論　文

種分化を導入したDifferential Evolutionによる複数解をもつ多峰性関数の最適化

柴坂美祐喜[†*]　原　章[†]　市村　匠[†]　高濱　徹行[†]

Species-Based Differential Evolution for Multimodal Function Optimization

Miyuki SHIBASAKA[†*], Akira HARA[†], Takumi ICHIMURA[†], and Tetsuyuki TAKAHAMA[†]

あらまし　複数の最適解をもつ多峰性問題に対し同時にすべての最適解を発見する研究が行われている．この研究の一つとして，探索空間を分割するために種分化を導入したDifferential Evolution (SDE) が提案されている．しかしこの方法は，最適解の分布状況や局所最適解の数，関数の形状などに依存するため，問題によっては，関数の評価回数が膨大になる場合がある．また種分化に使う種のなわばり半径を決めるために，予備実験による経験的なパラメータ設定が必要となる．そこで本論文では，関数評価回数を削減し，最適解の発見確率を高めるために，大域探索と局所探索を考慮した子個体の生成法，関数の峰の形状を利用する適応的なわばり半径決定法，次世代へ優良個体を残すエリート保存戦略の三つを取り入れた手法であるImproved SDE (ISDE) を提案する．代表的なテスト問題を用いて従来手法と性能を比較し，本研究の有効性を示す．

キーワード　多峰性関数最適化，Differential Evolution，種分化，適応的なわばり半径

1. まえがき

実数空間において関数の最大値若しくは最小値をとる解を求める最適化問題は，古くから研究されている．この問題の解決手法として，遺伝的アルゴリズム (GA) [1], [2] や鳥や魚の群れの行動を模倣したParticle Swarm Optimization (PSO) [3]，Differential Evolution (DE) [4] など，一般にメタヒューリスティクスといわれる解集団を用いた確率的な多点探索アルゴリズムが提案されている．しかし，現実には，最適解を一つだけ見つければよい問題ばかりではなく，非線形回路問題 [5] のようにすべての最適解やそのほかに存在する準最適解も含めた多くの可能性を列挙した方がよい問題もある．そのため，複数解をもつ関数においてすべての解を探索することは，問題解決において非常に重要であり，その問題を解決するために多くのアルゴリズムが研究されている [6]〜[8].

† 広島市立大学大学院情報科学研究科，広島市
Graduate School of Information Sciences, Hiroshima City University, 3–4–1 Ozuka Higashi, Asaminami-ku, Hiroshima-shi, 731–3194 Japan
* 現在，シャープビジネスコンピュータソフトウェア株式会社

しかし，多点探索アルゴリズムを複数の最適解をもつような関数に対して適用すると，発見した関数値の良い個体の方向へ探索が進んでしまい，すべての解を探索することができないという問題がある．また，探索方法として一つ若しくは二つの解を発見しながら，何度も試行を繰り返すことですべての最適解を発見することも考えられる．しかし，その手法ではすべての最適解を発見するまでにかなりの試行を繰り返さなければならない．その上，多くの試行を繰り返したからといって必ずすべての最適解が発見できるとはいえない．よって，1度の試行ですべての複数解を探索することは，非常に有効な方法であるといえる．

1度の試行ですべての最適解を探索する方法として，関数値の悪い個体にペナルティを与えるすみ分け型アルゴリズム [9], [10] や，クラスタリングを用いたkPSO [11]，種分化を導入したDifferential Evolution (Species-based DE, SDE) [12], Species-based GA [13], Species-based PSO [14], [15] などが提案されている．種分化とは関数値の良い個体を中心として，固定された半径 R のなわばり内の個体を種と認識し，種単位でオペレータを適用し探索を行う方法である．探索空間を分割し，それぞれの部分空間内にある

種を中心とした探索が行われるため，多様性を維持しながら最適化を行うことができる．またクラスタリングを用いて分類木を作ることで種を認識する種分化手法 [16] も提案されている．この方法では，ステップごとの個体の分布状況をもとに動的に探索空間を分割することで，適応的に最適化の状況に対する様々な大きさの種を形成することができる．

本研究では文献 [12] の種分化の方法をもとにしているが，収束速度を早め最適解の発見確率を高めるために，大域探索と局所探索を考慮して子個体を生成するための個体選択法，適応的なわばり半径決定法，エリート保存戦略の三つを取り入れた手法である Improved SDE (ISDE) を提案する．提案手法の有効性を検証するために，6 種類の関数最小化問題に適用し，従来手法である SDE [12] 及び kPSO [11] の実験結果と比較する．

2. では，本研究で用いた Differential Evolution を説明する．**3.** では，種分化を導入した最適化手法について，**4.** には本研究で提案する適応的なわばり半径をもつ種分化を導入した DE について述べる．**5.** で関数最適化実験の結果と考察を行い，最後の **6.** で今後の課題について述べる．

2. Differential Evolution

Differential Evolution [4] は，1995 年に提案された実数空間における最適化アルゴリズムである．そのアルゴリズムの基本的な構造は，遺伝的アルゴリズムとよく似ているが，子個体の生成部分や置換えなどの点において，方法が異なっている．$DE/base/num/cross$ は DE の表記方法で，以降で説明する DE の手法を示している．$<base>$ はベースベクトル $\boldsymbol{x}^{\boldsymbol{r_1}}$ の選択方法，$<num>$ は $\boldsymbol{x}^{\boldsymbol{r_2}}-\boldsymbol{x}^{\boldsymbol{r_3}}$ のような差分ベクトルの数，$<cross>$ は交叉方法を意味している．$\boldsymbol{x}^{\boldsymbol{r_1}}$, $\boldsymbol{x}^{\boldsymbol{r_2}}$, $\boldsymbol{x}^{\boldsymbol{r_3}}$ は子個体を生成するときに使われる個体で，詳細は後述する．DE のアルゴリズムを以下に示す．

（1） ランダムに個体を生成し集団を構成
（2） 個体の関数値を計算
（3） 子個体の生成
 (a) $\boldsymbol{x}^{\boldsymbol{r_1}}$, $\boldsymbol{x}^{\boldsymbol{r_2}}$, $\boldsymbol{x}^{\boldsymbol{r_3}}$ の選択
 (b) 変異ベクトル $\boldsymbol{v}$ の生成
 (c) 変異ベクトルと親個体を交叉
（4） 置換え
（5） 終了条件を満たしていなければ，（3）に戻る

問題の次元数である D 次元空間の中に NP 個の個体が存在するとし，DE の初期集団における各個体 $\boldsymbol{x}^{\boldsymbol{k}}= (x_1^k, x_2^k, \ldots, x_D^k)$ $k = 1, \ldots, NP$ は，問題の各次元における定義域内でランダムに初期化される．次に，D 次元の情報をもとにして，全 NP 個の個体の関数値 $f(\boldsymbol{x}^{\boldsymbol{k}})$ を計算する．

各々の個体 $\boldsymbol{x}^{\boldsymbol{P}}$ において，それぞれ一つずつ子個体を生成する．DE における子個体の生成は，GA のように親個体同士を交叉させるのではなく，親個体 $\boldsymbol{x}^{\boldsymbol{P}}$ と式 (1) で生成される変異ベクトル $\boldsymbol{v}$ を遺伝子単位で交叉させることで行われる．詳細は後に説明する．

$$\boldsymbol{v} = \boldsymbol{x}^{\boldsymbol{r_1}} + F \cdot (\boldsymbol{x}^{\boldsymbol{r_2}} - \boldsymbol{x}^{\boldsymbol{r_3}}) \tag{1}$$

DE/rand/1/exp では，式 (1) の $\boldsymbol{x}^{\boldsymbol{r_1}}$, $\boldsymbol{x}^{\boldsymbol{r_2}}$, $\boldsymbol{x}^{\boldsymbol{r_3}}$ は条件 $r1 \neq r2 \neq r3 \neq P$ のもとで全体からランダムに選択される．パラメータ F は，差分ベクトルの伸縮を意味するスケーリングファクタである．

図 1 に，DE/rand/1/exp の概要を示す．パラメータ CR は交叉率を，$U([0,1])$ は区間 $[0,1]$ の一様乱数を表している．DE では，変異ベクトルの生成と交叉は同時に行われる．すなわち，個体をランダムに選択し，交叉を始める遺伝子座 i をランダムに決め，$U([0,1])$ で発生させた乱数が交叉率 CR より小さくかつすべての遺伝子が交叉済みでなければ，親個体 $\boldsymbol{x}^{\boldsymbol{P}}$ をコピーした子個体 $\boldsymbol{x}^{\boldsymbol{child}}$ の遺伝子を変異ベクトル $\boldsymbol{v}$ の遺伝子に置換する．また，DE/best/1/exp では $\boldsymbol{x}^{\boldsymbol{r_1}}$ を集団全体の中で関数値が最良となる個体とする．

DE の置換えは，生成された子個体 $\boldsymbol{x}^{\boldsymbol{child}}$ と親個体 $\boldsymbol{x}^{\boldsymbol{P}}$ の関数値の比較を行い，子個体の関数値の方が良ければ置換えを行う．

なお，DE の $\boldsymbol{x}^{\boldsymbol{r_1}}$,$\boldsymbol{x}^{\boldsymbol{r_2}}$,$\boldsymbol{x}^{\boldsymbol{r_3}}$ の選択方法は，個体を全体からランダムに選択するため，少数の最適解に収束

```
親個体 x^P を選択する
x^P を x^child にコピーする
x^r1, x^r2, x^r3 を選択する (r1 ≠ r2 ≠ r3 ≠ P)
/*交叉開始点 i を決める*/
i = (0 から D − 1 までの整数値の一様乱数 );
L = 0;
/*親個体と変異ベクトルを交叉させる*/
do{
/*変異ベクトルを生成する*/
  v_i = x_i^r1 + F · (x_i^r2 − x_i^r3);
  x_i^child = v_i
  i = (i + 1)%D;
  L + +;
}while(U([0, 1]) < CR&&L < D);
```

図 1　子個体の生成
Fig. 1　Procedures for generating an offspring in DE.

し，複数の最適解を同時に発見することはできない．

上下限制約をもつ問題では，DE の交叉の結果，子個体の j 次元の値 x_j^{child} が定義域の外部に生成される場合がある．このような制約違反を解消するために，文献 [9] の鏡像反射という方法が提案されている．これは，上下限を超えた場合に超えた分だけ内側に反射させるという方法である．例えば，各次元 j の定義域の上限を $range_j^{max}$ とし $x_j^{child} > range_j^{max}$ となった場合，式 (2) により値を変換する．これにより定義域の制約は満たされる．

$$\begin{aligned} x_j^{'child} &= range_j^{max} - (x_j^{child} - range_j^{max}) \\ &= 2range_j^{max} - x_j^{child} \end{aligned} \tag{2}$$

3. 種分化を導入した最適化手法

種分化とは，生物が進化する過程で生じた集団の分化，つまり，新たな種が形成される過程のことを指す．これによって，集団ごとに異なる様々な進化が可能になる．この種分化の概念を多峰性関数の最適化問題に応用する．つまり種分化の特徴として，探索空間を部分空間に分け，各部分で最良解を得るような探索を行い，複数解を同時に発見する．このような手法を導入したものが，種分化を導入した DE [12] である．

3.1 種分化を導入した DE (SDE)

なわばりを張る種分化では，関数値の優良個体を種の支配者とし，支配者を中心としてあらかじめ定められた半径 R のなわばり内に存在する個体を種とみなす．この方法は種分化として最も一般的な方法で，実際に，文献 [12]〜[15], [17] などで用いられている．SDE のアルゴリズムを以下に示す．

（1） ランダムに個体を生成し集団を構成
（2） 個体の関数値を計算
（3） 種分化
 (a) 個体群を関数値の良い順にソート
 (b) 関数値の良い順に個体を種の支配者とし，種を形成する
（4） 子個体の生成
（5） 置換え
（6） 終了条件を満たしていなければ，（3）に戻る

種分化は DE アルゴリズムの子個体の生成の直前に適用される．種分化手法と SDE について，以降で説明する．

3.1.1 なわばり半径をもつ種分化

種分化について具体例で説明する．図 2 の **A**, **B**, **C** のような個体を種の支配者とする．種の支配者は，個体集団の関数値の良い上位個体から，つまり最小化問題では関数値の小さい個体から決められる．図 2 は最小化問題に適用し，関数値が $f(\mathbf{A}) < f(\mathbf{B}) < f(\mathbf{C})$ となっている例である．既に存在する種の支配者のなわばり半径 R の中に入っていない関数値の良い個体が新たな種の支配者 $\boldsymbol{x^s}$ となる．

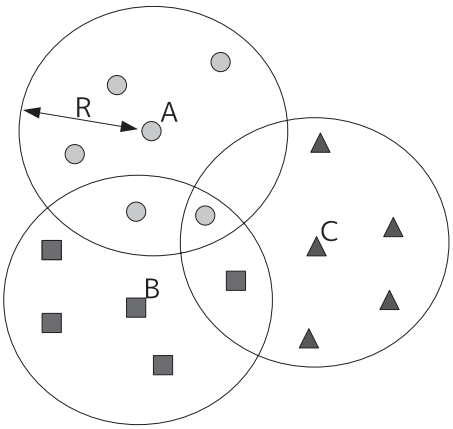

図 2 種となわばり
Fig. 2 Territories of respective species.

式 (3) になわばり判定条件を示す．個体 $\boldsymbol{x^i}$ と種の支配者 $\boldsymbol{x^s}$ とのユークリッド距離 d_{is} と，あらかじめ与えられているなわばり半径 R を用いてなわばりに含まれるかどうかを判定する．なわばり半径内の個体は $\boldsymbol{x^s}$ の種に属することになる．

$$d_{is} = \sqrt{\sum_{j=1}^{D}(x_j^i - x_j^s)^2} \leq R \tag{3}$$

ただし，複数のなわばりの交わり部分に属している個体は，属する種の支配者の関数値が最も良い支配者の種に属するものとし，他の種に重複して属することはない．**A**, **B** の 2 種類の種に属する個体は，より支配者の関数値が良い **A** の種に属することになる．

この方法は有効であるが，なわばり半径 R をどのようにして決定するべきかという問題がある．このため，予備実験によって関数の形状や最適解の分布状況を前もって調べ，関数ごとに最適ななわばり半径を決定する方法が多く用いられている．

3.1.2 SDE の設定

文献 [12] の SDE におけるなわばり半径 R は，$R = 0.5$ が使われており，式 (1) の変異ベクトル生成で使われる $\boldsymbol{x^{r_1}}$, $\boldsymbol{x^{r_2}}$, $\boldsymbol{x^{r_3}}$ は種の中からランダム

に選択される．この際，種に所属する最小個体数を維持するために種の最小個体数 $SMIN$ が定義されている．種の所属個体数が $SMIN$ 個に満たない場合は，$SMIN$ 個になるまで種のなわばり内に個体をランダムに増やしてから，三つの個体の選択が行われる．ただし，生成された個体は変異ベクトルの生成にのみ用いられ，次世代には残らない．また，置換えの際に子個体の関数値が所属する種の支配者の関数値と等しい場合は，ランダムに個体を生成して親個体と置き換えて種の中から類似した個体を省くという処理も行われている．

SDE の $\boldsymbol{x}^{r_1}$, $\boldsymbol{x}^{r_2}$, $\boldsymbol{x}^{r_3}$ の選択方法では種の中でのみ探索が進むので，複数の最適解や局所解をもつような関数では探索に時間がかかってしまう場合がある．

3.2 クラスタリングによる種分化

文献 [16] では，なわばり半径ではなくクラスタリングを用いることで種を形成するアルゴリズムが提案されている．

この方法は探索空間に分布している個体集団をユークリッド距離を用いたクラスタリングによって，二つの領域に再帰的に分割していくことで実現される．分割の停止条件として，生成されたクラスタが目的関数の形状をとらえているかどうかを調べる峰判定が用いられている．最小化問題において，あるクラスタ C_j に峰判定を適用したときの例を図 3 に示す．

図 3 の丸は種に属する個体を示し，クラスタ内の最良個体を x^* とする．$d_0, d_1, \ldots$ は x^* と最も近い個体との距離（ベース距離）をもとに計算された峰判定を行う判定区間の距離で，点線はそれぞれ判定区間の最小値を意味している．

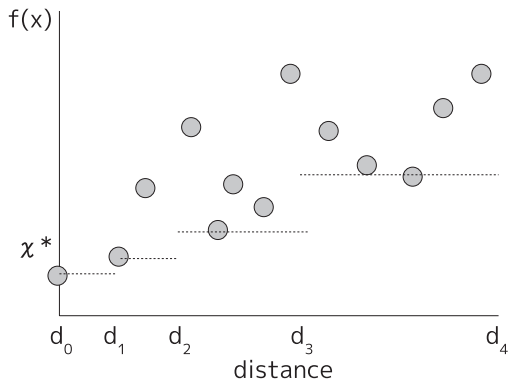

図 3 峰 判 定
Fig. 3 Judge of peaks of the function.

峰判定は，x^* を基準として行われる．x^* とクラスタ C_j に属する各個体とのユークリッド距離を計算し，個体を距離の昇順に並べ換える．そして峰判定を適用する判定区間 $d_0, d_1, \ldots$ を求め，区間内に存在する個体の関数最小値を記憶する．最後に，区間ごとの最小値を比較して，区間 $[d_i, d_{i+1}]$ の最小値が区間 $[d_{i-1}, d_i]$ の最小値より増加傾向であれば，クラスタ C_j は目的関数の形状をとらえているものとして，分割を停止する．そうでない場合は，まだ分割可能なクラスタであると判定され，更に領域分割が続けられる．こうすることで，関数の形状に沿うような形で様々な大きさの種を形成することができる．

4. 適応的なわばり半径をもつ種分化を導入した DE の提案

本研究は，従来の SDE より最適解への収束を早め，最適解の発見確率を向上し，関数の形状や最適解の分布状況に依存する種のなわばり半径などのパラメータを予備実験によらず最適化の過程で適応的に変化させ，最適なパラメータを自動で発見することを目的としている．このため，**3.1.1** の種分化をもとに，大域探索と局所探索を考慮した子個体の生成のための個体選択法，適応的なわばり半径決定法，エリート保存戦略の三つを提案し，DE に取り入れた．また本研究では，鏡像反射を用いた．

4.1 アルゴリズム

提案した三つの方法を取り入れたアルゴリズムを以下に示し，提案方法の部分を以降の節で説明する．

（1） ランダムに個体を初期化
（2） 個体の関数値を計算
（3） 種分化 (**4.3**)
  (a) 個体群を関数値の良い順にソート
  (b) 関数値の良い順に個体を種の支配者とし，種を形成する
  (c) 峰判定の実行
（4） 子個体の生成 (**4.2**)
（5） 置換え
（6） エリート保存 (**4.4**)
（7） 終了条件を満たしていなければ，（3）に戻る

4.2 子個体の生成のための個体選択方法

DE の子個体の生成は，**2.** に記述したとおり，親個体 $\boldsymbol{x}^{P}$ と $\boldsymbol{x}^{r_1}$, $\boldsymbol{x}^{r_2}$, $\boldsymbol{x}^{r_3}$ の 3 個体を用いて式 (1) で生成された変異ベクトルとを交叉させることで実現される．

本研究では最適解への収束を早めるため，種分化で得た種の情報を活用するよう，集団全体の最良個体を $\boldsymbol{x}^{\boldsymbol{r_1}}$ とする DE/best/1/exp とは異なり，所属している種の支配者を $\boldsymbol{x}^{\boldsymbol{r_1}}$ とする．また，種の外の情報も併せて大域的な探索も行うことができるよう，$\boldsymbol{x}^{\boldsymbol{r_2}}$, $\boldsymbol{x}^{\boldsymbol{r_3}}$ を全体からランダムに選択する．このように，DE と SDE の方法を組み合せた選択手法を提案する [17]．なお，$\boldsymbol{x}^{\boldsymbol{r_2}}$, $\boldsymbol{x}^{\boldsymbol{r_3}}$ は全体から選択するため，$SMIN$ は不要となる．

4.3 適応的なわばり半径決定法

本研究では，なわばり半径として最適な値を予備実験によって求めるのではない．まず，初期なわばり半径 R_{init} を求める．そして関数の形状をとらえ，種ごとに適応的ななわばり半径を求めるために **3.2** の峰判定を改良する．

4.3.1 初期なわばり半径

種の初期なわばり半径 R_{init} は，探索空間の大きさを考慮し，初期ステップで種が 10 個程度以上形成される値として式 (4) で求める．ここで，D は次元数，$range_j$ は各次元 j の定義域の幅である．S_{init} は整数値で，初期ステップでどれくらいの数の種を形成するのかを示すパラメータである．

$$R_{init} = \frac{1}{2}\sqrt[D]{\frac{\prod_{j=1}^{D} range_j}{S_{init}}} \tag{4}$$

4.3.2 峰 判 定

種ごとに最適ななわばり半径を求めるため，峰判定は初期なわばり半径を用いて仮の種を形成した後に適用される．最小化問題に対する峰判定の改良アルゴリズムを以下に示す．

（1） 支配者と所属個体間の距離を計算
（2） 距離の昇順に並べ換える
（3） ベース距離を求める

種の支配者に最も近い個体までの距離をベース距離とする．ベース距離は，峰判定の判定区間である比較間隔を求めるために使われる．ただし，種の支配者に最も近い個体までの距離が 0 だった場合は，種に属する個体の分布状況が狭くなっていると判定し，小規模な種でも大きな種と同様の判定ができるよう種の支配者に最も遠い個体までの距離の $1/BASE$ という小さな値をベース距離とする．$BASE$ は種が非常に小さい場合に有効なベース距離を求めるためのパラメータである．

（4） ベース距離の有効性をチェックする

この処理は，式 (5) で求めるしきい値 max_size とベース距離を比較する．max_size は，前世代の種の最大半径 R_{before} が 0 だった場合でもある程度の大きさを確保して複数個体で種を形成することができるよう，定数 R_{init}/DIV を用いている．DIV はベース距離の判定の際の最小距離を求めるためのパラメータである．

$$max\_size = R_{before} + \frac{R_{init}}{DIV} \tag{5}$$

このときベース距離 $> max\_size$ ならば，前世代の半径情報を有効に活用できず，種の形成に失敗したものとみなして，種の支配者以外の個体を種から除外する．この処理を行うことで，最適解同士が非常に近い距離にある場合，一つの種として混同するのを避けることができる．

（5） 比較間隔を求める

比較間隔は，ベース距離をもとに $d_{i+1} = 2 * d_i$ として種の支配者から最も遠い個体を含むところまで求める．

（6） 区間最小値を求める
（7） 単調増加判定を行う

各区間の最小値が増加傾向であるかどうかを判定し，増加傾向ではない区間以降に存在している個体を種から除外する．また，個体が存在しない区間が 3 以上連続していた場合，8 倍以上個体同士の距離に差があるため，種に属する個体同士が離れ過ぎていると判定して，それ以降の区間に存在する個体を種から除外する．

なお，（4）及び（7）で種から除外された個体は，いまだ種に所属していない個体であるので，**3.1.1** のアルゴリズムに沿って種分化における新たな種の支配者候補となり，すべての個体がいずれかの種に所属するまで種分化が続けられる．

峰判定を適用し，不適切な個体を種から除外することで，種の支配者から最も遠い個体までの距離が種のなわばり半径となる．こうして，種のなわばりは探索空間の種が存在する部分の関数の形状に適応するように定まる．

4.4 エリート保存戦略

4.2 及び **4.3** を実装した段階では，子個体との置換えによって優良な解付近を探索している個体が他の種へ移動し，種が消滅してしまうという事例が確認された．その結果，最適解の発見確率が悪くなりすべての最適解が発見できないことがあったので，優良な個体を次世代に残すエリート保存戦略を導入した．

しかしながら，本研究のエリート保存戦略は，従来

手法のエリート保存とは異なる．種分化を導入している本研究では，置換えによって消えてしまった種の支配者を仮エリート個体として記憶する．そして，仮エリート個体に対してのみ，エリート保存を行うかどうかを判定する処理を加える．仮エリート個体を次世代に残すかどうかの判定は，置き換える対象が個体集団中に存在しているかどうかで行う．その置換え対象となる個体は，以下の三つの処理で (a) (b) (c) の条件を最初に満足する個体である．仮エリート個体の置換え対象となった個体が重複することはない．

(a) 類似個体

仮エリート個体からの距離が前世代の種の最大半径 R_{before} 以内に存在する個体を類似個体とした．この個体の中で，関数値が最も悪い値を示すものと仮エリート個体を置き換えることで，より関数値が良い値を示す個体が次世代に残る．

(b) 所属個体が多い種の中で関数値が等しい個体

種の平均所属個体数（集団サイズ/種の総数）よりも所属個体数が多い種でのみ判定される．同じ種の中に等しい関数値をもつ個体が存在するということは，似ている個体が種の中に存在していると考えて，片方と仮エリート個体を置き換える．これによって種の中の類似個体が淘汰され，探索の方向性として優れている可能性のある仮エリート個体となった種の支配者を次世代に残し，探索の幅を広げることができる．

(c) 仮エリート個体より関数値が悪い個体

所属個体数などに依存せず，すべての種でチェックが行われる．種の中で関数値が最も悪い値を示す個体と関数値を比較し，仮エリート個体の方が優れている場合にのみ置換えを行う．こうして，淘汰すべき個体をより良い個体と置き換えることで，優良な個体を維持することができる．

5. 関数最適化実験

5.1 実験設定

SDE と比較実験を行う．文献 [12] と評価回数を合わせるため，個体数 $NP = 100$，ステップ数 $MAX = 500$ とし，SDE [12] のなわばり半径は，$R = 0.5$，種の最小個体数は $SMIN = 10$ とした．交叉率は $CR = 0.9$ で，スケーリングファクタ F は各ステップごとに，$0.4, 0.5, 0.6$ からランダムに選択される．予備実験より個体数が少ないと複数解をもつ関数での解の発見確率が下がり，個体数が多い，また，CR の値を下げると収束速度が遅くなったので，本研究ではこの値を用いた．また，本研究の手法で用いるパラメータはすべての関数で共通に用いることができるよう予備実験により $S_{init} = 10, BASE = 100, DIV = 10$ とした．試行回数を 50 回として実験を行った．

5.2 テスト関数

実験に使用したテスト関数を表 1 に示す．F1 は 2 次元で，18 個の最適解とその周辺に複数の局所解をもつ関数である．F2, F3, F4 は 2 次元の関数で，それぞれ 4, 2, 3 個の最適解をもつ．F5 は多次元の多峰性関数の最適化の結果を確認するために作成した関数で，各変数がすべて $(-1, \ldots, -1)$, $(0, \ldots, 0)$, $(1, \ldots, 1)$ となるときに最適解となる．本研究では，F5 を用いて 2, 3, 5, 10 次元でそれぞれ実験を行った．F6 は，4 次元の関数で一つの最適解をもつ関数で，最後の F7 は 2 次元で九つの最適解をもつ関数である．

5.3 評価方法

本論文では，精度，収束速度，発見個数の三つを評価し，それぞれ 50 回試行の平均と標準偏差を求めた．精度は，最終ステップで最適解と最適解に最も近い種の支配者との関数値の差の平均とし，発見個数の条件を満たすもののみで計算される．収束速度は，文献 [11] の収束速度の計算方法を参考にした．最適解から距離 10^{-1} 以内に存在する種の支配者で，最適解との関数値の差が 10^{-5} 未満となる状況を 10 ステップの間維持した場合の評価回数を求める．発見個数は，精度と同様に最終ステップで判定され，最適解から距離 10^{-3} に種の支配者が存在すれば，解を発見したものとする．

5.4 実験結果の評価

SDE と提案手法の実験結果を表 2 に示す．収束速度の括弧は，収束回数すなわち 50 回試行中何回収束したかを表し，“—” は 50 回試行で 1 度もすべての最適解に収束しなかったことを表している．なお，両手法を比較して優れている結果を太字で示した．

精度は，発見できた解の精度を計算するものなので同程度の結果が多かったが，関数 F1 において SDE より本研究の結果が優れていた．次に，収束回数では，SDE では 50 回すべての試行で完全には収束しない関数と F1, F5 の 10 次元，F7 など一度も収束しなかった関数があったが，提案手法ではすべて収束した．収束速度自体に関しても，ほとんどの関数で SDE より早くなった．発見個数を見ると，SDE では失敗している関数がいくつかあったが，提案手法ではすべての関数ですべての最適解を発見することができている．ま

表 1　テスト関数
Table 1　Test functions.

関数名	関数	定義域
Shubert (F1)	$\left(\sum_{i=1}^{5} i\cos[(i+1)x_1+i]\right)\times\left(\sum_{i=1}^{5} i\cos[(i+1)x_2+i]\right)$	$-10\le x_i\le 10$
Himmelblau (F2)	$-(200-({x_1}^2+x_2-11)^2-(x_1+{x_2}^2-7)^2)$	$-6\le x_i\le 6$
Six-Hump Camel Back (F3)	$4[(4-2.1{x_1}^2+{x_1}^4/3)\cdot {x_1}^2+x_1x_2+(-4+4{x_2}^2)\cdot {x_2}^2]$	$-1.9\le x_1\le 1.9$; $-1.1\le x_2\le 1.1$
Branin RCOS (F4)	$(x_2-\frac{5.1}{4\pi^2}\cdot x_1^2+\frac{5}{\pi}\cdot x_1-6)^2+10\cdot(1-\frac{1}{8\pi})\cdot\cos(x_1)+10$	$-5\le x_1\le 10$; $0\le x_2\le 15$
3-peaks function (F5)	$-\sum_{j=1}^{3} e^{-\frac{\sum_{i=1}^{D}(x_i-a_j)^2}{\sigma^2}}, ((a_1,a_2,a_3)=(-1,0,1),\sigma=0.3)$	$-2\le x_i\le 2$
Shekel function (F6)	$-\sum_{j=1}^{m}\left[\frac{1}{\sum_{i=1}^{4}(x_i-C_{ij})^2+\beta_j}\right]; m=5,7,10; (i=1,2,3,4)$ $\beta=[0.1,0.2,0.2,0.4,0.4,0.6,0.3,0.7,0.5,0.5]$ $C=\begin{bmatrix}4&1&8&6&3&2&5&8&6&7\\4&1&8&6&7&9&5&1&2&3.6\\4&1&8&6&3&2&3&8&6&7\\4&1&8&6&7&9&3&1&2&3.6\end{bmatrix}$	$0\le x_i\le 10$
Sinusoid function 1 (F7)	$\sum_{i=1}^{2}\sin(6.5x_i)$	$0\le x_i\le 3$

表 2　実 験 結 果
Table 2　An experiment result.

Function (dim)	精度	収束速度（収束回数）	発見個数
SDE の実行結果			
F1*(2)	6.96e-06±1.37e-05	—	15.38±1.31
F1(2)	2.09e-04±4.23e-04	—	1.84±1.17
F2(2)	3.51e-10±5.17e-26	**3471±641(50)**	4.00±0.00
F3(2)	5.30e-12±4.04e-27	**1759±118(50)**	2.00±0.00
F4(2)	4.15e-11±1.29e-26	4670±1325(50)	2.82±0.38
F5(2)	1.49e-10±1.03e-25	**1870±231(50)**	3.00±0.00
F5(3)	2.22e-15±2.37e-30	3555±140(50)	3.00±0.00
F5(5)	0.00e+00±0.00e+00	8288±773(40)	2.52±0.50
F5(10)	0.00e+00±0.00e+00	—	1.08±0.27
F6(4) m = 5	3.77e-09±4.14e-24	8038±589(50)	1.00±0.00
F6(4) m = 7	2.37e-09±2.90e-24	8683±1011(50)	1.00±0.00
F6(4) m = 10	2.90e-09±9.67e-10	8687±719(45)	0.90±0.30
F7(2)	3.39e-14±8.39e-14	—	7.58±0.49
本研究の実行結果			
F1(2)	4.24e-08±9.07e-10	**26789±1320**(50)	**18.00±0.00**
F2(2)	3.51e-10±5.17e-26	4943±97(50)	4.00±0.00
F3(2)	5.30e-12±4.04e-27	2259±237(50)	2.00±0.00
F4(2)	4.15e-11±1.29e-26	**3142±333(50)**	**3.00±0.00**
F5(2)	1.49e-10±1.03e-25	2290±112(50)	3.00±0.00
F5(3)	2.22e-15±2.37e-30	**3479±254(50)**	3.00±0.00
F5(5)	0.00e+00±0.00e+00	**5737±268(50)**	**3.00±0.00**
F5(10)	0.00e+00±0.00e+00	**13513±596**(50)	3.00±0.00
F6(4) m = 5	3.77e-09±4.14e-24	**7643±732(50)**	1.00±0.00
F6(4) m = 7	2.37e-09±2.90e-24	**7507±613(50)**	1.00±0.00
F6(4) m = 10	3.22e-09±1.65e-24	9603±1101(50)	**1.00±0.00**
F7(2)	1.83e-11±3.79e-11	**6060±409(50)**	**9.00±0.00**

た，表中の F1*(2) は文献 [12] で妥当な値とされていた個体数 300, ステップ数 1000, $R = 0.6$ における実験結果であるが，より厳しい評価方法を用いているため，すべての最適解を発見することはできなかった．

5.5 考　　察

5.5.1 関　数　F1

関数 F1 は 18 個の最適解を含めて 760 個の局所解をもつ複雑な関数で，他の関数とは違い最適解の分布状況が特殊である．計算によって求められた 18 個の最適解の最も近い最適解同士の距離が 0.88 で 2 個ずつのペアとなり，図 4 のように，定義域の中で均等に 9 箇所に配置されている．

図 5, 図 6 に SDE と本研究における最終ステップの種の形成状況を示した．四角が大域最適解で，丸は種のなわばりを示している．図 5 は SDE における $R = 0.5$ のときの結果である．一方，図 6 は ISDE においてなわばり半径が適応的に変化した結果，図 8 の最終ステップあたりにはなわばり半径が $R < 10^{-7}$ となっており複数の種が近距離に存在している．また，図 7 に種の総数の遷移を示す．最終的には 24 個程度の種に収束している．

SDE は，500 ステップ終了した時点でも 62 個の種の支配者が存在している．これは，種の中の情報のみを活用して探索を進めていくため探索の進行が遅く，また局所解に陥ったら抜け出すのが難しいためだと考えられる．その結果，ある程度最適解付近を探索していても，収束速度や発見個数の条件を満たすところまで探索が進まない．SDE は，最適解と局所解が多い複雑な関数に適用するには，個体数やステップ数を増やす必要があると考えられる．反対に提案手法では，100 ステップあたりですべての種の半径が小さくなり，最適解付近に探索が収束していた．大域的な情報と種の局所的な情報を利用することで局所解へ陥ることを避け，優良な方向に探索を進めることで収束を早めることができ，エリート保存によって種が維持されていると考えられる．

次に，SDE と提案手法の F1 における種のなわばり半径の遷移図を図 8 に示す．図の $R = 0.5$ は SDE

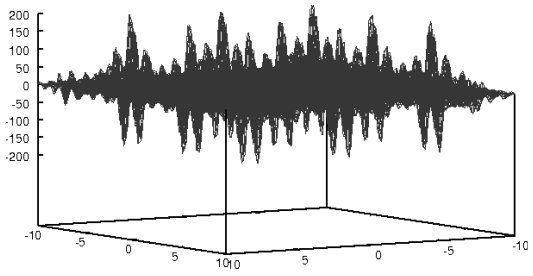

図 4　関数 F1 の 3 次元プロット
Fig. 4　3D plot of function on F1.

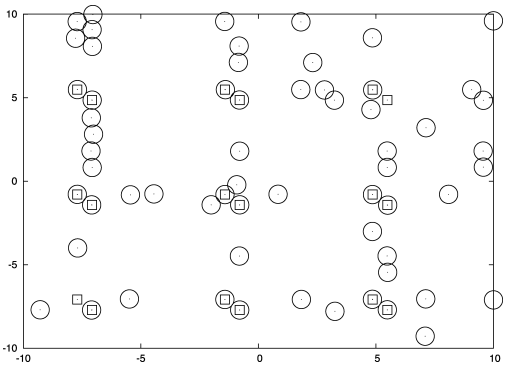

図 5　SDE における F1 の最終ステップ
Fig. 5　A snapshot in the last step of SDE on F1.

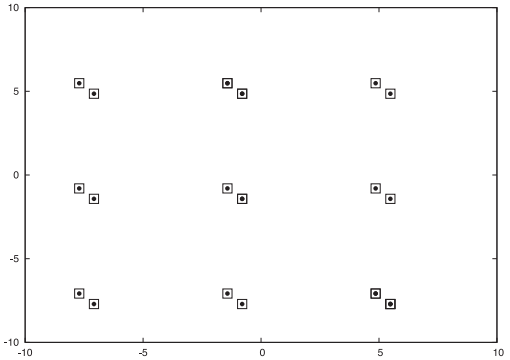

図 6　本研究における F1 の最終ステップ
Fig. 6　A snapshot in the last step of ISDE on F1.

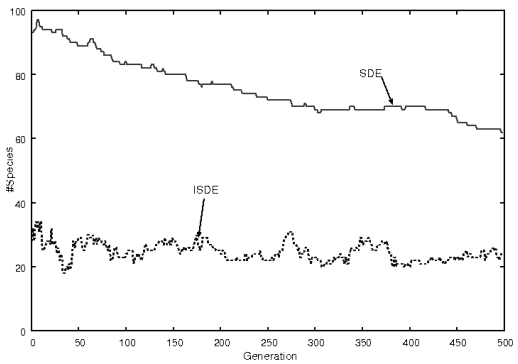

図 7　F1 の種の総数の遷移図
Fig. 7　The number of species on F1.

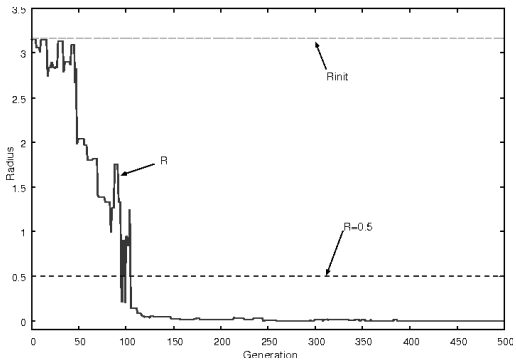

図 8　F1 の種のなわばり半径の遷移図
Fig. 8　Specie's territory radius on F1.

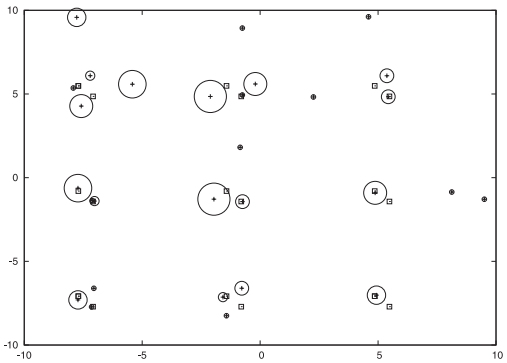

図 9　F1 の 25 ステップ目の種
Fig. 9　A snapshot in 25 step of ISDE on F1.

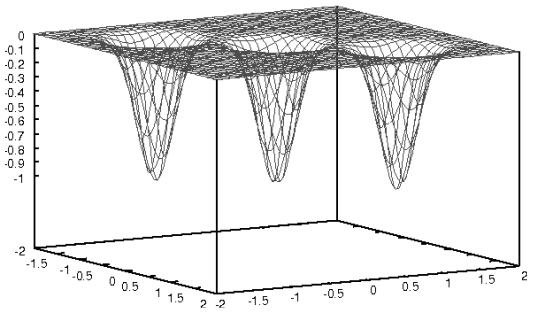

図 10　F5 の 3 次元プロット
Fig. 10　3D plot of function on F5.

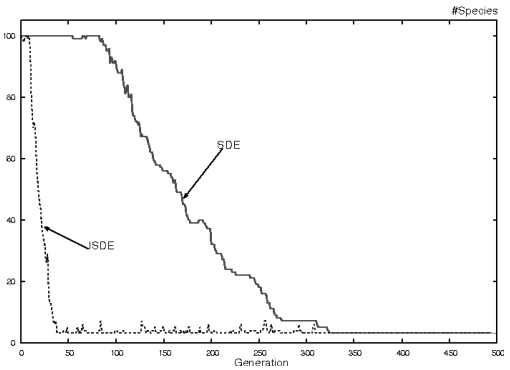

図 11　10 次元の F5 の種の総数の遷移図
Fig. 11　The number of species on F5 of 10D.

における種のなわばり半径で，R_{init} は本論文で用いた初期なわばり半径，R は峰判定を適用した後の各ステップごとの種の最大半径である．初期なわばり半径 R_{init} は最も近い最適解同士の距離 0.88 よりも大きくなっているが，種の最大半径 R は 100 ステップあたりまでに順調に減少しているということが分かる．これは，峰判定で判定区間を決めるベース距離が前世代の最大半径程度という制約をかけているため，探索がある程度収束しても複数の最適解を含んだ種が形成されないようになっているからだと考えられる．

図 6 と同じ試行の 25 ステップ目の種の形成状況を図 9 に示す．異なる大きさのなわばりをもつ種が形成されていることが分かる．

5. 5. 2　関　数　F5

2 次元の F5 は，三つの最適解をもつ関数で，図 10 のような形状をしている．

図 11 は，SDE と提案手法 (ISDE) における 10 次元の F5 関数の種の総数の遷移を示している．また，

表 3　10 次元の F5 の支配者情報
Table 3　Species-seeds on F5 of 10D.

アルゴリズム	$x_i = -1$	$x_i = 0$	$x_i = 1$
SDE	10(−0.996517)	76(−1.000000)	14(−0.999901)
本研究	34(−1.000000)	44(−1.000000)	22(−1.000000)

表 3 は，図 11 と同じ試行の SDE と本研究における種の支配者に属する個体の数と，括弧内に種の支配者の関数値を示している．

SDE では，種の数はゆっくりと収束していき，230 世代あたりで最適解周辺に収束していた．しかし，表 3 の種の支配者の関数値を見ると，$x_i = -1$ 若しくは $x_i = 1$ の場合で収束速度，発見個数ともに評価基準の判定条件を満たしていないことが分かる．SDE の探索の収束が遅いのは，2 次元の F5 とは違って 10 次元と探索空間が大きくなったにもかかわらず，同じなわばり半径 $R = 0.5$ を用いていたためだと考えられる．その結果，最適解周辺に種が形成されたとしても最適解に収束しきれず，終了ステップまでにすべての最適解を発見することができなかった．提案手法では，SDE

表 4 実験結果
Table 4 Experimental results.

Function (dim)	Algorithm	精度	収束速度（収束回数）	発見個数
F1(2)	SDE	2.09e-04±4.23e-04	—	1.84±1.17
	M1	4.16e-08±2.08e-09	40970±2779(46)	17.84±0.37
	M2	4.28e-08±9.44e-10	26567±1561(47)	17.84±0.37
	M3	4.24e-08±9.07e-10	**26789±1320**(50)	**18.00±0.00**
F5(10)	SDE	0.00e+00±0.00e+00	—	1.08±0.27
	M1	1.78e-17±4.98e-17	27702±1063(48)	2.96±0.20
	M2	7.18e-13±3.49e-12	30775±2893(50)	3.00±0.00
	M3	0.00e+00±0.00e+00	**13513±596**(50)	3.00±0.00

表 5 kPSO との比較表
Table 5 Comparison of kPSO.

Algorithm（個体数 ∗ ステップ数）	Shubert (F1)	Himmelblau (F2)	Camel back (F3)	Branin RCOS (F4)
kPSO (300*2000, 30*2000)	81194±45656	2259±539	1124±216	2084±440
kPSO (500*2000, 60*2000)	117503±77451	3713±570	2127±341	3688±717
本研究 (100*1000, 50*1000)	30580±4208	3016±219	1134±107	1852±134
本研究 (100*1000, 60*1000)	30580±4208	3345±175	1542±184	2198±301

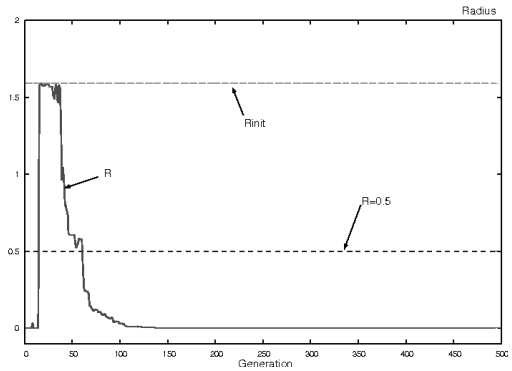

図 12 10 次元の F5 の種のなわばり半径の遷移図
Fig. 12 Specie's territory radius on F5 of 10D.

よりも早く最適解付近に収束し，すべての最適解を発見しているということが分かる．これは，探索空間を加味した初期なわばり半径の決定法を用いているため，初期ステップではある程度粗い探索を行うことができ，他の種の情報も活用できる大域探索的要素によるものだと考えられる．そのため，SDE より提案手法の方が収束が早く，発見個数も安定していると考えられる．

図 12 に SDE と提案手法の 10 次元の F5 における種のなわばり半径の遷移図を示す．図 11 の種の総数の減少にほぼ一致する形で，なわばり半径 R が遷移していることが分かる．また，初期のなわばり半径 R_{init} が小さいため初期ステップではすべての個体が種を形成しているが，大域探索も考慮することで個々の探索の進行を経て複数個体で種を形成しなわばり半径 R が広がり，早期に最適解に収束できるのだと考えられる．

5.6 提案手法の検証

表 4 に F1, F5 の 10 次元で各々SDE に子個体生成のための個体選択方法である r1, r2, r3 の選択方法 (M1)，適応的種分化 (M2)，エリート保存 (M3) を順に付け加えた実験結果を示す．

SDE に r1, r2, r3 の選択方法を付け加える (M1) ことで，SDE より最適解の発見確率が上がったが，すべての最適解を発見することはできなかった．M1 に適応的種分化を導入する (M2) と，F1 の収束速度が早くなり，F5 ではすべての最適解を発見することができた．M2 にエリート保存を付け加える (M3) と，F1 ですべての最適解を発見し，両関数とも収束速度が早くなった．この結果より，三つの方法を組み合わせることで，すべての試行で最適解を発見することができるようになったと考えられる．

5.7 他手法との比較

種分化を導入した DE と同様に，複数の最適解を同時に発見する方法として，kPSO [11] が提案されている．kPSO は，c ステップごとに複数の k の値で k-means 法 [18] によりクラスタリングを行い，その中で最適な k で求められたクラスタの情報を利用する PSO である．異なったアプローチによる最適化結果を比較するため，kPSO を提案した文献 [11] と本研究の収束速度の結果を表 5 に示す．kPSO では，F1 は個体数 300 あるいは 500，F2, F3, F4 では個体数 30 あるいは 60 でステップ数 2000 で実験を行っていた．本研究では，F1 は個体数 100，F2, F3, F4 では個体数 30 では個体が少なすぎて探索がうまく進まなかった

ため，個体数 50 あるいは 60 で，ステップ数は 1000 で実験を行った．kPSO (300∗2000, 30∗2000) は，それぞれ kPSO の（F1 の個体数とステップ数，F2, F3, F4 の個体数とステップ数）を示している．

kPSO (300∗2000, 30∗2000) と本研究 (100∗1000, 50∗1000) の結果を比較すると，F1 と F4 では本研究の方が早く収束し，F3 はあまり差がなく，F2 では kPSO の方が早くなった．F2 で本研究より kPSO の方が収束が早い要因は，F2 のような平たんな関数では，動的に峰判定で半径を調節する本研究よりクラスタリングを使った方法の方が最適解周辺の形状をとらえやすいためだと考えられる．逆に F1 のように複雑で局所解を多くもつような関数では，kPSO の半分以下で個体が最適解付近に収束した．本研究のように，局所探索と大域探索の要素を併せ持つような方法の方が局所解に陥りにくくなり，収束が早くなるのだと考えられる．

6. むすび

本研究は SDE を更に発展させて，問題ごとに重要なパラメータ R を調節することなく複数解をもつ問題の最適解を安定的に求める手法を考慮した．そのため，大域探索と局所探索を考慮した子個体生成のための個体の選択法，初期なわばり半径を利用した関数の峰の形状を利用する適応的なわばり半径決定法，次世代へ優良個体を残すエリート保存戦略の三つの方法を提案した．その結果，SDE のようになわばり半径を問題ごとに与えることなく探索の状況に応じた種のなわばり半径を活用し，共通のパラメータで多くのテスト関数で SDE より収束速度，発見個数ともに安定した結果が得られることを示した．今後は，F2 のような平たんな関数でも収束速度を早めるため，大域探索と局所探索を使い分ける方法を考えることが必要であると考えられる．またより多くのテスト関数に適用し，様々な形状で実験を行い，多くの関数の形状に適した種分化手法を考慮する必要もあると考えられる．

文　献

[1] J. Holland, Adaptation in natural and artificial systems, The University of Michigan Press, Ann Arbor, Michigan, 1975.

[2] D. Goldberg, Genetic Algorithms in search, Optimization, and Machine Learning, Addison Welsey Longman, Boston, 1989.

[3] J. Kennedy and R. Eberhart, “Particle swarm optimization,” Proc. IEEE International Conference on Evolutionaly Programming, pp.601–610, San Diego, Calif, USA, March 1998.

[4] R. Storn and K. Price, “Differential Evolution — A simple and efficient adaptive scheme for global optimization over continuous spaces,” Technical Report TR-95-012, International Computer Science Institute, Berkeley, 1995.

[5] 奥村浩士，坂本光男，木嶋　昭，“単体分割による非線形回路方程式の複数解の算出,” 信学論（A），vol.J70-A, no.3, pp.581–584, March 1987.

[6] S. Kurahashi and T. Terano, “A genetic algorithm with tabu search for multimodal and multiobjective function optimization,” Proc. 2nd Genetic and Evolutionary Computation Conf., pp.291–298, Las Vegas, Nevada, July 2000.

[7] J. Vesterstrom and R. Thomsen, “A comparative study of differential evolution, particle swarm optimization, and evolutionary algorithms on numerical benchmark problems,” Proc. 2004 IEEE Congress on Evolutionary Computation, vol.2, pp.1980–1987, Portland, Oregon, June 2004.

[8] J. Riget and J. Vesterstroem, “A diversity-guided particle swarm optimizer — The ARPSO,” Technical Report 2002-02, Department of Computer Science, University of Aarhus, Denmark, 2002.

[9] 島　孝司，“すみ分け型エボルーショナリー・アルゴリズムによる大域最適化,” システム制御情報学会論文誌，vol.8, no.2, pp.94–96, Feb. 1995.

[10] 島　孝司，“すみ分け型遺伝的アルゴリズムによる大域最適化,” システム制御情報学会論文誌，vol.8, no.5, pp.233–235, May 1995.

[11] A. Passaro and A. Starita, “Particle swarm optimization for multimodal functions: A clustering approach,” J. Artificial Evolution and Applications, vol.2008, Article ID 482032, 2008.

[12] X. Li, “Efficient differential evolution using speciation for multimodal function optimization,” Proc. 2005 Conference on Genetic and Evolutionary Computation, pp.873–880, Washington DC, June 2005.

[13] J.-P. Li, M.E. Balazs, G.T. Parks, and P.J. Clarkson, “A species conserving genetic algorithm for multimodal function optimization,” Evolutionary Computation, vol.10, no.3, pp.207–234, 2002.

[14] X. Li, “Adaptively choosing neighbourhood bests using species in a particle swarm optimizer for multimodal function optimization,” Proc. 2004 Conference on Genetic and Evolutionary Computation, pp.105–116, Seattle, Washington, June 2004.

[15] 岩松雅夫，“種分化するパーティクルスウォームによる多峰性関数の大域最適化,” 信学論（D-II），vol.J87-D-II, no.2, pp.771–774, Feb. 2004.

[16] A. Petrowski and M.G. Genet, “A classification tree for speciation,” Proc. 1999 Congress on Evolutionary Computation, vol.1, pp.204–211, Washington, DC, July 1999.

[17] M. Shibasaka, A. Hara, T. Ichimura, and T. Takahama, “Species-based differential evolution with switching search strategies for multimodal function optimization,” Proc. 2007 IEEE Congress on Evolutionary Computation, pp.1183–1190, Singapore, Sept. 2007.
[18] R.O. Duda and P.E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons, New York, 1973.

（平成 20 年 12 月 1 日受付，21 年 3 月 9 日再受付）

柴坂美祐喜

2007 広島市大・情報科学・知能情報システム卒．2009 同大大学院情報科学研究科博士前期課程了．現在，シャープビジネスコンピュータソフトウェア（株）に勤務．進化的計算と最適化アルゴリズムに興味をもつ．

原　　章　（正員）

1997 東工大・工・電気・電子卒，2002 同大大学院総合理工学研究科博士課程了．2002 広島市立大学情報科学部知能情報システム工学科助手．2007 より同大講師．生物の適応・学習のメカニズムに興味をもち，進化計算論や群知能，マルチエージェントシステムに関する研究に従事．情報処理学会，日本知能情報ファジィ学会，IEEE 各会員．博士（工学）．

市村　　匠　（正員）

1997 桐蔭横浜大学大学院工学研究科博士後期課程了．博士（工学）．同年広島市立大学情報科学部助手．2007 同大学院講師，現在に至る．ソフトコンピューティングの研究に従事．著書「実践 Linux」など．IEEE 等会員．

高濱　徹行　（正員）

1982 京大・工・電気第二卒．1987 同大大学院博士課程研究認定退学．同年福井大学工学部助手．1994 同大講師．1998 広島市立大学情報科学部知能情報システム工学科助教授．2005 より同大教授．ナチュラル・コンピューティング，進化的計算，最適化アルゴリズム，機械学習などに関する研究に従事．情報処理学会，人工知能学会，教育システム情報学会，言語処理学会，IEEE 各会員．工博．