two



(a) inertia adaptation function (b) acceleration adaptation function

Figure 2: Examples of adaptation functions

function were the following: (a) $\omega_1 = 0.7$, $\omega_2 = 0.5$, $\omega_3 = 0.3$, $A = 0.6$, $B = 2$, $C = 3$, $D = 4$, (b) $\eta_1 = 0.3$, $\eta_2 = 0.5$, $\eta_3 = 0.7$, $A = 0$, $B = 1$, $C = 3$, $D = 5$.

Using the inertia and acceleration adaptation functions we modify (9) in the following way:

$$\mathbf{v}_{n+1} = \omega(\|\mathbf{F}(\mathbf{z}_n)\|)\mathbf{v}_n + \eta(\|\mathbf{F}(\mathbf{z}_n)\|)\mathbf{N}(\mathbf{z}_n), \tag{15}$$

This approach allows to control changes in particle dynamics depending on $\|\mathbf{F}(\mathbf{z}_n)\|$.

4. Iteration processes

Iterative algorithms are widely used in many computational tasks, e.g., root finding [25], eigenvalue problem [27], solving of differential equations [26]. In these iterative methods usually the so-called Picard iteration is used, i.e., iteration of the following form:

$$\mathbf{z}_{n+1} = \mathbf{T}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \tag{16}$$

where $\mathbf{T}$ is some mapping and $\mathbf{z}_0$ is a starting point.

The literature on the approximate finding of fixed points is full of other types of iteration processes. These processes are divided into two types: explicit and implicit. In this paper we focus on the explicit ones. Let us recall some of the most widely used iteration method of this type.

1. The Mann iteration [28]:

$$\mathbf{z}_{n+1} = (1 - \alpha_n)\mathbf{z}_n + \alpha_n\mathbf{T}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \tag{17}$$

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

2. The Ishikawa iteration [29]:

$$\begin{aligned} \mathbf{z}_{n+1} &= (1 - \alpha_n)\mathbf{z}_n + \alpha_n\mathbf{T}(\mathbf{u}_n), \\ \mathbf{u}_n &= (1 - \beta_n)\mathbf{z}_n + \beta_n\mathbf{T}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \end{aligned} \tag{18}$$

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

3. The Agarwal iteration [30] (S-iteration):

$$\begin{aligned} \mathbf{z}_{n+1} &= (1-\alpha_n)\mathbf{T}(\mathbf{z}_n) + \alpha_n \mathbf{T}(\mathbf{u}_n), \\ \mathbf{u}_n &= (1-\beta_n)\mathbf{z}_n + \beta_n \mathbf{T}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \end{aligned} \tag{19}$$

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

4. The Das-Debata iteration [31]:

$$\begin{aligned} \mathbf{z}_{n+1} &= (1-\alpha_n)\mathbf{z}_n + \alpha_n \mathbf{T}^{II}(\mathbf{u}_n), \\ \mathbf{u}_n &= (1-\beta_n)\mathbf{z}_n + \beta_n \mathbf{T}^{I}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \end{aligned} \tag{20}$$

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

5. The Khan-Cho-Abbas iteration [32]:

$$\begin{aligned} \mathbf{z}_{n+1} &= (1-\alpha_n)\mathbf{T}^{I}(\mathbf{z}_n) + \alpha_n \mathbf{T}^{II}(\mathbf{u}_n), \\ \mathbf{u}_n &= (1-\beta_n)\mathbf{z}_n + \beta_n \mathbf{T}^{I}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \end{aligned} \tag{21}$$

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

6. The generalized Agarwal iteration [32]:

$$\begin{aligned} \mathbf{z}_{n+1} &= (1-\alpha_n)\mathbf{T}^{III}(\mathbf{z}_n) + \alpha_n \mathbf{T}^{II}(\mathbf{u}_n), \\ \mathbf{u}_n &= (1-\beta_n)\mathbf{z}_n + \beta_n \mathbf{T}^{I}(\mathbf{z}_n),\ n = 0, 1, 2, \ldots, \end{aligned} \tag{22}$$

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

The Picard, Mann, Ishikawa and the Agarwal iterations are used in finding fixed points of a single mapping $\mathbf{T}$, whereas the Das-Debata, Khan-Cho-Abbas and the generalized Agarwal iterations are used in finding common fixed points of two ($\mathbf{T}^I$, $\mathbf{T}^{II}$) or three ($\mathbf{T}^I$, $\mathbf{T}^{II}$, $\mathbf{T}^{III}$) mappings. Moreover, let us notice that some of the iterations reduce to other, e.g., the Mann iteration reduces to the Picard iteration when $\alpha_n = 1$ for all $n \in \mathbb{N}$. The possible conversions between the introduced iterations are presented in Tab. 1. The iteration in the column converts to the iteration in a row; "$-$" means that there is no possible conversion. More iteration processes for finding fixed points of a single mapping and their dependencies can be found in [33].

When we look at (8) we see that this algorithm uses the Picard iteration, where the mapping $\mathbf{T}$ is given by the following formula:

$$\mathbf{T}(\mathbf{z}_n) = \mathbf{z}_n + \mathbf{v}_{n+1}. \tag{23}$$

Mapping $\mathbf{T}$ is defined by three parameters: the function $\mathbf{F}$, the inertia adaptation function ω given by (13), and the acceleration adaptation function η given by (14).

In the algorithm we can replace the Picard iteration by the other presented iterations. In the case of several mappings we use the same function $\mathbf{F}$, but different inertia and acceleration adaptation functions to define the mappings.

Some of the iterations use an additional sample point $\mathbf{u}_n$ (reference point). In those cases, two steps are performed: the reference point $\mathbf{u}_n$ (a sample) is determined in the first step, and then a new position of the particle $\mathbf{z}_{n+1}$ is calculated. This allows to better control the particle's movement.

Table 1: Conversions of iterations (iteration in the column converts to the iteration in a row; "−" – no possible conversion)

	Picard	Mann	Ishikawa	Agarwal	Das-Debata	Khan-Cho-Abbas	generalized Agarwal
Picard	×	$\alpha = 1$	$\alpha = 1$ $\beta = 0$	$\alpha = 0$ or $\alpha = 1$ $\beta = 0$	$\alpha = 1$ $\beta = 0$	$\alpha = 0$ or $\alpha = 1$ $\beta = 0$	$\alpha = 0$ or $\alpha = 1$ $\beta = 0$
Mann	−	×	$\beta = 0$	−	$\beta = 0$	−	−
Ishikawa	−	−	×	−	$T^I = T^{II}$	−	−
Agarwal	−	−	−	×	−	$T^I = T^{II} = T^{III}$	$T^I = T^{II} = T^{III}$
Das-Debata	−	−	−	−	×	−	−
Khan-Cho-Abbas	−	−	−	−	−	×	$T^I = T^{III}$
generalized Agarwal	−	−	−	−	−	−	×

5. Visualization of the dynamics

The visualization of the dynamics of the proposed methods relays on the ideas used in polynomiography [34]. First, we select one of the iteration methods presented in Sec. 4. Depending on the selected iteration method we set the appropriate number of inertia and acceleration adaptation functions, i.e., ω, η for a single transformation $\mathbf{T}$, and from ω^I, η^I, ω^{II}, η^{II} up to ω^{III}, η^{III} – depending on the chosen iteration method – for transformations $\mathbf{T}^I$, $\mathbf{T}^{II}$, $\mathbf{T}^{III}$. Each of the functions is given by seven parameters: ω_1, ω_2, ω_3, A_ω, B_ω, C_ω, D_ω, for the ω function, and η_1, η_2, η_3, A_η, B_η, C_η, D_η for the η function. Moreover, we set the maximum number of iterations m, which the algorithm should perform, the accuracy of the computations $\varepsilon > 0$, and we also select a colouring function $C : \mathbb{N} \to \{0, 1, \ldots, 255\}^3$ and the area of interest $\mathbf{A}$. Then, for each $\mathbf{z}_0$ in $\mathbf{A}$ we use the algorithm. The computations are proceed till the convergence criterion is not satisfied:

$$\|\mathbf{z}_{n+1} - \mathbf{z}_n\| < \epsilon \tag{24}$$

or the maximum number of iterations is not reached. Finally, using the colouring function C we assign colour to $\mathbf{z}_0$ using the number of iterations that the algorithm has performed.

The pseudocode of the visualization algorithm is presented in Algorithm 1. The I_q in the algorithm is the selected iteration method and q is its vector of parameters. For simplicity we use the following notation $A^{\{I|II|III\}}_{\{\omega|\eta\}}$, where we give the number of the transformation in the superscript, the name of the function which this parameter defines in the subscript, and | denotes alternative.

The solution space $\mathbf{A}$ is defined in a D-dimensional space, thus the algorithm returns polynomiograph in this space. For $D = 2$, a single image is obtained. When $D > 2$ cross sections of $\mathbf{A}$ with a two-dimensional plane for visualization can be made.

6. Discussion on the research results

In this section we present the results of conducted research. In the research we visualized – using the method introduced in Sec. 5 – the dynamics of the algorithm from Sec. 2 with the adaptation functions described in Sec. 3 and various iteration processes from Sec. 4. Using the visualizations we analyse the behaviour of the algorithm depending on its different parameters.

A standard function used in the literature on root finding methods is the cubic complex polynomial $c^3 - 1$. We can transform this complex function into system of two real equations with two variables in the following way. Let $c = x + iy$ where $i = \sqrt{-1}$ and $x, y \in \mathbb{R}$. Moreover, let $p(c) = c^3 - 1$. To solve the following non-linear equation

$$p(c) = 0, \tag{25}$$

Algorithm 1: Visualization of the dynamics

Input: $\mathbf{F}$ – 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, $\omega_{\{1|2|3\}}^{\{I|II|III\}}$, $\eta_{\{1|2|3\}}^{\{I|II|III\}}$, $A_{\{\omega|\eta\}}^{\{I|II|III\}}$, $B_{\{\omega|\eta\}}^{\{I|II|III\}}$, $C_{\{\omega|\eta\}}^{\{I|II|III\}}$, $D_{\{\omega|\eta\}}^{\{I|II|III\}}$,– parameters defining functions $T^{\{I|II|III\}}$, C – colouring function, ϵ – accuracy

Output: visualization of the dynamics

```
foreach z_0 ∈ A do
    i = 0
    v_0 = [0, 0, ..., 0]
    while i ≤ m do
        z_{n+1} = I_q(z_n)
        if ||z_{n+1} − z_n|| < ε then
            break
        i = i + 1
    colour z_0 with C(i)
```

Figure 3: Colour map used in the experiments.

this equation can be written as follows:

$$0 = c^3 - 1 = (x + iy)^3 - 1 = x^3 - 3xy^2 - 1 + (3x^2y - y^3)i. \tag{26}$$

In this case, equation (26) can be transformed into system of two equations with two variables:

$$\mathbf{F}(x, y) = \begin{bmatrix} f_1(x, y) \\ f_2(x, y) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix} = \mathbf{0}, \tag{27}$$

where $f_1(x, y) = x^3 - 3xy^2 - 1$ and $f_2(x, y) = 3x^2y - y^3$. Of course, the set of solutions of this system has three elements which are the following: $[1, 0]$, $[-0.5, -0.866025]$, $[-0.5, 0.866025]$.

To solve (27) we used the method introduced in Sec. 2. The same colour map (Fig. 3) was used to colour all of the obtained polynomiographs. Furthermore, in every experiment we used the following common parameters: $\mathbf{A} = [-2.0, 2.0]^2$ ($[-0.5, 0.5]^2$ is used for the magnification of the centre parts of polynomiographs), $m = 128$, $r = 1.0e{-}2$, image resolution 800×800 pixels.

Because we need to give values of many parameters, so for simplicity we introduce the following convention for the description of the parameters of the adaptation functions. The shape of an adaptation function results directly from this description. All the parameters values must be specified for the correct

description of the adaptation function. We set the default value of 0.0 for all the parameters – it means that $\{\omega|\eta\}_{\{1|2|3\}}^{\{-|I|II|III\}} = 0.0$ and $\{A|B|C|D\}_{\{\omega|\eta\}}^{\{-|I|II|III\}} = 0.0$ ("$-$" means that superscript is omitted for a single transformation $\mathbf{T}$). Only values other than zero are given in the description of the parameters.

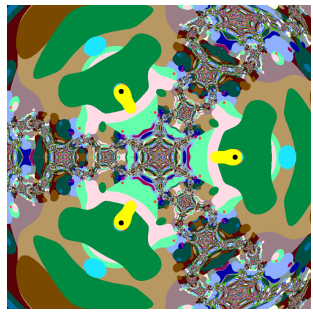
The algorithm used in the experiments was implemented in the C++ programming language. The experiments were conducted on a computer with the Intel Core i5-2520M processor, 4 GB RAM, and Linux 3.16.7-42-desktop openSUSE 13.2 (Harlequin 64-bits, KDE Platform Version 4.14.9).

6.1. The Picard iteration

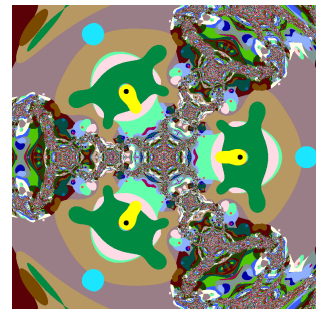
Fig. 4 presents polynomiographs obtained using the proposed method and the Picard iteration. The inertia adaptation function was modelled by using only the μ_1 membership function. This example also clarifies the convention of parameters description. Continuing the analysis of the example shown in Fig. 4, parameters A and B are required to determine the function μ_1. In the description of the example from Fig. 4a the value of A_ω (which is omitted in the description) is equal to 0.0 (by default) and B_ω is equal to 10.0, and parameter ω_1 (whose value is greater than zero and determines the use of μ_1) is equal to 0.7. These parameters indicate that the slope of the function μ_1 varies linearly from 0.7 to 0.0 in the range from $A_\omega = 0.0$ to $B_\omega = 10.0$ and the function μ_1 outside of this range is equal to 0.0. Because $\omega_2 = \omega_3 = 0$ the inertia function is determined only by using the membership function μ_1. By continuing, the parameters $\eta_1 = \eta_2 = 0$ (by default) so the acceleration function is determined only by the μ_3 membership function. Parameters C_η and D_η are equal to 0.0 (by default) and η_3 has value equal to 0.7 – these settings indicate that the acceleration function assumes a constant value equal to 0.7 in the whole range. In the example illustrated in Fig. 4b, the value of $A_\omega = 3.0$ is given in the description of the image and the remaining parameters are the same as in the example shown in Fig. 4a. It means that the inertia function returns a value equal to 0.7 in the range from 0.0 to $A_\omega = 3.0$, while the slope of the function μ_1 changes linearly from 0.7 to 0.0 in the range from $A_\omega = 3.0$ to $B_\omega = 10$, and outside of these ranges μ_1 takes the value equal to 0.0. The acceleration function is defined as in the example from Fig. 4a. The used convention of parameters specification greatly simplifies the description without affecting its quality. The parameter t in the image description denotes the average number of iterations required to create one point of polynomiograph.

In Fig. 4a and b we can observe the effect of changing the A_ω parameter in the function μ_1, and in Fig. 4c and d the effect of changing the B_ω parameter. Modelling of particle's dynamics by the ω function is realized for small values of $\|\mathbf{F}(\mathbf{z}_n)\|$. It results in visible changes in the central part of the polynomiographs.

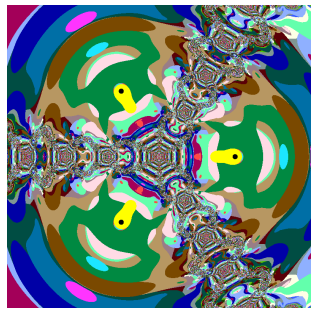
The effect of modelling the ω function, in which only the membership function μ_2 is used, i.e., $\omega_1 = \omega_3 = 0.0$ is shown in Fig. 5. The acceleration adaptation function η is the same as in the previous example. As it can be observed the μ_2 function significantly influences the polynomiographs smoothing. Areas



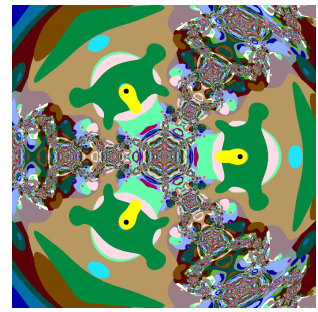
(a) $B_\omega = 10.0$, $t = 11.40$



(b) $A_\omega = 3.0$, $B_\omega = 10.0$, $t = 14.06$

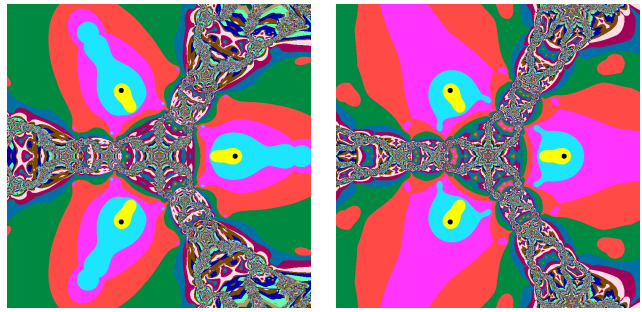


(c) $A_\omega = 1.0$, $B_\omega = 3.0$, $t = 10.80$



(d) $A_\omega = 1.0$, $B_\omega = 7.0$, $t = 12.29$

Figure 4: Polynomiographs of the Picard iteration for $\omega_1 = 0.7$, $\eta_3 = 0.7$ and varying A_ω and B_ω.



(a) $B_\omega = 2.0$, $C_\omega = 8.0$, $D_\omega = 10.0$, $t = 6.39$

(b) $B_\omega = 4.0$, $C_\omega = 6.0$, $D_\omega = 10.0$, $t = 5.58$

(c) $A_\omega = 0.2$, $B_\omega = 4.0$, $C_\omega = 6.0$, $D_\omega = 8.0$, $t = 5.42$

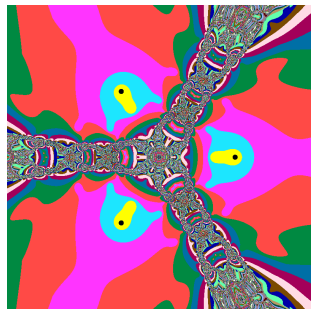
(d) $A_\omega = 0.5$, $B_\omega = 5.0$, $C_\omega = 6.0$, $D_\omega = 8.0$, $t = 5.50$

Figure 5: Polynomiographs of the Picard iteration for $\omega_2 = 0.7$, $\eta_3 = 0.7$ and varying A_ω, B_ω, C_ω, D_ω.

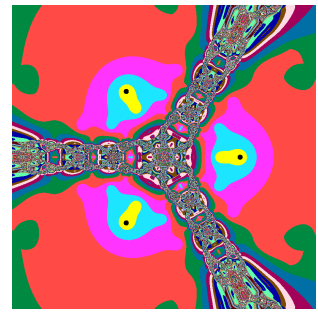
with high dynamics (areas presenting a wide range of changes) are not subject to numerous changes in comparison to the previous drawings.

In Fig. 6 the effect of modelling the ω function using only the μ_3 function ($\omega_1 = \omega_2 = 0.0$) is presented. As in the previous cases the same acceleration adaptation function η was used to obtain the polynomiographs. Modelling of particle's dynamics by the inertia adaptation function ω is within the range of high values of $\|\mathbf{F}(\mathbf{z}_n)\|$. We see that also in this case the use of different μ_3 membership functions significantly influences the smoothing of polynomiographs.

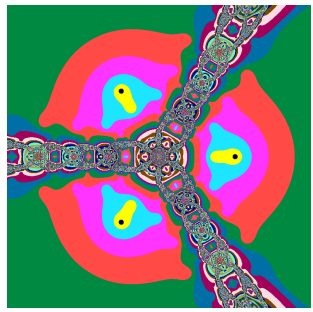
Modelling of dynamics by the acceleration adaptation function is special because the adaptation function should not have a value of 0 (it causes particle immobility) – the minimum and maximum values of the μ_1, μ_2, μ_3 membership functions are determined by the appropriate selection of coefficients. The membership functions μ_1 and μ_3 are used because of the specific selection of parameters for modelling acceleration adaptation function in the example presented in the Fig. 7 (this parameter description is consistent with the used convention, η_2 is equal to 0.0). Because the value of η_1 is higher than the value of η_3, so the μ_1 membership function has a greater impact on the dynamics shown in



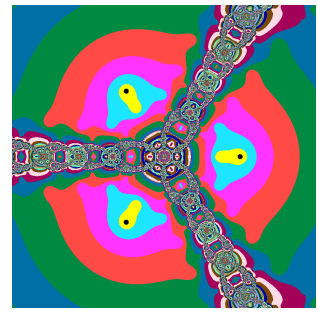
(a) $D_\omega = 5.0$, $t = 6.05$



(b) $C_\omega = 1.0$, $D_\omega = 5.0$, $t = 5.85$

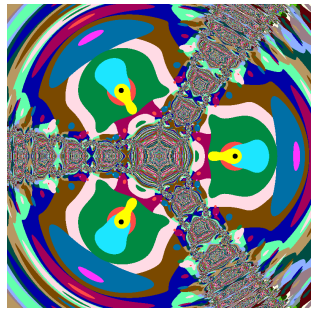


(c) $C_\omega = 3.0$, $D_\omega = 10.0$, $t = 5.54$

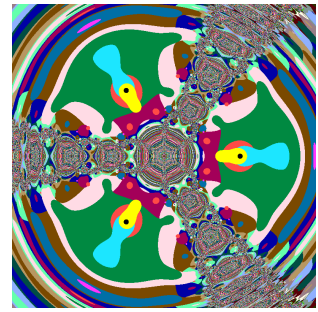


(d) $C_\omega = 8.0$, $D_\omega = 10.0$, $t = 5.64$

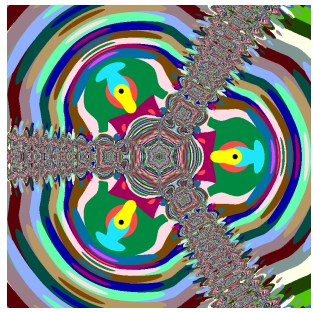
Figure 6: Polynomiographs of the Picard iteration for $\omega_3 = 0.7$, $\eta_3 = 0.7$ and varying C_ω and D_ω.



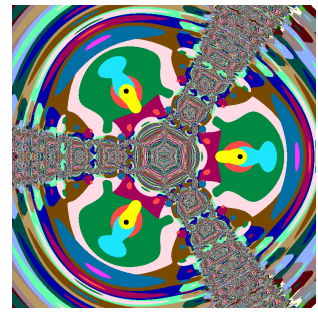
(a) $B_\eta = 10.0$, $D_\eta = 10.0$, $t = 12.46$



(b) $A_\eta = 3.0$, $B_\eta = 10.0$, $C_\eta = 3.0$, $D_\eta = 10.0$, $t = 12.51$



(c) $A_\eta = 1.0$, $B_\eta = 3.0$, $C_\eta = 1.0$, $D_\eta = 3.0$, $t = 15.19$



(d) $A_\eta = 1.0$, $B_\eta = 7.0$, $C_\eta = 1.0$, $D_\eta = 7.0$, $t = 13.45$

Figure 7: Polynomiographs of the Picard iteration for $\omega_3 = 0.5$, $\eta_1 = 0.7$, $\eta_3 = 0.1$ and varying A_η, B_η, C_η, D_η.

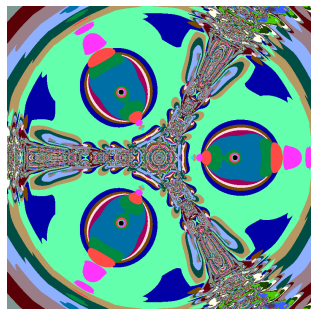
Fig. 7. The inertia adaptation function returns constant value equal to 0.5. For the images in Fig. 7a and b the parameter A_η changes, whereas for the images in Fig. 7c and d the parameter B_η is changing. The use of the η adaptation function causes a high acceleration of the particle for small values of $\|\mathbf{F}(\mathbf{z}_n)\|$.

The next example shows the modelling of acceleration adaptation function with the dominating influence of the μ_2 membership function (the shape of μ_2 is modelled by changing A_η, B_η, C_η, D_η parameters). For the examples presented in Fig. 8a and b, high acceleration values are obtained in the range of small and medium values of $\|\mathbf{F}(\mathbf{z}_n)\|$ (membership function μ_1 is omitted and η_3 has a small value) and in images in Fig. 8c and d in the medium range (η_1 and η_3 both have a small value equal to 0.1). The images of particle dynamics in Fig. 8c and d are more smoothed in comparison to drawings in Fig. 8a and b. All these images show interesting patterns.

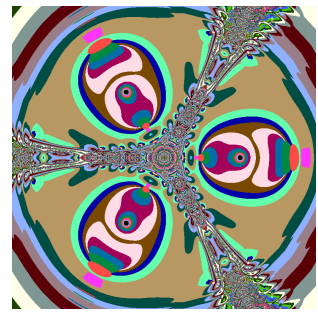
The modelling of acceleration adaptation function with the dominant influence of the μ_3 membership function because η_1 is equal to 0.1 and η_3 is equal to 0.7 (the μ_2 membership function is omitted) is shown in the next example. High values of particle acceleration shown in Fig. 9 are obtained for the high values

(a) $B_\eta = 2.0$, $C_\eta = 8.0$, $D_\eta = 10.0$, $t = 13.07$

(b) $B_\eta = 4.0$, $C_\eta = 6.0$, $D_\eta = 10.0$, $t = 21.44$

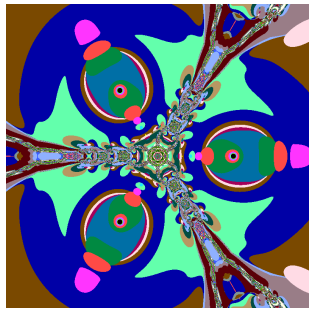


(c) $\eta_1 = 0.1$, $A_\eta = 0.2$, $B_\eta = 4.0$, $C_\eta = 6.0$, $D_\eta = 8.0$, $t = 12.76$

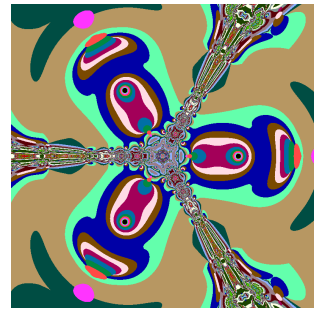


(d) $\eta_1 = 0.1$, $A_\eta = 0.5$, $B_\eta = 5.0$, $C_\eta = 6.0$, $D_\eta = 8.0$, $t = 13.18$

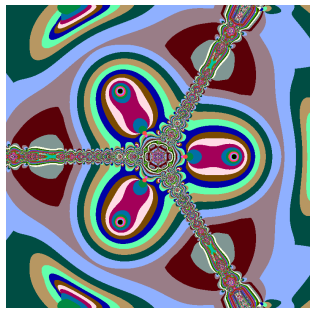
Figure 8: Polynomiographs of the Picard iteration for $\omega_3 = 0.5$, $\eta_2 = 0.7$, $\eta_3 = 0.1$ and varying A_η, B_η, C_η, D_η.



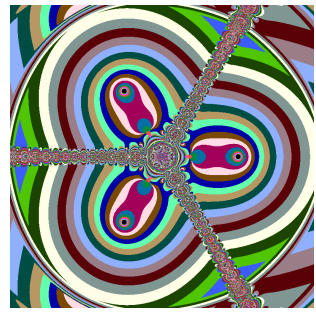
(a) $B_\eta = 5.0$, $D_\eta = 5.0$, $t = 10.38$



(b) $A_\eta = 1.0$, $B_\eta = 5.0$, $C_\eta = 1.0$, $D_\eta = 5.0$, $t = 11.70$



(c) $A_\eta = 3.0$, $B_\eta = 10.0$, $C_\eta = 3.0$, $D_\eta = 10.0$, $t = 13.69$



(d) $A_\eta = 8.0$, $B_\eta = 10.0$, $C_\eta = 8.0$, $D_\eta = 10.0$, $t = 15.29$

Figure 9: Polynomiographs of the Picard iteration for $\omega_3 = 0.5$, $\eta_1 = 0.1$, $\eta_3 = 0.7$ and varying A_η, B_η, C_η, D_η.

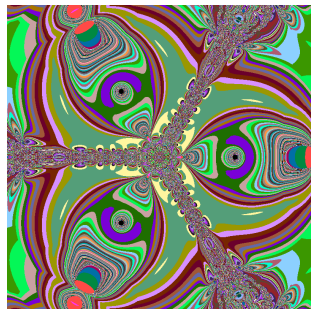
of $\|\mathbf{F}(\mathbf{z}_n)\|$. Modelling of particle dynamics presented on the polynomiographs in Figs. 9c and d is performed for higher values of $\|\mathbf{F}(\mathbf{z}_n)\|$ than for the images in Fig. 9a and b. All these pictures show interesting graphic designs.

The most interesting images are obtained by combining dynamics modelling by the ω and η adaptation functions. Polynomiographs in Fig. 10 present various images for different modelling parameters. They represent artistic designs.

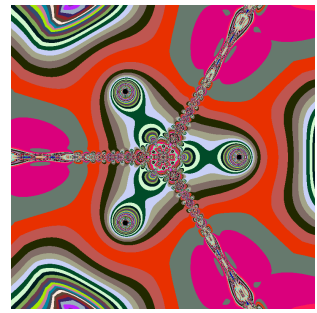
The central areas of the polynomiographs (areas of high dynamics) also create interesting mosaics. Magnification of the central parts of selected polynomiographs are shown in Fig. 11.

6.2. The Mann iteration

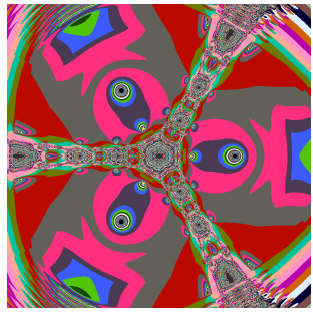
The Mann Iteration introduces the α parameter, which reduces the dynamics of the particle. The Picard iteration is obtained when $\alpha = 1$, and for the value equal to 0.0 the particle is in immobility. Fig. 12 shows images for the varying values of α, the remaining parameters are set as in the the example presented in Fig. 10a – it means that for $\alpha = 1$ we obtain the same image. Looking at the obtained images we see that the differences in particle dynamics



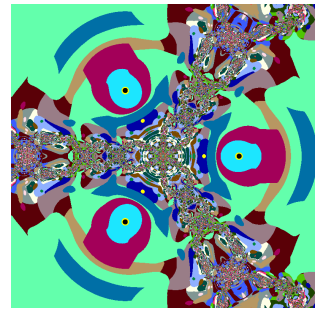
(a) $t = 77.75$



(b) $t = 110.95$

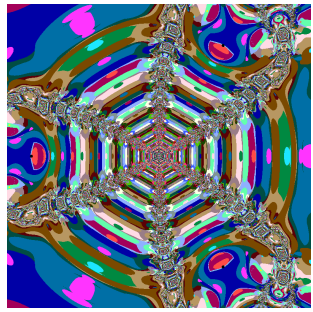


(c) $t = 26.48$

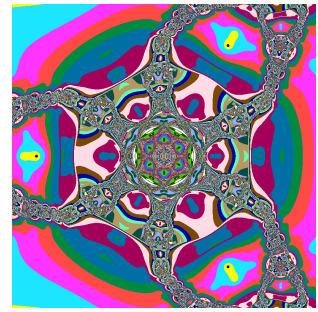


(d) $t = 12.49$

Figure 10: Polynomiographs of the Picard iteration for varying several parameters (parameters of examples presented in images: (a) $\omega_2 = 0.7$, $B_\omega = 0.5$, $C_\omega = 3.0$, $D_\omega = 10.0$, $\eta_3 = 0.7$, $D_\eta = 10.0$; (b) $\omega_2 = 0.5$, $B_\omega = 0.5$, $C_\omega = 3.0$, $D_\omega = 10.0$, $\eta_3 = 0.6$, $D_\eta = 10.0$; (c) $\omega_2 = 0.7$, $A_\omega = 0.2$, $B_\omega = 4.0$, $C_\omega = 6.0$, $D_\omega = 8.0$, $\eta_2 = 0.7$, $\eta_3 = 0.1$, $B_\eta = 2.0$, $C_\eta = 8.0$, $D_\eta = 10.0$; (d) $\omega_1 = 0.7$, $\omega_2 = 0.5$, $\omega_3 = 0.3$, $A_\omega = 0.6$, $B_\omega = 2.0$, $C_\omega = 3.0$, $D_\omega = 4.0$, $\eta_1 = 0.3$, $\eta_2 = 0.5$, $\eta_3 = 0.7$, $B_\eta = 1.0$, $C_\eta = 3.0$, $D_\eta = 5.0$).

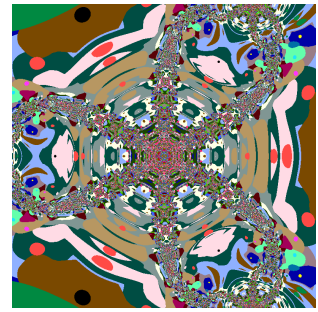


(a) Fig. 4c



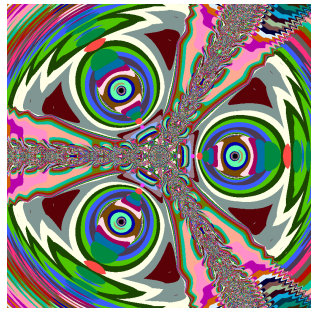
(b) Fig. 6b

(c) Fig. 8b

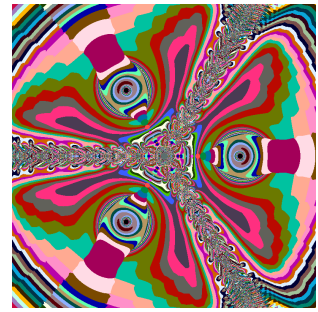


(d) Fig. 10d

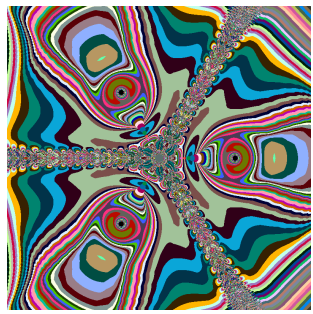
Figure 11: Magnification of the central part of selected polynomiographs of the Picard iteration.



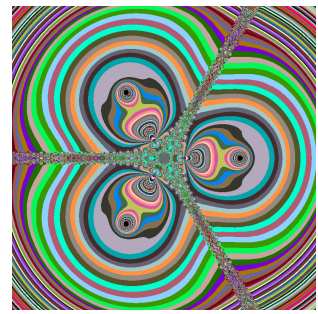
(a) $\alpha = 0.8$, $t = 20.93$



(b) $\alpha = 0.6$, $t = 24.96$



(c) $\alpha = 0.4$, $t = 32.71$

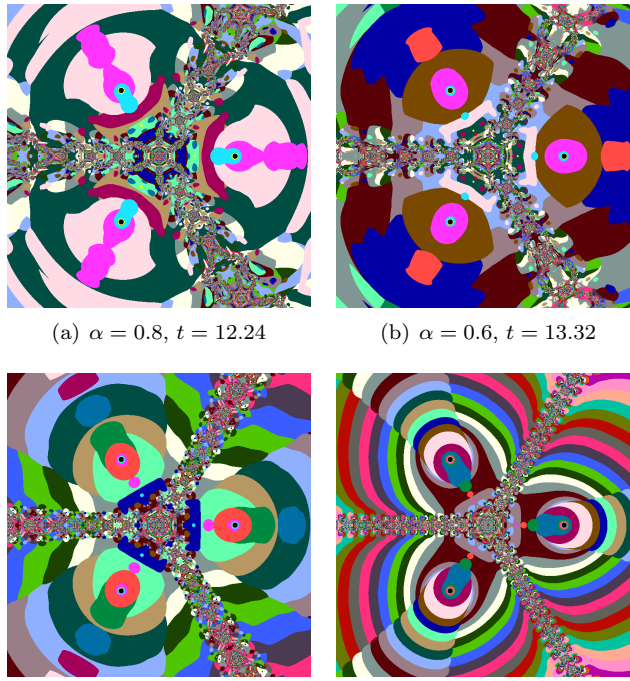


(d) $\alpha = 0.2$, $t = 75.02$

Figure 12: Polynomiographs of the Mann iteration for $\omega_1 = 0.7$, $A_\omega = 1.0$, $B_\omega = 3.0$, $\eta_2 = 0.7$, $\eta_3 = 0.1$, $B_\eta = 4.0$, $C_\eta = 6.0$, $D_\eta = 10.0$ and varying α.

are significant. Limiting the particle's dynamics increases the average number of iterations needed to create the polynomiographs. Moreover, comparing the images with the one obtained for the Picard iteration we see that the patterns differ in a significant way. The change in the shape is visible in all areas of the images.

Similar observations are made for the images in Fig. 13, but for other adaptation functions. In the previous example, the inertia adaptation function returned 0.7 in the range from 0.0 to $A_\omega = 1.0$ and then it decreased linearly to 0.0 for $B_\omega = 3.0$. Whereas the acceleration adaptation function grew linearly from 0.0 (for $A_\eta = 0.0$) to 0.7 (for $B_\eta = 4.0$), then (for $C_\eta = 6.0$) it decreased linearly to 0.1 (for $D_\eta = 10.0$), and for the arguments greater than D_η it returned 0.1. In the example from Fig. 13 the inertia adaptation function decreases from 0.7 to 0.3 (in the ranges defined by the values of A_ω, B_ω, C_ω and D_ω), whereas the acceleration adaptation function increases from 0.3 to 0.7 (in the ranges defined by A_η, B_η, C_η and D_η), so the parameters used in this example were the same as those used for the Picard iteration in Fig. 10d. It is not difficult to notice, by comparing the polynomiographs, that these adaptation functions cause different particle dynamics. Also for these images, with the decreasing value of α we notice a decrease in particle dynamics – it also results in the increase in the



(a) $\alpha = 0.8$, $t = 12.24$ (b) $\alpha = 0.6$, $t = 13.32$

(c) $\alpha = 0.4$, $t = 15.45$ (d) $\alpha = 0.2$, $t = 19.93$

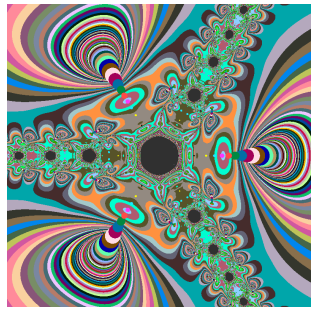
Figure 13: Polynomiographs of the Mann iteration for $\omega_1 = 0.7$, $\omega_2 = 0.5$, $\omega_3 = 0.3$, $A_\omega = 0.6$, $B_\omega = 2.0$, $C_\omega = 3.0$, $D_\omega = 4.0$, $\eta_1 = 0.3$, $\eta_2 = 0.5$, $\eta_3 = 0.7$, $B_\eta = 1.0$, $C_\eta = 3.0$, $D_\eta = 5.0$ and varying α.

average number of iterations needed to create the image.

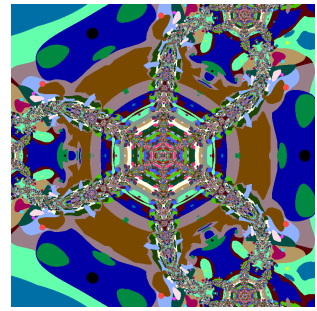
For the Mann iteration it is not difficult to see that, similarly to the case of Picard iteration, the particle dynamics paints attractive patterns. The magnification of the central parts of selected images is shown in Fig. 14. The obtained patterns are intriguing and have a high aesthetic value.

6.3. The Ishikawa and the Das-Debata iterations

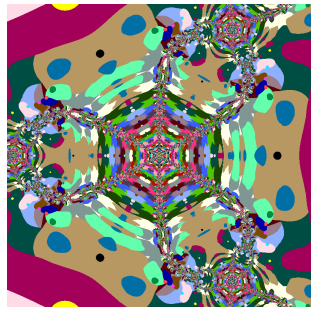
The Ishikawa iteration is a two step iteration. In the first step, a reference point is created using the Mann iteration. The β parameter is responsible for the dynamics of this point. In the second step, a new particle position is determined as the linear combination of the particle's position from the previous iteration and the processed reference point. The α parameter influences the particle dynamics. The Ishikawa iteration uses the same mappings during processing of the particle and the reference point – it also means that we have the same set of ω and η adaptation functions. The Das-Debata iteration is a generalization of the Ishikawa iteration by introducing two different mappings – it gives much wider range of possibilities for controlling particle dynamics.



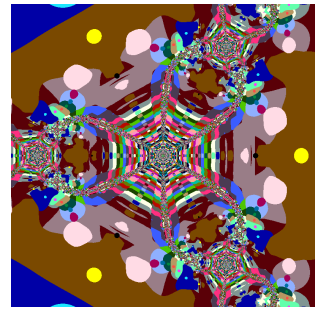
(a) Fig. 12d



(b) Fig. 13a

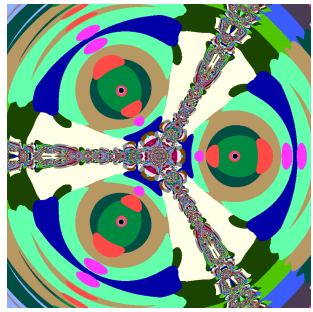


(c) Fig. 13b

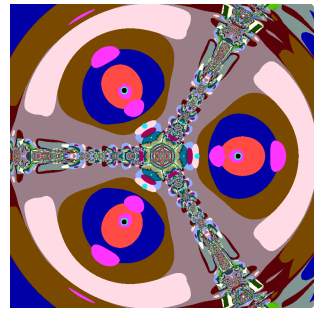


(d) Fig. 13c

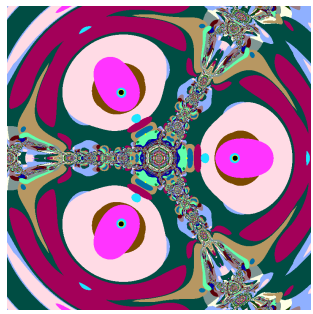
Figure 14: Magnification of the central part of selected polynomiographs of the Mann iteration.



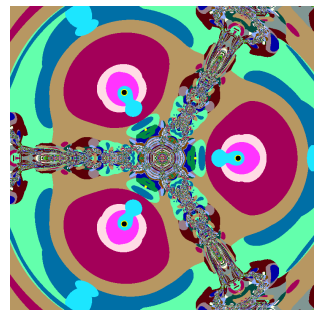
(a) $\beta = 0.2$, $t = 13.25$



(b) $\beta = 0.4$, $t = 11.10$



(c) $\beta = 0.6$, $t = 10.20$



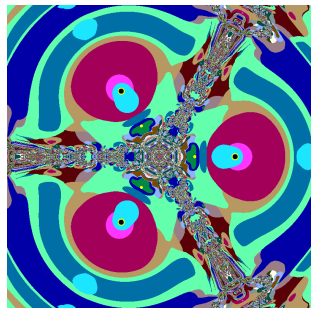
(d) $\beta = 0.8$, $t = 10.01$

Figure 15: Polynomiographs of the Das-Debata iteration for $\alpha = 0.9$, $\omega_1^I = 0.7$, $\omega_2^I = 0.5$, $\omega_3^I = 0.3$, $A_\omega^I = 0.6$, $B_\omega^I = 2.0$, $C_\omega^I = 3.0$, $D_\omega^I = 4.0$, $\eta_1^I = 0.3$, $\eta_2^I = 0.5$, $\eta_3^I = 0.7$, $B_\eta^I = 1.0$, $C_\eta^I = 3.0$, $D_\eta^I = 5.0$, $\omega_1^{II} = 0.7$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $\eta_2^{II} = 0.7$, $\eta_3^{II} = 0.1$, $B_\eta^{II} = 4.0$, $C_\eta^{II} = 6.0$, $D_\eta^{II} = 10.0$ and varying β.

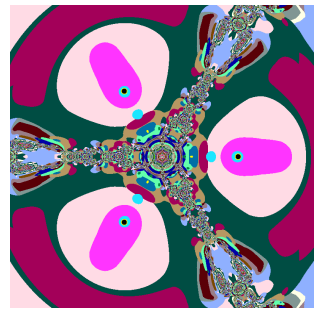
Images in Fig. 15 show the change in polynomiographs for the Das-Debata iteration in dependence on the β parameter. The parameters used for defining the $\mathbf{T}^I$ mapping are the same as in the example of the Picard iteration presented in Fig. 10a, and for the $\mathbf{T}^{II}$ mapping the same as in the example presented in the Fig. 10d. Although some settings are repeated in the description of the drawings, they are always given to facilitate their interpretation. The increase in the value of β increases the reference point dynamics. It results in decrease in the number of iterations needed to create the image.

Fig. 16 shows polynomiographs of the Das-Debata iteration for a varying α. The parameters that define the $\mathbf{T}^I$ and $\mathbf{T}^{II}$ transformations are the same as in the previous example. From the results we see that the decrease in particle dynamics by decreasing the value of the α parameter results in the increase in the average number of iterations needed to create the polynomiographs.

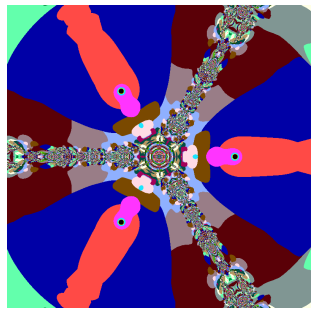
In the examples presented in Fig. 17 and 18 the parameters defining the $\mathbf{T}^I$ and $\mathbf{T}^{II}$ mappings have been changed. In the previous example, the values returned by the inertia adaptation function decreased from 0.7 to 0.3 and the values of the acceleration adaptation function increased from 0.3 to 0.7 (the



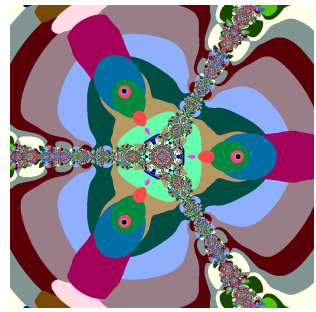
(a) $\alpha = 0.9$, $t = 9.89$



(b) $\alpha = 0.7$, $t = 10.07$

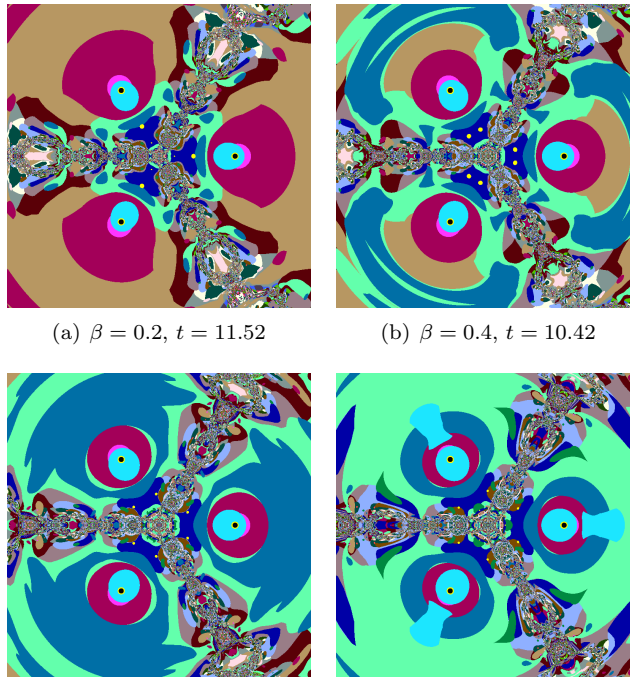


(c) $\alpha = 0.5$, $t = 11.52$



(d) $\alpha = 0.3$, $t = 13.84$

Figure 16: Polynomiographs of the Das-Debata iteration for $\beta = 0.9$, $\omega_1^I = 0.7$, $\omega_2^I = 0.5$, $\omega_3^I = 0.3$, $A_\omega^I = 0.6$, $B_\omega^I = 2.0$, $C_\omega^I = 3.0$, $D_\omega^I = 4.0$, $\eta_1^I = 0.3$, $\eta_2^I = 0.5$, $\eta_3^I = 0.7$, $B_\eta^I = 1.0$, $C_\eta^I = 3.0$, $D_\eta^I = 5.0$, $\omega_1^{II} = 0.7$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $\eta_2^{II} = 0.7$, $\eta_3^{II} = 0.1$, $B_\eta^{II} = 4.0$, $C_\eta^{II} = 6.0$, $D_\eta^{II} = 10.0$, and varying α.



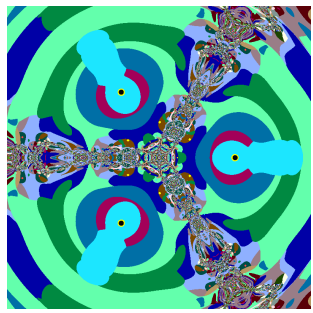
(a) $\beta = 0.2$, $t = 11.52$ (b) $\beta = 0.4$, $t = 10.42$

(c) $\beta = 0.6$, $t = 9.43$ (d) $\beta = 0.8$, $t = 10.15$

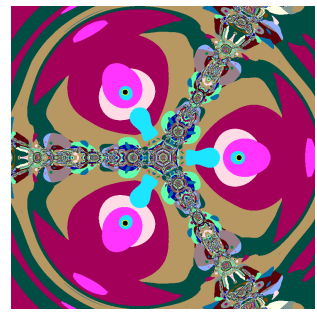
Figure 17: Polynomiographs of the Das-Debata iteration for $\alpha = 0.9$, $\omega_1^I = 0.7$, $A_\omega^I = 1.0$, $B_\omega^I = 3.0$, $\eta_2^I = 0.7$, $\eta_3^I = 0.1$, $B_\eta^I = 4.0$, $C_\eta^I = 6.0$, $D_\eta^I = 10.0$, $\omega_1^{II} = 0.7$, $\omega_2^{II} = 0.5$, $\omega_3^{II} = 0.3$, $A_\omega^{II} = 0.6$, $B_\omega^{II} = 2.0$, $C_\omega^{II} = 3.0$, $D_\omega^{II} = 4.0$, $\eta_1^{II} = 0.3$, $\eta_2^{II} = 0.5$, $\eta_3^{II} = 0.7$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 3.0$, $D_\eta^{II} = 5.0$ and varying β.

shapes of the functions result from the values of A, B, C and D parameters). In the given example, the particle inertia is high for the small values of $\|\mathbf{F}(\mathbf{z}_n)\|$ (it is due to the inertia adaptation function) and the particle acceleration is strongly stimulated for the medium values of $\|\mathbf{F}(\mathbf{z}_n)\|$ (it is due to the acceleration adaptation function). There is a noticeable difference between these examples showing the widening of the areas of high dynamics in the central part of polynomiographs. Fig. 17 shows lower dynamics for smaller values of β. It results in the increase in the average number of iterations. Similarly, the decrease in dynamics with decreasing values of α is observed in Fig. 17. In addition, it is possible to notice (by comparing the obtained polynomiographs) that the mixing of features of particle dynamic stimulation resulting from the shapes of ω and η adaptation functions is visible on polynomiographs.

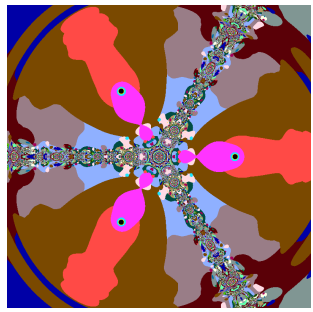
Selecting the appropriate parameters' values for the ω and η adaptation functions in the Das-Debata iteration gives the possibility to control the dynamics of the particle practically in the whole area of the polynomoiograph as it is presented in images in Fig. 19. Moreover, despite the limitations imposed by the Ishikawa iteration, a wide range of dynamic controls has been obtained –



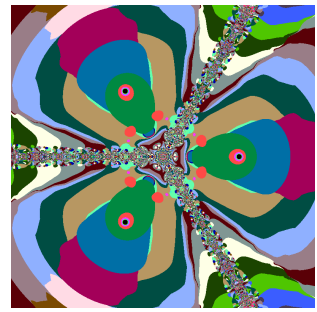
(a) $\alpha = 0.9$, $t = 9.36$



(b) $\alpha = 0.7$, $t = 10.08$



(c) $\alpha = 0.5$, $t = 10.70$



(d) $\alpha = 0.3$, $t = 12.44$

Figure 18: Polynomiographs of the Das-Debata iteration for $\beta = 0.9$, $\omega_1^I = 0.7$, $A_\omega^I = 1.0$, $B_\omega^I = 3.0$, $\eta_2^I = 0.7$, $\eta_3^I = 0.1$, $B_\eta^I = 4.0$, $C_\eta^I = 6.0$, $D_\eta^I = 10.0$, $\omega_1^{II} = 0.7$, $\omega_2^{II} = 0.5$, $\omega_3^{II} = 0.3$, $A_\omega^{II} = 0.6$, $B_\omega^{II} = 2.0$, $C_\omega^{II} = 3.0$, $D_\omega^{II} = 4.0$, $\eta_1^{II} = 0.3$, $\eta_2^{II} = 0.5$, $\eta_3^{II} = 0.7$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 3.0$, $D_\eta^{II} = 5.0$ and varying α.

it is shown in Fig. 20. Particle dynamics control gives the opportunity to create interesting graphic designs.

High particle dynamics contributes to creating very interesting patterns. Even fragments of polynomiographs can be an artistic inspiration. Fig. 21 shows the magnification of the central parts of some polynomiographs presented in the previous examples. We see that the patterns create interesting mosaics.

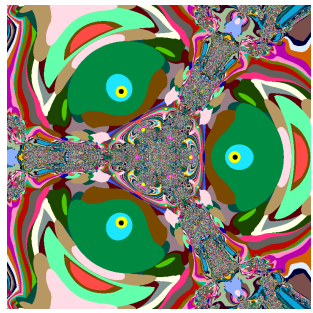
6.4. The Agarwal and the Khan-Cho-Abbas iterations

Compared to the Ishikawa iteration the Agarwal iteration extends the ability to control the particle dynamics by introducing additional transformation of $\mathbf{z}_n$ – see equation (19). In the following example (Fig. 22) the same parameters as in the example in Fig. 10 were used for defining mapping $\mathbf{T}$. This example illustrates the change in particle dynamics obtained by changing the dynamics of the reference point creation. The dynamics of creating a reference point depends on the β parameter. The decrease in the value of the β parameter reduces the dynamics of the reference point – it clearly affects the dynamics of the particle. The increase in the dynamics of the reference point creation decreases the average number of iterations required to create the polynomiograph (see Fig. 22).

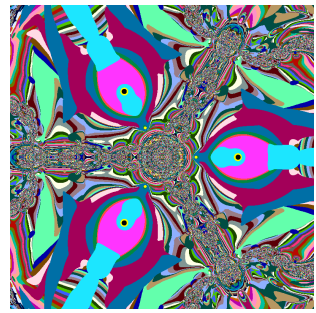
The next example – presented in Fig. 23 – shows the effect of changing the α parameter on particle dynamics for the same settings as in the previous example. As in the cases discussed above, reducing the particle dynamics by decreasing the value of α results in an increase in the average number of iterations needed to create the polynomiograph – as it is shown in Fig. 23. It is possible to note that α affects the linear combination of the processed particle position and the processed reference point position. Moreover, we can observe a remarkable resemblance between polynomiographs resulting from the shapes of the adaptation functions. The inertia adaptation function returns high values for small values of $\|\mathbf{F}(\mathbf{z}_n)\|$, and the acceleration adaptation function returns high values for medium values of $\|\mathbf{F}(\mathbf{z}_n)\|$. The polynomiographs shown in Figs. 22 and 23 are particularly similar in the area of extremes.

The Khan-Cho-Abbas iteration introduces two different mappings, which in comparison to the Agarwal iteration extends the ability of controlling the particle dynamics. In the example the parameters for the mappings $\mathbf{T}^I$, $\mathbf{T}^{III}$ were the same as in the case of the image presented in Fig. 10a, and for the $\mathbf{T}^{II}$ mapping the same as in Fig. 20a. Looking at the parameters used to generate the images in Fig. 24 we can observe that the increase in value of β reduces the average number of iterations needed for image creation. The obtained polynomiographs confirm that the introduction of two different mappings gives a wide range of possibilities to control the dynamics of the particle motion. It can be said that in this case particle dynamics paint images like a brush.

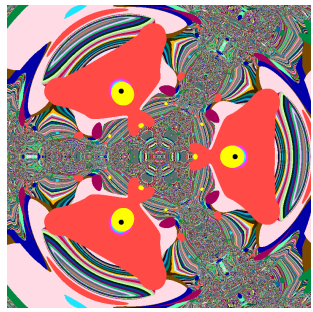
Images in Fig. 25 show the change of particle dynamics due to the value's change of the α parameter. In general, the decrease in the α value causes the increase in the average number of iterations required for the image creation, but for Fig. 25d, this value has decreased in relation to Fig. 25c. It can be concluded



(a) $t = 20.61$



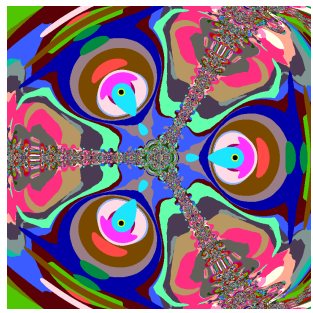
(b) $t = 15.19$



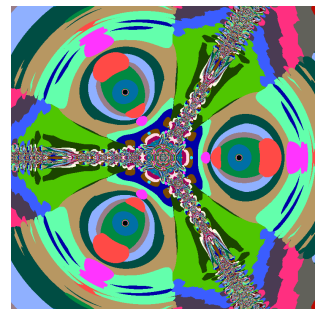
(c) $t = 34.66$

(d) $t = 16.70$

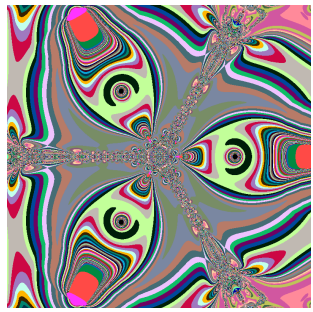
Figure 19: Polynomiographs of the Das-Debata iteration for varying several parameters (parameters of examples presented in images: (a) $\alpha = 0.9$, $\beta = 0.8$, $\omega_1^I = 0.7$, $\omega_2^I = 0.5$, $\omega_3^I = 0.9$, $A_\omega^I = 0.5$, $B_\omega^I = 2.0$, $C_\omega^I = 3.0$, $D_\omega^I = 40.0$, $\eta_1^I = 0.3$, $\eta_2^I = 0.5$, $\eta_3^I = 0.9$, $A_\eta^I = 0.1$, $B_\eta^I = 1.0$, $C_\eta^I = 3.0$, $D_\eta^I = 5.0$, $\omega_1^{II} = 0.9$, $\omega_2^{II} = 0.7$, $\omega_3^{II} = 0.5$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $C_\omega^{II} = 5.0$, $D_\omega^{II} = 10.0$, $\eta_1^{II} = 0.5$, $\eta_2^{II} = 0.9$, $\eta_3^{II} = 0.2$, $A_\eta^{II} = 1.0$, $B_\eta^{II} = 2.0$, $C_\eta^{II} = 5.0$, $D_\eta^{II} = 100.0$; (b) $\alpha = 0.9$, $\beta = 0.8$, $\omega_2^I = 0.7$, $B_\omega^I = 0.5$, $C_\omega^I = 3.0$, $D_\omega^I = 10.0$, $\eta_1^I = 0.5$, $\eta_2^I = 0.9$, $\eta_3^I = 0.2$, $A_\eta^I = 1.0$, $B_\eta^I = 2.0$, $C_\eta^I = 5.0$, $D_\eta^I = 100.0$, $\omega_1^{II} = 0.9$, $\omega_2^{II} = 0.7$, $\omega_3^{II} = 0.5$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $C_\omega^{II} = 5.0$, $D_\omega^{II} = 10.0$, $\eta_3^{II} = 0.7$, $D_\eta^{II} = 10.0$; (c) $\alpha = 0.9$, $\beta = 0.9$, $\omega_1^I = 0.7$, $\omega_2^I = 0.8$, $\omega_3^I = 0.9$, $A_\omega^I = 1.0$, $B_\omega^I = 3.0$, $C_\omega^I = 8.0$, $D_\omega^I = 40.0$, $\eta_1^I = 0.5$, $\eta_2^I = 0.7$, $\eta_3^I = 0.1$, $A_\eta^I = 0.1$, $B_\eta^I = 4.0$, $C_\eta^I = 6.0$, $D_\eta^I = 10.0$, $\omega_1^{II} = 0.7$, $\omega_2^{II} = 0.5$, $\omega_3^{II} = 0.3$, $A_\omega^{II} = 0.6$, $B_\omega^{II} = 2.0$, $C_\omega^{II} = 3.0$, $D_\omega^{II} = 4.0$, $\eta_1^{II} = 0.8$, $\eta_2^{II} = 0.5$, $\eta_3^{II} = 0.8$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 3.0$, $D_\eta^{II} = 5.0$; (d) $\alpha = 0.9$, $\beta = 0.9$, $\omega_1^I = 0.7$, $\omega_3^I = 0.7$, $B_\omega^I = 1.0$, $C_\omega^I = 1.0$, $D_\omega^I = 3.0$, $\eta_2^I = 0.7$, $\eta_3^I = 0.1$, $B_\eta^I = 4.0$, $C_\eta^I = 6.0$, $D_\eta^I = 10.0$, $\omega_1^{II} = 0.7$, $\omega_2^{II} = 0.5$, $\omega_3^{II} = 0.3$, $A_\omega^{II} = 0.6$, $B_\omega^{II} = 2.0$, $C_\omega^{II} = 3.0$, $D_\omega^{II} = 4.0$, $\eta_3^{II} = 0.5$, $D_\eta^{II} = 8.0$).



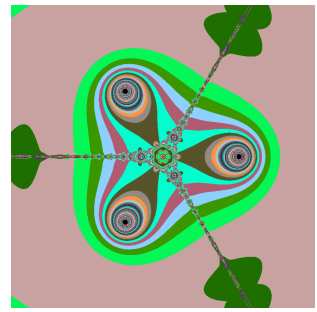
(a) $t = 17.19$



(b) $t = 14.34$

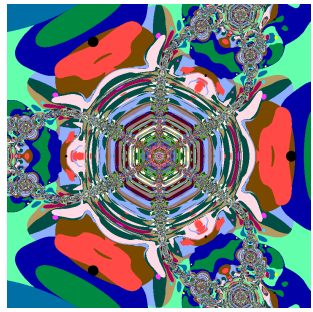


(c) $t = 46.25$

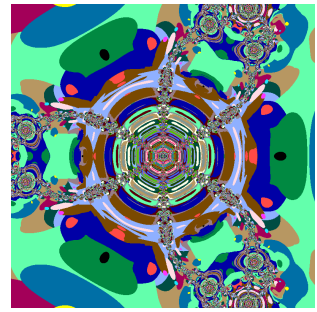


(d) $t = 76.42$

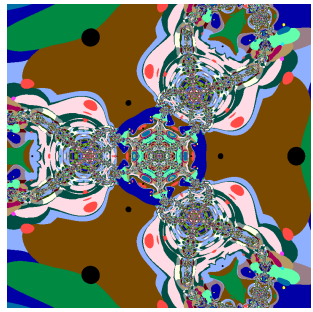
Figure 20: Polynomiographs of the Ishikawa iteration for $\alpha = 0.9$, $\beta = 0.9$ and varying several parameters (parameters of examples presented in images: (a) $\omega_1 = 0.9$, $\omega_2 = 0.7$, $\omega_3 = 0.8$, $A_\omega = 1.0$, $B_\omega = 3.0$, $C_\omega = 5.0$, $D_\omega = 10.0$, $\eta_1 = 0.2$, $\eta_2 = 0.1$, $\eta_3 = 0.5$, $A_\eta = 0.5$, $B_\eta = 1.0$, $C_\eta = 5.0$, $D_\eta = 10.0$; (b) $\omega_1 = 0.7$, $\omega_2 = 0.3$, $\omega_3 = 0.5$, $A_\omega = 0.6$, $B_\omega = 1.0$, $C_\omega = 2.0$, $D_\omega = 3.0$, $\eta_2 = 0.7$, $\eta_3 = 0.1$, $B_\eta = 4.0$, $C_\eta = 5.0$, $D_\eta = 6.0$; (c) $\omega_2 = 0.7$, $B_\omega = 0.5$, $C_\omega = 3.0$, $D_\omega = 10.0$, $\eta_3 = 0.7$, $D_\eta = 10.0$; (d) $\omega_3 = 0.7$, $D_\omega = 10.0$, $\eta_3 = 0.7$, $D_\eta = 10.0$).



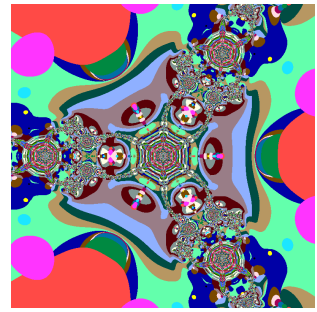
(a) Fig. 15d



(b) Fig. 16b

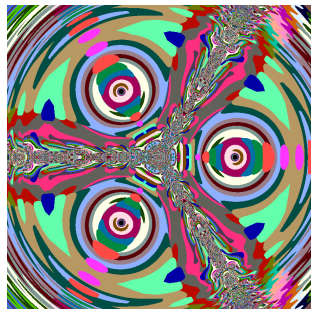


(c) Fig. 17a

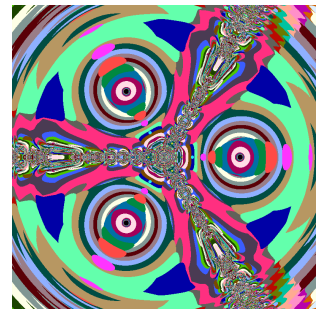


(d) Fig. 18d

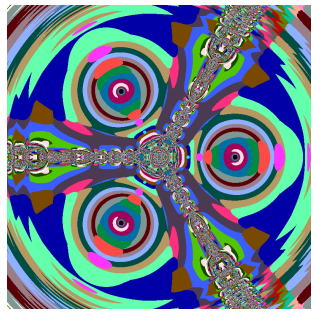
Figure 21: Magnification of the central part of selected polynomiographs of the Das-Debata iteration.



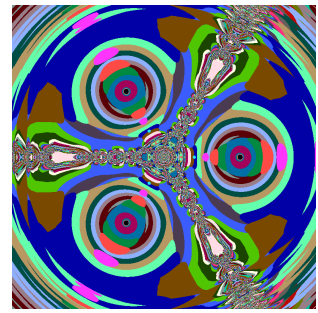
(a) $\beta = 0.2$, $t = 16.28$



(b) $\beta = 0.4$, $t = 15.29$

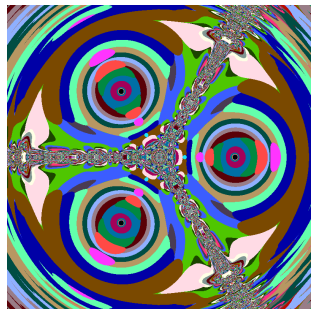


(c) $\beta = 0.6$, $t = 14.25$

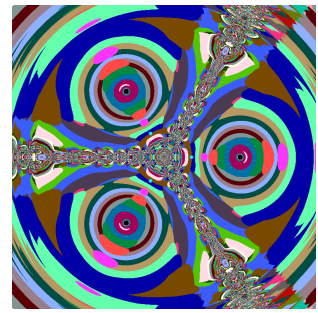


(d) $\beta = 0.8$, $t = 13.37$

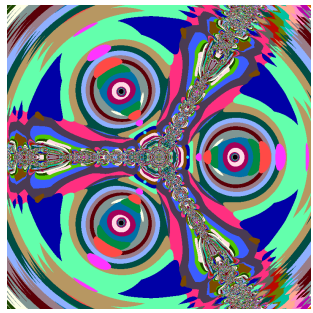
Figure 22: Polynomiographs of the Agarwal iteration for $\alpha = 0.5$, $\omega_1 = 0.7$, $A_\omega = 1.0$, $B_\omega = 3.0$, $\eta_2 = 0.7$, $\eta_3 = 0.1$, $B_\eta = 4.0$, $C_\eta = 6.0$, $D_\eta = 10.0$ and varying β.



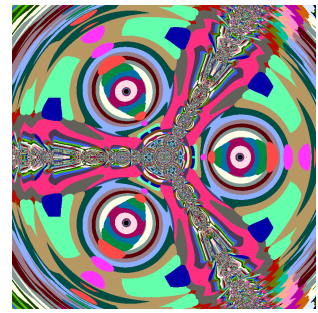
(a) $\alpha = 0.9$, $t = 12.95$



(b) $\alpha = 0.7$, $t = 13.73$

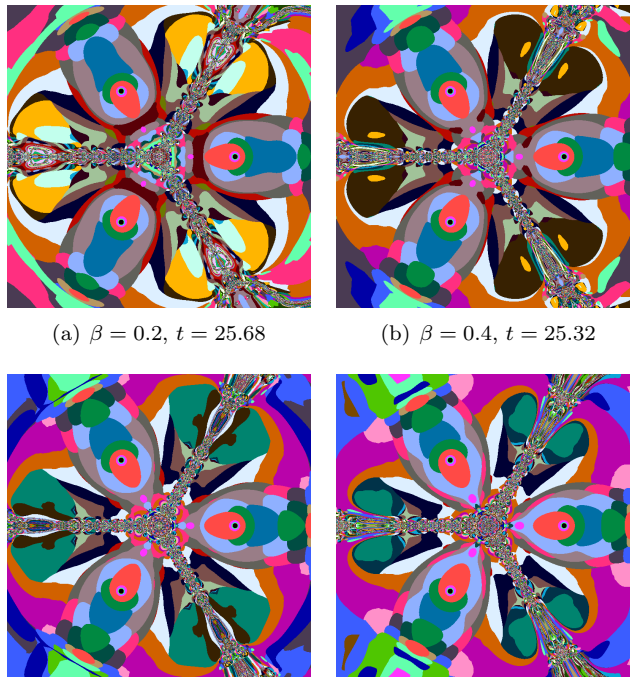


(c) $\alpha = 0.5$, $t = 14.87$



(d) $\alpha = 0.3$, $t = 15.75$

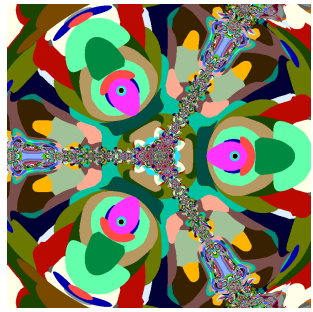
Figure 23: Polynomiographs of the Agarwal iteration for $\beta = 0.5$, $\omega_1 = 0.7$, $A_\omega = 1.0$, $B_\omega = 3.0$, $\eta_2 = 0.7$, $\eta_3 = 0.1$, $B_\eta = 4.0$, $C_\eta = 6.0$, $D_\eta = 10.0$ and varying α.



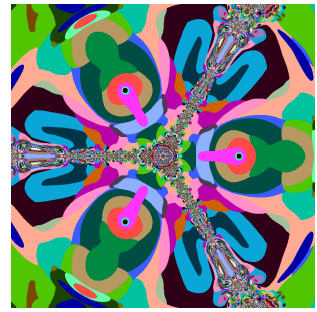
(a) $\beta = 0.2$, $t = 25.68$ (b) $\beta = 0.4$, $t = 25.32$

(c) $\beta = 0.6$, $t = 24.75$ (d) $\beta = 0.8$, $t = 23.98$

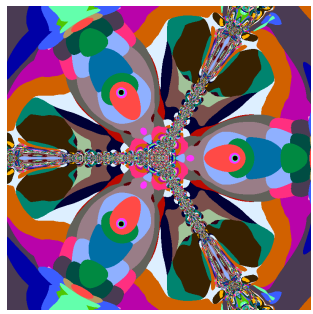
Figure 24: Polynomiographs of the Khan-Cho-Abbas iteration for $\alpha = 0.5$, $\omega_2^{\{I|III\}} = 0.7$, $B_\omega^{\{I|III\}} = 0.5$, $C_\omega^{\{I|III\}} = 3.0$, $D_\omega^{\{I|III\}} = 10.0$, $\eta_3^{\{I|III\}} = 0.7$, $D_\eta^{\{I|III\}} = 10.0$, $\omega_1^{II} = 0.9$, $\omega_2^{II} = 0.7$, $\omega_3^{II} = 0.8$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $C_\omega^{II} = 5.0$, $D_\omega^{II} = 10.0$, $\eta_1^{II} = 0.2$, $\eta_2^{II} = 0.1$, $\eta_3^{II} = 0.5$, $A_\eta^{II} = 0.5$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 5.0$, $D_\eta^{II} = 10.0$ and varying β.



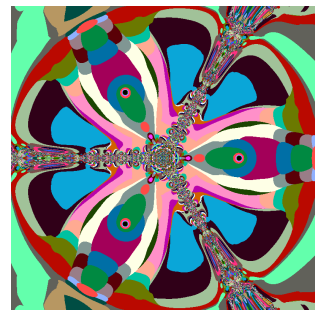
(a) $\alpha = 0.9$, $t = 23.39$



(b) $\alpha = 0.7$, $t = 24.14$



(c) $\alpha = 0.5$, $t = 24.87$



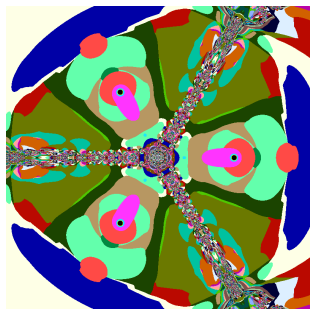
(d) $\alpha = 0.3$, $t = 24.38$

Figure 25: Polynomiographs of the Khan-Cho-Abbas iteration for $\beta = 0.5$, $\omega_2^{\{I|III\}} = 0.7$, $B_\omega^{\{I|III\}} = 0.5$, $C_\omega^{\{I|III\}} = 3.0$, $D_\omega^{\{I|III\}} = 10.0$, $\eta_3^{\{I|III\}} = 0.7$, $D_\eta^{\{I|III\}} = 10.0$, $\omega_1^{II} = 0.9$, $\omega_2^{II} = 0.7$, $\omega_3^{II} = 0.8$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $C_\omega^{II} = 5.0$, $D_\omega^{II} = 10.0$, $\eta_1^{II} = 0.2$, $\eta_2^{II} = 0.1$, $\eta_3^{II} = 0.5$, $A_\eta^{II} = 0.5$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 5.0$, $D_\eta^{II} = 10.0$ and varying α.

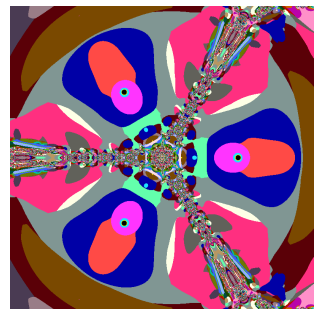
that the dependencies influencing the particle dynamics become more complex due to the different settings of parameters of the adaptation functions.

As in the examples for the Agarwal iteration, in both cases of dynamics modelling by changing β and α parameters the common features of the obtained images are shown. All these images can be an artistic inspiration.

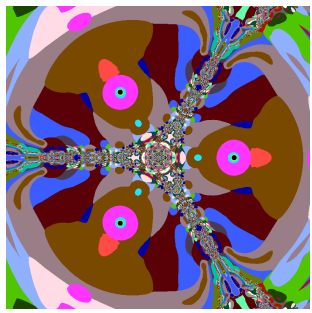
The generalized Agarwal iteration in comparison to the Khan-Cho-Abbas iteration extends the ability to control the particle by introducing a third mapping. In the following example the parameters used to define the three mappings were the following. For the $\mathbf{T}^I$ transformation they were the same as in the case of Fig. 10d, for the $\mathbf{T}^{II}$ transformation the ω, η adaptation functions were defined as in Fig. 10d and 12, respectively, and for the $\mathbf{T}^{III}$ transformation the parameters were the same as in the case of Fig. 20a. The obtained polynomiographs are presented in Fig. 26 and 27. As in the previous examples, the decrease in the value of β and α causes the increase in the average number of iterations needed for creating the polynomiograph. Due to the different mappings, the common features of the images are blurred. Polynomiographs created with high particle dynamics look like they were painted.



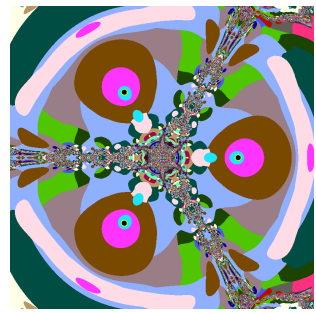
(a) $\beta = 0.2$, $t = 18.41$



(b) $\beta = 0.4$, $t = 15.48$

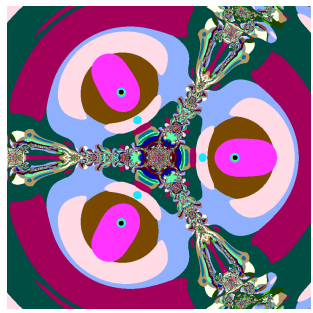


(c) $\beta = 0.6$, $t = 14.15$

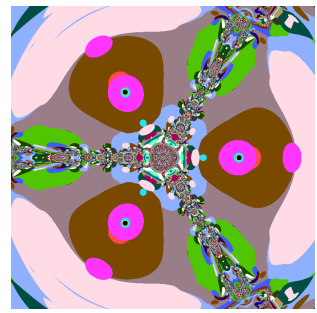


(d) $\beta = 0.8$, $t = 13.19$

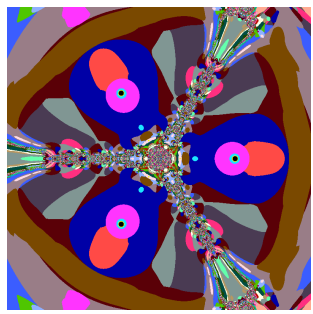
Figure 26: Polynomiographs of the generalized Agarwal iteration for $\alpha = 0.5$, $\omega_1^I = 0.7$, $\omega_2^I = 0.5$, $\omega_3^I = 0.3$, $A_\omega^I = 0.6$, $B_\omega^I = 2.0$, $C_\omega^I = 3.0$, $D_\omega^I = 4.0$, $\eta_1^I = 0.3$, $\eta_2^I = 0.5$, $\eta_3^I = 0.7$, $B_\eta^I = 1.0$, $C_\eta^I = 3.0$, $D_\eta^I = 5.0$, $\omega_1^{II} = 0.7$, $\omega_2^{II} = 0.5$, $\omega_3^{II} = 0.3$, $A_\omega^{II} = 0.6$, $B_\omega^{II} = 2.0$, $C_\omega^{II} = 3.0$, $D_\omega^{II} = 4.0$, $\eta_2^{II} = 0.7$, $\eta_3^{II} = 0.1$, $B_\eta^{II} = 4.0$, $C_\eta^{II} = 6.0$, $D_\eta^{II} = 10.0$, $\omega_1^{III} = 0.9$, $\omega_2^{III} = 0.7$, $\omega_3^{III} = 0.8$, $A_\omega^{III} = 1.0$, $B_\omega^{III} = 3.0$, $C_\omega^{III} = 5.0$, $D_\omega^{III} = 10.0$, $\eta_1^{III} = 0.2$, $\eta_2^{III} = 0.1$, $\eta_3^{III} = 0.5$, $A_\eta^{III} = 0.5$, $B_\eta^{III} = 1.0$, $C_\eta^{III} = 5.0$, $D_\eta^{III} = 10.0$ and varying β.



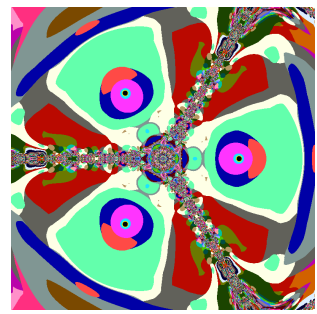
(a) $\alpha = 0.9$, $t = 10.53$



(b) $\alpha = 0.7$, $t = 12.01$



(c) $\alpha = 0.5$, $t = 14.09$



(d) $\alpha = 0.3$, $t = 17.57$

Figure 27: Polynomiographs of the generalized Agarwal iteration for $\beta = 0.5$, $\omega_1^I = 0.7$, $\omega_2^I = 0.5$, $\omega_3^I = 0.3$, $A_\omega^I = 0.6$, $B_\omega^I = 2.0$, $C_\omega^I = 3.0$, $D_\omega^I = 4.0$, $\eta_1^I = 0.3$, $\eta_2^I = 0.5$, $\eta_3^I = 0.7$, $B_\eta^I = 1.0$, $C_\eta^I = 3.0$, $D_\eta^I = 5.0$, $\omega_1^{II} = 0.7$, $\omega_2^{II} = 0.5$, $\omega_3^{II} = 0.3$, $A_\omega^{II} = 0.6$, $B_\omega^{II} = 2.0$, $C_\omega^{II} = 3.0$, $D_\omega^{II} = 4.0$, $\eta_2^{II} = 0.7$, $\eta_3^{II} = 0.1$, $B_\eta^{II} = 4.0$, $C_\eta^{II} = 6.0$, $D_\eta^{II} = 10.0$, $\omega_1^{III} = 0.9$, $\omega_2^{III} = 0.7$, $\omega_3^{III} = 0.8$, $A_\omega^{III} = 1.0$, $B_\omega^{III} = 3.0$, $C_\omega^{III} = 5.0$, $D_\omega^{III} = 10.0$, $\eta_1^{III} = 0.2$, $\eta_2^{III} = 0.1$, $\eta_3^{III} = 0.5$, $A_\eta^{III} = 0.5$, $B_\eta^{III} = 1.0$, $C_\eta^{III} = 5.0$, $D_\eta^{III} = 10.0$ and varying α.

To illustrate various visual effects resulting from the dynamic control capabilities by means of the ω and η adaptation functions, in Fig. 28 the Ishikawa iteration, the Khan-Cho-Abbas iteration and the generalized Agarwal iteration are presented for varying several parameters. The obtained images confirm the possibility to create artistic patterns. The magnification of some selected polynomiographs (Fig. 29) also shows artistic mosaics.

7. Conclusions

In this paper we proposed an algorithm that is based on the Newton-Raphson root finding method and which uses an adaptation mechanics. The adaptation mechanics is based on a linear combination of some membership functions and allows a better control of algorithm's dynamics. The proposed approach allows to visualize the adaptation mechanics impact on the operation of the algorithm. Moreover, various iteration processes and their operation mechanics were discussed. The presented linear combination method is very simple. Its use shows how the dynamics of a particle can be controlled. Similar approaches are used in control systems [35]. It is possible to optimize the adaptation functions, e.g., by means of a genetic algorithm. The presented images are also of aesthetic importance. They can be used as art projects.

References

[1] E. Polak, Optimization Algorithms and Consistent Approximations, Springer-Verlag, New York, 1997. doi:10.1007/978-1-4612-0663-7.

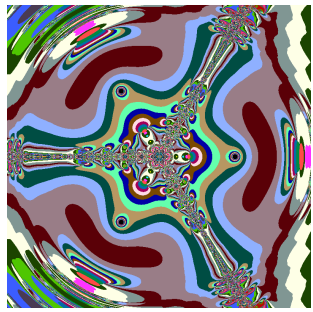
[2] I. Gosciniak, Discussion on semi-immune algorithm behaviour based on fractal analysis, Soft Computing 21 (14) (2017) 3945–3956. doi:10.1007/s00500-016-2044-y.

[3] I. Gosciniak, Immune algorithm in non-stationary optimization task, in: 2008 International Conference on Computational Intelligence for Modelling Control Automation, 2008, pp. 750–755. doi:10.1109/CIMCA.2008.181.

[4] T. Weise, Global Optimization Algorithms – Theory and Application, 2nd Edition, Online available at http://www.it-weise.de/projects/book.pdf, 2009.

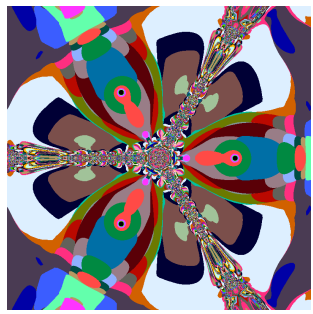
[5] Y. Shi, R. Eberhart, A modified particle swarm optimizer, in: Proceedings of IEEE International Conference on Evolutionary Computation, IEEE Computer Society, Washington, DC, USA, 1998, pp. 69–73. doi:10.1109/ICEC.1998.699146.

[6] R. Eberhart, Y. Shi, Tracking and optimizing dynamic systems with particle swarms, in: Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), Vol. 1, 2001, pp. 94–100. doi:10.1109/CEC.2001.934376.

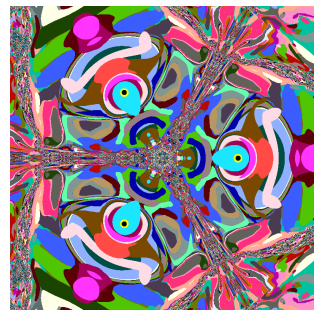


(a) $t = 15.36$

(b) $t = 21.71$

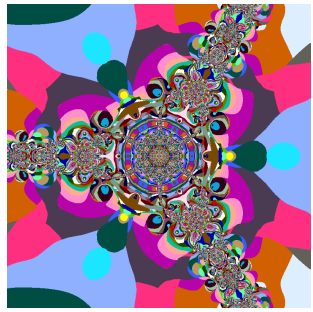


(c) $t = 23.73$

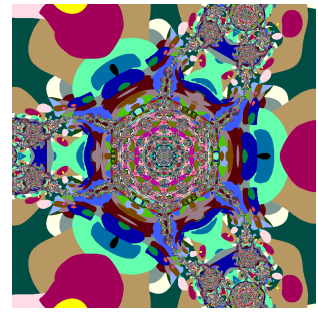


(d) $t = 19.65$

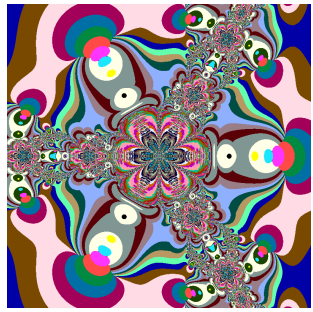
Figure 28: Polynomiographs of the Agarwal iteration (a, b), the Khan-Cho-Abbas iteration (c) and generalized Agarwal iteration (d) for varying several parameters (parameters of examples presented in images: (a) $\alpha = 0.5$, $\beta = 0.2$, $\omega_1 = 0.7$, $\omega_2 = 0.1$, $\omega_3 = 0.7$, $B_\omega = 1.0$, $C_\omega = 1.0$, $D_\omega = 3.0$, $\eta_2 = 0.7$, $\eta_3 = 0.1$, $B_\eta = 4.0$, $C_\eta = 6.0$, $D_\eta = 8.0$; (b) $\alpha = 0.2$, $\beta = 0.5$, $\omega_1 = 0.9$, $\omega_2 = 0.7$, $\omega_3 = 0.1$, $B_\omega = 4.0$, $C_\omega = 6.0$, $D_\omega = 8.0$, $\eta_1 = 0.7$, $\eta_3 = 0.7$, $B_\eta = 1.0$, $C_\eta = 1.0$, $D_\eta = 3.0$; (c) $\alpha = 0.4$, $\beta = 0.9$, $\omega_2^{\{I|III\}} = 0.7$, $B_\omega^{\{I|III\}} = 0.5$, $C_\omega^{\{I|III\}} = 3.0$, $D_\omega^{\{I|III\}} = 10.0$, $\eta_3^{\{I|III\}} = 0.7$, $D_\eta^{\{I|III\}} = 10.0$, $\omega_1^{II} = 0.9$, $\omega_2^{II} = 0.7$, $\omega_3^{II} = 0.8$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $C_\omega^{II} = 5.0$, $D_\omega^{II} = 10.0$, $\eta_1^{II} = 0.2$, $\eta_2^{II} = 0.1$, $\eta_3^{II} = 0.5$, $A_\eta^{II} = 0.5$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 5.0$, $D_\eta^{II} = 10.0$; (d) $\alpha = 0.5$, $\beta = 0.9$, $\eta_1^I = 0.9$, $\eta_2^I = 0.7$, $\eta_3^I = 0.1$, $B_\eta^I = 4.0$, $C_\eta^I = 6.0$, $D_\eta^I = 8.0$, $\omega_1^I = 0.7$, $\omega_3^I = 0.7$, $B_\omega^I = 1.0$, $C_\omega^I = 1.0$, $D_\omega^I = 3.0$, $\omega_1^{II} = 0.9$, $\omega_2^{II} = 0.7$, $\omega_3^{II} = 0.8$, $A_\omega^{II} = 1.0$, $B_\omega^{II} = 3.0$, $C_\omega^{II} = 5.0$, $D_\omega^{II} = 10.0$, $\eta_1^{II} = 0.2$, $\eta_2^{II} = 0.1$, $\eta_3^{II} = 0.5$, $A_\eta^{II} = 0.5$, $B_\eta^{II} = 1.0$, $C_\eta^{II} = 5.0$, $D_\eta^{II} = 10.0$, $\omega_1^I = 0.9$, $\omega_2^I = 0.7$, $\omega_3^I = 0.8$, $A_\omega^I = 1.0$, $B_\omega^I = 3.0$, $C_\omega^I = 5.0$, $D_\omega^I = 10.0$, $\eta_1^I = 0.2$, $\eta_2^I = 0.1$, $\eta_3^I = 0.5$, $A_\eta^I = 0.5$, $B_\eta^I = 1.0$, $C_\eta^I = 5.0$, $D_\eta^I = 10.0$).



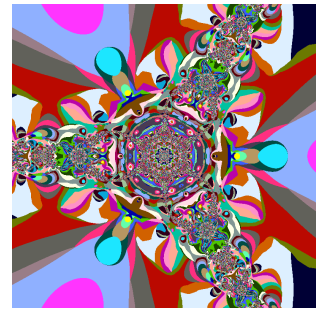
(a) Fig. 24c



(b) Fig. 26d



(c) Fig. 28a



(d) Fig. 28c

Figure 29: Magnification of the central part of selected polynomiographs of the Agarwal and Khan-Cho-Abbas iterations.

[7] Y. Shi, R. Eberhart, Empirical study of particle swarm optimization, in: Proceedings of the 1999 Congress on Evolutionary Computation-CEC99 (Cat. No. 99TH8406), Vol. 3, 1999, pp. 1945–1950. doi:10.1109/CEC.1999.785511.

[8] R. Malik, T. Rahman, S. Hashim, R. Ngah, New particle swarm optimizer with sigmoid increasing inertia weight, International Journal of Computer Science and Security 1 (2007) 35–44.

[9] W. Al-Hassan, M. Fayek, S. Shaheen, PSOSA: An optimized particle swarm technique for solving the urban planning problem, in: 2006 International Conference on Computer Engineering and Systems, 2006, pp. 401–405. doi:10.1109/ICCES.2006.320481.

[10] K. Lei, Y. Qiu, Y. He, 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, 2006, pp. 977–980. doi:10.1109/ISSCAA.2006.1627487.

[11] G. Chen, X. Huang, J. Jia, Z. Min, Natural exponential inertia weight strategy in particle swarm optimization, in: 2006 6th World Congress on Intelligent Control and Automation, Vol. 1, 2006, pp. 3672–3675. doi:10.1109/WCICA.2006.1713055.

[12] H. Li, Y. Gao, Particle swarm optimization algorithm with exponent decreasing inertia weight and stochastic mutation, in: 2009 Second International Conference on Information and Computing Science, Vol. 1, 2009, pp. 66–69. doi:10.1109/ICIC.2009.24.

[13] Y. Gao, X. An, J. Liu, A particle swarm optimization algorithm with logarithm decreasing inertia weight and chaos mutation, in: 2008 International Conference on Computational Intelligence and Security, Vol. 1, 2008, pp. 61–65. doi:10.1109/CIS.2008.183.

[14] Y. Shi, R. Eberhart, Fuzzy adaptive particle swarm optimization, in: Proceedings of the 2001 Congress on Evolutionary Computation (IEEE Cat. No.01TH8546), Vol. 1, 2001, pp. 101–106. doi:10.1109/CEC.2001.934377.

[15] A. Saber, T. Senjyu, N. Urasaki, T. Funabashi, Unit commitment computation – a novel fuzzy adaptive particle swarm optimization approach, in: 2006 IEEE PES Power Systems Conference and Exposition, 2006, pp. 1820–1828. doi:10.1109/PSCE.2006.296189.

[16] X. Yang, J. Yuan, J. Yuan, H. Mao, A modified particle swarm optimizer with dynamic adaptation, Applied Mathematics and Computation 189 (2) (2007) 1205–1213. doi:10.1016/j.amc.2006.12.045.

[17] M. Arumugam, M. Rao, 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, Applied Soft Computing 8 (1) (2008) 324–336. doi:10.1016/j.asoc.2007.01.010.

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

[19] Z. Qin, F. Yu, Z. Shi, Y. Wang, Adaptive inertia weight particle swarm optimization, in: L. Rutkowski, R. Tadeusiewicz, L. Zadeh, J. Żurada (Eds.), Artificial Intelligence and Soft Computing – ICAISC 2006, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006, pp. 450–459.

[20] K. Suresh, S. Ghosh, D. Kundu, A. Sen, S. Das, A. Abraham, Inertia-adaptive particle swarm optimizer for improved global search, in: 2008 Eighth International Conference on Intelligent Systems Design and Applications, Vol. 2, 2008, pp. 253–258. doi:10.1109/ISDA.2008.199.

[21] A. R. Jordehi, J. Jasni, Parameter selection in particle swarm optimisation: a survey, Journal of Experimental Theoretical Artificial Intelligence 25 (4) (2013) 527–542. doi:10.1080/0952813X.2013.782348.

[22] A. Sengupta, V. Mishra, Time varying vs fixed acceleration coefficient PSO driven exploration during high level synthesis: Performance and quality assessment, in: 2014 International Conference on Information Technology, 2014, pp. 281–286. doi:10.1109/ICIT.2014.16.

[23] J. Bansal, P. Singh, M. Saraswat, A. Verma, S. Jadon, A. Abraham, Inertia weight strategies in particle swarm optimization, in: 2011 Third World Congress on Nature and Biologically Inspired Computing, 2011, pp. 633–640. doi:10.1109/NaBIC.2011.6089659.

[24] H. Broer, F. Takens, Dynamical Systems and Chaos, Springer-Verlag, New York, 2011.

[25] B. Kalantari, Polynomial Root-Finding and Polynomiography, World Scientific, Singapore, 2009. doi:10.1142/9789812811837.

[26] W. Cheney, D. Kincaid, Numerical Mathematics and Computing, 6th Edition, Brooks/Cole, Pacific Groove, CA, 2007.

[27] J. Solomon, Numerical Algorithms: Methods for Computer Vision, Machine Learning, and Graphics, CRC Press, Boca Raton, 2015.

[28] W. Mann, Mean value methods in iteration, Proceedings of the American Mathematical Society 4 (3) (1953) 506–510. doi:10.1090/S0002-9939-1953-0054846-3.

[29] S. Ishikawa, Fixed points by a new iteration method, Proceedings of the American Mathematical Society 44 (1) (1974) 147–150. doi:10.1090/S0002-9939-1974-0336469-5.

[30] R. Agarwal, D. O'Regan, D. Sahu, Iterative construction of fixed points of nearly asymptotically nonexpansive mappings, Journal of Nonlinear and Convex Analysis 8 (1) (2007) 61–79.

[31] G. Das, J. Debata, Fixed points of quasinonexpansive mappings, Indian Journal of Pure and Applied Mathematics 17 (11) (1986) 1263–1269.

[32] S. Khan, Y. Cho, M. Abbas, Convergence to common fixed points by a modified iteration process, Journal of Applied Mathematics and Computing 35 (1) (2011) 607–616. doi:10.1007/s12190-010-0381-z.

[33] K. Gdawiec, W. Kotarski, Polynomiography for the polynomial infinity norm via Kalantari's formula and nonstandard iterations, Applied Mathematics and Computation 307 (2017) 17–30. doi:10.1016/j.amc.2017.02.038.

[34] K. Gdawiec, W. Kotarski, A. Lisowska, Polynomiography based on the nonstandard Newton-like root finding methods, Abstract and Applied Analysis 2015 (2015) Article ID 797594. doi:10.1155/2015/797594.

[35] J. Lilly, Fuzzy control and identification, John Wiley & Sons, Inc., Hoboken, NJ, 2010. doi:10.1002/9780470874240.