3点および4点比較ベースの対話型差分進化と差分進化

裴, 岩
九州大学大学院芸術工学府

高木, 英行
九州大学大学院芸術工学研究院

出版情報：第3回進化計算学会研究会, pp.74-84, 2012-09. 進化計算学会
バージョン：
権利関係：

3点および4点比較ベースの対話型差分進化と差分進化

裴　岩[†]　　高木英行[††]

九州大学大学院芸術工学府[†]，九州大学大学院芸術工学研究院[††]

1　はじめに

対話型進化計算（IEC）は人間の知識，経験，好み等を最適化過程に組み込むアプローチで，人間の評価系に基づいて解探索を行う．人間そのものを最適化系に組み込むことで，評価系構築が困難，あるいは評価計測が困難な多くのタスクにも適用できるようになる．例えば補聴器フィッティング[29]や人工内耳フィッティング[11]では，音の聴こえの機器計測ができずユーザの主観評価が唯一の指標となるため，IECに適したタスクと言える．IECは，音楽やグラフィックス生成のようなアート応用，音や画像の信号処理，制御・ロボット，人工現実感，データマイニング，メディアデータベース検索，などの工学応用，地質学応用，教育，ゲームなどその他の分野の多数のタスクに応用されてきた[27]．

枠組み的には，fitness関数を人間の評価に置き換えればどの進化計算（EC）アルゴリズムでもIECを構成できる．対話型遺伝的アルゴリズム（IGA）[4]，対話型遺伝的プログラミング[23]，対話型進化的戦略[8]，人間ベース遺伝的アルゴリズム[10]，対話型particle swarm optimization [14]，本論文で扱う対話型差分進化（IDE）[26]など各種EC技術がIECに用いられている．しかし人間の評価にノイズ混入は避けられず，評価値ノイズに敏感なECアルゴリズムをそのままIECにしても性能が出ないため，実用的な観点からの対策が必要である[13]．

IEC研究には，応用の拡大，IEC枠組みの拡大[30]，人間をリバースエンジニアリング的に解析する人間科学のための展開[31]，探索の高速化，IECインタフェースの改善など，色々な取り組みがあるが，主要な残された課題の1つがIECユーザの疲労問題である[27]．疲労軽減を図る取組としては，これら探索の各種高速化とIECインタフェースの改善も疲労軽減の研究の一環であるし，評価値入力方法の改善，IECとECの組合せ，ユーザの探索への関与，IECユーザの評価モデルの導入，新しいIEC枠組みの構築，多数個体の同時比較に代わる対比較の導入などが取り組まれてきている．

本論文は高速化によるIECユーザに疲労軽減を図る研究の1つである．高速化方法には，勾配法，山登り法，その他のメタヒューリスティックな手法との組合せ法など[16]や，探索景観の近似[34]がある．探索景観の近似も，探索景観を単峰性関数で近似して最適解近傍の推定をしたり[28]，その近似を低次元関数で簡略化して高速化を図ったり[17, 15]，Fourier変換を利用した景観近似と最適解近傍の推定[18]，など色々な取り組みが行われてきた．

本論文の目的は，opposition-based learning（OBL）を差分進化（DE）に組み込んだIEC高速化手法を提案し評価することにある．第2節で後述するように，OBLをDEに組込む手法は初期値設定と世代交代の2つの段階で使われている．これに対し第3節では，新しいOBLのIDE組込み法を提案する．提案法は，従来の対比較ベースのIDEが2個体比較を繰り返すのに対し，3個体比較，あるいは，4個体比較を行うため1回当たりのIDEユーザの評価疲労は増える．しかしこのユーザは3個体のような比較的個体記憶が可能な範囲では大きく判断負荷が増えないであろうとの仮説と，収束が高速化できればトータルとしてユーザの疲労軽減につながるとの考えに基づく提案である．第4節ではIDEのシミュレーション実験を通して提案法の評価を行う．また本手法はIDEに特化したものでもなくもちろんDEにも利用できる．そこで第5節で24のベンチマーク関数を用いての評価も行う．

Triple and Quadruple Comparison-Based Interactive Differential Evolution and Differential Evolution

† Yan PEI(peiyan@ieee.org)

†† Hideyuki TAKAGI (http://www.design.kyushu-u.ac.jp/~takagi/)

Graduate School of Design, Kyushu University (†)

Faculty of Design, Kyushu University (††)

2　従来の技術

2.1　差分進化

DEは個体ベース探索法の 1 つである[24, 19]．個体群の分布サイズの間接的情報を持つ個体間の差分情報を使ってベースとなる個体周辺を探索することで一親個体に該当する一子個体を生成する方法である．子個体が当該親個体を上回る場合のみ親子個体を入れ替えることでエリート戦略あるいは山登り法に似た特性も持つ．差分進化の特長はその比較的簡単な演算と強力な探索性能にある．

Fig. 1の左側が個体群，右側の等高線図が探索景観，探索景観上の○印が個体としよう．DEアルゴリズムは次のように表される．

(1) 1 個体をtarget vectorとして取り出す.

(2) 残りの個体から2 個体をparameter vectors としてランダムに選択し両者の差分vector を得る．

(3) 残りの個体からランダムに，あるいは，最良個体を選び，base vectorとする．

(4) base vectorに重み付けした差分vector を加え，mutant vector を求める．

(5) target vector と mutant vector を交差させて trial vectorを生成する．

(6) target vectorとtrial vectorを比べ，良い方を次世代の個体とする．

(7) 以下(1)に戻って他の個体についても同様に次世代個体を生成する．

上記(1)〜(4)を1行にまとめて，mutant vectorを
base vector + F×(parameter vector2 - parameter vector1)
と書けるので，DEアルゴリズムがいかに容易に実現できるかが理解できよう．ここでFはscale factorと呼ばれる．上記(1)の差分vectorの個数，上記(3)のbase vectorの選択方法，上記(5)の交差方法で色々なDEバリエーションがある．

探索が進むにつれて個体分布は狭くなり，その狭くなる個体分布にしたがって差分vectorも平均的に短くなるため，自動的にexplorationとexplotationのバランスを取ることができるアルゴリズムである．ランダムに選択した個体間の差分vectorは向きも長さもランダムであり，上記(4)から，大まかにはbase vectorの周辺を個体分布（探索の収束）にしたがって狭めながら探索するアルゴリズムと見ることができる．さらに詳細に言えば，上記(5)からその点を各target vector側にバイアスをかけて探索するアルゴリズムであると言える．

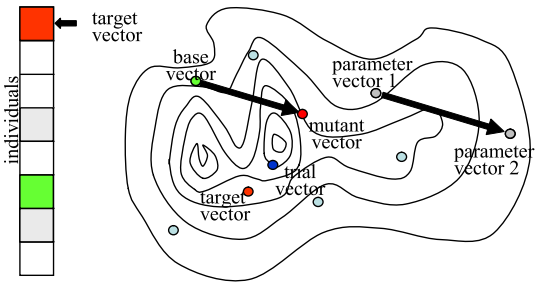

Fig. 1　差分進化のアルゴリズム

探索性能向上のために，例えば，SaDE [20]，JADE [21]，jDE [3]，JASaDE [5] 等，色々な改良DEが提案されている．これらのDEは，DEパラメータ値の確率的な設定，過去の探索更新履歴からより良い探索戦略の切り替え，新しいDE演算法の組込み，これらの戦略の組合せ，など独自の探索戦略を採用している．探索では，パラメータ調整，探索戦略の設定法（戦略プール構造），戦略選択法，の3つについて考慮する必要がある．

2.2　対比較の対話型差分進化

IGAに代表される多くのIECでは，全個体をIECユーザに提示し，ユーザに個体評価を求める．静止画像のように空間比較できる場合では全個体を比較することができるため，全個体を同時に評価することも可能であり，多くのIEC応用はこの提示・評価方法を採用している．

しかし，音や動画を評価する場合，全個体比較をして評価する方法は記憶判断の負荷が大きい．人間の脳情報処理能力にはいろいろな限界があり，5〜9以上の異なる情報を同時に処理することができないことが指摘されている[6]．通常のIECの個体数はこの記憶の限界を超えることが多く，10〜20個体の音や動画をIECユーザに提示する方法には限界がある．

この解決のために提案された方法が対比較に基づくIECであり，全個体比較を対比較にすることによってIECユーザ疲労軽減効果が期待できる．最初に考えられた方法は，トーナメントGA [9, 12] である．N個体に対して，$N-1$回の対比較を繰り返し，勝ち抜き数や評価差を反映させたfitnessを各個体に与える．この方法の欠点は，

本来全個体比較をして決定したfitnessに基づいて選択演算を行うことが本来のアルゴリズムであるにもかかわらず，トーナメントは全個体比較をしない簡略比較であるためfitnessにノイズを含む点である．このノイズは選択演算時に影響し，探索性能劣化につながる．

これに対しDEでは，上述アルゴリズム(5)でtarget vectorとtrial vectorを対比較する．この点に着目した対比較ベースIDE [26] では，DEアルゴリズムに変更を加えずにIECユーザに対比較を提供するため，ユーザ疲労軽減には有望な手法と考えられている．本論文は，この対比較ベースIDEに次節のOBLを組み込んだ新しい収束加速を狙う．

2.3 Opposition-based Learning

opposition-based learning (OBL) [32] は機械学習 [33] や最適化の高速化（OBL最適化）に利用される．$x \in [a,b]$を実数値とすると，$[a,b]$の中心を鏡像点とするxの対称点$OP(x) = a + b - x$が得られる．これを多次元に拡張すると，n次元実数空間上の1点$X = (x_1, x_2, ..., x_n)$（$x_i \in [a_i, b_i], i = 1, 2, ..., n; a_i, b_i \in R$）の対称点$OP(X)$が式(1)と式(2)で定義される．

$$OP(X) = \{OP(x_1), OP(x_2), ..., OP(x_n)\} \quad (1)$$

$$OP(x_i) = a_i + b_i - x_i \quad (2)$$

この対称点を進化計算の加速に利用する方法がOBL最適化であり，初期個体生成と各世代での子個体生成の2種類の高速化手法が広く使われている．前者は，進化計算の初期個体を乱数で生成後，各初期個体の対称点と比較して良い個体を第1世代の個体とする方法である．特に個体数が少ない場合，乱数初期個体の分布は偏りが大きくなりがちで局所最適解に陥る危険があるが，OBLによる初期化はこの危険を軽減し，探索空間を大局的に見て良い部分空間から探索を始めやすくする働きがある．後者は一定確率（ジャンプ率と呼ばれる）以下で，通常の進化的演算で子個体を生成する代わりに，親個体の対称点を生成してより良い方を次世代子個体とする方法である．

最適化に対称点を導入する考えは，多くの最適化アルゴリズムが確率的探索に基づいており，現在の探索点が大局的最適解から離れているのであれば探索点の対称点の方が大局的最適解に近づくという仮説と，現在の探索点もその対称点も相手より良い確率が50%である，という考えに基づいている．

OBLは各種ECに適用され，通常DEに前述の2つの技術を組み込んだ対称点ベース差分進化（OBDE）も提案されている．第1の組込み技術は対称点ベースの個体初期化であり，第2の組込み技術はジャンピング率に基づく対称点ベースの個体選択法である [22]．

その後，OBDEのshuffled DEへの適用 [1]，対称探索空間と4つの対称点生成法を導入して拡張した汎用OBDEの提案 [35]，対称演算子をmutation vectorに適用し勝者個体を残す新しいDEを提案 [7]，など色々なOBDEのバリエーションが提案されてきた．

本論文では，特に対話型差分進化でのユーザ疲労を大きく増加させることなく高速化を図るための新たなOBLの導入法を提案する．

3 新しいOBDE法の提案

本論文で提案するOBDE法と比較のための従来法をTable 1に示す．第2.3節の第1の技術（OBLを用いた初期値決定法）は本論文で比較するすべての手法に適用できるため比較評価対象とはせず，評価実験の従来法2では前節の第2の技術（ジャンピング率に基づく対称点ベースの個体選択法）のみを比較対象とする．

Table 1　2点比較ベースの従来法と3点，4点比較ベースの提案法．target-OBとtrial-OBはtarget vectorとtrial vectorの対称点vectorで，対象点を求める鏡像点は個体分布の中心点（Tune）と探索空間全体の中心点（Whole）との2種類を用いる．

従来法1	通常DE [19] ／対比較ベースIDE [26]（target, trialの2点比較）
従来法2	ジャンプ率に応じてtarget, target-OBの2点比較と通常DE/対比較ベースIDEの切替 [22]
提案法1	target, trial, target-OBの3点比較
提案法2	target, trial, trial-OBの3点比較
提案法3	target, trial, target-OB, trial-OBの4点比較

進化計算の高速化では，(a) fitness計算負荷，(b) 高速化手法の処理計算負荷，(c) 収束速度の兼ね合い，を考慮する必要がある．fitness計算負荷が大きなタスクでは，高速化のための処理に

時間がかかろうとも収束が早くなった方が有利になるため，目標値に達するまでのトータルの計算時間で評価されることが多い．

一方，IECではIECユーザの疲労問題が大きな評価要素になる．1回の評価負荷が同じ場合は，IECユーザの疲労は目標値に達するまでのトータルの計算時間と正の比例関係があると言えるが，手法が異なるがために1回の評価負荷が異なる場合，必ずしもこの関係が成り立たない．評価が容易なのでトータルの評価時間が長くなっても疲労負荷が少ない場合もあれば，1回の評価負荷が増えても収束が早くなって全体としては探索が楽になる場合もある．したがって1回の評価負荷と収束の2要因についてシミュレーションで解析し，その後人間の被験者を使った確認実験を行う必要がある．本論文は，前段階としてのシミュレーション解析を扱う．

提案法は，target vector, trial vector 以外に各々の対称点も毎回探索に利用しようとするものである．提案法1は，ジャンプ率に応じて切り替えている従来法2を同時に実施するので，従来法2を機能的に包含していると言える．同様に，提案法3は提案法1と提案法2を機能的に包含していると言える．

対話型でないDEへ提案法を応用する場合は1回の比較当たりのfitness計算が従来法の2回が3回あるいは4回になるため，上記(a)の計算負荷は1.5倍もしくは2倍になる．本手法の評価は，それでも収束が早まってトータルの計算時間が短くなるかどうかである．

一方，IDEの場合，IDEユーザの疲労はこの個体比較回数に比例しない．空間比較が可能な静止画の場合，2枚の画像から良い方を選ぶ疲労と3〜4枚の中の1枚を選ぶ疲労を比較すると後者の方が精神的負荷は多くなるが，1.5倍または2倍になるわけではない．空間比較ができない音や動画提示タスクの場合負荷比率は上がると思われるものの，それでも1.5倍または2倍になるとは限らない．一般的に言えば，提示個体が記憶できる程度の少ない個体数比較の場合は比較的疲労増加が少なく，その比較個数を超えると急激に精神的負荷が増える．本論文の提案法の3個体比較，4個体比較はこの点に着目し，比較時の疲労増加を犠牲にしても収束を高速化することでトータルの疲労増加を少なくできれば有効な高速化手法となる，との考えに基づいた提案である．

文献[22]（従来法2）では個体分布の中心を鏡像点としている．この方法は，収束していくにつれて対称点も含めた個体分布も小さくなっていくDEには適した方法のように思われる．しかし，探索空間内で大きな個体分布移動が収束の加速に効果がある特に若い探索世代では，探索空間全体を見渡して対称点を求める方法が効果を発揮するかもしれない．本論文では，この効果も調べるため，提案法1〜3では，動的に変化する個体分布の中心を鏡像点として対称点を求める手法（以後Tuneと記述）と探索空間全体の固定中心鏡像点として対称点を求める手法（以後Wholeと記述）も比較評価する．

4 IDEでの評価実験と解析

4.1 IDEユーザモデルと実験条件

IECタスクは難易度の高いベンチマーク関数や騙し問題のような極端な探索景観を持つわけではない．IECユーザは弁別閾以下の細かな違いは評価上区別ができないことと，それでも実用的な解が得られることから考えて，探索景観は比較的単純であることが期待できる．

混合ガウス関数モデルはこのIECの探索景観を大局的には大谷構造の単純な多峰性特性である，と仮定して実現するモデルで[13]，異なる平均，分散，ピーク高のガウス関数を複数組み合わせることで多次元の大谷構造の単純な多峰性特性を実現する．

本節ではこの混合ガウス関数モデルを評価に用いる．具体的には，4つのガウス関数（$k=4$）から成る3, 5, 7, 10次元の関数を用意し，式(3)で表す特性を用いる．Fig. 2は4次元混合ガウス関数モデルの3次元表示例である．

$$GMM(\mathbf{x}) = \sum_{i=0}^{k} a_i \exp(-\sum_{j=0}^{n} \frac{(x_{ij}-\mu_{ij})^2}{2\sigma_{ij}^2}) \qquad (3)$$

ここで，

$$\sigma = \begin{pmatrix} 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 & 1.5 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 \end{pmatrix}$$

$$\mu = \begin{pmatrix} -1 & 1.5 & -2 & 2.5 & -1 & 1.5 & -2 & 2.5 & -1 & 1.5 \\ 0 & -2 & 3 & 1 & 0 & -2 & 3 & 1 & 0 & -2 \\ -2.5 & -2 & 1.5 & 3.5 & -2.5 & -2 & 1.5 & 3.5 & -2.5 & -2 \\ -2 & 1 & -1 & 3 & -2 & 1 & -1 & 3 & -2 & 1 \end{pmatrix}$$

$$a_i = \begin{pmatrix} 3.1, & 3.4, & 4.1, & 3.0 \end{pmatrix}^T$$

通常のベンチマーク関数と異なり，IECユーザのシミュレーションを行うにはちょっとした工夫

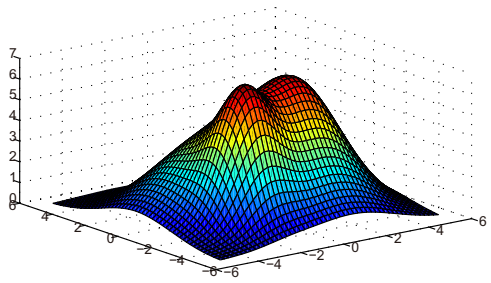

Fig. 2　4次元ガウス混合関数の3次元景観

が必要である．IECユーザは個体間の差がわずかである時には区別が困難であるので，この点をシミュレーションに組み込むことが通常の評価関数と異なる点である．本節のIDEユーザのシミュレーションでは，trail vectorとtarget vectorの比較時に，これらのfitnessの差が一定値以下であるならば（本論文での実験ではこの閾値を個体群の最大fitnessと最小fitnessの差の1/50とする），疑似IDEユーザは区別できないものとして，ランダムにtrail vectorかtarget vectorを選択して次世代に渡す．

評価実験は各次元の混合ガウス関数の1,000世代までの実験を30試行行い，最初の20世代まで各世代に符号検定で提案法の有無の効果を評価し，20世代目と1,000世代目ではWilcoxonの符号検定を適用して両者の収束性能の有意差を確認する．

IDEのパラメータを以下のように設定する．個体数20は実際の人間のIECユーザが実験を行う場合を想定した値である．

個体数	20
パラメータ探索範囲	$[-5.12, 5.12]$
scale factor F	0.3
交差率	0.7
DE演算法	DE/best/1/bin
最大探索世代 MAX_{NFC}	1,000
混合ガウス関数の次元数 D	3,5,7,10
試行数	30

4.2　提案法の評価実験

通常DEと提案法との比較のいくつかをFig. 3に示す．これらの収束曲線は，世代数比較であればいかに提案法が通常IDEよりも収束が速いかを示している．

Table 2(a)は実用的な条件（20個体で20世代目）での収束比較結果である．3次元混合ガウス関数での収束実験では，提案法の有意な効果はわずかに提案法2が従来法1（通常IDE）に対して見られるだけであり，他の手法はほぼ従来法と同等の効果であった．それに対し，次数が上がりタスクの難易度が上がると提案手法の優位性が顕著になった．この提案法の優位性は，従来法1（通常IDE）に対しても従来法2（従来のIDEへのOBL適用）に対しても同様である．

この理由はFig. 3から容易に想像できる．すなわち．難易度の低い3次元混合ガウス関数では従

Table 2　3, 5, 7, 10次元混合ガウス関数（疑似IDEユーザ）用いた時の平均fitness値．従来法2はジャンプ率37%にした対称点ベース世代ジャンプ法[22]，従来法と提案法の詳細，および，TuneとWholeはTable 1を参照．太字と† は各々，Wilcoxonの符号検定で提案法が従来法1（通常IDE），および，従来法2よりも危険率5%で有意に良いことを示す．

(a) 20世代目の平均fitness

手法	3D	5D	7D	10D
従来法1	-5.71	-3.42	-2.98	-2.40
従来法2	-5.70	-3.46	-3.03	-2.48
IDE-提案法1-Tune	-5.72	**-3.78**†	**-3.22**†	**-2.98**†
IDE-提案法2-Tune	-5.67	-3.62	-3.10	-2.54
IDE-提案法3-Tune	-5.72	**-3.73**†	**-3.25**†	**-3.14**†
IDE-提案法1-Whole	-5.68	-3.56	**-3.13**†	**-2.66**†
IDE-提案法2-Whole	-5.70	**-3.59**†	**-3.14**†	**-2.91**†
IDE-提案法3-Whole	-5.66	**-3.63**†	**-3.13**†	**-2.61**

(b) 1,000世代目の平均fitness

手法	3D	5D	7D	10D
従来法1	-5.71	-3.42	-3.00	-2.44
従来法2	-5.70	-3.47	-3.04	-2.49
IDE-提案法1-Tune	-5.72†	**-3.78**†	**-3.22**†	**-3.00**†
IDE-提案法2-Tune	-5.67	-3.62	-3.11†	-2.58
IDE-提案法3-Tune	-5.72	**-3.73**†	**-3.25**†	**-3.18**†
IDE-提案法1-Whole	-5.68	-3.56	**-3.14**†	**-2.70**†
IDE-提案法2-Whole	-5.70	**-3.61**†	**-3.15**†	**-3.04**†
IDE-提案法3-Whole	-5.66	**-3.63**†	**-3.17**†	-2.63

Table 3　fitness評価回数600回目と800回目の平均fitness値．太字と† は各々，Wilcoxonの符号検定で提案法が従来法1（通常IDE），および，従来法2よりも危険率5%で有意に良いことを示す．記号はTable 1を参照．

(a) fitness評価回数600回目の平均fitness値

手法	3D	5D	7D	10D
従来法1	-5.71	-3.42	-2.99	-2.43
従来法2	-5.70	-3.47	-3.04	-2.49
IDE-提案法1-Tune	-5.72	**-3.78**†	**-3.22**†	**-2.99**†
IDE-提案法2-Tune	-5.67	-3.62	-3.10	-2.54
IDE-提案法3-Tune	-5.72	**-3.73**†	**-3.22**†	**-3.08**†
IDE-提案法1-Whole	-5.68	-3.56	**-3.13**†	**-2.67**†
IDE-提案法2-Whole	-5.70	**-3.59**†	**-3.14**†	**-2.93**†
IDE-提案法3-Whole	-5.66	-3.63	-3.11	-2.56

(b) fitness評価回数800回目の平均fitness値

手法	3D	5D	7D	10D
従来法1	-5.71	-3.42	-3.00	-2.43
従来法2	-5.70	-3.47	-3.04	-2.49
IDE-提案法1-Tune	-5.72	**-3.78**†	**-3.22**†	**-3.00**†
IDE-提案法2-Tune	-5.67	-3.62	-3.11	-2.57
IDE-提案法3-Tune	-5.72	**-3.73**†	**-3.25**†	**-3.14**†
IDE-提案法1-Whole	-5.68	-3.56	**-3.14**†	**-2.69**†
IDE-提案法2-Whole	-5.70	**-3.60**†	**-3.15**†	**-2.99**†
IDE-提案法3-Whole	-5.66	-3.63	**-3.13**†	-2.61

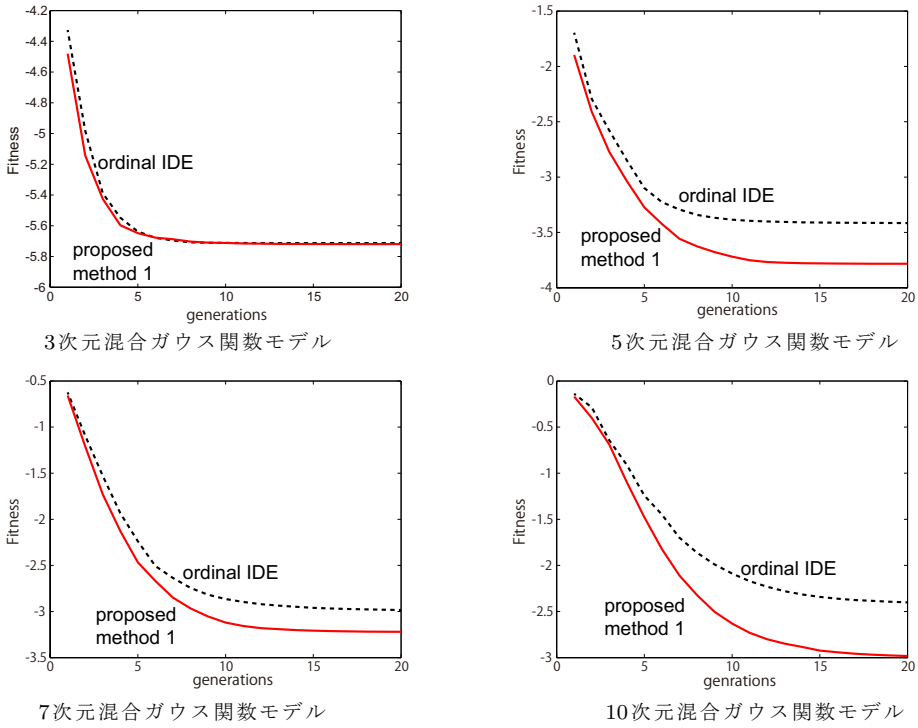

Fig. 3　3, 5, 7, 10次元混合ガウス関数モデルの最適探索30試行の平均収束曲線．破線は従来法1（通常DE），実践は提案法1.

来法，提案法とも早い世代から収束し切ってしまっており，20世代目では両者の差が見られないのに対し，難易度が上がるにつれて20世代ではまだ収束途中であり，収束の速い提案法の効果が顕著になったからである，と結論付けられる．

Table 2(b)の1,000世代目での評価はIDEユーザの評価条件ではなく第5節で行うDEでの性能評価と言えるが，傾向はIDEのTable 2(a)と同様である．

従来法1と2が共に対比較ベースの探索であることに対し，提案法 1 と2は3点比較，提案法3は4点比較であるので，Table 2(a)だけの比較では公平ではない．そこで，fitness評価回数を同じにした場合の収束状況を比較した結果がTable 3である．対比較の従来法は毎世代40回（＝20個体×2回比較）のfitness計算を行うので，fitness評価回数600回目と800回目は，15世代目と20世代目での評価になる．3点比較の提案法1と2では10世代目と14世代目での評価に，4点比較の提案法3では8世代目と10世代目での評価になる．

結果はTable 2と同様で，3次元混合ガウス関数のように早くから収束している場合には提案法の効果は見られないが，次元数が上がり難易度が上がるにつれて，同じfitness評価回数でも提案法が有意に早く収束している．

なお，Table 3(a) の5次元混合ガウス関数で，提案法2-Wholeと提案法3-Wholeの平均fitness値では後者の方が良いのに，従来法と比較した場合の有意差は前者にのみ見られる点に違和感を持たれるかもしれない．これは有意差検定はデータの分散に依存するためである（Wilcoxonの符号検定の各々のp値は，0.0056と0.1782）．同様の結果は2(b) の5次元関数での提案法2-Tuneと提案法2-Wholeの間にもみられる．

4.3　対称点計算の鏡像点に関する考察

Table 2(a)(b)共に共通した傾向であり，動的に対称点計算のための鏡像点が変化するTuneも探索空間の中心点を鏡像点とするWholeも，ほぼ同様の性能を示しており，いずれが有効かは言い難い．

この1つの理由として，IDEユーザモデルとして用いた混合ガウス関数の大局最適解が探索空間の中心にあることが考えられる．探索空間の中心を個体分布の中心として徐々に分布が狭くなっていくため，TuneもWholeも探索の進行に関わらず鏡像点がほぼ中央であると思われるからである．しかし第5節の実験結果からはこれを支えるだけの根拠は得られていない．

提案法2だけは結果が特異であり，Whole方式の方がTune方式よりも良い．

5 DEでの評価実験と解析

5.1 評価関数と実験条件

提案法は1回当たりの評価に多少の疲労増加を招いても探索の高速化を実現することで，結果としてIDEユーザ疲労軽減を狙ったものである．しかし，提案手法はIDEに限定されるものではなく，fitness関数を用いる通常のDE探索の高速化にも利用できる．そこで，24ベンチマーク関数[25]を使った最小値探索問題で従来法（通常DEとOBDE）とを比較して提案手法の高速化を評価する．Table 4にベンチマーク関数の定義，探索範囲，大局的最適解，特性を示す．

Table 4 評価実験に用いる24ベンチマーク関数[25]．最小値探索問題となるよう符号調整している．(Uni=単峰性，Mul=多峰性，Sh=シフト，Rt=回転，GB=最適解が探索空間の境界上にある関数，HC=両特性を備えた関数，NS=変数相互依存の関数，S=変数独立の関数．)

No.	関数名	探索範囲	最適解	特性
F1	Sh Sphere	[-100,100]	-450	Sh-Uni-S
F2	Sh Schwefel 1.2	[-100,100]	-450	Sh-Uni-NS
F3	Sh Rt Elliptic	[-100,100]	-450	Sh-Rt-Uni-NS
F4	F2 with Noise	[-100,100]	-450	Sh-Uni-NS
F5	Schwefel 2.6 GB	[-100,100]	-310	Uni-NS
F6	Sh Rosenbrock	[-100,100]	390	Sh-Mul-NS
F7	Sh Rt Griewank	[0,600]	-180	Sh-Rt-Mul-NS
F8	Sh Rt Ackley GB	[-32,32]	-140	Sh-Rt-Mul-NS
F9	Sh Rastrigin	[-5,5]	-330	Sh-Mul-Sep
F10	Sh Rt Rastrigin	[-5,5]	-330	Sh-Rt-Mul-NS
F11	Sh Rt Weierstrass	[-0.5,0.5]	90	Sh-Rt-Mul-NS
F12	Schwefel 2.13	$[-100, 100]$	-460	Mul-NS
F13	Sh Expanded F8F2	[-3,1]	-130	Sh-Mul-NS
F14	Sh Rt Scaffer F6	[-100,100]	-300	Sh-Rt-Mul-NS
F15	HC Function	[-5,5]	120	HC-S
F16	Rt HC F1	[-5,5]	120	Rt-HC-NS
F17	F16 with Noise	[-5,5]	120	Rt-HC-NS
F18	Rt HC F2	[-5,5]	10	Rt-HC-NS
F19	F18 with Basin	[-5,5]	10	Rt-HC-NS
F20	F18 with GB	[-5,5]	10	Rt-HC-NS
F21	Rt HC F3	[-5,5]	360	Rt-HC-NS
F22	F21 with NM	[-5,5]	360	Rt-HC-NS
F23	NC Rt F2	[-5,5]	360	HC-NS
F24	Rt HC F4	[-5,5]	260	Rt-HC-NS

第4節のIDE評価と異なり，IDEユーザ疲労を考える必要がないDEへの応用では，収束が収束閾値（VTR）に達するまでのfitness計算回数（NFC）で提案法と従来法の収束速度，および，収束閾値に達する成功で比較評価する．NFCが少ない程収束が速いことになる．評価は$MAX_{NFC} = 1{,}000$世代までの収束実験を30試行行う．式(4)で定義する収束閾値（value-to-reach: VTR）に辿り着いた時を収束したとする．収束速度比較のため，最終世代（1,000世代目）のNFCに基づく式(7)の加速率ARを全関数で用いる．$AR > 1$は提案法が通常DEに比べて収束が早いことを意味する．また，各評価関数で収束閾値VTRに辿り着いた回数から式(6)の収束成功率SRを定義する．さらに，評価関数全体の平均加速率と平均成功率を求め最終結果に用いる．

$$\text{収束閾値}VTR = MAX_{NFC}\text{世代での各手法の平均}fitness\text{値} \quad (4)$$

$$NFC = VTR\text{に達するまでの平均}fitness\text{計算回数} \quad (5)$$

$$\text{収束成功率}SR = \frac{VTR\text{への到達回数}}{\text{試行数}} \quad (6)$$

$$\text{加速率}AR = \frac{NFC_{\text{通常}DE}}{NFC_{\text{提案法}}} \quad (7)$$

DE評価実験のパラメータを以下のように設定する．各変数の探索範囲を広く設定した10次元関数を50個体で探索するよう設定してタスクの難易度を上げた条件にしている．

個体数	50
scale factor F	0.3
交差率	0.7
DE演算法	DE/best/1/bin
最大探索世代MAX_{NFC}	1,000世代
収束閾値VTR	Table 5のVTR値
ベンチマーク関数の次元数 D	10
試行数	30

5.2 提案法の評価

Table 5に24ベンチマーク関数を用いた収束特性を示す．表中の数値は，1,000世代目における30試行平均のfitness値である．後述のTable 6の収束成功率SRで用いる収束閾値VTRもこの表に記載する．演算量比較はTable 6で行う．

同じ世代までの探索であれば，一部の関数を除いて提案法は従来法より有意に収束が速い．効果が見られないF12関数は解探索ができていないことが理由である．F12関数の最適解は-460であるにも変わらず，他のベンチマーク関数の収束とは異なって1,000世代目でも大きな正数fitness値であり，まったく収束していないことが判る．10次元のF12関数を50個体程度で探索するには難し過ぎることが理由であろう．F18関数系（F18およびその変形であるF19, F20）も従来手法，提案手法ともに収束が悪い．10次元関数のF12とF18関数は50個体で探索するには難しすぎ

Table 5　1,000世代目におけるベンチマーク関数（F1〜F24）の30試行平均fitness値．太字と† は各々，Wilcoxonの符号検定で提案法が従来法1（通常IDE），および，従来法2よりも危険率5%で有意に良いことを示す．

方法	F1	F2	F3	F4	F5	F6	F7	F8	F9	F10	F11	F12
従来法1	-354.71	-250.18	192113.06	-141.32	2359.79	733669.47	-170.87	-119.60	-313.48	-302.79	95.21	3374.60
従来法2	**-413.53**	**-359.01**	308946.65	**-448.03**	26.70	287810.81	**-179.41**	**-136.56**	**-320.56**	**-320.56**	**91.16**	18078.54
DE-提案法1-Tune	**-440.88**†	**-438.45**†	67357.98	**-416.28**†	**-39.89**	**10528.84**†	**-179.74**†	**-137.68**†	**-323.89**†	**-323.89**†	**90.89**	12610.21
DE-提案法2-Tune	**-444.75**†	**-441.15**†	**26538.41**†	**-433.12**†	**-127.78**†	**7668.20**†	**-179.84**†	**-138.29**†	**-323.44**†	**-323.44**†	**90.92**	13725.32
DE-提案法3-Tune	**-447.59**†	**-440.02**†	**12532.29**†	**-443.62**†	**-120.31**†	**1042.11**†	**-179.87**†	**-138.72**†	**-324.02**†	**-324.02**†	**90.73**†	14801.29
DE-提案法1-Whole	**-450.00**†	**-450.00**†	**-450.00**†	**-450.00**†	**-310.00**†	**1919.73**†	**-179.95**†	**-140.00**†	**-326.68**†	**-326.68**†	**90.00**†	5819.92
DE-提案法2-Whole	**-450.00**†	**-450.00**†	**-450.00**†	**-450.00**†	**-310.00**†	**712.98**†	**-179.94**†	**-140.00**	**-326.52**†	**-326.52**†	**90.00**†	14088.20
DE-提案法3-Whole	**-450.00**†	**-450.00**†	**-450.00**†	**-450.00**†	**-310.00**†	**1363.79**†	**-178.53**	**-135.63**	**-326.75**†	**-326.75**†	**90.00**†	14452.89
VTR	-419.52	-384.87	126782.99	-404.05	247.03	130589.49	-178.52	-135.81	-321.51	-320.17	91.11	12118.87
方法	F13	F14	F15	F16	F17	F18	F19	F20	F21	F22	F23	F24
従来法1	-127.92	-296.71	498.21	299.84	306.39	990.61	990.25	992.80	1463.61	1253.51	1546.75	1077.70
従来法2	-128.18	**-297.98**	**158.22**	**158.22**	**157.13**	992.86	1006.69	1005.34	**414.45**	**414.45**	**400.56**	**911.72**
DE-提案法1-Tune	**-128.99**†	**-298.90**†	**148.68**†	**148.68**†	**147.75**†	959.60	958.27	959.23	**403.18**†	**403.18**†	**385.85**†	**354.11**†
DE-提案法2-Tune	**-129.09**†	**-298.97**†	**147.01**†	**147.01**†	**145.43**†	962.36	970.57	965.20	**400.62**†	**400.62**†	**386.21**†	**339.78**†
DE-提案法3-Tune	**-129.06**†	**-299.01**†	**143.99**†	**143.99**†	**144.87**†	966.31	965.93	964.67	**398.10**†	**398.10**†	**384.86**†	**310.60**†
DE-提案法1-Whole	**-128.48**	**-299.84**†	**138.89**†	**138.89**†	**136.72**†	989.40	988.78	994.29	**408.30**	**408.30**	**399.27**	**823.33**†
DE-提案法2-Whole	127.98	**-299.83**†	**137.14**†	**137.14**†	**137.05**†	990.10	990.82	991.63	**396.67**†	**396.67**†	**394.83**	**348.72**
DE-提案法3-Whole	**-128.53**	**-297.97**†	**136.09**†	**136.09**†	**134.86**†	1007.10	1007.10	1007.22	**397.39**†	**397.39**†	**394.83**	**279.92**†
VTR	-128.53	-298.62	188.53	163.73	163.77	982.36	984.80	985.04	535.29	509.03	536.65	555.74

Table 6　ベンチマーク関数（F1〜F24）を用いた30試行平均の評価指標．fitness計算回数NFC，収束成功率SR，加速率ARは式5，6，7を参照．

	従来法1（通常DE）			DE-提案法1-Tune			DE-提案法2-Tune			DE-提案法3-Tune		
関数	NFC	SR	AR	NFC	SR	AR	NFC	SR	AR	NFC	SR	AR
F1	54240	0.46		11830	0.92	4.58	11530	0.92	4.70	2160	0.99	25.11
F2	70757	0.29		3045	0.98	23.24	2925	0.98	24.19	4007	0.98	17.66
F3	36897	0.63		26670	0.82	1.38	6670	0.96	5.53	2180	0.99	16.93
F4	84290	0.16		25225	0.83	3.34	20630	0.86	4.09	5067	0.97	16.64
F5	96707	0.03		22170	0.85	4.36	7205	0.95	13.42	10127	0.95	9.55
F6	47373	0.53		6385	0.96	7.42	1395	0.99	33.96	1640	0.99	28.89
F7	87030	0.13		1740	0.99	50.02	1545	0.99	56.33	1907	0.99	45.65
F8	100000	0		16615	0.89	6.02	1675	0.99	59.70	8687	0.96	11.51
F9	86920	0.13		29475	0.8	2.95	33730	0.78	2.58	39727	0.8	2.19
F10	100000	0		14805	0.9	6.75	23890	0.84	4.19	26727	0.87	3.74
F11	100000	0		41970	0.72	2.38	42280	0.72	2.37	49527	0.75	2.02
F12	45890	0.54		123070	0.18	0.37	129360	0.14	0.35	173673	0.13	0.26
F13	57950	0.42		22945	0.85	2.53	22205	0.85	2.61	36100	0.82	1.61
F14	100000	0		64130	0.57	1.56	65970	0.56	1.52	71580	0.64	1.40
F15	100000	0		2535	0.98	39.45	2525	0.98	39.60	4080	0.98	24.51
F16	100000	0		17835	0.88	5.61	8300	0.94	12.05	5160	0.97	19.38
F17	100000	0		13720	0.91	7.29	3925	0.97	25.48	19413	0.9	5.15
F18	70393	0.30		90865	0.39	0.77	85890	0.43	0.82	90765	0.39	0.78
F19	63820	0.36		85830	0.43	0.74	90645	0.40	0.70	95670	0.36	0.67
F20	63827	0.36		85825	0.43	0.74	85785	0.43	0.74	95670	0.36	0.67
F21	100000	0		1000	0.99	100.00	950	0.99	105.26	1300	0.99	76.92
F22	100000	0		1215	0.99	82.30	1140	0.99	87.72	1567	0.99	63.83
F23	100000	0		810	0.99	123.46	755	0.99	132.45	1087	0.99	92.02
F24	96700	0.03		6860	0.95	14.10	11780	0.92	8.21	9073	0.95	10.66
平均		0.14			0.87	20.47		0.89	26.19		0.90	19.88

	従来法2（OBDE）			DE-提案法1-Whole			DE-提案法2-Whole			DE-提案法3-Whole		
関数	NFC	SR	AR	NFC	SR	AR	NFC	SR	AR	NFC	SR	AR
F1	51530	0.62	16.26	1640	0.99	33.07	1635	0.99	33.17	1987	0.99	27.30
F2	65536	0.52	15.25	2480	0.98	28.53	2475	0.98	28.59	3360	0.98	21.06
F3	64892	0.53	10.86	1570	0.99	23.50	1570	0.99	23.50	2027	0.99	18.21
F4	12262	0.91	13.43	2805	0.98	30.05	2655	0.98	31.75	3547	0.98	23.77
F5	20249	0.85	38.26	2235	0.99	43.27	2305	0.98	41.96	3047	0.98	31.74
F6	19997	0.85	21.43	1400	0.99	33.84	1270	0.99	37.30	1613	0.99	29.36
F7	15550	0.89	42.79	1445	0.99	60.23	1415	0.99	61.51	62147	0.69	1.40
F8	42493	0.69	37.62	1465	0.99	68.26	1465	0.99	68.26	101260	0.49	0.99
F9	65787	0.52	13.25	9700	0.94	8.96	9420	0.94	9.23	13860	0.93	6.27
F10	43475	0.68	22.21	8765	0.94	11.41	8420	0.94	11.88	12687	0.94	7.88
F11	65532	0.52	19.54	2335	0.98	42.83	2320	0.98	43.10	3033	0.98	32.97
F12	96959	0.29	5.38	95985	0.36	0.48	134105	0.11	0.34	178320	0.11	0.26
F13	74797	0.45	8.66	72370	0.52	0.80	106425	0.29	0.54	70580	0.65	0.82
F14	110669	0.19	4.4	26600	0.82	3.76	26180	0.83	3.82	40033	0.8	2.50
F15	6823	0.95	44.39	3420	0.98	29.24	3340	0.98	29.94	4693	0.98	21.31
F16	43292	0.68	23.78	6050	0.96	16.53	5285	0.96	18.92	7940	0.96	12.59
F17	39127	0.71	22.42	5935	0.96	16.85	5595	0.96	17.87	8107	0.96	12.34
F18	96439	0.30	0.73	105410	0.30	0.67	105425	0.30	0.67	160307	0.20	0.44
F19	87433	0.36	0.73	105470	0.30	0.61	105395	0.30	0.61	160300	0.20	0.40
F20	87443	0.36	0.73	110270	0.26	0.58	105390	0.30	0.61	160300	0.20	0.40
F21	1420	0.99	75.43	1145	0.99	87.34	990	0.99	101.01	1293	0.99	77.32
F22	1525	0.99	70.19	1315	0.99	76.05	1225	0.99	81.63	1567	0.99	63.83
F23	1302	0.99	81.54	915	0.99	109.29	815	0.99	122.70	1040	0.99	96.15
F24	51535	0.62	32.71	16830	0.89	5.75	11720	0.92	8.25	2327	0.99	41.56
平均		0.73	11.18		0.92	30.49		0.91	32.38		0.89	22.12

るのであろう．F15〜F17，F21〜F24関数は1,000世代ではまだ大局的最適解には行き着いていないものの，平均fitnessをみるとかなり大局的最適解に近づいており，解探索ができていない従来法1と比べて大きな改善である．総論として，実験条件に対して難易度が高いと思われるF12関数とF18関数系（F18〜F20）を除いて，いずれの評価関数の場合でも，提案法は従来法よりも有意に収束速度が加速できている．

Table 6は演算量（fitness計算回数）比較と収束閾値への到達成功率を比較するものである．従来法1（通常DE）では高々50個体で10次元のベンチマーク関数の探索をすることは重荷すぎて，収束閾値VTRへ辿り着ける収束成功率SRは24関数平均で14%，収束成功率SRが5割を超えるのは24関数中わずかに3関数にしか過ぎない．従来法2（OBDE）の平均成功率は73%だが，提案法は90%前後に上る．収束していないF12関数を除いてどの関数に対しても高い収束達成率を示している．

また収束閾値VTRに達するまでのfitness関数の平均計算回数（NFS）は従来法1や従来法2に比べて大幅に少なっている．すなわち，Table 5の世代だけでなくTable 6のNFSで見ても，早く少ない演算量で収束閾値に達しているので，提案手法は高速化を実現していると言える．

収束閾値VTRに辿り着く世代を求める場合は，

$$\text{世代数} = NFC/(\text{個体数} \times p)$$

で求める．ここで，pは1回に比較する個体数で，対比較ベースの従来法は$p = 2$，3点比較ベースの提案法1と2は$p = 3$，4点比較ベースの提案法3は$p = 4$である．例えば，F4関数の場合，収束閾値に辿りつく平均世代数は，従来法1（842.9世代），従来法2（122.6世代），提案法1-Tune（168.2世代），提案法2-Tune（137.5世代），提案法3-Tune（25.3世代）と計算できる．

実験結果のまとめとして，すべての関数で，提案法1〜3 × TuneまたはWholeの6通りの少なくともいずれかは従来法1〜2よりも少ないfitness関数計算で収束閾値に達し，当然収束世代数も早く，到達成功率，加速率のいずれをみても同等以上の収束性能を示している．

5.3 対称点計算の鏡像点に関する考察

Table 6の動的鏡像点法（Tune）と探索空間の中心鏡像点法（Whole）とを比較すると，全体的に提案法の間で大きな違いはない．10倍以上の差が見られるのは，(1) F3関数の提案法1，(2) F7関数の提案法3，(3) F8関数の提案法1と3，(4) F11関数の提案法1〜3，だけである．これらの共通点は，収束成功率SRの違いが，収束閾値VTRに達成するまでのfitness計算回数の違いを生み，その結果加速率ARに違いが生じている点である．

対称点を求めるための鏡像点を常に探索空間の中心に固定するWholeと動的に鏡像点を変えるTuneは，大局的最適解が探索空間の中心にあるかどうかで性能に影響があるように思われるが，実際にはこの違いは現れていない．24ベンチマーク関数中に，探索空間の境界に最適解がある関数は，F5, F8, F20関数であるが，これらの関数でどちらかの手法が共通して良くなっているわけではない．

収束閾値VTRが異なればこの見かけも大きく異なる．

Table 5で最終世代の1,000世代目で両手法を比較すると，Table 6以上に顕著な違いが見られるのはF3関数である．動的鏡像点法（Tune）では平均収束fitness値が数万の値であるのに探索空間の中心鏡像点法（Whole）では-450と大局的最適解に辿り着いている．しかし収束閾値が$VTR = 126,782.99$と相当高いため，収束閾値での評価を行っているTable 6では大きな違いが見られなかったと言える．F3関数で$VTR = 100$として追加実験を行うと，Tune法では収束成功率が13%と33%で低調であったのに対し，Whole法ではすべて98%と高い収束性を示していた．以上のTable 5と追加実験から，F3関数ではWholeの収束特性はTuneの特性よりも早く収束するという違いが見られたが，他の23関数ではこのような違いが見られない．

結論として，今回の24ベンチマーク関数だけでは，対称点を決定する鏡像の位置による顕著な影響は見られなかった．

5.4 機能包含と収束性能に関する考察

第3節で述べたように，提案法1は従来法2を，提案法3は従来法1と2を機能的に包含していると言えるにもかかわらず，実験結果は概ね包含している手法が包含されている手法より良いが，すべてに対して上回っている訳ではない．この点について考察する．

これは，best-best-bestと各々の比較で最良個体を選択することが全体のbestになることが保証

されている訳ではない，すなわち最適性の原理[2]が成り立たないためである．良い個体周辺領域はそうでない個体の周辺領域よりは，最適解に向かう探索上有利であることが確率的には期待される．しかし，多峰性のタスクでは最適性の原理の成立条件の1つである単調性が成り立たたず，3点比較，4点比較を行い対称vectorが選択された場合，その比較3個体あるいは4個体の間では選択された対称vectorが一番良くても，選択された対称vectorの領域が選択されなかったvectorsの領域に比べて大局的最適解に行き着くために有利な領域かどうかは保証されない．また一旦探索パスが異なれば以降の探索パスも異なる．

これら2点から，手法Aが機能的に手法Bを包含する時，手法Aは確率的に手法Bよりも収束が高速であることが期待できるが，その期待を保証するものではないと言える．Table 5と6の実験結果はこの考察傾向に沿ったものである．

6 結論と今後の課題

本論文では，opposition-based learningをDEに取り込む新しいOBDEを提案し評価した．通常の対比較ベースDEや従来のOBDEのように対比較をしながら次世代個体を生成する方法に比べて，提案手法は3点比較，4点比較をする手法であるので，1回当たりの個体比較の負担あるいはfitness計算コストが高い．しかし，同じ世代での比較はもちろんのこと，IDEのシミュレーション評価，および，24ベンチマーク関数での評価で，同じfitness計算回数での比較や収束閾値に辿り着くまでのfitness計算コストを比較すると，少ないコストで高い収束性能を示すことができる．

3〜4個体を記憶できるIDEタスクは多々あると思われる．この場合のIDEユーザの比較疲労は，2個体を比較する疲労の1.5倍，2倍になるわけではないことが期待でき，上述の収束高速化と合わせて，特にIDEには有効な手法であると結論付けられる．

本論文では，同じ実験比較条件で複数の初期値による複数手法の特性を比較するためシミュレーション実験を行った．次のステップとして，記憶できる個体個数前後でのIECユーザの負担を定量化し，本論文の収束高速化実験と合わせることで本提案手法のIDEにおける性能評価の基礎データを揃える必要がある．その上で最終評価として，人間のIDEユーザを用いた主観評価実験を行い，比較疲労と収束高速化の総合評価を行うことで本提案手法の評価を完了させる．

謝辞

本研究は科学研究費（課題番号23500279）の助成を受けたものである．筆者の裴岩は吉田奨学会からの奨学金を受けて本研究を遂行した．ここに感謝する．

参考文献

1) Ahandani, M.A. and Alavi-Rad, H., “Opposition-based learning in the shuffled differential evolution algorithm”, Soft Computing, pp. 1–35, (2012).
2) Bellman, Richard *Dynamic Programming*, Princeton University Press (1957).
3) Brest, J., Greiner, S., Boskovic, B., Mernik, M. and Umer, V. , “Self-adapting control parameters in differential evolution: A comparative study on numerical benchmark problems”, IEEE Trans. on Evolutionary Computation, vol.10, no.6, pp.107–125, (Feb., 2008).
4) Dawkins, Richard, *The Blind Watchmaker*. Essex: Longman, 1986.
5) Gong, W., Cai Z., Ling, C. X., and Li, H., “Enhanced differential evolution with adaptive strategies for numerical optimization”, IEEE Trans. on System, Man and Cybernetics, Part B, vol.41, no.2, pp.397–413, (Apr., 2011).
6) Miller, George Armitage, “The magical number seven, plus or minus two: Some limits on our capacity for processing information”, The Psychological Review, vol.63, pp.81–97(1956).
7) Iacca, G., Neri, F., and Mininno, E., “Opposition-based learning in compact differential evolution”, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) vol. 6624 LNCS, no. PART 1, pp.264–273 (2011).
8) Herdy, Michael, “Evolutionary optimisation based on subjective Selection - evolving blends of coffee”, 5th Eourpean Cong. on Itellengt Techniques and Soft Computing (EUFIT’97). pp.640–644 (1997).
9) Johanson, Brad, “Automated fitness raters for the GP-music system,” Univ. of Birmingham, Master’s Degree Final Project (Sept., 1997).
10) Kosorukoff, Alex, “Human-based genetic algorithm”, IEEE Int. Conf. on Systems, Man and Cybernetic, Tucson, Arizona, USA, pp.3464–3469 (Oct., 2001).
11) Legrand, P., Bourgeois-Republique, C., et. al., “Interactive Evolution for Cochlear Implants Fitting,” Genetic Programming and Evolvable Machines, vol.8, no.4, pp.319–354 (2007).
12) Lim, I. S. and Thalmann, D., “Tournament selection for browsing temporal signals,” ACM Symp. on Applied Computing (SAC2000), Como, Italy, pp.570–573, March, (2000).
13) Nakano, Y. and Takagi, H., “Influence of quantization noise in fitness on the performance of

interactive PSO," IEEE Congress on Evolutionary Computation (CEC2009), Trondheim, Norway, pp.2146–2422 (2009).

14) Mader, J., Abonyi, J., and Szeifert, F., "Interactive particle swarm optimization," Int. Conf. on Intelligent Systems Design and Applications (ISDA2005), pp.2314–319, Wroclaw, Poland (2005).

15) Pei, Y. and Takagi, H., "Accelerating evolutionary computation with elite obtained in projected one-dimensional spaces", 5th Int. Conf. on Genetic and Evolutionary Computing, Kinmen/Taiwan and Xiamen/China, pp.89–92 (Aug./Sept., 2011).

16) Pei, Y. and Takagi, H., "A survey on accelerating evolutionary computation approaches", 3rd Int. Conf. on Soft Computing and Pattern Recognition (SoCPaR2011), pp.201–206, Dalian, China (Oct., 2011).

17) Pei, Y. and Takagi, H., "Comparative evaluations of evolutionary computation with elite obtained in reduced dimensional spaces", 3rd Int. Conf. on Intelligent Networking and Collaborative Systems (INCoS2011), pp.35–40, Fukuoka, Japan (Nov./Dec., 2011).

18) Pei,Y. and Takagi, H., "Fourier analysis of the fitness landscape for evolutionary search acceleration", 2012 IEEE Congress on Evolutionary Computation (IEEE CEC 2012), Brisbane, Australia pp.2934–2940 (June, 2012).

19) Price, K., Storn, R., and Lampinen, J., "Differential evolution: A practical approach to global optimization," Berlin, Germany: Springer-Verlag, (2005).

20) Qin, A. K., Huang, V. L., and Suganthan, P. N., "Differential evolution algorithm with strategy adaptation for global numerical optimization", IEEE Trans. on Evolutionary Computation, vol.13, no.2, pp.398–417, (Apr., 2009).

21) Qin, A. K., Huang, V. L., and Suganthan, P. N., "JADE: adaptive differential evolution with external achieve", IEEE Trans. on Evolutionary Computation, vol.13, no.5, pp.945–958, (Oct., 2009).

22) Rahnamayan, S., Tizhoosh, H. R., and Salama, M. M., "Opposition-based differential evolution algorithms," IEEE Trans. on Evolutionary Computation, vol.12, no.1, pp.64–79, (Feb., 2008).

23) Sims, Karl, "Ariticial evolution for computer graphic", Computer Graphics, vol.25, no.4, Proc. of Siggraph 91, pp.319–328, (Jul., 1991).

24) Storn, R. and Price, K., "Differential evolution — A simple and efficient heuristic for global optimization over continuous spaces", Journal of Global Optimization 11. Norwell, MA: Kluwer, pp.341–359 (1997).

25) Suganthan, P., Hansen, N., Liang, J., Deb, K., Chen, Y., Auger, A. and Tiwari, S., "Problem definitions and evaluation criteria for the CEC 2005 special session on Real-Parameter Optimization", Technical Report, Nanyang Technological University, Singapore, "$http://www.ntu.edu.sg/home/EPNSugan$" (2005).

26) 「対比較ベース対話型差分進化」第3回進化計算シンポジウム，那覇，pp.245–251 (2009年12月).

27) Takagi, Hideyuki, "Interactive evolutionary computation: Fusion of the capabilities of EC optimization and human evaluation", Proceedings of the IEEE, vol. 89, no. 9, pp. 1275–1296 (2001).

28) 「単峰性関数当てはめによるGA収束高速化」知能と情報（日本知能情報ファジィ学会誌）, vol.15, no.2, pp.219–229 (2003).

29) Takagi, H. and Ohsaki, M., "Interactive Evolutionary Computation-based Hearing-aid Fitting," IEEE Trans. on Evolutionary Computation, vol.11, no.23, pp.414–427 (2007).

30) Takagi, Hideyuki, "New IEC research and framkworks", in Aspects of Soft Computing, Intelligent Robotic and Control, ed.by J. Fodor and J. Kacprzyk, Springer-Verlag, Berlin Heidelberg, pp.65–78 (2009).

31) Takagi, Hideyuki, "Interactive evolutionary computation for analyzing human aware mechanism," Applied Computational Intelligence and Soft Computing, vol. 2012, Article ID 694836 (2012).

32) Tizhoosh, H. R., "Opposition-based learning: A new scheme for machine intelligence", Int. Conf. on Computational Intelligence for Modelling Control and Automation (CIMCA2005), vol. I, pp. 695–701, Vienna, Austria, (2005).

33) Tizhoosh, H. R., "Reinforcement learningbased on actions and opposite Actions", ICGST Int. Conf. on Artificial Intelligence and Machine Learning (AIML-05), Cairo, Egypt, (Dec., 2005).

34) Jin, Yaochu, "A comprehensive survey of fitness approximation in evolutionary computation", Soft Computing, Springer, vol.9, no.1, pp.3–12 (2005).

35) Wang, H., Wu, Z., and Rahnamayan, S., "Enhanced opposition-based differential evolution for solving high-dimensional continuous optimization problems", Soft Computing, vol.15, no.11, pp.2127–2140 (Nov., 2011).