Available online at www.sciencedirect.com
ScienceDirect
Journal of Computational and Applied Mathematics 206 (2007) 1070–1081
JOURNAL OF COMPUTATIONAL AND APPLIED MATHEMATICS
www.elsevier.com/locate/cam

An optimization algorithm based on chaotic behavior and fractal nature

Mohammad Saleh Tavazoei, Mohammad Haeri*

Advanced Control System Laboratory, Electrical Engineering Department, Sharif University of Technology, Tehran, Iran

Received 2 May 2006; received in revised form 27 July 2006

Abstract

In this paper, we propose a new optimization technique by modifying a chaos optimization algorithm (COA) based on the fractal theory. We first implement the weighted gradient direction-based chaos optimization in which the chaotic property is used to determine the initial choice of the optimization parameters both in the starting step and in the mutations applied when a convergence to local minima occurred. The algorithm is then improved by introducing a method to determine the optimal step size. This method is based on the fact that the sensitive dependence on the initial condition of a root finding technique (such as the Newton–Raphson search technique) has a fractal nature. From all roots (step sizes) found by the implemented technique, the one that most minimizes the cost function is employed in each iteration. Numerical simulation results are presented to evaluate the performance of the proposed algorithm.

MSC: 90C30; 34C28; 37F50

Keywords: Nonlinear optimization; Chaos search; Fractals; Julia set; Newton–Raphson method

1. Introduction

Fractals and chaos, as relatively new branches of physics and mathematics, have provided us a new way of viewing the universe and are important tools for understanding the world we live in. They can be traced all around us and describe ordered systems that are seemingly random objects and patterns. A fractal is an object or quantity that displays self-similarity, in a somewhat technical sense, on all scales. The object need not exhibit exactly the same structure at all scales, but the same type of structures must appear on all scales [33]. Chaos is mathematically defined as a semi-randomness behavior generated by nonlinear deterministic systems. In general, chaos has three important dynamic properties [30]:

- the sensitive dependence on initial conditions;
- the quasi-stochastic property;
- ergodicity.

* Corresponding author. Tel.: +98 21 66165964; fax: +98 21 66023261.
E-mail address: haeri@sina.sharif.edu (M. Haeri).

0377-0427/$ - see front matter
doi:10.1016/j.cam.2006.09.008

Recently, the idea of using chaotic systems instead of random processes has been noticed in several fields. One of these fields is the optimization theory. In random-based optimization algorithms, the role of randomness can be played by chaotic dynamics instead of random processes. Experimental studies assert that the benefits of using chaotic signals instead of random signals are often evident even though a general rule cannot be formulated [3]. Taking properties of chaos like ergodicity, some new searching algorithms called chaos optimization algorithms (COAs) are presented [4,10,19–22,31,32,34,36,37]. COA can more easily escape from local minima than the stochastic optimization algorithms [19]. The random-based algorithms often escape from local minima by admitting some unacceptable solutions with a certain probability. On the contrary, a COA searches on the regularity exist in a chaotic motion to escape from local minima.

In addition to quasi-stochastic property, the other property of chaos that can be advantageous in the optimization is the sensitivity to the initial condition. Assume that the final result of a search technique is very sensitive to initial point. Consequently, if one chooses “proper” initial point and search in small vicinity of this point, all of the possible search results may be achieved. The Newton–Raphson method is a search technique that is sensitive to initial point [14]. This sensitivity has a fractal nature which can be utilized to find all solutions to a nonlinear equation [15]. Based on utilizing sensitive fractal areas to locate all of the solutions along one direction in a variable space, a method for searching of global minima in optimization problems was introduced [16].

In this paper, our aim is to propose an optimization algorithm based on two mentioned properties of chaos, quasi-stochastic property and sensitivity to initial condition. To do the job, we choose a COA method structured on semi-randomness property of chaotic maps and modify it by fractal nature of basin of attraction in Newton–Raphson method. In other words, the first property of the chaos is used to initialize a new optimization search and the second property is used to find the optimal step size. The search directions are determined by the weighted gradient direction method.

This paper is organized as follows. Section 2 formulates nonlinear programming problems that are dealt with in this paper. The weighted gradient direction method is also explained in this section for its use in the next sections. In Section 3, the COA is described. Section 4 summarizes some definitions that are required in the successive sections. Section 5 describes the Newton–Raphson method and the appearance of the fractal nature in its solution. Section 6 presents Julia set properties and the way to find a point in this set. In Section 7 we propose our modification on the algorithm presented in Section 3 based on the concepts described in the previous sections. By numerical simulation results that are given in Section 8, we evaluate the modified algorithm. Conclusions in Section 9 close the paper.

2. Problem formulation

Global optimization is the task of seeking the absolutely best set of parameters to maximize or minimize an objective function. Linear programming [26], quadratic programming [24], integer programming [26], dynamic programming [1], combinatorial optimization [5], nonlinear programming [24], infinite-dimensional optimization [23] and stochastic programming [2] are some major approaches to the optimization problem. Some chaotic-based algorithms have been proposed for combinatorial optimization problems [4,20,27]. In a combinatorial optimization problem, the set of feasible solutions is discrete or can be reduced to a discrete one. Therefore, combinatorial optimization algorithms are effective in discrete problems like traveling salesman problem [6], minimum weight spanning tree problem [6] or eight queens puzzle [29] but these algorithms are not simply usable in the domain of the continuous optimization problems. Now, we consider a continuous problem in the class of nonlinear programming and try to find an optimization algorithm based on chaotic behavior and fractal nature to solve this problem.

Nonlinear programming algorithm: Formally, a nonlinear programming (NLP) problem with inequality constraints can be stated as:

$$\begin{array}{ll} \text{Minimize} & f(x) \\ \text{subject to} & g_i(x) \leqslant 0, \quad i = 1, 2, \ldots, m. \\ & a_i \leqslant x_i \leqslant b_i, \quad i = 1, 2, \ldots, n, \end{array} \tag{1}$$

where $x = (x_1, x_2, \ldots, x_n)^{\mathrm{T}} \in R^n$, $f(x)$ is the objective function and $g_i(x)$'s $(i = 1, 2, \ldots, m)$ are the inequality constraints defined on R^n. We assume that functions $f(x)$ and $g_i(x)$'s are differentiable in R^n. The nonlinear programming stated in (1) can be converted as follows to another nonlinear programming that has only bound constraints:

$$\begin{aligned} &\text{Minimize} \quad P(x,\sigma) = f(x) + \sigma \sum_{i=1}^{m} \max(0, g_i(x)) \\ &\text{subject to} \quad a_i \leqslant x_i \leqslant b_i, \quad i = 1, 2, \ldots, n, \end{aligned} \tag{2}$$

where $f(x)$ is the objective function of the original constrained problem and the positive coefficient σ is the penalty parameter. In [35], it was proved that the minima of the non-differentiable exact penalty function (2) converge to the minima of the original constrained problem (1), if penalty parameter σ is chosen sufficiently large.

Weighted gradient direction: We study the weighted gradient direction method that was proposed in [11,12]. It was shown in [22] that the implementation of this method would enhance improvements that can be obtained employing the chaos optimization algorithm. Let

$$Q = \{x \in R^n | g_i(x) \leqslant 0, \quad i = 1, 2, \ldots, m\}. \tag{3}$$

For an individual x, if $x \in Q$, moving x along the negative gradient direction of the objective function $-\nabla f(x)$, the objective function may be improved. If $x \notin Q$, it denotes that x is out of the feasible domain. Assume that,

$$I(x) = \{i | g_i(x) > 0, \quad i = 1, 2, \ldots, m\}. \tag{4}$$

For $i \in I$, moving x along the negative gradient direction $-\nabla g(x)$, $g_i(x)$ can be decreased and may satisfy $g_i(x) \leqslant 0$. Based on these facts, the weighted gradient direction is defined as follows:

$$d(x) = -\nabla f(x) - \sum_{i=1}^{n} w_i \nabla g_i(x), \tag{5}$$

where w_i is the weight of the gradient direction and defined as follows:

$$w_i = \begin{cases} 0, & g_i(x) \leqslant 0, \\ \delta_i, & g_i(x) > 0, \end{cases} \tag{6}$$

where

$$\delta_i = \frac{1}{1 - G_i(x)/(G_{\max}(x) + \varepsilon)}, \tag{7}$$

$$G_i(x) = \frac{g_i(x)}{\|\nabla g_i(x)\|} \tag{8}$$

and

$$G_{\max}(x) = \max\{0, G_i(x), \quad i = 1, 2, \ldots, m\}. \tag{9}$$

ε in (7) is a very small positive number. Then $x^{(k+1)}$ is generated from $x^{(k)}$ by updating along the weighted gradient direction $d(x)$ and is described by

$$x^{(k+1)} = x^{(k)} + \lambda d(x^{(k)}), \tag{10}$$

where λ is a positive step-size parameter that satisfies

$$\frac{\mathrm{d}}{\mathrm{d}\lambda} P(x^{(k)} + \lambda d(x^{(k)}), \sigma) = 0. \tag{11}$$

From (5)–(8), it is obvious that for $g_i(x) > 0$, increasing of $g_i(x)$ causes increasing the weight of gradient direction. $d(x)$ is an effective search direction because if $x \in Q$, then $d(x) = -\nabla f(x)$ and it moves along the direction of

$-\nabla f(x)$; consequently, the objective function may be improved. Also, if $x \notin Q$, the bigger the $g_i(x)$, the farther apart x is from the feasible domain Q, the high weight w_i can be obtained in order to move into feasible domain. Therefore, x converges to the local optimal solution by the weighted gradient direction.

3. Chaos optimization algorithm

In the previous section, the weighted gradient direction method is described as a local optimization technique for constrained nonlinear problems. By means of ergodicity, regularity and quasi-stochastic property of chaos, the optimal solution migrates in a chaotic way among the local minima and finally converges to the global optimal solution with a high probability.

In most of the COA methods, chaos variables are generated by the logistic map. This map is defined by function $M(.)$:

$$\gamma^{(l+1)} = M(\gamma^{(l)}); \quad M(\gamma^{(l)}) = a\gamma^{(l)}(1-\gamma^{(l)}), \tag{12}$$

where $a = 4$.

Fig. 1 shows flowchart of the weighted gradient direction-based chaos optimization algorithm. Now, we explain the components of this flowchart.

Chaos search by using the first carrier wave

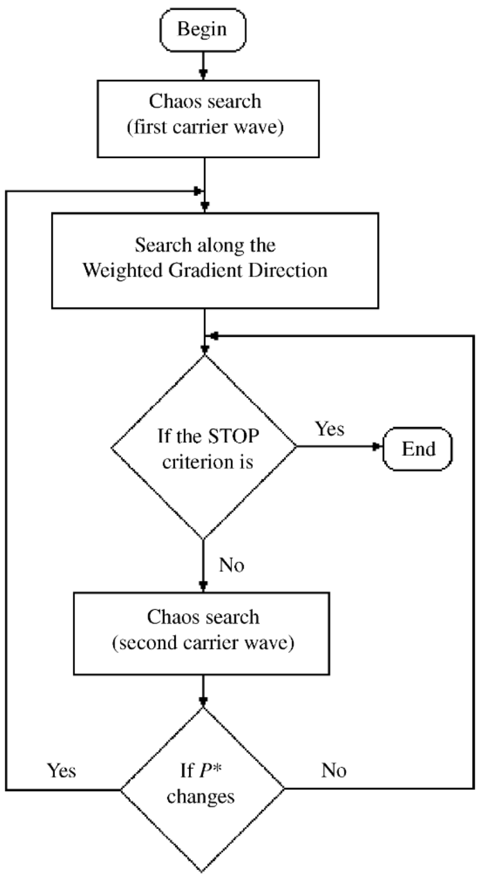

Fig. 1. Flowchart of the weighted gradient direction-based chaos optimization algorithm.

Step 1: Initialize the number of the first chaos search S_1, the number of the second chaos search S_2, penalty parameter σ, initial value of chaos variables $0<\gamma_i^{(0)}<1$ $(i=1,2,\ldots,n)$ which have small differences, adjusting parameter for small ergodic ranges around solution $\alpha_i>0$ $(i=1,2,\ldots,n)$ and adjusting parameter for $\alpha t_3>1$.

Step 2: Set $l=0$ and $P^*=\infty$.

Step 3: Map chaos variables $\gamma_i^{(l)}$, $i=1,2,\ldots,n$, into the variance range of the optimization variables by the following equation:

$$x_i^{(l)}=a_i+\gamma_i^{(l)}(b_i-a_i);\quad x^{(l)}=[x_1^{(l)},x_2^{(l)},\ldots,x_n^{(l)}]^{\mathrm{T}}.$$

Step 4: If $P(x^{(l)},\sigma)\leqslant P^*$ then $x^*=x^{(l)}$ and $P^*=P(x^{(l)},\sigma)$.

Step 5: Generate next values of chaos variables by a chaotic map function (M):

$$\gamma_i^{(l+1)}=M(\gamma_i^{(l)}).$$

Step 6: If $l<S_1$, $l\leftarrow l+1$ and go to step 3, else stop the first chaos search process.

Search along the weighted gradient method

Step 1: Initialize step length $\lambda>0$ and choose step size parameters $t_1>1$ and $0<t_2<1$. Also, set $k=0$.

Step 2: $\bar{x}^{(0)}=x^*$, where $x^*=[x_1^*,x_2^*,\ldots,x_n^*]^{\mathrm{T}}$.

Step 3: Calculate $d(\bar{x}^{(k)})$ from (5).

Step 4: $x=\bar{x}^{(k)}+\lambda d(\bar{x}^{(k)})$.

Step 5: If $P(x,\sigma)\leqslant P^*$ then $\bar{x}^{(k+1)}=x$, $P^*=P(x,\sigma)$ and $\lambda\leftarrow t_2\lambda$ else $\lambda\leftarrow t_1\lambda$.

Step 6: While P^* improves go to step 4.

Step 7: While $d(\bar{x}^{(k)})$ does not become sufficiently small, $k\leftarrow k+1$ and go to step 3.

Step 8: $x^*=\bar{x}^{(k)}$.

Chaos search by using the second carrier wave

Step 1: $l'=0$.

Step 2: Generate next values of chaos variables by the chaotic map function (M):

$$\gamma_i^{(l+1)}=M(\gamma_i^{(l)}),\quad i=1,2,\ldots,n.$$

Step 3: $l\leftarrow l+1$.

Step 4: $x_i^{(l)}=x_i^*+\alpha_i(\gamma_i^{(l)}-0.5)$.

Step 5: If $P(x^{(l)},\sigma)\leqslant P^*$ then $x^*=x^{(l)}$, $P^*=P(x^{(l)},\sigma)$ and stop.

Step 6: If $l'<S_2$, $l'\leftarrow l'+1$ and go to step 2.

Step 7: $\alpha_i\leftarrow t_3\alpha_i$ and stop the second chaos search process.

α_i is a very important parameter which adjusts small ergodic ranges around x^*. It is difficult to determine the appropriate value of α_i and usually is chosen heuristically [37]. Initial value of this parameter is usually set to $0.01(b_i-a_i)$ [22]. The optimization program is stopped when α_i becomes more than a specific constant (for example $0.1(b_i-a_i)$).

Computation of λ is a problem in the mentioned algorithm. Equation $P'(x^{(k)}+\lambda d(x^{(k)}))=0$ may have more than one root but the presented algorithm does not have the ability to determine all the roots of (11). Also, in this algorithm, two parameters t_1 and t_2 have been used to update λ. To choose these parameters is very difficult and problem dependent. To improve the presented algorithm, we should find another way to determine all roots of the nonlinear equation. Based on the fractal theory, a method to locate all the solutions of the nonlinear equation will be introduced in the next sections.

4. Some definitions

A *fixed point* x_0 of a function $f:X\to X$ is a point that remains constant upon application of that function, i.e., $f(x_0)=x_0$. Also, a point x_0 is said to be a *periodic point* of function f of period p if $f^p(x_0)=x_0$, where $f^0(x)=x$ and $f^p(x)$ is defined recursively by $f^p(x)=f(f^{p-1}(x))$. In other words, a point x_0 is called a periodic point of period p of f if it is a fixed point of f^p. A sequence $x_0, f(x_0), f^2(x_0),\ldots$ generated by repeated application of function f is said *forward orbit* and it is denoted by $O^+(x_0)$. If this sequence has only p points, the forward orbit is called *periodic orbit* of period p and $\lambda_{x_0} = (f^p)'(x_0)$ is the *eigenvalue* of this periodic orbit. By the chain rule, each point in the periodic orbit has the same eigenvalue. A periodic orbit $O^+(x_0)$ is

- super attracting if $\lambda_{x_0} = 0$ and x_0 is called a super attracting periodic point;
- attracting if $0 < |\lambda_{x_0}| < 1$ and x_0 is called an attracting periodic point;
- neutral if $|\lambda_{x_0}| = 1$ and x_0 is called a neutral periodic point;
- repelling if $|\lambda_{x_0}| > 1$ and x_0 is called a repelling periodic point.

If a point x_0 is attracting or super attracting periodic point, then there is an open interval U around x_0 such that $\lim_{p\to\infty} f^p(x) = x_0$ for every $x \in U$. The set of all points whose orbits converge to x_0 is called the *basin of attraction* of x_0. Also, if x_0 is repelling periodic point then $\lim_{p\to\infty} f^p(x) \neq x_0$ unless $x = x_0$ [7].

Julia set is one of the definitions that we deal with in this paper. The closure of all repelling points of rational function $f : C \to C$ is called Julia set and denoted by $J(f)$ [18].

5. Newton–Raphson method and fractal nature

Newton's method, also called the Newton–Raphson method, is a root-finding algorithm that uses the first two terms of the Taylor series of a function $f(x)$ in the vicinity of a suspected root. With a good initial choice of the root's position (x_1), the algorithm can be applied iteratively as follows:

$$x_{k+1} = x_k - [f'(x_k)]^{-1} f(x_k) \tag{13}$$

for $k = 1, 2, 3, \ldots$. In other words, Newton–Raphson's method is the functional iteration of $N_f(x) = x - f(x)/f'(x)$ or $x_{k+1} = N_f(x_k)$ [28]. Furthermore, if p is a simple root of f, then p is a super attracting fixed point of N because

$$\begin{aligned} N_f(p) &= p - \frac{f(p)}{f'(p)} = p - 0 = p, \\ N'_f(p) &= 1 - \frac{[f'(p)]^2 - f(p)f''(p)}{[f'(p)]^2} = 1 - 1 + 0 = 0. \end{aligned} \tag{14}$$

Applying Newton–Raphson's method to the roots of any polynomial of degree two or higher yields a rational map of C, and the Julia set of this map is a fractal whenever there are three or more distinct roots. Also, fractals typically arise from non-polynomial maps as well [8]. Fig. 2 shows basins of attraction for $z^5 - 1 = 0$ using Newton–Raphson's method in the complex plane [38].

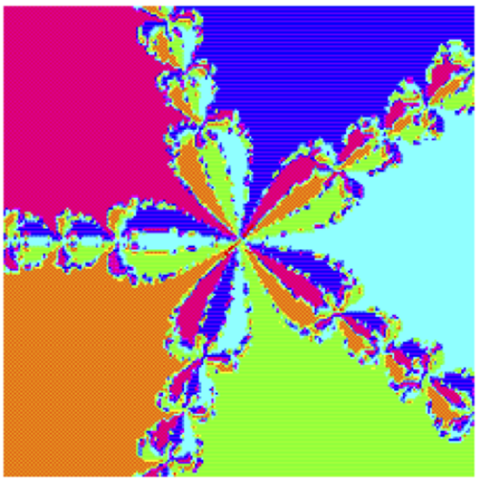

Fig. 2. Basins of attraction for $z^5 - 1 = 0$ using Newton–Raphson's method.

6. Julia set: gates to all the roots

Julia set: By noting an interesting property of Julia set, we begin this section. A point on the boundary of one of the domains of attraction must be on the boundary of all of them [9]. In other words, Julia set is the boundary of domain of attraction of each root.

$$J(N) = \partial A(z_1) = \cdots = \partial A(z_n) \tag{15}$$

where $\partial A(z_i)$ is the boundary of the domain of attraction of root z_i. This property results from Montel's theorem [25]. Therefore, the Julia set contains all the gates to all the roots of the nonlinear equation. In other words, by finding a point that belongs to Julia set of the Newton–Raphson function and iterating on a small neighborhood of this point, all of the roots are reachable. Now, we state another property of Julia set by the following theorem [18].

Theorem. *If $J(f)$ is the Julia set for a rational function f and $z \in J(f)$, then $\bigcup_p f^{(-p)}(z)$ is dense in $J(f)$.*

The above theorem means that the Julia set of a function is backward invariant under iteration of the function. Hence, all the points that iterate to repelling points also belong to the Julia set. In the next paragraph, a method has been presented to locate a point in the Julia set. We use this point as a gate to obtain all the roots of an equation.

Finding a point in the Julia set: As it is said in the previous paragraph, the points that iterate back to the repelling points of a function belong to Julia set. Based on this fact, in [13] it has been shown that any solution x_c of equation $f'(x) = 0$ belongs to the Julia set of the Newton–Raphson function $N_f(x)$. Therefore, to find all roots of a nonlinear equation $f(x) = 0$ one can find any solution x_c of $f'(x) = 0$ and use values in the region $|x - x_c| \leqslant \varepsilon$, where ε is an arbitrary small positive value, as initial values of the iteration process. x_c being in the Julia set guarantees that all roots of an equation $f(x) = 0$ will be achievable by this procedure. To find a solution of $f'(x) = 0$ a number of numerical techniques are available, for example Newton–Raphson method itself. As an initial value in the Newton–Raphson method, one can choose points on a circle with very small radius around a Julia set point. To become clear, consider the following example: assume that we want to seek all roots of equation $x^3 - 3x = 0$. $x = 1$ is a root of equation $(x^3 - 3x)' = 3x^2 - 3 = 0$, hence it belongs to Julia set of the Newton–Raphson function $x^3 - 3x$. Now, we select some points around $x = 1$ as initial values of Newton–Raphson to find all roots of equation $x^3 - 3x = 0$. These points are chosen on the circle $|x - 1| = 10^{-4}$ with offset $\Delta\theta = 45°$. Fig. 3 shows selected initial values according to the final values of the iteration process. The final values $(-\sqrt{3}, 0, \sqrt{3})$ are all roots of equation $x^3 - 3x = 0$.

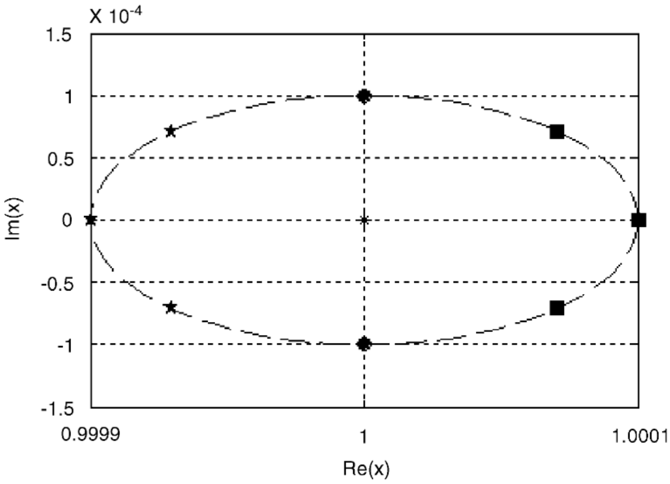

Fig. 3. Pentagram points: final value of the iteration process is $-\sqrt{3}$, circle points: final value of the iteration process is 0 and square points: final value of the iteration process is $\sqrt{3}$.

Another way of finding a point in the Julia set has been described in [15]. The aim of this method is to find an unstable (repelling) point with period 2. It mathematically equals to finding a solution to the following equation with constraint:

$$N_f^2(x) = x \quad \text{subject to} \quad |(N_f^2)'(x)| > 1. \tag{16}$$

Contrary to the previous case, Newton–Raphson method is not a good tool to solve (16) since most of the initial points will lead to stable points of $N^2(x) = x$. The proposed method in [15] to find a solution of (16) is based on this fact: with simple iteration of inverse function $(N_f^2)^{-1}(x)$, the stability of the existing fixed point of $N_f^2(x)$ is reserved. The proposed method is described as follows. First locate one of the roots of $f(x)$ (for example by Newton–Raphson method). Then, construct $N_f^{*2}(x_k)$ and $\varphi^*(x_k)$ as

$$N_f^{*2}(x_k) = x_k - \mu(x_k - N_f^2(x_k)), \tag{17}$$

$$\varphi^*(x_k) = x_k + \frac{x_k - N_f^{*2}(x_k)}{(N_f^{*2})'(x_k)},$$

where $0 < \mu < 1$ is a control parameter. Now iterate $\varphi^*(x_k)$ with located root as initial seed. This iteration will converge to an unstable periodic point [15].

7. To reinforce the chaos optimization search

In Section 3, we described the algorithm of chaos optimization search based on the weighted gradient direction. As we said before, the phase "Search along the Weighted Gradient Method" of this algorithm is not very effective. This phase does not have the ability of finding all the roots of Eq. (11). Furthermore, the phase requires some parameters which are problem dependent. We modify this section of the algorithm using Julia set theory. Fig. 4 shows the flowchart of the modified algorithm. Now, we explain the components of this flowchart.

Chaos search by using the first carrier wave

Step 1: Initialize the number of the first chaos search S_1, the number of the second chaos search S_2, penalty parameter σ, initial value of chaos variables $0 < \gamma_i^{(0)} < 1$ $(i = 1, 2, \ldots, n)$ which have small differences, adjusting parameter for small ergodic ranges around solution $\alpha_i > 0$ $(i = 1, 2, \ldots, n)$ and adjusting parameter for α $t_3 > 1$.

Step 2: Set $l = 0$ and $P^* = \infty$.

Step 3: Map chaos variables $\gamma_i^{(l)}$, $i = 1, 2, \ldots, n$, into the variance range of the optimization variables by the following equation:

$$x_i^{(l)} = a_i + \gamma_i^{(l)}(b_i - a_i); \quad x^{(l)} = [x_1^{(l)}, x_2^{(l)}, \ldots, x_n^{(l)}]^{\mathrm{T}}.$$

Step 4: If $P(x^{(l)}, \sigma) \leqslant P^*$ then $x^* = x^{(l)}$ and $P^* = P(x^{(l)}, \sigma)$.

Step 5: Generate next values of chaos variables by a chaotic map function (M):

$$\gamma_i^{(l+1)} = M(\gamma_i^{(l)}).$$

Step 6: If $l < S_1$, $l \leftarrow l + 1$, and go to step 3, else stop the first chaos search process.

Search along the weighted gradient method (*using the julia set theory*)

Step 1: Calculate $d(x^*)$ from (5).

Step 2: Find the minima (λ's) of function $P(\lambda) = P(x^* + \lambda d(x^*), \sigma)$. In other words, find all solutions of equation $P'(\lambda) = 0$. To find the solutions, first locate a Julia set point and then use Newton–Raphson method to iterate from a neighborhood around this point. If no minimum is found stop this phase.

Step 3: Among all solutions of step 2, find λ that makes $P(x^* + \lambda d(x^*), \sigma)$ minimum, then set $x^* = x^* + \lambda d(x^*)$ and $P^* = P(x^*, \sigma)$.

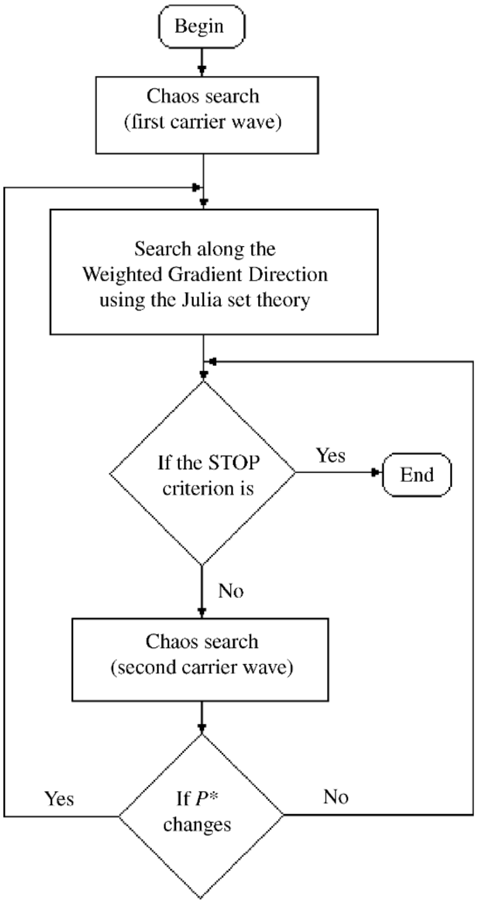

Fig. 4. Flowchart of the modified algorithm.

Step 4: Go to step 1 until $d(x^*)$ becomes sufficiently small.
Chaos search by using the second carrier wave
Step 1: $l' = 0$.
Step 2: Generate next values of chaos variables by the chaotic map function (M):

$$\gamma_i^{(l+1)} = M(\gamma_i^{(l)}), \quad i = 1, 2, \ldots, n.$$

Step 3: $l \leftarrow l + 1$.
Step 4: $x_i^{(l)} = x_i^* + \alpha_i(\gamma_i^{(l)} - 0.5)$.
Step 5: If $P(x^{(l)}, \sigma) \leqslant P^*$ then $x^* = x^{(l)}$, $P^* = P(x^{(l)}, \sigma)$ and stop.
Step 6: If $l' < S_2$, $l' \leftarrow l' + 1$ and go to step 2.
Step 7: $\alpha_i \leftarrow t_3 \alpha_i$ and stop the second chaos search process.

8. Numerical experiments

In this section, we apply chaos optimization algorithm based on search along the weighted gradient method and its modification to solve two benchmark optimization problems. For each algorithm, 100 trials are performed running on a Pentium IV 3 GHz PC.

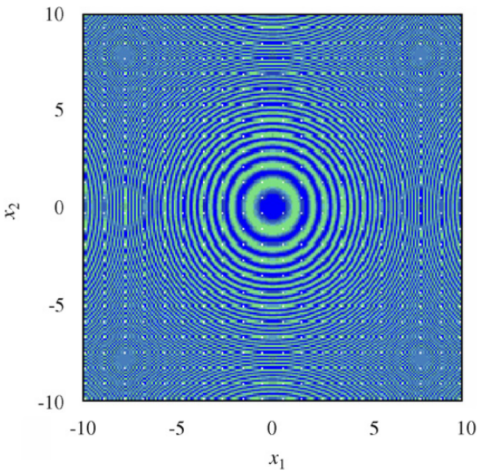

Fig. 5. Contours of f for Problem 1.

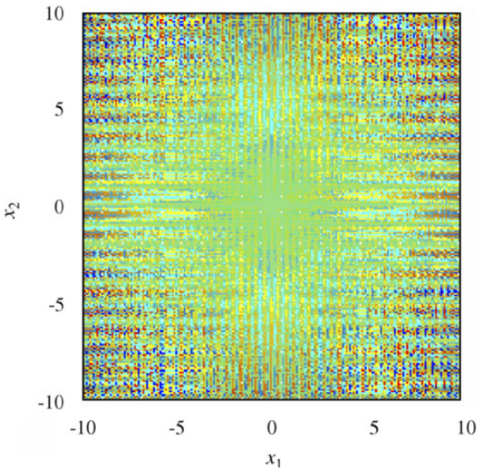

Fig. 6. Contours of f for Problem 2.

Problem 1 [20]:

$$\begin{aligned} &\text{Minimize} \quad f(x) = -\left(0.5 - \frac{\sin^2\sqrt{x_1^2+x_2^2} - 0.5}{(1+0.001(x_1^2+x_2^2))^2}\right) \\ &\text{subject to} \quad -100 < x_1 < 100, \\ &\qquad\qquad\quad -100 < x_2 < 100. \end{aligned}$$

Fig. 5 indicates contours of f for this problem. As it is seen, the given function contains numerous numbers of extrema. The number of convergence to the global optimum for chaos optimization algorithm based on the ordinary weighted gradient method is 85% and for chaos optimization algorithm based on the modified version is 98%.

Problem 2 [17]:

$$\begin{aligned} &\text{Minimize} \quad f(x) = -[x_1 \sin(9\pi x_2) + x_2 \cos(25\pi x_1) + 20] \\ &\text{subject to} \quad x_1^2 + x_2^2 \leqslant 81, \\ &\qquad\qquad\quad -10 < x_1 < 10, \\ &\qquad\qquad\quad -10 < x_2 < 10. \end{aligned}$$

Table 1
Simulation results (Problem 2)

Method	Worst solution	Medium of solution	Best solution
COA-based linear search along the weighted gradient	−31.7881	−32.4098	−32.7179
COA-based Julia set theory	−32.3310	−32.6844	−32.7179

Because of high oscillatory behavior of functions $\sin(9\pi x_2)$ and $\cos(25\pi x_1)$, the objective function of Problem 2 is very complex with high numbers of extrema (Fig. 6). Minimum value of this objective function with above constraint is -32.7179 and concentrates on four points on the circle $x_1^2 + x_2^2 = 81$. Table 1 presents the worst solution, medium of the solutions and the best solution for each method. Same as in the previous problem, here the modified method has better result.

9. Conclusion

Do chaotic behaviors complicate our problems? This paper shows that a chaotic behavior is not always a troublemaker! Logistic map demonstrates chaotic behavior, but based on this behavior, we used the map as a search pattern in optimization problem. On the other hand, the sensitive dependence of the Newton–Raphson method to the initial point seems to be a negative issue in the first view, but this property helps us to find all roots of a nonlinear equation efficiently. These facts along with some other similar observations demonstrate that a chaotic behavior can act as a powerful factor in our problems provided that it is employed in the proper situation.

In this paper, we modified the weighted gradient direction-based chaos optimization algorithm with fractal-based algorithm that is used to find the optimal step size and proposed a new optimization algorithm. The new method can be applied to any NLP problem such that its objective function and derivative of this objective function are known. Numerical experiments showed that this algorithm has better results than the ordinary weighted gradient direction-based chaos optimization algorithm.

References

[1] D.P. Bertsekas, Dynamic Programming and Optimal Control, second ed., Athena Scientific, 2000.
[2] J. Birge, F. Louveaux, Introduction to Stochastic Programming, Springer, Berlin, 1997.
[3] M. Bucolo, R. Caponetto, L. Fortuna, M. Frasca, A. Rizzo, Does chaos work better than noise?, Circuits and Systems Magazine, IEEE 2 (3) (2002) 4–19.
[4] L. Chen, K. Aihara, Optimization by chaotic simulated annealing, Proceedings of International Conference of Sino-Japanese Young Scientist, vol. 3, 1995, pp. 57–59.
[5] W.J. Cook, W.H. Cunningham, W.R. Pulleyblank, A. Schrijver, Combinatorial Optimization, first ed., Wiley, New York, 1997.
[6] T.H. Cormen, C.E. Leiserson, R.L. Rivest, C. Stein, Introduction to Algorithms, second ed., MIT Press, McGraw-Hill, Cambridge, MA, New York, 2001.
[7] R.M. Crownover, Introduction to Fractals and Chaos, Jones and Bartlett, 1995.
[8] R.M. Dickau, Compilation of iterative and list operations, Mathematica J. 7 (1997) 14–15.
[9] K. Falconer, Fractal Geometry, Wiley, Chichester, England, 1990.
[10] Y. Hu, Y.C. Li, J.X. Yu, H.D. Chao, Steeped-up chaos optimization algorithm and its application, J. System Eng. 17 (1) (2002) 41–44.
[11] T. Jiang, D. Wang, A hybrid generic algorithm for a type of nonlinear programming problem, Internat. J. Comput. Math. Appl. 36 (5) (1998) 11–21.
[12] T. Jiang, D. Wang, Evaluation-based generic algorithm for non-linear programming problem, Control Decis. 15 (5) (2000) 573–576.
[13] V.T. Jovanovic, Chaotic descent method and fractal conjecture, Internat. J. Numer. Methods Eng. 48 (2000) 137–152.
[14] V.T. Jovanovic, K. Kazerounian, Chaos in Computational Kinematics, in: Proceedings of the Fourth National Applied Mechanism and Robotics Conference, Ohio, Cincinnati, December 1995.
[15] V.T. Jovanovic, K. Kazerounian, Using Chaos to Obtain Global Solutions in Computational Kinematics, ASME J. Mech. Des. 120 (2) (1998) 299–304.
[16] V.T. Jovanovic, K. Kazerounian, Optimal design using chaotic descent method, ASME J. Mech. Des. 122 (3) (2000) 265–270.
[17] L. Kang, Z. Kang, Y. Li, P. Liu, Y. Chen, Asynchronous parallelization of Gue's algorithm for function optimization, in: Proceedings of the 2000 Congress on Evolutionary Computation, vol. 1, 2000, pp. 783–789.
[18] L. Keen, Julia Sets, Proc. Sympos. Appl. Math. 39 (1989) 57–74.
[19] B. Li, W. Jiang, Optimization complex functions by chaos search, Internat. J. Cybernet. Systems 29 (4) (1998) 409–419.
[20] Y.D. Li, S.Y. Li, A new generic chaos optimization combination method, Control Theory Appl. 19 (1) (2002) 143–145.

[21] W. Li, X.M. Liang, A hybrid algorithm based on chaos optimization and steepest decent, Algorithm, Comput. Techn. Automat. 22 (2) (2003) 12–14.
[22] S.S. Liu, Z.J. Hou, Weighted gradient direction based chaos optimization algorithm for nonlinear programming problem, in: Proceedings of the Fourth World Congress on Intelligent Control and Automation, 2002, pp. 1779–1783.
[23] D. Luenberger, Optimization by Vector Space Methods, Wiley, New York, 1997.
[24] J. Nocedal, S.J. Wright, Numerical Optimization, Springer, Berlin, 1999.
[25] H.-O. Peitgen, D. Saupe, The Science of Fractal Images, Springer, Berlin, 1988.
[26] A. Schrijver, Theory of Linear and Integer Programming, Wiley, New York, 1998.
[27] I. Tokuda, K. Aihara, T. Nagashima, Adaptive annealing for chaotic optimization, Phys. Rev. E 58 (1998) 5157–5160.
[28] J.L. Varona, Graphic and numerical comparison between iterative methods, Math. Intelligence 24 (2002) 37–46.
[29] J.J. Watkins, Across the Board: The Mathematics of Chess Problems, Princeton University Press, Princeton, NJ, 2004.
[30] X.X. Wu, Z. Chen, Introduction of chaos theory, Shanghai Science and Technology, Bibliographic Publishing house, 1996.
[31] L. Xu, S.O. Zhou, H.M. Zhang, A hybrid chaos optimization method and its application, System Eng. Electron. 25 (2) (2003) 226–228.
[32] H.P. Xu, H. Zhu, T. Zhang, Z.C. Wang, Application of mutation scale chaos algorithm in power plant and units economics dispatch, J. Harbin Inst. Tech. 32 (4) (2000) 55–58.
[33] M. Yamaguti, M. Hata, J. Kigami, Mathematics of Fractals, American Mathematical Society, Providence, RI, 1997.
[34] J. Yang, J. Zhou, W. Wu, F. Liu, C. Zhu, G. Cao, A chaos algorithm based on progressive optimality and Tabu search algorithm, in: Proceedings of the Fourth Conference on Machine Learning and Cybernetics, 2005, pp. 2977–2981.
[35] Y. Yuan, W. Sun, Optimization Theory and Methods, Science Press, China, 1999.
[36] C. Zhang, X. Li, H. Shaho, Chaos optimization algorithm based on linear search and its application to nonlinear constraint optimization problems, Control Decis. 16 (1) (2001) 120–125.
[37] T. Zhang, H. Wang, Z. Wang, Mutative scale chaos optimization algorithm and its application, Control Decis. 14 (3) (1999) 285–288.
[38] ⟨http://mathforum.org/advanced/robertd/newtons.html⟩.