IEICE Proceeding Series

An idea of Novel Optimizer using Swarm of Chaotic Dynamical Particles

Yoshikazu Yamanaka, Tsubone Tadashi

Vol. 2 pp. 240-243
Publication Date: 2014/03/18
Online ISSN: 2188-5079

2013 International Symposium on Nonlinear Theory and its Applications
NOLTA2013, Santa Fe, USA, September 8-11, 2013

An idea of Novel Optimizer using Swarm of Chaotic Dynamical Particles

Yoshikazu Yamanaka† and Tsubone Tadashi‡

†‡Nagaoka University of Technology, 1603-1 Kamitomioka, Nagaoka, Niigata, 940-2188, Japan
Email: †y_yama@stn.nagaokaut.ac.jp, ‡tsubone@vos.nagaokaut.ac.jp

Abstract—In this paper we propose a novel Optimization method called Optimizer using Swarm of Chaotic Dynamical Particles. Our proposed method is based on particles which follow chaotic dynamics. The chaotic dynamics makes the particle complex motion by stretching and holding mechanism even tough the dynamics does not contain any stochastic elements. A swarm consists of the particles which share the information of their own position which is candidate solution like Particle Swarm Optimization(PSO). By the particle's complex behavior and sharing information mechanism, our proposed method has ability to search optimum solution. Our proposed method does not contain any stochastic elements and has only 3 system parameters, therefor, it is easy to implement the algorithm, to analyze particle dynamics and to design system parameters. We compared our proposed method with other deterministic PSO and conventional PSO for well-known benchmark functions and proposed method shown better search ability in almost all of situation.

1. introduction

Particle Swarm Optimization (PSO) is a evolutionary computation method developed by Kennedy and Eberhart [1]. The method optimizes an object function using population of particles which is based on social behavior like birds flock and fish school. In recent years PSO is studied because of their simple concept and efficient performance.

The particles updates their position and velocity at each time step. The particle's position which denotes a coordinates of a candidate solution is evaluated by an object function. The position is calculated by the velocity toward the best position in own searching history and toward the best position in all particle's searching history. The velocity is calculated with stochastic terms.

The stochastic terms are important factor because they cause particle's variety motion which helps to escape from local best position and to find an optimum solution. However, the stochastic terms also make difficult to analyze particle's dynamics.

In order to improve searching ability, analyzing particle's behavior is an important topic. Clerc and Kennedy proposed a deterministic PSO method, called simple PSO [2]. simple PSO is removed the stochastic terms from PSO to analyze particle's trajectory easily. Comparing simple PSO and PSO, however, PSO shows much better searching ability than simple PSO.

Shindo and Jin'no also proposed deterministic PSO called RDPSO which is implemented re-acceleration particle's velocity mechanism with simple PSO [3]. RDPSO has five system parameters. Three of them are needed for the re-acceleration mechanism and others are for simple PSO dynamics. RDPSO makes their particle's behavior variety than simple PSO's one by the additional mechanism. RDPSO shown better searching ability than simple PSO, however, the performance of PSO is still better than RDPSO [3].

In this paper, we propose novel optimization method using swam of particles which follow chaotic dynamics. In the swarm, the particles share the searching experience like PSO. We are concentrating on implementation an optimization method which is able to theoretically analyze particle's behavior and denotes superior performance than other deterministic PSOs. In order to develop our propose method, we focused on three features of chaotic dynamics.

First of all, chaotic dynamics causes complex phenomena. Following the chaotic dynamics, particles have variety motion with only three system parameters. It is said that proposed method has an ability which helps for particles leaving local beset position and searching an solution without any additional mechanism.

Second, size of chaotic attractor is controllable. The size of chaotic attractor denotes particle's searching range. We designed that the size of attractor depends on own searching experience. The particle whose best position is far away from swarm's best position roughly searches an optimal solution in large area. Likewise, The particle whose best position is near by swarm's best position searches an optimal solution in narrow area with high density.

Third, chaotic dynamics does not contain any stochastic terms. Therefore, in proposed method, particle's behavior is observed in deterministic system. It is helpful for analyzing particle's behavior and for theoretical designing system parameters.

The proposed method is described in Sec. 2. Experimental results for four benchmark functions are shown In Sec. 3. Proposed method was compared with other deterministic PSOs and it shown effective performance than other deterministic PSOs. Proposed method was also compared with conventional PSO and it denoted superior searching ability in multimodal functions.

2. Optimizer using Swam of Chaotic Dynamical Particles

2.1. Particle's information

Our method searches an optimal solution using swam which contains particles. Every particles has position, velocity and a point which is center point of their searching. At each time step, the particle's position is updated by chaotic dynamics described in Sec. 2.3. Following the dynamics, the particle behaves like hovering around own center point. The particle's searching center point is also updated by the dynamics which is described in Sec. 2.2.

In N-dimensional space, these elements of variables vector of i-th particle are described as following:

$$\boldsymbol{x_i(t)} = \{x_{i1}(t),\ x_{i2}(t),\ ....,\ x_{iN}(t)\}, \quad (1)$$

$$\boldsymbol{v_i(t)} = \{v_{i1}(t),\ v_{i2}(t),\ ....,\ v_{iN}(t)\}, \quad (2)$$

$$\boldsymbol{fp_i(t)} = \{fp_{i1}(t),\ fp_{i2}(t),\ ....,\ fp_{iN}(t)\}. \quad (3)$$

Every particles have own evaluated value given by an object function f. i-th particle has a N-dimensional vector $\boldsymbol{pbest_i}$. $\boldsymbol{pbest_i}$ is denoted particle's position which shows best evaluated value in his searching history. The swarm has a N-dimensional vector $\boldsymbol{gbest}$. $\boldsymbol{gbest}$ is also denoted particle's position which shows best evaluated value in all particle's searching history.

2.2. Dynamics of searching center point

The $\boldsymbol{fp_i(t)}$ is updated by $\boldsymbol{pbest_i}$ and $\boldsymbol{gbest}$ each time step as following:

$$\boldsymbol{fp_i(t+1)} = \boldsymbol{fp_i(t)} + c\{(\boldsymbol{pbest_i} - \boldsymbol{fp_i(t)}) + (\boldsymbol{gbest} - \boldsymbol{fp_i(t)})\}, \quad (4)$$

where c is positive constants. $\boldsymbol{fp_i(0)}$ is set to $\boldsymbol{x_i(0)}$. The parameter c is set to the range $[0, 1]$ to guarantee that $\boldsymbol{fp_i(t)}$ converges on $\frac{1}{2}(\boldsymbol{pbest_i} + \boldsymbol{gbest})$ in sufficiently long time.

2.3. Particle's chaotic dynamics

The particles follows chaotic dynamics. To describe this dynamics simply, i-th particle's position which is linearly mapped by $\boldsymbol{fp_i(t)}$ is denoted $\boldsymbol{y_i(t)}$ as following:

$$\boldsymbol{y_i(t)} = \boldsymbol{x_i(t)} - \boldsymbol{fp_i(t)}. \quad (5)$$

The i-th particle's position and velocity for j-th element are updated by

$$\begin{bmatrix} y_{ij}(t+1) \\ v_{ij}(t+1) \end{bmatrix} = R \begin{bmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{bmatrix} \begin{bmatrix} y_{ij}(t) \\ v_{ij}(t) \end{bmatrix}, \quad (6)$$

where R is a damping parameter and θ is a degree parameter.

Two thresholds, y_{th1ij} and y_{th2ij}, are considered as following. If $(y_{ij}(t) < y_{th1ij})$ and $(v_{ij}(t) \geq 0)$ are satisfied, status $(y_{ij}(t),\ v_{ij}(t))$ jumps to $(2y_{th1ij} - y_{ij}(t),\ 0)$. And if $(y_{ij}(t) > y_{th2ij})$ and $(v_{ij}(t) = 0)$ are satisfied, $(y_{ij}(t),\ v_{ij}(t))$ jumps to $(2y_{th2ij} - y_{ij}(t),\ 0)$. y_{th1ij} is set in the negative direction from the origin and the value $|y_{th1ij}|$ is the average of $gbest_j$ and $pbest_{ij}$. y_{th2ij} is also set in the positive direction and the value is My_{th1ij}, where M is $(1 - R^{\frac{\pi}{\theta}})^{-1}$.

Figure 1 and 2 shows typical particle's behavior, where we assume that y_{th1ij} is constant. The particle follows this dynamics, and exhibits chaotic attracter as shown in Fig. 1. Focusing on the particle's position, $y_{ij}(t)$, the chaotic attractor produces time series as shown in Fig. 2. Using this time series, the particles search optimum solution.

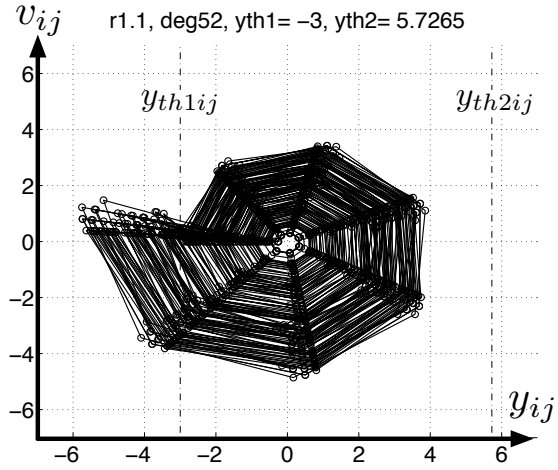

Figure 1: A particle's trajectory, $\theta = 52$[deg], $R = 1.1, yth_1 = -3$, drawn 29000 to 30000 iteration.

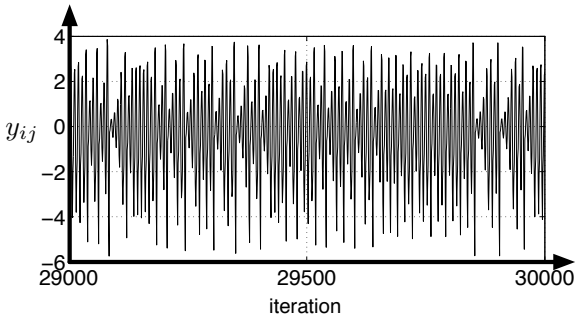

Figure 2: A time-serias of particle's positon, $\theta = 52$[deg], $R = 1.1, yth_1 = -3$, drawn 29000 to 30000 iteration.

The size of chaotic attractor is designed as variable depending on y_{th1ij}. Size of attractors exhibit search range of particles from $fp_{ij}(t)$. the particle's search area adapts depend on the length between $pbest_{ij}$ and $gbest_j$. This means that a particle having an element, $pbest_{ij}$, which is far away from $gbest_j$, roughly searches optimum solution around $fp_{ij}(t)$ and a particle having near $pbest_{ij}$ searches with high density, respectively.

2.4. Algorithm of proposed method

The algorithm of proposed method in pseudocode follows.

Initialize Population with random values
Initialize $\boldsymbol{gbest}$, $\boldsymbol{pbest}$, and $\boldsymbol{fp}$
repeat
  Evaluate all populations
  if better position is found **then**
    Update $\boldsymbol{pbest}$ and $\boldsymbol{gbest}$ if necesarry
    Calculate $\boldsymbol{y_{th1}}$ and $\boldsymbol{y_{th2}}$
  end if
  for $i = 1$ to Number of particles **do**
    for $j = 1$ to Dimension **do**
      update fp_{ij}
      $y_{ij} = x_{ij} - fp_{ij}$
      update y_{ij} and v_{ij}
      $x_{ij} = y_{ij} + fp_{ij}$
    end for
  end for
until termination criterion is met

3. Experimental Results

3.1. Experiment condition

The performance of proposed method was compared with conventional PSO [1] and other deterministic PSO, simple PSO [2] and RDPSO [3], by well known benchmark functions as shown in table 1.

PSO's parameters were set to $\omega = 0.729$ and $c1 = c2 = 1.49445$ [4]. For simple PSO, $\omega = 0.9025$ and $\psi = 2.2646$ were selected [3]. RDPSO's parameters were set to $v_{\text{boundary}} = 10^{-6}$, $\alpha = 0.5$, $\omega = 0.9025$ and $\psi = 2.2646$ [3]. In RDPSO, $x_{\text{SearchRange}}$ was same as Initial x Range in table 1. For proposed method, $R = 1.1$, $\theta = 52$[deg], $c = 0.94$ were selected by simulating with some patterns of parameters.

To be fair, in all methods, particle's positions were initialized to a set of coordinates at each independent runs. The set of coordinates is generated by uniformed distribution with Initial x Range in table 1. We compared methods for 30 dimensional object functions using 20 particles. So the set of coordinates contained 20×30 elements. 100 sets of coordinates were provided for 100 independent runs.

3.2. Results

Table 2 shows experimental results, averaged best evaluations at the end of iteration and the standard deviations with blanket by 100 independent runs using 20 populations. The best averaged evaluated value for each iteration is denoted with bold.

For unimodal function, Sphere, PSO shown best performance in all methods. Our proposed method shown better performance comparing with simple PSO and RDPSO.

For multimodal functions, proposed method performed better than other deterministic PSOs. Proposed method also performed better than PSO without Rosenbrock at 3000 iteration and Griewank at 1000 iteration.

In table 2, it seems that even though proposed method does not contain stochastic terms, our proposed method shown better performance for almost all of condition to Rosenbrock, Rastrigin and Griewank function comparing PSO. This result suggests that particle's variety of behavior operated chaotic dynamics has superior ability in optimizer which solves multimodal functions.

4. Conclusion

In this paper, we proposed a novel optimization method using swarm of chaotic dynamical particles.

The proposed method denoted better performance than PSO in almost all situation even tough our method is deterministic system. In proposed method, the particle's variety motion is given by chaotic dynamics without any additional mechanism. Using a chaotic dynamics's feature which is that size of chaotic attractor is controllable, our method effectively searches an optimal solution depending on each particle's searching experience.

The proposed method is easy to implement the algorithm and to analyze particle dynamics because the method has only three system parameters and does not contain any stochastic terms. Analyzing particle's behavior and theoretical designing system parameters are future topics.

References

[1] J. Kennedy and R. Eberhart, "Particle Swarm Optimization," *in Proc. IEEE Int. Conf. Neural Networks*, pp. 1942-1948, 1995.

[2] M. Clerc and J. Kennedy, "The particle swarm - explosion, stability, and convergence in a multidimensional complex space," *IEEE Trans. Evol. Comput.* , Vol. 6, No. 1, pp. 58-73, 2002.

[3] Takuya Shindo and Kenya Jin'no, "On a Reacceleration Mechanism for Particle Swarm Optimizer," *Journal of Signal Processing*, Vol. 15, No. 6, pp. 407-516, November 2011.

[4] R. C. Eberhart and Y. Shi, "Comparing inertia weights and constriction factors in particle swarm optimization," *in Proc. 2000 Congr. Evol. Comput.* , San Diego, CA, July 2000, pp. 84-88.

Table 1: Benchmark functions

f	Function	Initial x Range
Sphere	$f_S(x) = \sum_{i=1}^{N} x_i^2$	±20
Rosenbrock	$f_{RO}(x) = \sum_{i=1}^{N-1} 100\left((x_{i+1}^2 - x_i^2)^2 + (1 - x_i)^2\right)$	±10
Rastrigin	$f_{RA}(x) = \sum_{i=1}^{N}(x_i^2 - 10\cos(2\pi x_i) + 10)$	±5.12
Griewank	$f_G(x) = \frac{1}{4000}\sum_{i=1}^{N} x_i^2 - \prod \cos(\frac{x_i}{\sqrt{i}})$	±300

Table 2: Experimental results for 30 dimensional functions with 20 populations.
Each averaged best evaluated value is calculated with 100 independent runs. The values with bold denote the best performance of evaluation for each iteration.

Function	iteration	PSO	simple PSO	RDPSO	OSCDP
Sphere	1000	$\mathbf{1.252 \times 10^{-5}}$ (1.212×10^{-4})	188.7 (67.39)	7.182 (23.54)	2.491×10^{-2} (9.229×10^{-2})
	2000	$\mathbf{3.848 \times 10^{-14}}$ (2.704×10^{-13})	188.7 (67.39)	5.622 (23.26)	2.309×10^{-7} (7.703×10^{-7})
	3000	$\mathbf{1.032 \times 10^{-24}}$ (8.183×10^{-24})	188.7 (67.39)	5.518 (23.22)	3.579×10^{-12} (1.851×10^{-11})
Rosenbrock	1000	45.41 (37.58)	3.430×10^4 (3.580×10^4)	483.7 (1618)	**36.13** (24.94)
	2000	31.74 (28.89)	3.430×10^4 (3.580×10^4)	389.1 (1597)	**29.20** (15.01)
	3000	**24.97** (24.98)	3.430×10^4 (3.580×10^4)	365.7 (1591)	26.86 (8.783)
Rastrigin	1000	74.71 (20.14)	170.8 (25.13)	76.99 (24.29)	**70.87** (18.75)
	2000	77.75 (20.06)	170.8 (25.13)	66.43 (22.00)	**59.95** (17.23)
	3000	71.12 (17.17)	170.8 (25.13)	62.72 (21.41)	**59.76** (17.20)
Griewank	1000	**0.08713** (0.1627)	11.16 (3.989)	1.960 (1.121)	0.1829 (0.151)
	2000	0.04534 (0.05466)	11.16 (3.989)	1.149 (0.9553)	**0.02181** (0.02427)
	3000	0.06223 (0.07212)	11.16 (3.989)	0.9103 (1.003)	**0.02082** (0.02334)