

# 時変関数に適応するためのABCアルゴリズムの修正

正員 西田 健\*

## Modification of ABC Algorithm for Adaptation to Time-Varying Functions

Takeshi Nishida\*, Member

(2011年8月31日受付, 2011年12月26日再受付)

Although ABC (artificial bee colony) algorithm is effective tool for finding solutions to static optimization problems, application to the dynamical problem which includes the time-varying element has been not assumed. A modification of ABC algorithm for adaptation to time-varying functions is proposed. To adjust to the change in the function, the procedure for reevaluating the bee at each time is introduced. It is shown that the proposed modification does not influence the search performance of the conventional algorithm. The efficiency of the modified ABC algorithm is demonstrated and verified by numerical simulations.

キーワード: artificial bee colony アルゴリズム, 時変関数

**Keywords:** artificial bee colony algorithm, time-varying function

### 1. はじめに

確率的な最適解探索法の一つであるABC (artificial bee colony) アルゴリズムは, 蜂の群知能を模倣した非線形計画問題の効率的な解法である<sup>(1)</sup>。ロバストかつ高速に非線形問題, 微分不能問題および多峰問題などの最適解探索を行う性能が数多くの研究によって検証され, 特に目的関数が高次元である場合の大局解探索性能において優れているという報告がある<sup>(2)</sup>。さらに, 様々な最適化問題に対する性能を強化するために改良がなされ<sup>(3)-(5)</sup>, 現在では数多くの応用事例への適用がなされている<sup>(6)(7)</sup>。しかし, ABCアルゴリズムは時不変関数の大局解探索を目的として提案されており, 多くの実用で生ずる観測ノイズや外乱による目的関数の時間変化は, いずれの研究でも考慮されていない。そこで本論文では, 時変関数に対する適応性能をABCアルゴリズムに付加するための修正を提案する。また, この修正はABCアルゴリズムの時不変目的関数に対する大局解探索性能に影響を及ぼさないこと, 修正後のABCアルゴリズムは単峰性および多峰性時変関数に対する適応性能を有することを数値シミュレーションにより示す。さらに, 各パラメータと探索性能の関係性についての検証を行う。

本論文の構成は次の通りである。まず, 2章においてABCアルゴリズムの処理手順を示し, 3章において, 時変関数に適応するための修正を提案する。さらに, 4章において数値シミュレーションにより提案する修正の有効性を評価する。そして最後に5章において本論文の結論を述べる。

### 2. Artificial Bee Colony アルゴリズム

**〈2・1〉 概要** ABCアルゴリズムでは, 摂食活動を行う3種類の人工蜂群, すなわち働き蜂群 (employed bees), 傍観蜂群 (onlooker bees) および斥候蜂群 (scout bees) と餌場 (food sources) を基本要素として, 目的関数の大局解探索問題をモデル化する。コロニー (colony) とは一つの蜂群を意味し, この蜂群の目的は最も評価の高い餌場の探索である。コロニーの半数は働き蜂と斥候蜂が占め, 残りは傍観蜂が占める。働き蜂は記憶している餌場の近くを飛び回ると同時に, 餌に関する情報を傍観蜂に送る。傍観蜂は働き蜂の情報に基づいて餌場の中から良いものを選択し重点的に探索する。餌場に関する情報が一定期間更新されなかった場合, そこに関連付けられた働き蜂は餌場を捨て, 斥候蜂に変化して新しい餌場へ移動する。

ABCアルゴリズムはこれらのモデルによって, 関数  $g: \mathbb{R}^D \rightarrow \mathbb{R}$  についての最適化問題

$$\min_{\mathbf{x}} g(\mathbf{x}), \text{ subj. to } \mathbf{x} \in \mathbb{R}^D \dots \dots \dots (1)$$

を解くことを目的とする。ここで  $\mathbf{x} \in \mathbb{S}$  であり, 目的関数  $g$  は探索空間  $\mathbb{S} \subseteq \mathbb{R}^D$  上で定義される。本研究では,  $\mathbb{S}$  を

\*九州工業大学  
〒804-8550 福岡県北九州市戸畑区仙水町1-1  
Kyushu Institute of Technology  
1-1, Sensui, Tobata, Kitakyushu, Fukuoka 804-8550,  
Japan

$\mathbb{R}^D$  内の  $D$  次元超立方体  $\{\mathbf{x} \in \mathbb{R}^D \mid l \leq x_j \leq u, \forall j\}$  で定義する。ここで  $j \in J = \{1, 2, \dots, D\}$  であり,  $l$  と  $u$  はそれぞれ  $x_j$  の下限と上限を表す。

**(2.2) 処理手順** この最適化問題を解くための ABC の手順は次のように表される†。

**given** 働き蜂の数を  $n_e$ , 傍観蜂の数を  $n_o$  とし, コロニーサイズを  $N = n_e + n_o$  とする。働き蜂から斥候蜂への変化を制御する値  $limit$  を設定する‡。総反復回数  $C_{max}$  を設定する。探索空間の範囲を限定する  $l$  と  $u$  を設定する。

**step 0**

- (1) 反復回数を  $c = 1$  とし, 餌場もしくは働き蜂の位置を  $\mathbf{x}_i \in \mathbb{R}^D$  と表す††。ここで  $i \in I = \{1, 2, \dots, n_e\}$  は働き蜂の番号である。働き蜂が連続して更新されなかった回数を  $s_i$  と表し, まず  $s_i = 0$  ( $\forall i$ ) と設定する。また, 働き蜂を一様乱数を用いて  $\mathbb{S}$  内に配置する。
- (2) 初期配置における適合度を計算する。

$$f_i := \begin{cases} \frac{1}{1 + g(\mathbf{x}_i)} & \text{if } g(\mathbf{x}_i) \geq 0 \\ 1 + |g(\mathbf{x}_i)| & \text{otherwise} \end{cases} \dots\dots (2)$$

ここで  $:=$  は代入を表す。また, 初期配置における最良解とその関数値を以下のように保持する。

$$\mathbf{x}_{best} := \mathbf{x}_{i_b} \dots\dots\dots (3)$$

$$f_{best} := f_{i_b} \dots\dots\dots (4)$$

ここで  $i_b = \arg \max_i f_i$  である。

**step 1** このステップは働き蜂による探索に相当する。

- (1) 新しい餌場の候補  $\mathbf{v}_i \in \mathbb{R}^D$  ( $i \in I$ ) を生成する。

$$\mathbf{v}_{ij} := \begin{cases} x_{ih} + \phi(x_{ih} - x_{mh}) & \text{if } j = h \\ x_{ij} & \text{otherwise} \end{cases} \dots\dots\dots (5)$$

ここで,  $\mathbf{v}_{ij}$  と  $x_{ij}$  はそれぞれ  $\mathbf{v}_i$  と  $\mathbf{x}_i$  の第  $j$  要素,  $m \in I$  と  $h \in J$  はそれぞれ  $I$  と  $J$  からランダムに選択された値,  $\phi = [-1, 1]$  は一様乱数である。

- (2)  $\mathbf{v}_i$  の適合度を以下のように求める。

$$\hat{f}_i := \begin{cases} \frac{1}{1 + g(\mathbf{v}_i)} & \text{if } g(\mathbf{v}_i) \geq 0 \\ 1 + |g(\mathbf{v}_i)| & \text{otherwise} \end{cases} \dots\dots (6)$$

- (3) 適合度に基づいて働き蜂の情報を次のように更新する。

$$\mathbf{x}_i := \begin{cases} \mathbf{v}_i & \text{if } \hat{f}_i > f_i \\ \mathbf{x}_i & \text{otherwise} \end{cases} \dots\dots\dots (7)$$

† ここで示す実行手順は, ABC アルゴリズムの提案者が開設しているウェブサイト<sup>(8)</sup>において公開されている C 言語ソースコードに基づく。

††  $limit = 0.1 \cdot D \cdot N$  と設定する<sup>(2)</sup>。

††† 働き蜂と餌場は概念上別のものを指すが処理手順では完全に一致する。

$$f_i := \begin{cases} \hat{f}_i & \text{if } \hat{f}_i > f_i \\ f_i & \text{otherwise} \end{cases} \dots\dots\dots (8)$$

$$s_i := \begin{cases} 0 & \text{if } \hat{f}_i \geq f_i \\ s_i + 1 & \text{otherwise} \end{cases} \dots\dots\dots (9)$$

**step 2** このステップは傍観蜂による探索に相当する。

- (1) 適合度に基づいて相対確率を計算する。

$$p_i = f_i / \sum_{n=1}^{n_e} f_n \dots\dots\dots (10)$$

- (2) 相対確率に基づくルーレット選択によって  $J$  から働き蜂の番号を選択し, その番号の働き蜂に対して **step 1** を適用する。これを  $n_o$  回繰り返したら **step 3** へ移る。

**step 3**  $f_{i_b} > f_{best}$  を満たせば, 解とその関数値を更新する。

$$\mathbf{x}_{best} := \mathbf{x}_{i_b} \dots\dots\dots (11)$$

$$f_{best} := f_{i_b} \dots\dots\dots (12)$$

ここで  $i_b = \arg \max_i f_i$  である。

**step 4** このステップは斥候蜂による探索に相当する。

規定回数更新されなかった働き蜂, すなわち

$$m \in I_m = \{m \mid s_m \geq limit, m \in I\} \dots\dots\dots (13)$$

の番号を持つ働き蜂を一様乱数によって  $\mathbb{S}$  内に再配置して  $s_m = 0$  とし, 適合度  $f_m$  を計算し保持する。

**step 5**  $c = C_{max}$  であれば終了する。そうでなければ  $c := c + 1$  として **step 1** へ戻る。

**(2.3) 問題点** 例えば Fig.1 のように, ある時刻に目的関数が  $g(\cdot)$  から  $g^*(\cdot)$  へと変化する場合を考えてみよう。ここで,  $g(\cdot)$  と  $g^*(\cdot)$  の大局解をそれぞれ  $\mathbf{x}_{best}$  と  $\mathbf{x}_{best}^*$  とする。この図から,  $g^*(\cdot)$  を利用した解の評価が可能であれば,  $g^*(\mathbf{x}_{best}^*) < g^*(\mathbf{x}_{best})$  が明らかなので, その大局解  $\mathbf{x}_{best}^*$  の探索が可能であることがわかる。しかし  $g(\cdot)$  を利用した探索を続ける場合には, **step 3** の処理手順から  $g(\mathbf{x}_{best}^*) < g(\mathbf{x}_{best})$  もしくは  $f_{best}^* > f_{best}$  が満たされない限り解  $\mathbf{x}_{best}$  は更新されないで, 解は不適切な位置に拘束されたままになる。

この例からも明らかなように, ABC アルゴリズムでは目的関数  $g(\cdot)$  の時間変化を想定していないため, 最適解の

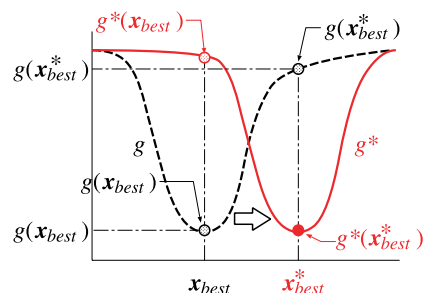


Fig. 1. Conceptual diagram of change of the global minimum on a time-varying function.

移動や目的関数の変化などの環境変化に適応した探索が行われない。

### 3. ABC アルゴリズムの修正

時変関数の大局解探索を目的とする ABC アルゴリズムを構成するために、従来型 ABC アルゴリズムの一部を以下のように修正する。まず、step 1 の (2) と (3) の間に次の手順を挿入する。

step 1 (2-2)  $x_i$  の適合度を以下のように計算する。

$$f_i = \begin{cases} \frac{1}{1+g(x_i)} & \text{if } g(x_i) \geq 0 \\ 1+|g(x_i)| & \text{otherwise} \end{cases}$$

この手順によって、目的関数の変化によって生ずる適合度の変化を、step 1(3)における働き蜂の更新に反映させることができるようになる。目的関数が時不変の場合には  $f_i$  の値に変化は生じ無いため、この手順の導入は時不変の目的関数の最適解の探索結果には影響を与えない。

次に、step 3 を次のように変更する。

step 3\* 探索解とその適合度を更新する。

$$x_{best} := x_{i_b} \dots\dots\dots (14)$$

$$f_{best} := f_{i_b} \dots\dots\dots (15)$$

ここで  $i_b = \arg \max_i f_i$  である。

この手順では、step 3 から  $f_{i_b} > f_{best}$  という条件を削除した。これにより、全時刻を通じてではなく、各時刻ごとの最大の適合度を求めることになり、目的関数の変化に伴う適合度の変化に適応することが可能になる。また一方、上述の step 1 (2-2) の導入に依らず、この手順の導入のみによっても、以下の流れによって  $f_{best}$  および  $x_{best}$  が大局解の移動に対して適応できる可能性がある：(1) 探索開始から一定時間で大域解が発見され、多くの働き蜂がその周辺に集まる：(2) 目的関数の変化により大域解が移動する：(3) 過去の大局解の周辺に配置された働き蜂は(2・3)項に示した理由により、さらに良い解を発見できないために更新が滞り、 $s_i$  の値が増加してゆく：(4) ある程度  $s_i$  の値が増加すると、働き蜂は step 4 の手順によって斥候蜂に変化し、探索空間内にランダムに再配置される：(5) すべての働き蜂が再配置され、それに伴ってすべての  $f_i$  の値が再初期化される：(6) 新たに算出された  $f_i$  の中から最大の  $f_{best}$  が選択され、移動した大局解に対する探索が再び進行する。しかし、以上の流れで再探索が進行し始めるには、すべての働き蜂が step 4 によって再初期化されるまで待つ必要があるため、その適応の速度は極めて遅い。この step 3\* の導入のみを行った場合の適応性能は後述のシミュレーションにより検証する。

以上より、時変目的関数に適応した最適解探索を目的として上述の 2 つの処理手順を導入する ABC アルゴリズムには、従来型 ABC アルゴリズムと比較して次に挙げるような特徴がある：(1) 時変関数の大局解の変化に対応して適合度  $f_i$  および  $f_{best}$  を逐次的に再評価するので、これらの

値が単調増加とならない；(2) 新たな設計パラメータの増加が無い；(3) 修正は簡潔で計算量の増加が抑えられている。

### 4. 数値シミュレーション

ここでは従来型 ABC アルゴリズムと提案アルゴリズムの性能を比較する。

〈4・1〉 時不変関数に対する探索性能 文献(2)などにおいて利用されている一般的な 4 種のベンチ関数、すなわち Rosenbrock 関数、Sphere 関数、Rastrigin 関数および Griewank 関数 (Table 1 および付録 1 参照) を用いて、従来型および修正アルゴリズムによる大局解を探索するシミュレーションを行った。コロニーサイズは 50, 100 および 200, 目的関数の次元数は 50, 100 および 150 について、すべての組み合わせに対するシミュレーションを行い、探索解の関数値  $g_p(x_{best})$  ( $p = 1, \dots, 4$ ) の推移を検証した。ただし、各シミュレーションにおいて同一の疑似乱数の系列を用いた。その結果を Fig. 2 に示す。いずれの関数を用いたシミュレーションにおいても、従来型 ABC アルゴリズムと修正アルゴリズムによる探索結果の推移は完全に一致した。これらの結果より、提案するアルゴリズムの修正は、時不変関数の大局解探索性能およびその結果に対して影響を及ぼさないことが確かめられた。

#### 〈4・2〉 時変関数に対する適応性能

〈4・2・1〉 単峰性時変関数 ここでは、以下に示す単峰性時変関数の大局解探索を考える。

$$g_5(x_1, x_2, k) = 1 - \exp \left[ - \frac{(x_1 - 250 - 125 \sin \alpha k)^2}{2 \cdot 40^2} - \frac{(x_2 - 250 + 125 \cos \alpha k)^2}{2 \cdot 40^2} \right]$$

この関数の形状を Fig. 3 に示す。この時変関数は、ビジュアルトラッキング問題を模擬したものである。すなわち、ガウス関数で表現される対象の特徴量の分布が画像中で連続して移動し、それを追跡するような問題を想定している。大局解における関数値は 0 であり、その位置は  $(x_1, x_2) = (250, 250)$  を中心とした半径 125 の円周上を移動し、およそ  $(2\pi/\alpha)$  ステップで元の位置に戻る。ここで、 $k$  は離散時刻を表し、この値が 1 増加する間に ABC アルゴリズムの 1 ステップが実行されるとする。 $\alpha$  の値を大きく設定するほど 1 ステップ毎の変化量が大きくなる。本シミュレーションにおける各アルゴリズムの目的は、一定速度で移動し続ける  $g_5(x_1, x_2, k)$  の大局解の位置を  $x_{best}$  として捉え続けることである。ここでは、文献(1)(2)で用いられた働き蜂と傍観蜂の割合を引用して  $n_e = n_o = 100$  とし、目的関数の変化を制御するパラメータを  $\alpha = 0.01$  と設定した。

従来型 ABC アルゴリズム、step 3\* の導入のみを行った不完全な修正アルゴリズム、提案する修正アルゴリズムの三種類を上述の時変目的関数に適用し、各アルゴリズムに

Table 1. Numerical benchmark functions.

function	ranges	minimum value	name
$g_1(\mathbf{x}) = \sum_{n=1}^{D-1} [100(x_n^2 - x_{n+1})^2 + (x_n - 1)^2]$	$-100 \leq x_n \leq 100$	$g_1(\mathbf{1}, 0, 0) = 0$	Rosenbrock function
$g_2(\mathbf{x}) = \sum_{n=1}^D x_n^2$	$-100 \leq x_n \leq 100$	$g_2(\mathbf{0}) = 0$	Sphere function
$g_3(\mathbf{x}) = \sum_{n=1}^D (x_n^2 - 10 \cos(2\pi x_n) + 10)$	$-5.12 \leq x_n \leq 5.12$	$g_3(\mathbf{0}) = 0$	Rastrigin function
$g_4(\mathbf{x}) = \sum_{n=1}^D \frac{x_n^2}{4000} - \prod_{n=1}^D \cos\left(\frac{x_n}{\sqrt{n}}\right) + 1$	$-600 \leq x_n \leq 600$	$g_4(\mathbf{0}) = 0$	Griewank function

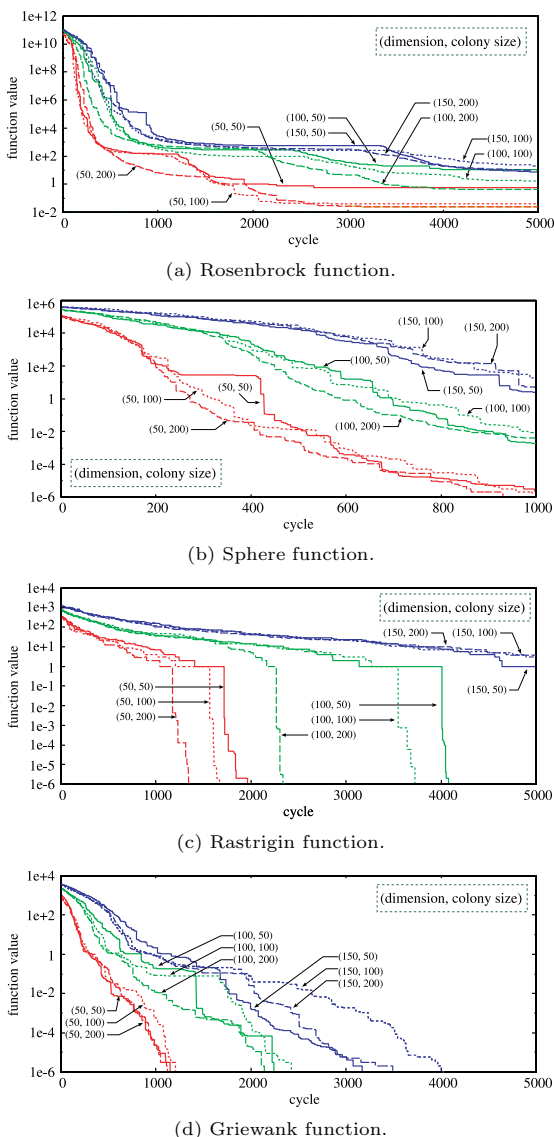


Fig. 2. Evolution of function value. The search results of original and modified algorithm have overlapped completely.

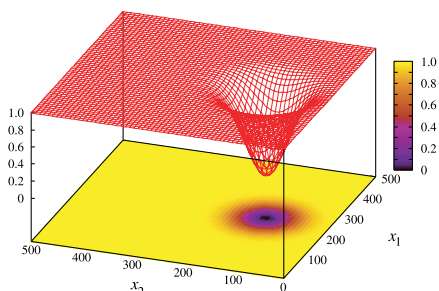


Fig. 3. Time-varying function  $g_5(x_1, x_2, k)$ .

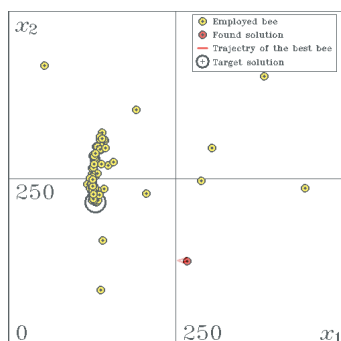
よって得られた  $c = 500$  の時点での働き蜂の分布, 探索解の位置およびそれまでの  $\mathbf{x}_{best}$  の軌跡を, それぞれ Fig. 4 (a) から (c) に示す。また, 大局解と  $\mathbf{x}_{best}$  のユークリッド距離の時間推移を Fig. 4 (d) に示す。まず Fig. 4 (a) および (d) より, 従来型 ABC アルゴリズムでは探索開始直後に  $f_{best}$  の値が収束したことで探索解  $\mathbf{x}_{best}$  の更新が止まっていることがわかる。次に同図 (b) および (d) より, 不完全な修正アルゴリズムでは大局解の移動に対する探索解の追跡は脈動的であり, また探索解の移動軌跡は歪んでいることがわかる。これは, **step 3\*** の手続きによって, 斥候蜂の発生と各種パラメータの再初期化による適応的な探索が行われたが, その速度が遅いことが原因となって生じた現象である。一方, 同図 (c) および (d) より, 修正アルゴリズムによる  $\mathbf{x}_{best}$  の軌跡には歪みが少なく, 目的関数の変化に適応した大局解の追従が高精度に行われたことがわかる。これらの結果より, 本研究で提案する修正によって, 連続的に変化する単峰性時変関数の大局解に対して, 適応的に追従することが可能になることが確かめられた。

**4・2・2) 多峰性時変関数** ここでは, 以下に示す多峰性時変関数の大局解探索を考える。

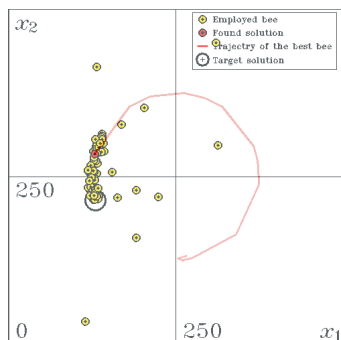
$$g_6(x_1, x_2, k) = 1 - \left[ \frac{\sin \beta k + 1}{2} \exp \left\{ -\frac{1}{2} \left( \frac{(x_1 - 125)^2}{40^2} + \frac{(x_2 - 375)^2}{40^2} \right) \right\} + \frac{\sin(-\beta k) + 1}{2} \exp \left\{ -\frac{1}{2} \left( \frac{(x_1 - 375)^2}{40^2} + \frac{(x_2 - 125)^2}{40^2} \right) \right\} \right]$$

関数の概形を Fig. 5 に示す。この時変関数は, 上述の関数と同様にビジュアルトラッキング問題を模擬したものである。すなわち, ガウス関数で表現される複数の対象の特徴量の分布が画像中で出現と消失を繰り返して不連続に移動し, それらを追跡するような問題を想定している。大局解における関数値は 0 から 0.5 の間で推移し, その位置は一定周期で (125, 375) と (375, 125) とで切り替わる。本シミュレーションにおける各アルゴリズムの目的は, 一定周期で切り替わる  $g_6(x_1, x_2, k)$  の大局解の位置を  $\mathbf{x}_{best}$  として捉え続けることである。また,  $\beta$  の値を大きく設定するほど, 1 ステップ毎の目的関数の変化量が大きくなる。ここで, 働き蜂と傍観蜂の数は前述のシミュレーションと同じく  $n_e = n_o = 100$  とし, 目的関数の変化を制御するパラメータを  $\beta = 0.05$  とした。本シミュレーションでは, 2000 回の試行の内に大局解の移動は 31 回発生した。

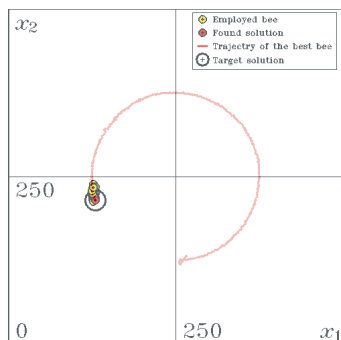
従来型 ABC アルゴリズム, **step 3\*** の導入のみを行った不完全な修正アルゴリズム, 提案する修正アルゴリズムの三種類を上述の時変目的関数に適用し, 各アルゴリズムに



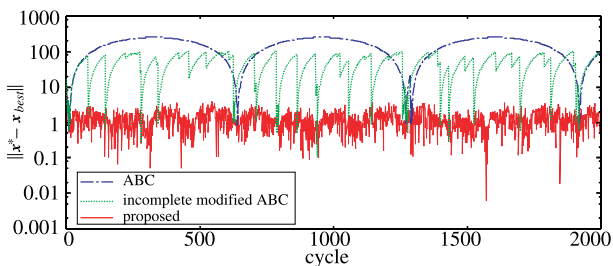
(a) Distribution of bees by original ABC at  $c = 500$ .



(b) Distribution of bees by incomplete modified ABC at  $c = 500$ .



(c) Distribution of bees by proposed ABC at  $c = 500$ .



(d) Time evolution of Euclidean distance between the global solution and the found solution.

Fig. 4. Results of the tracking simulation of the unimodal time-varying function  $g_5(x_1, x_2, k)$ .

よって得られた  $c = 500$  の時点での働き蜂の分布, 探索解の位置およびそれまでの  $x_{best}$  の軌跡を, それぞれ Fig. 6 (a) から (c) に示す。また, 大局解と  $x_{best}$  のユークリッド距離の時間推移を Fig. 6 (d) に示す。まず Fig. 6 (a) および (d) より, 従来型 ABC アルゴリズムでは  $x_{best}$  が一度だけ (125, 375) から (375, 125) に推移したが, その後は更新が止まり, 大局解の位置の変化に適応できなかったことがわかる。すなわち, 従来型 ABC アルゴリズムにより発見

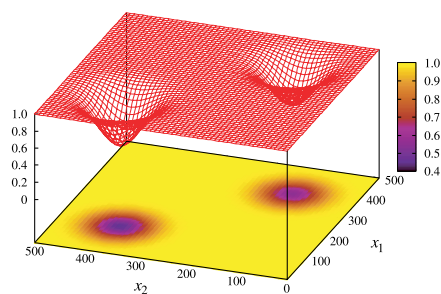
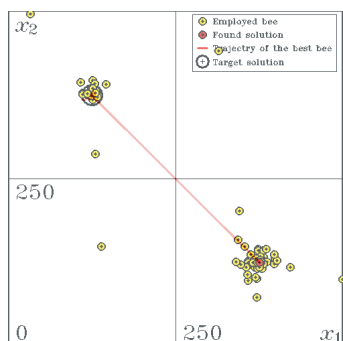


Fig. 5. Time-varying function  $g_6(x_1, x_2, k)$ .

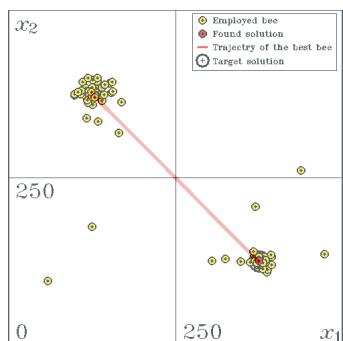
された解は (375, 125) に留まったままであったため, 大局解とのユークリッド距離は大局解が (375, 125) に移った場合には 0, (125, 375) に移った場合には  $250\sqrt{2}$  となり, これらの二つの値で振動的に推移した。次に同図 (b) および (d) より, 不完全な修正アルゴリズムでは, やはり適応的な探索が行われなかったことがわかる。これは, **step 3\*** の手続きによる適応的な探索の速度が遅いことが原因となって生じた結果である。一方, 同図 (c) より, 修正アルゴリズムでは, 大局解の位置の移動に対して  $x_{best}$  が適応できたことがわかる。これらの結果より, 本研究で提案するアルゴリズムの修正によって, 不連続的に変化する多峰性時変関数の大局解に対して適応的に追従し探索することが可能であることが確かめられた。

**〈4・3〉 コロニーサイズと適応性能** ここでは, コロニーサイズが修正アルゴリズムの探索性能に与える影響について検証する。前述の時変目的関数  $g_5(x_1, x_2, k)$  と  $g_6(x_1, x_2, k)$  を用いて, コロニーサイズを 10, 50 および 100 としてシミュレーションを行った結果を Fig. 7 に示す。 $\alpha = \beta = 0.01$  とし,  $n_e$  の値はコロニーサイズの半数とした。これらの結果より, コロニーサイズが  $N = 10$  と小さい場合は移動する最適解を適切に追跡できていないことがわかる。また, コロニーサイズを大きく設定することで, 適応性能を向上できることがわかる。

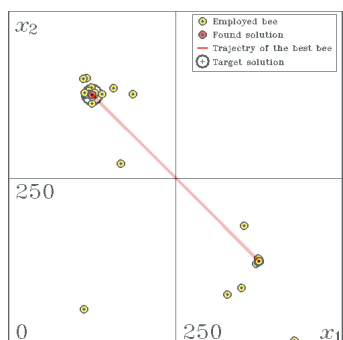
**〈4・4〉 目的関数の移動速度と適応性能** ここでは, 〈4・2・1〉項で用いた単峰性時変関数と〈4・2・2〉項で用いた多峰性時変関数を用いたシミュレーションにより, 目的関数の移動速度と修正アルゴリズムの適応性能の関係性について検証する。前述の時変関数  $g_5(x_1, x_2, k)$  と  $g_6(x_1, x_2, k)$  の変化速度を調整する  $\alpha$  と  $\beta$  の値を 0.01, 0.05 および 0.1 と変更して, 同様のシミュレーションを行った。コロニーサイズは 200 とし,  $n_e = 100$  と設定した。シミュレーションの結果得られた大局解と探索解のユークリッド距離の時間推移を Fig. 8 に示す。また,  $g_5(x_1, x_2, k)$  に対して  $\alpha = 0.05, 0.1$  とした場合の探索解の軌道と, 時刻  $k = 100$  および  $k = 50$  での働き蜂の配置を Fig. 9 に示した。これらの結果より, いずれの速度に対しても目的関数の変化に伴う大局解の移動に適応できていることがわかるが, 目的関数の変化速度の増加に伴い大局解と探索解のユークリッド距離が大きくなる傾向があることがわかる。また Fig. 9 より, 目的関数の変化速度の増加に伴い大局解の移動軌道に対する探索解



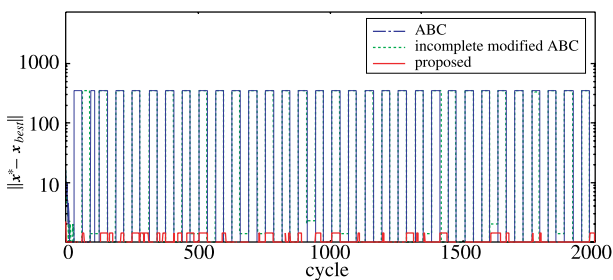
(a) Distribution of bees by original ABC at  $c = 500$ .



(b) Distribution of bees by incomplete modified ABC at  $c = 500$ .



(c) Distribution of bees by proposed ABC at  $c = 500$ .



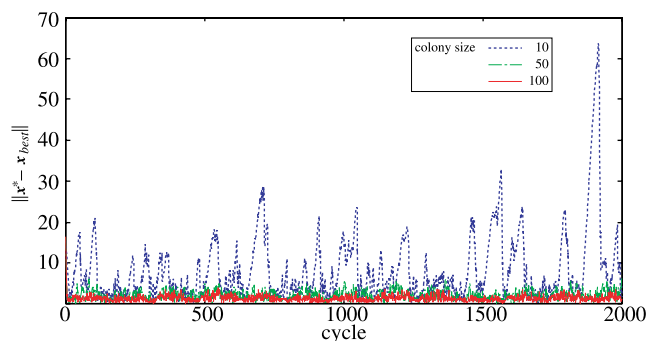
(d) Time evolution of Euclidean distance between the global solution and the found solution.

Fig. 6. Results of the tracking simulation of the multi-modal time-varying function  $g_6(x_1, x_2, k)$ .

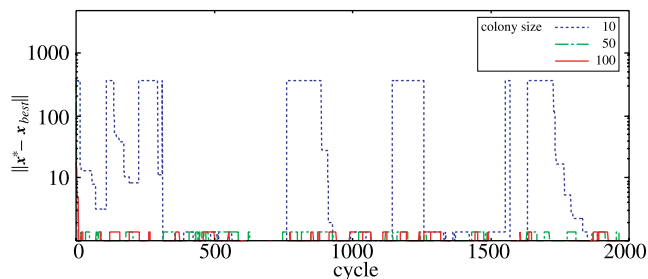
の軌道の歪みが大きくなることわかる。

〈4・5〉 次元数と適応性能 ここでは、以下に示す  $D$  次元単峰性時変関数の大局解探索を考える。

$$g_7(\mathbf{x}, k) = 1 - \exp \left\{ -\frac{1}{2} \sum_{n=1}^D \left( \frac{x_n - 125 \sin \alpha k}{40} \right)^2 \right\} \dots \dots \dots (16)$$

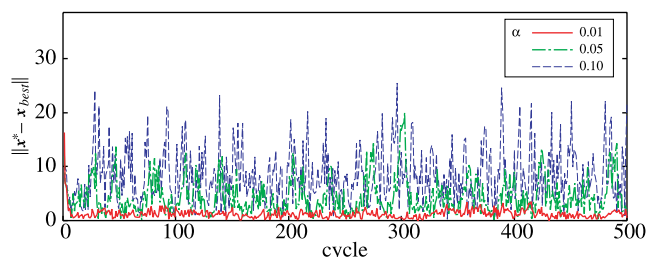


(a)  $g_5(x_1, x_2, k)$

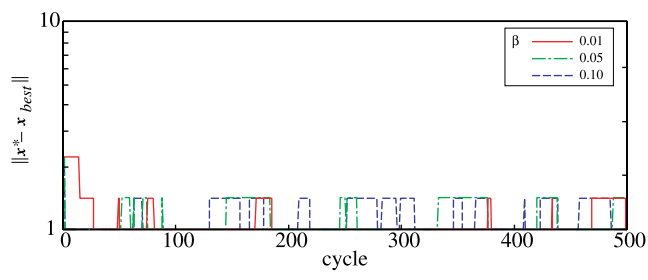


(b)  $g_6(x_1, x_2, k)$

Fig. 7. Time evolution of the mean squared error vs. the colony size.



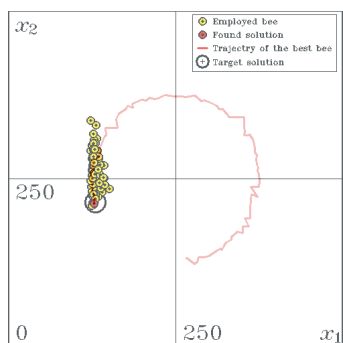
(a)  $g_5(x_1, x_2, k)$



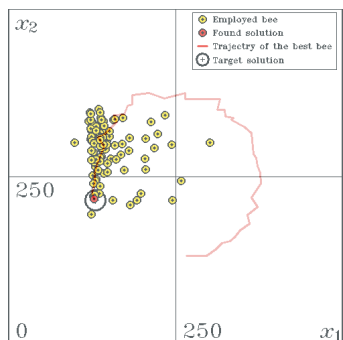
(b)  $g_6(x_1, x_2, k)$

Fig. 8. Time evolution of the mean squared error vs. the changing speed of the function.

ここで  $x_n$  は  $\mathbf{x} \in \mathbb{R}^D$  の第  $n$  要素を表し、探索領域は  $-500 \leq x_n \leq 500$  で張られる超立方体の内部とする。この関数は  $x_n = \sin \alpha k$  ( $\forall n$ ) において最小値 0 をとる。ここでは  $\alpha = 0.01$  および  $n_e = n_o = 100$  として提案アルゴリズムによる最小解探索を行った。ただし、〈4・1〉に示した結果からわかるように、対象とする関数の次元が増加すると、解探索の収束までにより多くの繰り返し回数が必要となるため、ここでは目的関数の離散時間  $k$  が 1 ステップ進む間に、アルゴリズムの繰り返しステップ  $c$  を次元数と同じ回

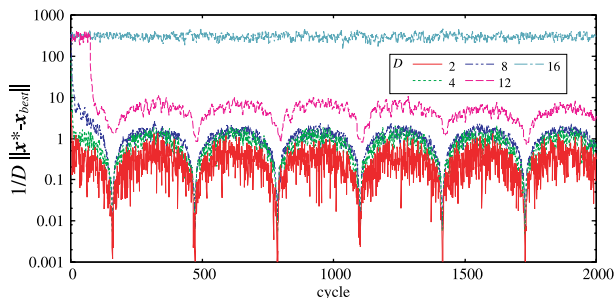


(a)  $\alpha = 0.05$

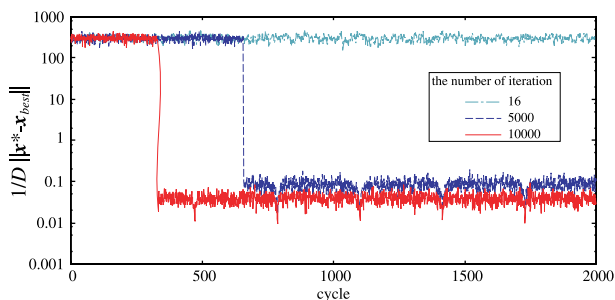


(b)  $\alpha = 0.1$

Fig. 9. Trajectory of bees at (a)  $k = 100$  and (b)  $k = 50$  for tracking of  $g_5(x_1, x_2, k)$ .



(a) The number of dimensions vs. search performance.



(b) The number of iteration vs. search performance ( $D = 16$ ).

Fig. 10. Time evolution of distance between  $\mathbf{x}_{best}$  and the optimal solution  $\mathbf{x}^*$ .

数だけ増加させる, すなわち探索を  $D$  回繰り返すこととした。また, 探索解と最適解の位置のユークリッド距離を次元数で割った値  $1/D \|\mathbf{x}^* - \mathbf{x}_{best}\|$  により, 追跡精度を評価した。次元数を  $D = 2, 4, 8, 16$  と変化させた場合の結果を Fig. 10 (a) に示す。この結果より, 次元数の増加に伴う解の探索精度と収束速度の低下が見られる。特に次元数が

$D = 16$  の場合には, 大局解の追跡に失敗していることがわかる。次に, 次元数が  $D = 16$  の場合において, 関数の離散時刻  $k$  が 1 ステップ進む間の提案アルゴリズムの繰り返し回数を 16 回, 5000 回, 10000 回と変化させた場合の解の探索結果を Fig. 10 (b) に示す。この結果より, 次元数の増加に伴ってアルゴリズムの適用回数を大幅に増加することで大局解の発見と追跡が可能となり, さらに追跡精度を向上できることがわかる。

### 5. おわりに

本論文では, まず ABC アルゴリズムが目的関数の時間変化に適応できない原因は, 働き蜂の適合度  $f_i (\forall i)$  と探索解の適合度  $f_{best}$  が単調増加するアルゴリズム構造にあることを指摘した。次に, この問題を解消するための修正方法を提案し, 数値シミュレーションによりその性質を検証した。提案した修正アルゴリズムは, 目的関数の変化に応じて適合度  $f_i$  および  $f_{best}$  の値を各時刻で再評価することで, 目的関数の時間変化を陽に考慮する構造を持つ。また, 新たなパラメータの導入を必要とせず, 計算量の大幅な増加やアルゴリズム構造の複雑化を生じさせない。

現在までに提案されている数多くの進化的アルゴリズム (EAs: evolutionary algorithms) の枠組みにおいて, ABC アルゴリズムの特徴は高次元の最適探索問題における優位性にある<sup>(2)</sup>。本研究ではまず, 提案アルゴリズムが高次元の静的目的関数の探索問題に対して従来型 ABC アルゴリズムと同等の性能を有することをシミュレーションにより示し, 提案手法がその優位性を継承することを示した。次に, 単峰性および多峰性の多次元時変関数を用いて, 提案手法がこれらに対する適応的な探索性能を有することを示した。さらに, より高次元の時変目的関数への適用の可能性について検証した。

EAs の多くは探索中の環境変化を前提としておらず, 動的目的関数の最適探索に利用できないが, 近年, 動的環境変への適応のために各種 EAs の改良が提案されている<sup>(10)(11)</sup>。しかし一方で, これらの改良版 EAs が前提とする動的問題は様々であり体系化がなされていない。したがって, 動的問題への適用を前提する他の EAs と提案手法の性能比較は今後の課題である。また今後の発展として, 現在までに数多く提案されている改良型 ABC アルゴリズムへの本修正の適用と性能検証が考えられる。さらに, 具体的もしくは実際の時変システムに対する本手法の適用が考えられる。

### 文 献


- (1) D. Karaboga: "An idea based on honeybee swarm for numerical optimization", Technical Report TR06, Erciyes University, Engineering Faculty, Computer Engineering Department (2005)
- (2) D. Karaboga and B. Basturk: "On the performance of artificial bee colony (ABC) algorithm", Applied Soft Computing, Vol.8, pp.687-697 (2007)
- (3) F. Kang, J. Li, and Z. Ma: "Rosenbrock artificial bee colony algorithm for accurate global optimization of numerical func-

- tions”, Information Sciences, Vol.181, pp.3508–3531 (2011)
- (4) W. Gao and S. Liu: “Improved artificial bee colony algorithm for global optimization”, Information Processing Letters, Vol.111, pp.871–882 (2011)
  - (5) W. Gao and S. Liu: “A modified artificial bee colony algorithm”, Computers & Operations Research, Letters, Vol.39, pp.687–697 (2012)
  - (6) M. Horng: “Multilevel thresholding selection based on the artificial bee colony algorithm for image segmentation”, Computers & Operations Research, Expert Systems with Applications, Vol.38, pp.13785–13719 (2011)
  - (7) W. Y. Szeto, Y. Wu, and S. C. Ho: “An artificial bee colony algorithm for the capacitated vehicle routing problem”, European Journal of Operational research, Vol.215, No.1, pp.126–135 (2011)
  - (8) Artificial Bee Colony (ABC) Algorithm Home Page: <http://mf.erciyes.edu.tr/abc/>
  - (9) M. Doi, T. Kamiya, and Y. Mori: Evolutionary Algorithms: Genetic Programming, Particle Swarm Optimization, Memetic Algorithm, Cma-Es, Gaussian Adaptation, Harmony Search, Evolutionary Algorithm, Mutation Testing, Neuroevolution, Differential Evolution, Learning Classifier System, Evolution Strategy, Books LLS, Memphis (2010)
  - (10) M. Kominami and T. Hmagami: “A New Genetic Algorithm with Diploid Chromosomes by Using Probability Decoding for Adaptation to Various Environments”, *IEEJ Trans. EIS*, Vol.128, No.3, pp.381–387 (2008) (in Japanese)  
小南 学・濱上和樹:「環境の変化に応じた多様体維持を可能にする二倍体遺伝的アルゴリズム」, 電学論 C, Vol.128, No.3, pp.381–387 (2008)
  - (11) T. Nishida and T. Sakamoto: “Adaptive PSO for Online Identification of Time-Varying Systems”, *IEEJ Trans. EIS*, Vol.131, No.9, pp.1642–1649 (2011) (in Japanese)  
西田 健・坂本哲三:「時変システムのオンライン同定のための適応 PSO」, 電学論 C, Vol.131, No.9, pp.1642–1649 (2011)

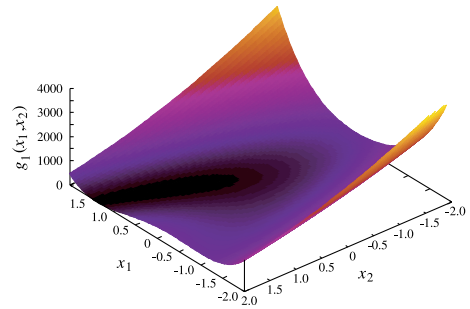
付 録

1. テスト用時不変関数

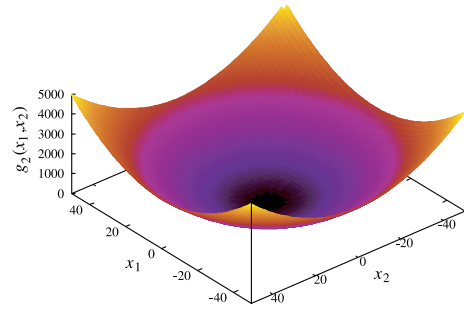
時不変関数に対する提案手法の大局解探索性能の評価に用いたテスト関数の 2 次元における概観を app. Fig. 1 に示す。



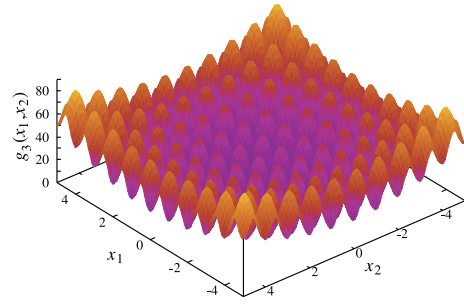
**西田 健** (正員) 1998 年九工大・工・設計生産工卒。2002 年九工大大学院博士後期課程修了。同年より九工大・機械知能工学・助手。2007 年より助教, 博士 (工学)。屋外移動ロボットに関する研究に従事。日本ロボット学会, 計測自動制御学会, 日本神経回路学会, 電子情報通信学会などの会員。



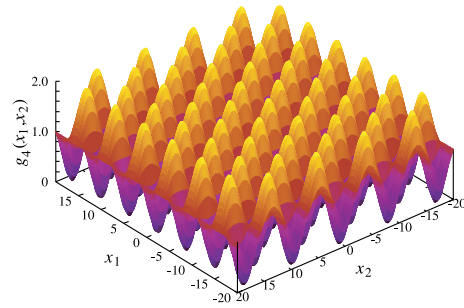
(a) Rosenbrock function.



(b) Sphere function.



(c) Rastrigin function.



(d) Griewank function.

app. Fig. 1. Test functions on 2nd dimentions.