動的問題に対する ABC 系アルゴリズムの適用に関する一研究

○宇谷　明秀＊1
土方　拓也＊2

キーワード：Artificial Bee Colonyアルゴリズム　差分進化　時変関数

１．はじめに

実システムの大規模化・複雑化に伴い，多くの工学設計問題が多数の局所解を有する高次元連続型多峰性関数の最適化問題として定式化されるようになってきた．特に文献[1],[2]のような自律分散型ネットワークの設計問題は設計変数間に依存関係はないが，今後の大規模領域でのネットワーク運用を考慮に入れると設計変数(最適化問題の次元数)は数百のオーダになることが想定される．既往の研究では高次元最適化問題に対する解探索性能に優れたArtificial Bee Colony(ABC)アルゴリズム[3]の改良法が提案され，代表的なベンチマーク関数を用いた数値実験を通してその有効性が検証されている[4],[5].

時不変関数の最適化を目的として提案されたこのArtificial Bee Colony(ABC)アルゴリズムは，近年，動的な実システムへの適用を考慮した改良が加えられ，観測ノイズの乱入や対象とするシステムの時間変化に起因する環境変化,すなわち探索空間の変化に対して適用可能なABCアルゴリズムの改良法が開発されている[6].

本研究では，文献[5]の差分進化型 ABC アルゴリズムに対して時変関数に適用するための機能を付加し，それに伴う改善策を加えた動的問題のための差分進化型 ABC アルゴリズムを提案する．提案アルゴリズムの有効性は代表的な時変関数に対する数値実験を通して検証する.

２．差分進化型ABCアルゴリズムと問題点

以下，差分進化型ABCアルゴリズムについて概説した後，差分進化アルゴリズムの動的環境へ適用することで発生する問題点について述べる.

２．１　差分進化型ABCアルゴリズムの概要

文献[4]の手法のベースとなるABCアルゴリズムは多次元解探索空間に配置された探索点と３種類の探索群(empl-oyed bees，onlookers，scouts) を基本要素として構成されている．ここで，各employed beesはある一つの探索点と関係づけられており，解探索過程で関連づけられた探索点の更新を試みる．onlookersによる解探索では相対的に価値の高い探索点の更新が繰返し試みられる．scoutsは遺伝的アルゴリズムにおける突然変異に相当する役割を担っている．しかし，ABCアルゴリズムには幾つかの問題点が内在している．文献[4]で指摘した問題点を以下に示す.

・適合度算出方法
・onlookersによる探索における探索点の選択方法
・参照点(更新を求める際の基準点)の選択方法
・scoutsによる探索における探索実行条件

文献[4]の手法では上記四つの問題点に対する改善策が導入され，高次元最適化問題に対する解探索性能の向上を実現している．また，文献[5]の手法では，文献[4]の手法に対し，大域的探索性向を有し，また多峰性関数の最適化問題において最も効果のある以下の変異ベクトル生成式(1)をemployed beesの探索, すなわちそれらの更新候補点の算出のために利用する.

$$\boldsymbol{v}_i = \boldsymbol{x}_{r_1} + F(\boldsymbol{x}_{r_2} - \boldsymbol{x}_{r_3}) \qquad (i = 1, \ldots, SN) \tag{1}$$

ここで，$\boldsymbol{v}_i\,(i = 1,\ldots, SN)$は各個体$\boldsymbol{x}_i\,(i = 1,\ldots, SN)$の変異ベクトルを表す．$r_1$, r_2, r_3は個体番号(i)以外の中からランダムに選ばれた互いに異なる個体番号であり，SNは総個体数を表し，Fは増幅率と呼ばれる更新量制御パラメータである．この変異ベクトル生成式の導入により，各探索点に関係づけられた各employed beesの解探索において，探索点の初期配置の影響が緩和され，解探索空間内をより大域的に探索できるようになる．employed beesの大域的探索能力の向上とonlookersによる相対的に価値の高い探索点の集中的な局所探索によって，文献[5]の手法の高次元最適化問題に対する解探索性能は文献[4]の手法よりも精度が向上することが確認されている.

２．２　動的環境へ適用への問題点

単峰性時変関数の変化前と1ステップ変化した後を図1に示す．図1のように，ある時刻に対する目的関数が$f(\cdot)$から$f^*(\cdot)$に変化する場合を考える．$f(\cdot)$に対する大域的最適解をx_{best}とし，変化後の$f^*(\cdot)$に対する大域的最適解をx^*_{best}とする．図1から$f^*(\cdot)$を利用した解を評価することが可能であれば，$f^*(x^*_{best}) < f(x_{best})$が明らかなので，その大域的最適解$x^*_{best}$の探索が可能であることがわかる．全体最良解の更新に関して $f_{best}^{k+1} > f(\boldsymbol{x}_{ib}^{k+1})$が満たされない限り解$x_{best}$は更新されないので，解は不適切な位置に拘束されたままである.

この例より，差分進化型ABCアルゴリズムは目的関数$f(\cdot)$の時間変化を想定していないため，目的関数の移動や環境の変化等に適用した探索が行われない.

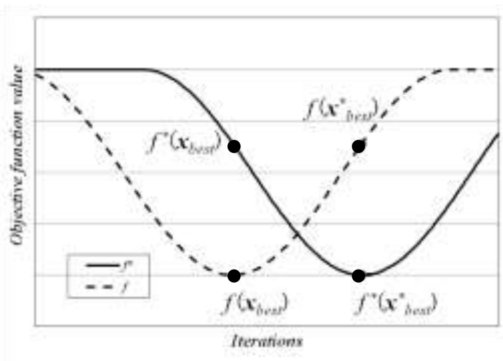

図1 単峰性時変関数の時刻変動

３．提案アルゴリズム

差分進化型ABCアルゴリズムに対して以下の３点を改善策を導入する．まず１点目はemployed beesによる解探索の始めに，前時刻の各探索点を現時刻の評価関数で再評価を行う．この手順によって，目的関数の変化によって生じる評価値の変化を，現時刻の各employed beeの更新に反映させることができるようになる．

次に２点目は全体最良解の更新において，$f_{best}^{k+1} > f(\boldsymbol{x}_{ib}^{k+1})$という条件を削除する．これにより，各時刻の全体最良解を求めることになるため，解が不適切な位置に拘束されたままになることを防ぎ，目的関数の変化に適用することが可能になる．

また３点目は目的関数の変化によって，解探索の収束状況判定も変化していくため探索初期段階の最低繰り返し回数($T1_{\mathrm{min}}$)を取り除く．

以上より，時変目的関数に適用した最適解探索を目的として上記に示した３つの処理手順を導入した差分進化型ABCアルゴリズムには次のような特徴がある．(1)時変目的関数の大域的最適解の変化に対用して評価値f_iおよび f_{best}を逐次的に再評価を行うため，これらの値が単調減少とならない．(2)設計パラメータの削減を実現させている．

３．１　提案アルゴリズムの処理手順

動的問題に対する改善策を加えた差分進化型ABCアルゴリズムの具体的な解探索手順は次の通りである．

［**Step0**（準備）］

・コロニーサイズ(N)及び探索点総数(SN)を設定する．ここで，employed beesの総数は探索点総数(SN)と同数設定し，onlookersの総数はコロニーサイズから探索点総数を引いた数($N-SN$)になる．

・適合度上位の探索点の数（α）を設定する．

・解への収束状況判定パラメータ(dr)を設定する．

・許容限界値(f_{bound})を設定する．

・f_{bound}に対する精度値($f_{accuracy}$)を設定する．

・総繰返し回数(T_{max})を設定する．

・更新量制御パラメータ(F)を設定する．

［**Step1**（初期化）］

・繰返し回数のカウンタを$k = 1$にする．

・探索段階2への切換判定値(f_{judge})を初期化する($f_{judge} = 0$)．

・各探索点の初期位置ベクトル($\boldsymbol{x}_i^1$)を乱数によって生成する．ここで，下付$i \in \{1, 2, \ldots, SN\}$は探索点番号を表し，上付($k =$)1は繰返し回数を表す．

・初期状態における全体最良解($\boldsymbol{best}^1$)を決定する．

$$i_b = arg\ \min_i\ f(\boldsymbol{x}_i^1), \quad i = 1, \ldots, SN$$

$$\boldsymbol{best}^1 = \boldsymbol{x}_{i_b}^1$$

・初期状態での全探索点の平均評価値(f_{init})を計算する．

［**Step2**（employed beesによる解探索）］

１）前時刻の各探索点を現時刻の評価関数で再評価を行う．

２）すべての探索点[$\boldsymbol{x}_i^k (i = 1, \ldots, SN)$]に対して，更新候補点[$\boldsymbol{v}_i^k (i = 1, \ldots, SN)$]を生成する．

$$v_{ih}^k = x_{r_1h}^k + F(x_{r_2h}^k - x_{r_3h}^k), \quad i = 1, \ldots, SN$$

$$v_{ij}^k = x_{ij}^k, \quad i = 1, \ldots, SN$$

ここで，r_1, r_2, r_3は探索点番号(i)以外の中からランダムに選択された互いに異なる探索点番号を表す．また，$h \in \{1, 2, \ldots, D$ (次元数)$\}$は探索点ごとにランダムに選択された一つの変数番号を表し，$j \in \{1, 2, \ldots, D\}$は選択された番号($h$)以外の残りの変数番号を表す．

３）各探索点($\boldsymbol{x}_i^k$)を更新する．

$$I_1 = \left\{ i \mid f(\boldsymbol{v}_i^k) < f(\boldsymbol{x}_i^k), \quad i = 1, \ldots, SN \right\}$$

とし，次のように更新する．

$$\boldsymbol{x}_i^k = \begin{cases} \boldsymbol{v}_i^k, & i \in I_1 \\ \boldsymbol{x}_i^k, & i \notin I_1 \end{cases}$$

［**Step3**（onlookersによる解探索）］

１）onlookersの探索カウンタを$l = 1$にする．

２）各探索点($\boldsymbol{x}_i^k$)の適合度(fit_i^k)を計算する．

$$fit_i^k = \begin{cases} \dfrac{1}{f(\boldsymbol{x}_i^k) - f_{bound}}, & f(\boldsymbol{x}_i^k) - f_{bound} \geq f_{accuracy} \\ \dfrac{1}{f_{accuracy}}, & f(\boldsymbol{x}_i^k) - f_{bound} < f_{accu} \end{cases}$$

$$(i = 1, \ldots, SN)$$

３）$f_{judge} \geq dr$であれば下記4)へ行き，そうでない場合は次式によってf_{judge}の値を更新する．

$$f_{judge} = \frac{f_{init} - f(\boldsymbol{best}^k)}{f_{init} - f_{bound}}$$

そして，この更新値が$f_{judge} \geq dr$になれば下記4)へ行き，そうでない場合は下記5)へ行く．

４）各探索点($\boldsymbol{x}_i^k$)の相対価値確率(P_i^k)を算出する．

$$P_i^k = fit_i^k \Bigg/ \sum_{n=1}^{SN} fit_n^k$$

5）$(f_{judge} \geq dr)$の場合は相対価値確率(P_i^k)に基づくルーレット選択から一つの探索点$(\boldsymbol{x}_c^k)$を選択し，そうでない場合は適合度の高い上位αの探索点の中からランダムに一つの探索点$(\boldsymbol{x}_c^k)$を選択する．ここで，$c \in \{1, 2, \dots, SN\}$は選択された探索点番号を表す．そして，この選択された探索点$(\boldsymbol{x}_c^k)$についてのみ，その更新候補点$(\boldsymbol{v}_c^k)$を生成する．

$$\boldsymbol{v}_{ch}^k = \boldsymbol{x}_{ch}^k + \phi_{ch}^k(\boldsymbol{x}_{ch}^k - \boldsymbol{x}_{mh}^k)$$
$$\boldsymbol{v}_{cj}^k = \boldsymbol{x}_{cj}^k$$

ここで，$h \in \{1, 2, \dots, D\}$はランダムに選択された一つの変数番号を表し，$j \in \{1, 2, \dots, D\}$は選択された番号(h)以外の残りの変数番号を表す．また，ϕ_{ch}^kは[-1,1]の一様乱数であり，$m \in \{1, 2, \dots, SN\}$については，$(f_{judge} \geq dr)$の場合は探索点番号$(c)$以外で相対価値確率$(P_i^k)$に基づくルーレット選択によって選択された探索点番号を表し，そうでない場合は探索点番号(c)以外で適合度の高い上位αの探索点の中からランダムに選択された探索点番号を表す．

6）探索点$(\boldsymbol{x}_c^k)$を更新する．

$$\boldsymbol{x}_c^k = \begin{cases} \boldsymbol{v}_c^k, & f(\boldsymbol{v}_c^k) < f(\boldsymbol{x}_c^k) \\ \boldsymbol{x}_c^k, & f(\boldsymbol{v}_c^k) \geq f(\boldsymbol{x}_c^k) \end{cases}$$

7）$l = N\text{−}SN$であれば**Step4**へ行く．

そうでない場合は$l = l+1$として上記5)へ戻る．

［**Step4**（全体最良解の更新）］

全体最良解を更新する．

$$\boldsymbol{x}_i^{k+1} = \boldsymbol{x}_i^k, \quad i = 1,\dots,SN$$
$$i_b = \arg\min_i f(\boldsymbol{x}_i^{k+1})$$
$$\boldsymbol{best}^{k+1} = \boldsymbol{x}_{i_b}^{k+1}$$

［**Step5**（終了判定）］

$k = T_{max}$であれば解探索を終了する．

そうでなければ$k = k+1$として**Step2**へ戻る．

4．数値実験

ここでは時間に応じて環境が変化する状況を考慮した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　ベンチマーク問題と実験設定

本研究では，以下に示す時間によって環境が変化する単峰性時変関数を用いて，提案アルゴリズムの有効性を検証する．

この関数の形状を図2に示す．この時変関数は大局解の位置が$(\boldsymbol{x}_1, \boldsymbol{x}_2)$=(250, 250)を中心点とした半径125の円周上を移動し，それを追跡するよう探索を行う．今回の実験では目的関数の変化を制御するパラメータをα=0.01と設定した．この関数の最適解における評価値は0であり，大局解の位置は628ステップで元の位置に戻る．αの値を大きくするほど1ステップごとの変化量が大きくなる．ここでkは離散時刻を表し，k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．

・単峰性時変関数

$$\boldsymbol{f}_1(\boldsymbol{x}_1, \boldsymbol{x}_2, \boldsymbol{k}) = 1 - \exp\left[-\frac{(\boldsymbol{x}_1 - 250 - 125\sin\alpha k)^2}{2 \cdot 40^2} - \frac{(\boldsymbol{x}_2 - 250 + 125\cos\alpha k)^2}{2 \cdot 40^2} \right]$$

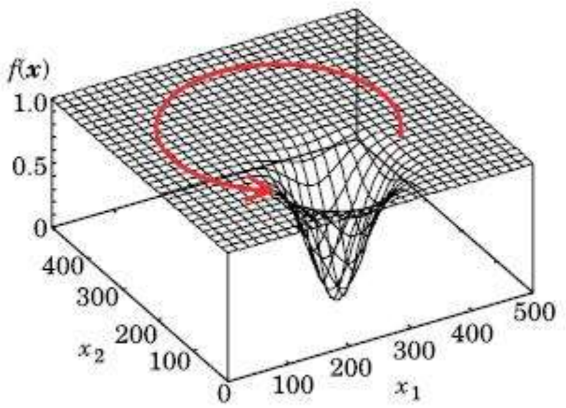

図2　単峰性時変関数

比較対象のアルゴリズムを以下に列挙する．

(1) ABCアルゴリズム

(2) 動的環境を考慮したABCアルゴリズム

以下，上記(1) ABCアルゴリズムをOriginal ABC，上記(1)動的環境を考慮したABCアルゴリズムをABCとし，本提案アルゴリズムをProposalと表記する．

数値実験における”ABC”及び提案アルゴリズムの設定値を表1に示す．

表1　実験で用いた設定値

	ABC	*Proposal*
colony size	*200*	*200*
employed bees	*50% of colony size*	*50% of colony size*
onlookers(N-SN)	*50% of colony size*	*50% of colony size*
α	――――――	*0.3×SN*
dr	――――――	*0.99*
fbound	――――――	*0*
faccuracy	――――――	*10×10-16*
Tmax	*3000*	*3000*
F	――――――	*1*

4．2　実験結果及び考察

ABCアルゴリズムに動的環境を考慮した文献[5]の手法と提案アルゴリズムを上述の時変関数に適用し，各アルゴ

リズムによって得られたx_{best}と大局解のユークリッド距離の時間推移を図3に示す．また，図4は各手法の収束過程を比較した実験結果の一例であり，単峰性時変関数(D=2)の繰り返し回数3000までの平均値の収束過程が示されている．提案した改善策によって，時間変化する目的関数の探索が可能になっていることがわかる．さらに図3に示すように提案アルゴリズムが文献[5]の手法と比較して，目的関数の変化に対しより高精度に適用できていることから提案アルゴリズムの有効性が確認できる．図4の結果よりOriginal ABCが収束しているように見えるが，図3より大域的最適解との距離が離れていることから環境の変化に対して追従できていないことがわかる．一方，本提案アルゴリズムは図4，図3から環境の変化に対して追従できていることが確認できる．

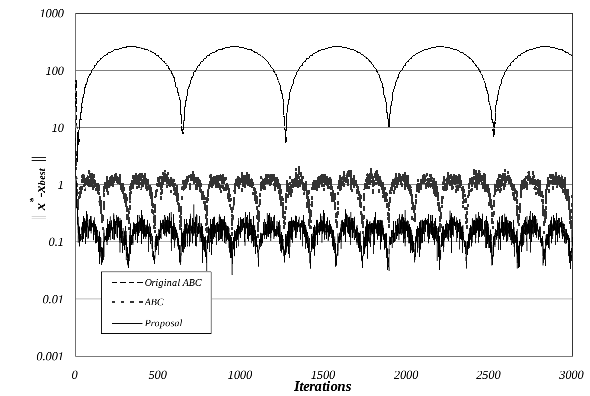

図3　距離による性能比較

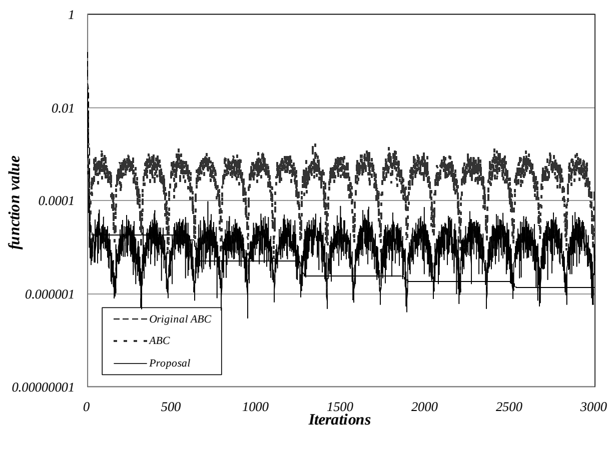

図４　評価値による性能比較

５．おわりに

本研究では，動的に環境が変化する問題に対する差分進化型 ABC アルゴリズムを提案した．提案アルゴリズムの有効性は時変関数に対する数値実験を通して検証した．動的問題のための ABC アルゴリズム[6]との比較を通して，提案アルゴリズムの優位性を確認した．提案アルゴリズムは，例えば大規模ネットワークの動的制御問題に対して効果的に用いることができる．これらの分野への提案アルゴリズムの導入(適用法)に関しては今後の研究課題とする．

［参考文献］

1) 宇谷明秀，山本尚生，”複数の許容解を探索する Particle Swarm Optimization とその複数シンク無線センサネットワークにおけるシンクノード配置問題への適用,”信学論(D),　vol.J93-D, no.5，pp.555-567，May 2010.
2) 長島淳也，宇谷明秀，山本尚生，”複数許容解探索型粒子群最適化手法の無線センサネットワークへの適用―フラッディング効率化のための各センサノードの送信電力調整,” 日本知能情報ファジィ学会誌，vol,23，no.1，pp.65-77，2011.
3) D. Karaboga and B. Basturk, “A powerful and efficient algorithm for numerical function optimization: artificial bee colony (ABC) algorithm,” J. Global Optimization, vol.39, pp.459-471, 2007
4) 宇谷明秀，長島淳也，午膓隆太，山本尚生，“Artificial Bee Colony(ABC)アルゴリズムの高次元問題に対する解探索性能の強化，信学論(D), vol.J94-D，no.2，pp.425-438，Feb. 2011
5) 香川卓哉，宇谷明秀，山本尚生：高次元最適化問題のための差分進化型改良 ABC アルゴリズム，電学論(A), vol.J95-A, no.6, pp. 514-518, 2012.
6) 西田健:時変関数に適応するための ABC アルゴリズムの修正，電学論(C), vol.132, no.4, pp.584-591, 2011.

＊1 東京都市大学　教授　博士(工学)
＊2 東京都市大学大学院　博士前期課程