数理解析研究所講究録
第1734巻 2011年 188-195

Differential Evolutionによる非線形最適化 —直交ベクトルを用いた回転不変性を有する交叉の実現—

広島修道大学商学部　阪井　節子 (Setsuko Sakai)
Faculty of Commercial Sciences, Hiroshima Shudo University
広島市立大学大学院 情報科学研究科　高濱　徹行 (Tetsuyuki Takahama)
Department of Intelligent Systems, Hiroshima City University

1 はじめに

進化的アルゴリズム (Evolutionary Algorithm, EA) は，生物進化の過程をモデル化した最適化アルゴリズムの総称であり，遺伝的アルゴリズム (Genetic Algorithm, GA)，進化戦略 (Evolution Strategy, ES)，差分進化 (Differential Evolution, DE)[1, 2, 3, 4] など多くのアルゴリズムが提案されている．特にDEはStorn and Priceによって1995年に提案された比較的新しいEAである．DEは他のEAと同様に解集団による確率的な多点探索を行う．DEの特徴としては，突然変異におけるステップ幅を探索点の収束に比例して決めることによりステップ幅の制御を不要にしたことが挙げられる．DEは非線形問題，微分不可能な問題，非凸問題，多峰性問題など様々な最適化問題に適用されており，これらの問題に対して高速で頑健なアルゴリズムであることが示されてきている [5, 6, 7, 8, 9].

最適化アルゴリズムに求められる性質の一つに回転不変性 (rotation-invariant) がある．回転不変性を有するアルゴリズムは，回転していない問題を解くのと同じように変数間に強い依存性がある問題を解くことができるという望ましい性質を持つ．DEで行われる2つの遺伝的操作の内，突然変異は回転不変であるが，交叉は回転不変ではない．したがって，DEは回転不変なアルゴリズムではなく，変数間に強い依存性がある問題に対する性能は低下すると考えられる．

本研究では，回転不変性を有する新たな交叉を提案する．DEでは突然変異によって集団から選択された3つ以上の親個体から変異ベクトル (個体) を生成する．交叉では，一つの親個体と変異ベクトルから，確率的に選択した遺伝子を組み合わせて一つの子個体を生成する．通常の交叉は，固定された直交座標系に従って遺伝子を選択するため，回転不変性を持たない．本研究では，回転不変性を実現するために，固定された座標系を用いる替わりに，現在の解集団に属する探索点から新たな直交座標系を構成することを提案する．新しい直交座標系を構成する方法としては，Gram-Schmidtの直交化法を用いて生成した直交ベクトルを基底ベクトルとして新たな座標系を構成する．回転不変性を有する交叉の有効性は，よく知られた13のテスト問題を解くことによって示す．

以下，2.では交叉における回転不変性について説明する．3.で回転不変性を有する新しい交叉を提案し，4.でこの交叉を導入したDEについて説明する．5.でテスト問題に対する実験結果を示し，6.はまとめである．

2 最適化問題と回転不変性

2.1 最適化問題

本研究では，決定変数の上下限制約のみを有する次のような最適化問題 (P) を扱う．

$$(P)\ \text{minimize}\quad f(\boldsymbol{x}) \qquad \text{subject to}\ \ l_i \le x_i \le u_i,\ i = 1, \ldots, n \tag{1}$$

ここで，$\boldsymbol{x} = (x_1, \cdots, x_n)$ は n 次元決定変数ベクトル，$f(\boldsymbol{x})$ は目的関数であり，f は線形あるいは非線形の実数値関数である．l_i, u_i はそれぞれ，n 個の決定変数 x_i の下限値，上限値である．さらに，以下では全ての上下限制約を満足する領域を探索空間 ($\mathcal{S}$) と呼ぶことにする．

2.2 回転不変交叉

EAにおいて，交叉は非常に重要な役割を果たす．回転不変性の観点から，実数値GAにおける2つの親による代表的な交叉を調べると，次のようになる．ここで，交叉する親個体を$\boldsymbol{x}$と$\boldsymbol{y}$，生成された子個体を$\boldsymbol{z}$とする．

- 算術交叉
  算術交叉では，以下のように2つの親個体の一次結合によって，1つの子個体を生成する．

$$z_i = rx_i + (1-r)y_i \tag{2}$$

  ここで，rは区間$[0,1]$上の一様乱数である．図1に算術交叉を示す．●が親個体，○が子個体を表す．与えられた問題が回転し探索点が回転したとき，算術交叉では，回転後の新たな親に対する子は灰色の丸になり，子○を回転したものと一致する．よって，算術交叉は回転不変交叉である．

- 一様交叉
  一様交叉では，以下のように一番目の親と二番目の親から等確率で遺伝子を取り出して子を生成する．

$$z_i = \begin{cases} x_i & \text{with prob. } 0.5 \\ y_i & \text{with prob. } 0.5 \end{cases} \tag{3}$$

  図2に一様交叉を示す．●が親個体，○が子個体を表す．一様交叉では，回転後の新たな親に対する子は薄い灰色の丸のいずれかになるが，これは子○が回転した濃い灰色の丸と一致しない．よって，一様交叉は回転不変交叉ではない．DEにおける交叉も一様交叉と同様の性質を持つ．

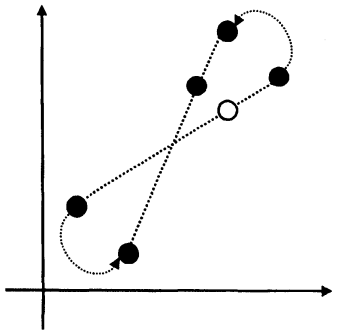

図1: 算術交叉と回転

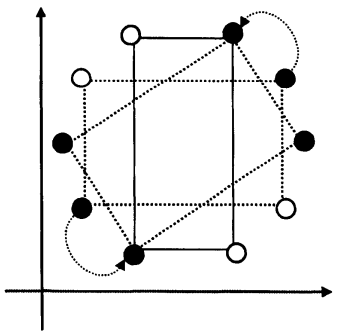

図2: 一様交叉と回転

3 回転不変交叉 (Rotation-Invariant Crossover)

DEにおける交叉である二項交叉や指数交叉は，一様交叉と同様に，親個体と変異ベクトルが対角線の頂点となる超直方体の1つの頂点を選択することに対応する．そのため，二項交叉や指数交叉もまた，回転不変性を有しない．これらの交叉に対して回転不変性を実現する一つの方法として，各時点における探索点の分布を反映した座標系を定義することが考えられる．

3.1 座標系と座標ベクトル

1つの直交座標系は，互いに直交する単位ベクトルの組によって定義される．本研究では，解集団$P = \{\boldsymbol{x}^i, i = 1,2,\cdots,N\}$から選択したベクトルの組に対してGram-Schmidtの直交化法を用いて，これらの直交単位ベクトル$\{\boldsymbol{b}_k, k = 1,2,\cdots,n\}$を次のように生成する．

1) 探索点の図心を求める

$$\boldsymbol{c} = \frac{1}{N}\sum \boldsymbol{x}^i \tag{4}$$

2) 図心から探索点への方向ベクトルを求める

$$\boldsymbol{d}_i = \boldsymbol{x}^i - \boldsymbol{c}, \quad i = 1, 2, \cdots, N \tag{5}$$

3) 方向ベクトルの中から，n 個のベクトルを選択する．本研究では，無作為に選択する．

$$V = \{\boldsymbol{v}_k | k = 1, 2, \cdots, n, \boldsymbol{v}_k \in \{\boldsymbol{d}_i\}\} \tag{6}$$

4) Gram-Schmidt の直交化法を用いて，選択された方向ベクトルから直交単位ベクトルを生成する．

$$\boldsymbol{b}_1 = \frac{\boldsymbol{v}_1}{\| \boldsymbol{v}_1 \|}, \quad \boldsymbol{b}_i = \frac{\boldsymbol{v}_i - \sum_{k=1}^{i-1} (\boldsymbol{v}_i, \boldsymbol{b}_k)\boldsymbol{b}_k}{\| \boldsymbol{v}_i - \sum_{k=1}^{i-1} (\boldsymbol{v}_i, \boldsymbol{b}_k)\boldsymbol{b}_k \|}, \quad (i = 2, \cdots n) \tag{7}$$

ここで，$(\boldsymbol{v}, \boldsymbol{b})$ は $\boldsymbol{v}$ と $\boldsymbol{b}$ の内積である．

3.2 座標ベクトルによる交叉

二項交叉や指数交叉では，親 $\boldsymbol{x}^i$ と変異ベクトル $\boldsymbol{x}'$ のいずれか一方の成分が選択される．すなわち，$\boldsymbol{x}^i$ から $\boldsymbol{x}'$ までの差分ベクトル $\boldsymbol{y} = \boldsymbol{x}' - \boldsymbol{x}^i$ が n 成分に分解され，確率的に選択される．これらの交叉は次のように定義できる．

$$\boldsymbol{x}^{child} = \boldsymbol{x}^i + \sum_{k \in K} (\boldsymbol{y}, \boldsymbol{e}_k)\boldsymbol{e}_k, \tag{8}$$

ここで，K は選択された成分の添字，$\boldsymbol{e}_k$ は第 k 成分が 1 で他の成分がすべて 0 である単位ベクトルである．

本研究では，各座標ベクトル $\boldsymbol{e}_k$ を，式 (7) によって求められた座標ベクトル $\boldsymbol{b}_k$ に置き換えることによって，回転不変二項交叉および回転不変指数交叉を実現する．図 3 に回転不変交叉のアルゴリズム，図 4 に 2 次元空間における回転不変交叉の例を示す．

```
y=x'-x^i;
x^child=x^i;
回転不変二項交叉
j=randint(1,n);
for(k=1; k≤n; k++) {
  if(k == j || u(0,1) < CR)
    x^child=x^child+(y,b_k)b_k;
}
回転不変指数交叉
k=1; j=randint(1,n);
do {
    x^child=x^child+(y,b_j)b_j;
    k=k+1; j=(j+1)%n;
} while(k≤n && u(0,1) < CR);
```

図 3: 回転不変交叉のアルゴリズム．randint(p, q) は区間 $[p, q]$ の整数乱数を，$u(l, r)$ は区間 $[l, r]$ 上の一様乱数を生成する関数である．

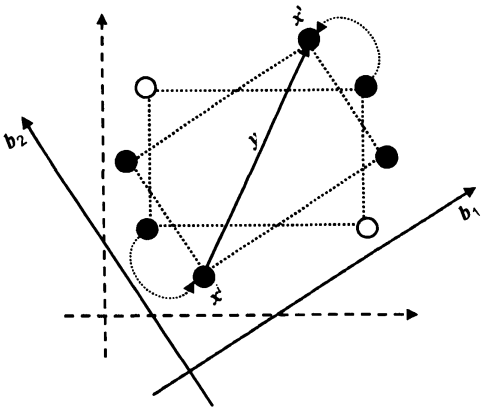

図 4: 回転不変交叉 (2 次元)

4 回転不変交叉を有する DE のアルゴリズム

4.1 Differential Evolution

DE では，探索空間中にランダムに初期個体を生成し，初期集団を構成する．各個体は決定ベクトルに対応し，n 個の決定変数を遺伝子として持つ．各世代において，全ての個体を親として選択する．各親に対して，

```
DE/rand/1/exp()
{
// Initialize an population
 P=N individuals generated randomly in S;
 for(t=1; FE ≤ FE_max; t++) {
  for(i=1; i ≤ N; i++) {
// DE operation
     x^p1=Randomly selected from P(p1 ≠ i);
     x^p2=Randomly selected from P(p2 ≠ i ≠ p1);
     x^p3=Randomly selected from P
                              (p3 ≠ i ≠ p1 ≠ p2);
     x'=x^p1+F(x^p2 − x^p3);
     x^child=trial vector is generated from
           x^i and x' by exponential crossover;
// Survivor selection
     if(f(x^child) ≤ f(x^i)) z^i=x^child;
     else                    z^i=x^i;
     FE=FE+1;
  }
  P={z^i, i = 1,2,···,N};
 }
}
```

図 5: DE の擬似コード．FE は関数評価回数，$FE_{\max}$ は最大評価回数である．

```
RIDE/rand/1/exp()
{
// Initialize an population
 P=N individuals generated randomly in S;
 for(t=1; FE ≤ FE_max; t++) {
  B=obtained coordinate vectors;
  for(i=1; i ≤ N; i++) {
   for(k=1; k<=2; k++) {
// DE operation
    x^p1=Randomly selected from P(p1 ≠ i);
    x^p2=Randomly selected from P(p2 ≠ i ≠ p1);
    x^p3=Randomly selected from P
                              (p3 ≠ i ≠ p1 ≠ p2);
    x'=x^p1+F(x^p2 − x^p3);
    if(k==1)
     x^child=trial vector is generated from
           x^i and x' by exponential crossover;
    else
     x^child=trial vector is generated from
           x^i and x' by rotation-invariant
           exponential crossover using B;
    FE=FE+1;
// Survivor selection
    if(f(x^child) ≤ f(x^i)) {
      x^i=x^child;
      break;
    }
   } } } }
```

図 6: RIDE の擬似コード

次のような処理が行われる．突然変異のために，選択された親を除く個体群から互いに異なる $1+2\,num$ 個の個体を選択する．最初の個体が基本ベクトルとなり，残りの個体対が num 個の差分ベクトルとなる．差分ベクトルは F(scaling factor) が乗算され基本ベクトルに加えられる．その結果得られたベクトルと親が交叉し，CR(crossover factor) により指定された確率で親の遺伝子をベクトルの要素で置換することにより，子のベクトル (trial vector) が生成される．最後に，生存者選択として，子が親よりも良ければ，親を子で置換する．

本研究では，差分ベクトル数を 1 ($num = 1$) とした DE/rand/1/exp を用いる．図 5 に擬似コードを示す．

4.2 RIDE

回転不変交叉を有する DE(RIDE) のアルゴリズムについて説明する．図 6 に，RIDE/rand/1/exp の擬似コードを示す．標準的な DE との違いは以下の点である．

1) 連続世代モデルを採用する．DE において通常用いられる離散世代モデルでは親より優れた子が次世代に生き残るが，連続世代モデルではすぐに親と子を置き換える．

2) 標準交叉と回転不変交叉によって最大 2 個の子を生成する．先ず標準的交叉を用いて子を生成しその子が親より良くなければ，回転不変交叉を用いてさらに 1 個の子を生成する．2 種類の異なる交叉によって異なるタイプの子が生成され探索点の多様性が増加するため，探索過程の頑健性が改良できる．

3) 境界を外れた解に対する鏡映操作．上下限制約を維持するために，探索空間の外部にある点を内部に動かす操作が必要である．本研究では，鏡映を用いる．

$$x_{ij} = \begin{cases} l_i + (l_i - x_{ij}) - \left\lfloor \frac{l_i - x_{ij}}{u_i - l_i} \right\rfloor (u_i - l_i) & (x_{ij} < l_i) \\ u_i - (x_{ij} - u_i) + \left\lfloor \frac{x_{ij} - u_i}{u_i - l_i} \right\rfloor (u_i - l_i) & (x_{ij} > u_i) \\ x_{ij} & (\text{otherwise}) \end{cases} \tag{9}$$

ここで，$\lfloor z \rfloor$ は z 以下の最大の整数である．なお，$F < 1$ ならば $\lfloor \cdot \rfloor$ の部分は 0 である．鏡映は，DE 操作によって新しい個体が生成されたときに適用する．

5 実験

本研究では，よく知られている 13 個のテスト問題を，RIDE，連続世代モデルを用いた DE(CDE)，離散世代モデルを用いた標準 DE(SDE) を用いて解き，結果を比較検討する．

5.1 テスト問題と実験条件

表 1 に示す 13 個の問題をテスト問題として使用する．f_1 は Sphere 関数，$f_2 \sim f_4$ は Schwefel 2.22, Schwefel 1.2, Schwefel 2.21 関数で，いずれも連続単峰性関数である．f_5 は Rosenbrock 関数で，2 次元及び 3 次元では単峰性関数であるが，それ以上の高次元では多くの局所解をもっている．f_6 は Step 関数，f_7 は雑音のある 4 次関数である．f_8 は Schwefel 2.26 関数，f_9 は Rastrigin 関数，f_{10} は Ackley 関数，f_{11} は Griewank 関数，$f_{12} \sim f_{13}$ はペナルティ付き関数である．関数 $f_8 \sim f_{13}$ は多峰性関数である．

表 1: D 次元のテスト問題 [10]

テスト問題	上下限制約
$f_1(\boldsymbol{x}) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$
$f_2(\boldsymbol{x}) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$
$f_3(\boldsymbol{x}) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^D$
$f_4(\boldsymbol{x}) = \max_i \{ \lvert x_i \rvert \}$	$[-100, 100]^D$
$f_5(\boldsymbol{x}) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^D$
$f_6(\boldsymbol{x}) = \sum_{i=1}^{D} \lfloor x_i + 0.5 \rfloor^2$	$[-100, 100]^D$
$f_7(\boldsymbol{x}) = \sum_{i=1}^{D} i x_i^4 + rand[0, 1)$	$[-1.28, 1.28]^D$
$f_8(\boldsymbol{x}) = \sum_{i=1}^{D} -x_i \sin \sqrt{\lvert x_i \rvert} + D \cdot 418.98288727243369$	$[-500, 500]^D$
$f_9(\boldsymbol{x}) = \sum_{i=1}^{D} \left[x_i^2 - 10 \cos(2\pi x_i) + 10 \right]$	$[-5.12, 5.12]^D$

テスト問題	上下限制約
$f_{10}(\boldsymbol{x}) = -20 \exp\left(-0.2 \sqrt{\frac{1}{D} \sum_{i=1}^{D} x_i^2} \right) - \exp\left(\frac{1}{D} \sum_{i=1}^{D} \cos(2\pi x_i) \right) + 20 + e$	$[-32, 32]^D$
$f_{11}(x) = \frac{1}{4000} \sum_{i=1}^{D} x_i^2 - \prod_{i=1}^{D} \cos\left(\frac{x_i}{\sqrt{i}} \right) + 1$	$[-600, 600]^D$
$f_{12}(x) = \frac{\pi}{D} [10 \sin^2(\pi y_1) + \sum_{i=1}^{D-1} (y_i - 1)^2 \{1 + 10 \sin^2(\pi y_{i+1})\} + (y_D - 1)^2] + \sum_{i=1}^{D} u(x_i, 10, 100, 4)$ where $y_i = 1 + \frac{1}{4}(x_i + 1)$ and $u(x_i, a, k, m) = \begin{cases} k(x_i - a)^m & x_i > a \\ 0 & -a \le x_i \le a \\ k(-x_i - a)^m & x_i < -a \end{cases}$	$[-50, 50]^D$
$f_{13}(x) = 0.1[\sin^2(3\pi x_1) + \sum_{i=1}^{D-1} (x_i - 1)^2 \{1 + \sin^2(3\pi x_{i+1})\} + (x_D - 1)^2 \{1 + \sin^2(2\pi x_D)\}] + \sum_{i=1}^{D} u(x_i, 5, 100, 4)$	$[-50, 50]^D$

実験条件は次の通りである．テスト問題の次元数を $D = 30$ に設定し，f_1 から f_{13} の関数を最適化する．DE の設定は，個体数 $N = 50$, $F = 0.7$, $CR = 0.9$ である．個体数は $2D$ から $10D$ 程度が良いとされているが，効率性を重視し少し小さい値とした．各関数について 30 回独立に試行を行い，近似解に到達するまでに要する関数評価回数を比較する．比較対象は，標準的な DE(SDE)，連続世代モデルを利用した DE(CDE) および本研究で提案した回転不変交叉を導入した DE(RIDE) である．近似解の許容誤差は，ノイズ付きの関数 f_7 については 1.0×10^{-2}，それ以外の関数については 1.0×10^{-7} とし，目的関数値が許容誤差以内の解が見つかったときに試行を打ち切る．

5.2 実験結果

表2に実験結果を示す．各関数について，近似解を見つけるために必要な関数評価回数の平均値および標準偏差を上段に，SDEの平均値に対する比率を下段の括弧内に示した．なお，最良の結果は太字で示した．

表 2: 実験結果

	SDE	CDE	RIDE
f_1	74077.8 ± 1122.4	72487.5 ± 1173.9	**37240.4 ± 925.0**
	(1.000)	(0.979)	**(0.503)**
f_2	104488.9 ± 943.8	103042.5 ± 1074.8	**61856.6 ± 1309.8**
	(1.000)	(0.986)	**(0.592)**
f_3	478759.6 ± 9712.3	474079.7 ± 8517.1	**108957.7 ± 3107.4**
	(1.000)	(0.990)	**(0.228)**
f_4	516741.9 ± 6518.7	516057.8 ± 6973.5	**126985.2 ± 3008.5**
	(1.000)	(0.999)	**(0.246)**
f_5	225852.0 ± 4456.2	216904.8 ± 4116.9	**196354.2 ± 8873.7**
	(1.000)	(0.960)	**(0.869)**
f_6	29307.4 ± 916.0	29126.7 ± 822.4	**14259.0 ± 796.6**
	(1.000)	(0.994)	**(0.487)**
f_7	306553.6 ± 49304.1	281086.1 ± 45917.1	**36215.1 ± 17642.3**
	(1.000)	(0.917)	**(0.118)**
f_8	89118.6 ± 1906.1	87545.3 ± 1730.2	**81902.8 ± 3470.6**
	(1.000)	(0.982)	**(0.919)**
f_9	161042.2 ± 4386.0	**159441.5 ± 4542.2**	221820.5 ± 9815.3
	(1.000)	**(0.990)**	(1.377)
f_{10}	111665.3 ± 1315.7	109890.0 ± 1487.0	**56898.7 ± 1111.1**
	(1.000)	(0.984)	**(0.510)**
f_{11}	82194.5 ± 4992.5	82564.4 ± 6351.8	**43910.4 ± 1298.2**
	(1.000)	(1.005)	**(0.534)**
f_{12}	66451.2 ± 1278.8	65036.7 ± 960.7	**36106.5 ± 1201.2**
	(1.000)	(0.979)	**(0.543)**
f_{13}	71270.6 ± 1240.0	69919.4 ± 932.6	**38248.5 ± 1085.0**
	(1.000)	(0.981)	**(0.537)**

まず，連続世代モデルを使用するCDEと標準的な離散世代モデルを使用するSDEを比較する．f_{11} 以外の全ての問題においてCDEはSDEよりも速く近似解を発見している．したがって，CDEの方が効率的なアルゴリズムであると考えられるが，その差は僅かであり，大きな差はないと考えられる．次に，SDEとRIDEを比較する．RIDEは，SDEと比較して f_7 では約10倍の速さで，f_3, f_4 では約5倍の速さで，$f_1, f_6, f_{10}, f_{11}, f_{12}, f_{13}$ では約2倍の速さで，f_2 では約1.5倍の速さで，f_5, f_8 では約10%高速に近似解を発見している．しかし，f_9 については，約40%効率が低下した．したがって，RIDEは単峰性関数については非常に有効であるといえる．多峰性関数についても多くの場合有効性が認められるが，逆に探索効率を低下させる場合もあることが分かった．

探索経過を示すために，関数評価回数に対する目的関数値の変化を図7に示す．図には，関数評価回数150,000回までの変化を示した．f_9 以外の関数については，RIDEが最も速く目的関数値が減少していることが分かる．

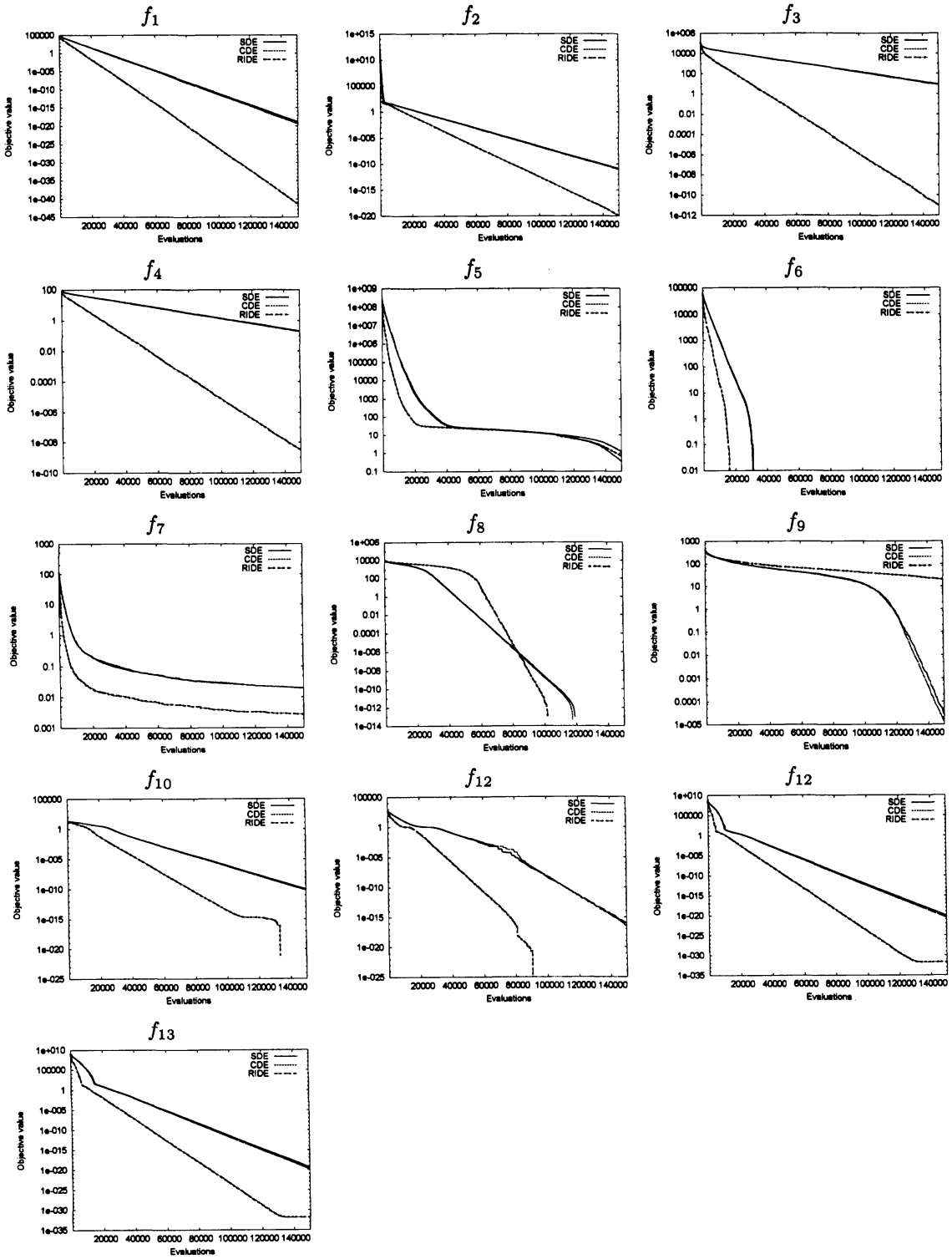
f_1
f_2
f_3
f_4
f_5
f_6
f_7
f_8
f_9
f_{10}
f_{12}
f_{12}
f_{13}
SDE
CDE
RIDE
Objective value
Evaluations

図 7: 目的関数値の変化

6 終わりに

DEは単純で高速な最適化アルゴリズムであり，多峰性問題に対しても頑健なアルゴリズムである．しかし，変数間の依存関係が強い問題に対して性能が低下するという問題があった．本研究ではこの問題に対応するために，回転不変性を有する交叉を提案し，その交叉を有効に用いたRIDEを提案した．様々の問題に適用することにより，RIDEは従来のDEよりも高速に近似解を発見できることを示した．

今後は，RIDEで効率が低下した関数に対する対策を考えるとともに，より多峰性の関数に対応できるように種分化などにより探索領域を棲み分けるアルゴリズムについて研究する予定である．

謝辞 この研究の一部は，日本学術振興会科学研究費補助金 基盤研究 (c) (No. 20500138,22510166) および広島市立大学特定研究費（一般研究）7111 の援助を受けた．

参考文献

[1] Storn, R. and Price, K.: Minimizing the Real Functions of the ICEC'96 Contest by Differential Evolution, in *Proc. of the International Conference on Evolutionary Computation*, pp. 842–844 (1996).

[2] Storn, R. and Price, K.: Differential evolution – A simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization*, Vol. 11, pp. 341–359 (1997).

[3] 高濱徹行, 阪井節子, 原章：低精度の近似モデルを用いた比較推定法による Differential Evolution における関数評価回数の削減, 電子情報通信学会論文誌, Vol. J91-D, No. 5, pp. 1275–1285 (2008).

[4] Takahama, T. and Sakai, S.: Fast and Stable Constrained Optimization by the ε Constrained Differential Evolution, *Pacific Journal of Optimization*, Vol. 5, No. 2, pp. 261–282 (2009).

[5] Price, K. V., Storn, R. M. and Lampinen, J. A.: *Differential Evolution: A Practical Approach to Global Optimization*, Springer (2005).

[6] Chakraborty, U. K. ed.: *Advances in Differential Evolution*, Springer (2008).

[7] 高濱徹行, 阪井節子：低精度近似モデルを利用した ε 制約 Differential Evolution による効率的な制約付き最適化, 人工知能学会論文誌, Vol. 24, No. 1, pp. 34–45 (2009).

[8] Takahama, T. and Sakai, S.: Efficient Constrained Optimization by the ε Constrained Adaptive Differential Evolution, in *Proc. of the 2010 IEEE Congress on Evolutionary Computation*, pp. 2052–2059 (2010).

[9] Takahama, T. and Sakai, S.: Constrained Optimization by the Epsilon Constrained Differential Evolution with an Archive and Gradient-Based Mutation, in *Proc. of the 2010 IEEE Congress on Evolutionary Computation*, pp. 1680–1688 (2010).

[10] Yao, X., Liu, Y., Liang, K.-H. and Lin, G.: Fast evolutionary algorithms, in Ghosh, A. and Tsutsui, S. eds., *Advances in evolutionary computing: theory and applications*, pp. 45–94, Springer-Verlag New York, Inc., New York, NY, USA (2003).