数理解析研究所講究録
第1912巻 2014年 121-130

比較推定法におけるGradient Boostingの利用に関する一考察

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)
Graduate School of Information Sciences, 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] など多くのアルゴリズムが提案されている．EAは，最適化の対象である目的関数の値だけを利用して解を求めることができる直接探索法であり，アルゴリズムの実装が容易であることから，様々な最適化問題を解くために利用されている．

しかし，EAは確率的な多点探索を行うため，比較的局所解に陥りにくいが，関数評価回数が多くなりがちである．近年，最適化問題が大規模化し，目的関数の評価コストが増大してきているため，目的関数の評価回数の削減は大きな課題となってきている．本研究の目的は，制約なし最適化問題において，精度の高い解を少ない関数評価回数で探索する効率的なアルゴズムを提案することである．

関数評価回数を削減するための代表的な方法として，関数の近似モデルを構成し，近似値を利用して探索を行う研究が進化的アルゴリズムを中心に活発に行われている [3, 4, 5, 6, 7, 8, 9, 10, 11]．精度の高い近似モデルを構築できれば，関数評価回数を大きく削減できるが，汎化性能の高い近似モデルを学習することは非常に困難であり，訓練データに対して近似精度の高い近似モデルの汎化性能が高いとは限らないこともよく知られている．また，試行錯誤的に学習パラメータを調整しなければならないという問題や学習のためにかなりの時間量および記憶量を要するという問題があるため，比較的コストの低い関数の最適化に利用すると学習コストが評価回数の削減効果を超えてしまうこともある．

これに対して，低精度の近似モデル (approximation model with low accuracy) を用いて，最適化において関数評価回数を削減する比較推定法 (estimated comparison method) が提案されている [12, 13, 14, 15, 16, 17, 18]．低精度の近似モデルは，真の関数値に対する近似値 (推定値) の誤差は大きいが，近傍の点の大小関係をある程度表現しているため，誤差を考慮して近傍の点の比較に利用することは可能である．比較推定法では，ある解と新しい解のそれぞれの推定値により大小関係を求め，解が改良されていると推定される場合は目的関数を評価し真の関数値を求めるが，そうでない場合には評価を省略することにより，目的関数の評価回数を削減する．比較推定法は，学習が不要な低精度の近似モデルを有効に利用することにより，学習パラメータの調整が不要となり，広範囲の問題に簡単に利用できる．さらに，学習のための時間が不要となり，評価コストがそれほど高くない目的関数の最適化問題にも適用可能な方法である．

本研究では，低精度近似モデルであるポテンシャルモデルを弱い学習器とし，勾配ブースティング [19] によって近似精度を向上させることにより，集団的降下法の探索効率をさらに向上する方法について検討する．本手法をDEに適用し，ベンチマーク関数を最適化することにより本手法の有効性を示す．

以下，**2.** で比較推定法とポテンシャルモデルを簡潔に紹介する．**3.** で，勾配ブースティングについて説明する．**4.** で，本手法のアルゴリズムを説明する．**5.** で他の方法と比較した性能を示す．**6.** はまとめである．

2 比較推定法

2.1 集団的降下法と比較推定法

DEでは，個体により集団を形成し，各個体から交叉と突然変異により新しい個体を生成し，新しい個体が良ければ元の個体と置換する．このようにDEでは，集団の各要素が降下法により個別に最適化されるため，集団が急速に特定の解に集中することが少なく，網羅性の高い探索が行われる．また，各要素が新しい解を生成する際に集団の情報を利用するため，一点による降下法と比較すると局所解に陥りにくい効率の良い探索が行われる．本研究では，一般にこのようなアルゴリズムを集団的降下法と呼ぶ．

比較推定法では，新しい解と古い解の関数値を推定し，新しい解が良いと推定されれば，評価・更新を行い，そうでなければ，新しい解を評価することなく拒否することにより関数評価回数を削減する．比較推定法を導入した集団的降下法のアルゴリズムは一般に以下のように記述できる．アルゴリズム中の`EstimatedBetter(`$\boldsymbol{x}$`,` $\boldsymbol{y}$`)`は，推定値を用いた比較である比較推定を実現する関数であり，$\boldsymbol{x}$が$\boldsymbol{y}$より良いかどうかを，関数値を使用せず，推定値に基づいて判定する．

1. 初期化: 集団に属する解をランダムに発生する
2. 評価: 全ての解を評価する
3. 終了判定: 終了条件を満足すれば終了する
4. 各解に対して，
   (a) 生成: 各解と集団の情報に基づき新しい解を生成する
   `if EstimatedBetter(`新しい解，古い解`) then`
   (b) 評価: 新しい解を評価する
   (c) 更新: 新しい解が古い解より良ければ，古い解を新しい解で置換する
   `endif`
5. 3. へ戻る

比較推定が常に真を返せば，通常の集団的降下法となり，関数評価回数は減少しない．比較推定が常に正しい判定を行うという理想的な場合には，真の関数値を求める評価回数は大幅に削減される．

2.2 ポテンシャルモデル

ポテンシャルエネルギーとは，物体がある位置に存在することで物体にたくわえられる位置エネルギーである．例えば，質量mの物体が存在すれば，その周りには重力ポテンシャルU_gが発生し，その物体から距離r離れた質量m'の物体には引力F_gが働く．

$$U_g = -G\frac{m}{r}, \quad F_g = G\frac{mm'}{r^2} \tag{1}$$

ここで，Gは万有引力定数である．

ポテンシャルモデルでは，ある解$\boldsymbol{x}$が存在することにより，その解からr離れた位置に，目的ポテンシャルU_o (potential for objective) と混雑ポテンシャルU_c (potential for congestion) が発生すると仮定する．

$$U_o = \frac{f(\boldsymbol{x})}{r^p}, \quad U_c = \frac{1}{r^p} \tag{2}$$

ここでは，簡単のため，比例係数を 1 とした．また，距離のベキ乗数 p は正数であり，線形性の強い関数の近似では 1，そうでない関数では 2 とするのが通常である．

解集合 $X = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \cdots, \boldsymbol{x}_N\}$ が存在し，関数値 $f(\boldsymbol{x}_i), i = 1, 2, \cdots, N$ が既知であるとする．集団的降下法では，解集合 X 中の古い解 $\boldsymbol{x}_i$ から新しい解 $\boldsymbol{x}_i'$ を生成する．このとき，$\boldsymbol{x}_i$ と $\boldsymbol{x}_i'$ におけるポテンシャルは，以下のように古い解を除いた解集合によるポテンシャルで求める．

$$U_o(\boldsymbol{x}_i^{(\prime)}) = \sum_{j \neq i} \frac{f(\boldsymbol{x}_j)}{d(\boldsymbol{x}_j, \boldsymbol{x}_i^{(\prime)})^p}, \quad U_c(\boldsymbol{x}_i^{(\prime)}) = \sum_{j \neq i} \frac{1}{d(\boldsymbol{x}_j, \boldsymbol{x}_i^{(\prime)})^p} \tag{3}$$

ただし，$d(\boldsymbol{x}, \boldsymbol{y})$ は点 $\boldsymbol{x}$ と点 $\boldsymbol{y}$ 間の距離である．

点 $\boldsymbol{x}_i^{(\prime)}$ における関数の推定値 $\hat{f}(\boldsymbol{x}_i^{(\prime)})$ は，これらの商で与えられる．

$$\hat{f}(\boldsymbol{x}_i^{(\prime)}) \quad = \quad U_o(\boldsymbol{x}_i^{(\prime)}) / U_c(\boldsymbol{x}_i^{(\prime)}) \tag{4}$$

関数値の推定に $\boldsymbol{x}_i$ 自体を含めると，$\boldsymbol{x}_i$ における推定値は真値に一致する．このとき推定値は $\boldsymbol{x}_i$ の近傍で急激に真値に近づくため，推定値の変化の滑らかさを壊し，関数全体の概形の近似が困難になることがある．また，$\boldsymbol{x}_i$ における推定値の精度は高くなるが $\boldsymbol{x}_i'$ における推定値の精度は低いため，比較の際の精度が却って下がるという問題もある．このため，比較推定法では $\boldsymbol{x}_i$ を除いたポテンシャルを用いる．

2.3 比較推定

関数 `EstimatedBetter` は推定誤差を考慮するため，推定の困難さに応じてある程度の余裕を与える必要があり，以下のように定義する [15].

$$\mathrm{EstimatedBetter}(\boldsymbol{x}_i', \boldsymbol{x}_i) \;\Leftrightarrow\; \hat{f}(\boldsymbol{x}_i') \leq \hat{f}(\boldsymbol{x}_i) + \delta\, \mathcal{DS}(X) \tag{5}$$

ただし，$\mathcal{DS}(X)$ は解集合 X における推定の困難さを示す推定困難度であり，$\delta \geq 0$ は誤差の余裕を調整するパラメータである．

推定する関数面が複雑ならば，関数値の推定は困難となり，余裕を大きく取る必要がある．関数面の複雑さは探索の進展，すなわち解集合 X の状態により変化する．一般に，探索の初期段階では大域的な推定が必要であるため推定困難度は高いが，探索が進むにつれて解集合は 1 点に収束し，推定困難度は低下すると考えられる．通常，推定困難度として，解集合 X における関数値の標準偏差を用いる．

$$\mathcal{DS}(X) = \sigma(X) = \sqrt{\frac{1}{N} \sum_{\boldsymbol{x} \in X} (f(\boldsymbol{x}) - \bar{f})^2}, \quad \bar{f} = \frac{1}{N} \sum_{\boldsymbol{x} \in X} f(\boldsymbol{x}) \tag{6}$$

標準偏差が大きい解集合は関数値の変化が大きく，標準偏差が小さい解集合は関数値の変化が小さいため，標準偏差によって関数面の複雑さを表現できると考えられる．

また，余裕パラメータ δ については，δ が 0 の場合は，推定値に基づく比較となり，多数のベクトルを拒否するため，良いベクトルも拒否してしまう可能性が高くなる．δ を大きくすると，推定値が少し悪いベクトルも受け入れるため，良いベクトルを拒否する可能性は低くなるが，逆に，拒否する数が減少する．したがって，δ は適当な大きさにとる必要がある．

さらに，各世代において集団に存在する解の次元毎の幅，すなわち最大値と最小値の差により正規化した距離を用いる．これは，探索の進行により解の存在する探索領域が扁平的になることがあり，このような場合にも探索領域における関数形状をより的確に表現するためである．

$$d(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_j \frac{(x_j - y_j)^2}{(\max_i x_{ij} - \min_i x_{ij})^2}} \tag{7}$$

3 勾配ブースティングによる関数近似

3.1 勾配ブースティング

教師付き学習とは，入出力関係を推定するために，訓練データを使用して入出力関係を近似する関数を求めることである．すなわち，入力 $\boldsymbol{x} = (x_1\, x_2 \cdots x_n)$ から出力 y への写像を表現する関数 $F(\cdot)$ を，訓練データ $\{(\boldsymbol{x}_i, y_i)|i = 1, 2, \cdots, N\}$ を使用して求めることになる．ただし，n は入力データの次元数，N は訓練データの数である．

教師付き学習法の一つにブースティング (boosting) がある．ブースティングは，教師付き学習を行うためのメタアルゴリズムであり，弱い学習器 (weak learner) あるいは基本学習器 (base learner) を複数使用することにより強い学習器を生成する手法である．弱い学習器は，何らかの学習法により，真の入出力関係をある程度満足するように学習したものである．ブースティングでは，訓練データの分布に従って弱い分類器を学習し，それを最終的な強い分類器の一部として追加することを繰り返す．弱い分類器を追加する際，学習器の正確さを反映するように重みを付けることが多い．弱い学習器の追加に応じて，誤差の大きいデータの重みを増加させ，小さいデータの重みを減少させるように，データの重み付けが更新される．これにより，新たに追加される弱い学習器は以前の学習器において誤差の大きいデータを重視することになる．

ブースティングの一つとして，Friedman が勾配ブースディング (gradient boosting)[19] を提案している．勾配ブースティングでは，真の関数 $F(\boldsymbol{x})$ に対する損失の期待値を最小にする近似関数 $F^*(\boldsymbol{x})$ を学習する．

$$F^*(\boldsymbol{x}) = \arg\min_{F(\boldsymbol{x})} E_{y,\boldsymbol{x}}\Psi(y, F(\boldsymbol{x})) \tag{8}$$

ただし，E は期待値，$\Psi(\cdot,\cdot)$ は損失関数である．ブースティングにより，基本学習器の重み付き和を用いて真の関数を近似する．

$$F(\boldsymbol{x}) = \sum_{m=0}^{M} \beta_m h(\boldsymbol{x}; \boldsymbol{a}_m) \tag{9}$$

ただし，関数 $h(\boldsymbol{x}; \boldsymbol{a})$ はパラメータ $\boldsymbol{a}$ を持つ $\boldsymbol{x}$ の関数である基本学習器，β_m は基本学習器の重み，M は基本学習器の数である．初期の推定 $F_0(\boldsymbol{x})$ から，$m = 1, 2, \cdots, M$ に対して，次のように学習器を構成する．

$$(\beta_m, \boldsymbol{a}_m) = \arg\min_{\beta,\boldsymbol{a}} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\boldsymbol{x}_i) + \beta h(\boldsymbol{x}_i; \boldsymbol{a})) \tag{10}$$

$$F_m(\boldsymbol{x}) = F_{m-1}(\boldsymbol{x}) + \beta_m h(\boldsymbol{x}; \boldsymbol{a}_m) \tag{11}$$

勾配ブースティングでは，任意の損失関数について，式 (10) を以下の 2 段階で近似的に解く．

1. 最小二乗法による基本学習器のパラメータ決定

$$\boldsymbol{a}_m = \arg\min_{\boldsymbol{a},\rho} \sum_{i=1}^{N} (\tilde{y}_{im} - \rho h(\boldsymbol{x}_i; \boldsymbol{a}))^2 \tag{12}$$

$$\tilde{y}_{im} = -\left[\frac{\partial \Psi(y_i, F(\boldsymbol{x}_i))}{\partial F(\boldsymbol{x}_i)}\right]_{F(\boldsymbol{x})=F_{m-1}(\boldsymbol{x})} \tag{13}$$

ここで，$\tilde{y}_{im}$ は疑似残差 (pseudo residual) と呼ばれ，全訓練データ点の関数値で構成されるベクトル空間における損失関数の最急降下方向を表現する．この最急降下方向を基本学習器によって近似することになる．

2. 1変数最適化による重み決定

$$\beta_m = \arg\min_{\beta} \sum_{i=1}^{N} \Psi(y_i, F_{m-1}(\boldsymbol{x}_i) + \beta h(\boldsymbol{x}_i; \boldsymbol{a}_m)) \tag{14}$$

これにより，最急降下方向への最適なステップ幅が基本学習器の重みとなる．

3.2 2乗誤差回帰

損失関数を2乗誤差，すなわち $\Psi(y, F(\boldsymbol{x})) = \frac{1}{2}(y - F(\boldsymbol{x}))^2$ とする場合を考えると次のようになる．

$$\tilde{y}_{im} = y_i - F_{m-1}(\boldsymbol{x}_i) \tag{15}$$

$$\Psi(y_i, F_{m-1}(\boldsymbol{x}_i) + \beta h(\boldsymbol{x}_i; \boldsymbol{a}_m)) = \frac{1}{2}(\tilde{y}_{im} - \beta h(\boldsymbol{x}_i; \boldsymbol{a}_m))^2 \tag{16}$$

固定的な学習器，すなわち，基本学習器のパラメータ $\boldsymbol{a}$ が存在しないか固定の場合，β_m は下記のように決定することができる．

$$\frac{\partial \sum_{i=1}^{N} (\tilde{y}_{im} - \beta_m h(\boldsymbol{x}_i; \boldsymbol{a}))^2}{\partial \beta_m} = 0 \tag{17}$$

$$\sum_{i=1}^{N} (\tilde{y}_{im} - \beta_m h(\boldsymbol{x}_i; \boldsymbol{a})) h(\boldsymbol{x}_i; \boldsymbol{a}) = 0 \tag{18}$$

$$\beta_m = \frac{\sum_{i=1}^{N} \tilde{y}_{im} h(\boldsymbol{x}_i; \boldsymbol{a})}{\sum_{i=1}^{N} h(\boldsymbol{x}_i; \boldsymbol{a})^2} \tag{19}$$

4 比較推定法を導入したDE

Differential Evolution にポテンシャルモデルを学習器とする勾配ブースティングに基づく比較推定法を適用したアルゴリズムである DEgb を提案する．

4.1 Differential Evolution

Differential evolution (DE) は ES の一つであり，Storn & Price [1, 2] によって提案された．DE は確率的な直接探索法であり，解集団を用いた多点探索を行う．DE は非線形問題，微分不可能な問題，非凸問題，多峰性問題などの様々な最適化問題に適用されてきており，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．

DE には幾つかの形式が提案されており，DE/best/1/ bin や DE/rand/1/exp などがよく知られている．これらは，DE/*base*/*num*/*cross* という記法で表現される．"*base*" は基本ベクトルとなる親の選択方法を指定する．例えば，rand は基本ベクトルのための親を集団からランダムに選択し，best は集団の最良個体を選択する．"*num*" は基本ベクトルを変異させるための差分ベクトルの個数を指定する．"*cross*" は子を生成するために使用する交叉方法を指定する．例えば，bin は一定の確率で遺伝子を交換する交叉 (binomial crossover) を用い，exp は，指数関数的に減少する確率で遺伝子を交換する交叉 (exponential crossover) を用いる．

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．

4.2 DEgb

DE/rand/1/exp に基づく比較推定法のアルゴリズムを以下に示す.

```
DE with estimated comparison/rand/1/exp()
{
  P=Generate N individuals {x_i} randomly;
  Evaluate x_i, i = 1,2,...,N;
  for(t=1; t < T_max; t++) {
    σ=standard deviation of estimated error;
    for(i=1; i ≤ N; i++) {
    // 探索点の生成
      (p1,p2,p3)=select randomly in [1,N]\{i}
          s.t. p_j ≠ p_k(j,k = 1,2,3, j ≠ k);
      x'_i=x_i ∈ P;
      j=select randomly from [1,n];
      k=1;
      do {
        x'_ij=x_p1,j+F(x_p2,j − x_p3,j);
        j=(j + 1)%n;
        k++;
      } while(k ≤ n && u(0,1) < CR);
    // 生存者選択
      updated=0;
      if(EstimatedBetter(x'_i, x_i)) {
        if(f(x'_i) ≤ f(x_i)) updated=1;
      }
      if(updated) z_i=x'_i; else z_i=x_i;
    }
    P={z_i};
  } }
```

5 実験

5.1 テスト問題

本実験では，非線形最適化問題 f_1 から f_6 として，単峰性関数である Sphere, Schwefel 2.22, Schwefel 1.2, 多峰性関数である Rastrigin, Ackley, Griewank を用いる [20]. 表 1 に，関数定義とその初期化領域を示す. なお，D は次元数を表している.

表 1: テスト関数 (sphere, Schwefel 2.22, Schwefel 1.2, Rastrigin, Ackley, and Griewank [20])

Test functions	Bound constraints
$f_1(\boldsymbol{x}) = \sum_{i=1}^{D} x_i^2$	$[-100, 100]^D$
$f_2(\boldsymbol{x}) = \sum_{i=1}^{D} \lvert x_i \rvert + \prod_{i=1}^{D} \lvert x_i \rvert$	$[-10, 10]^D$
$f_3(\boldsymbol{x}) = \sum_{i=1}^{D} \left(\sum_{j=1}^{i} x_j \right)^2$	$[-100, 100]^D$
$f_4(\boldsymbol{x}) = \max_i \{ \lvert x_i \rvert \}$	$[-100, 100]^D$
$f_5(\boldsymbol{x}) = \sum_{i=1}^{D-1} \left[100(x_{i+1} - x_i^2)^2 + (x_i - 1)^2 \right]$	$[-30, 30]^D$
$f_6(\boldsymbol{x}) = \sum_{i=1}^{D} \lfloor x_i + 0.5 \rfloor^2$	$[-100, 100]^D$

5.2 実験条件

次元数 $n = 30$ に設定し，f_1 から f_6 の関数を最適化する．rand 戦略を用いる標準の DE，ポテンシャルモデルによる比較推定法を用いた DEpm，ポテンシャルモデルを基本学習器とする勾配ブースティングを用いた比較推定法を用いた DEgb の 3 つのアルゴリズムの性能を比較する．DE の設定は，個体数 $N = 50$, F と CR の組合せを $(0.7, 0.9)$ とした．比較推定法の設定は，距離の次数 $p = 2$，余裕パラメータ $\delta = 0.01$ とした．勾配ブースティングの設定は，基本学習器の数 $M = 1, 2, 5, 10$ と変化させ，$\beta_m = 1$ に固定した．各関数について 30 回の試行を行い，結果を考察する．個体数は $2n$ から $10n$ 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.

5.3 実験結果

実験結果を表2に示す．許容精度を満足した回数を "< $1e-7$"，実際に関数を評価した回数とその標準偏差を eval，そのうち子ベクトルが良かった回数を succ，悪かった回数を fail, 成功率を rate に示した．

単峰性関数 f_1 から f_3 では，全ての方法が全試行で最適解を発見した．最適解を発見するまでの評価回数については，DEgb($M=10$)，DEgb($M=5$)，DEgb($M=2$)，DEpm，DEgb($M=1$)，DE の順に少なくなっている．DEgb($M=10$) では，50%以上の関数評価を削減することができた．DEpm と DEgb($M=1$) はいずれもポテンシャルモデルに基づく推論を一度行うが，DEpm の方が効率が良い．DEpm は，ある世代の処理において，個体が更新されれば，更新後のベクトルとその値を用いて推論を行う．しかし，DEgb では世代の処理を始める前に，前世代のベクトルとその値を用いて基本学習器を構成するため，個体が更新されてもそのベクトルを使用することができない．このため，最新の情報を利用できる DEpm の方が効率が良くなったと考えられる．

多峰性関数 f_4 から f_6 では，DE，DEpm，DEgb($M=1$) は全ての試行で最適解を発見した．しかし，DEgb($M=2$) は f_4 で1試行 f_6 で4試行，DEgb($M=5$) では f_4 で1試行 f_6 で1試行，DEgb($M=10$) では f_6 で4試行，最適解の発見に失敗した．最適解を発見するまでの評価回数については，単峰性関数と同様に DEgb($M=10$)，DEgb($M=5$)，DEgb($M=2$)，DEpm，DEgb($M=1$)，DE の順に少なくなっている．ただし，f_4 については DEgb($M=1$) が DE より少ない．

また，実際に関数評価が行われた際に子ベクトルが親ベクトルより良好であった割合は，比較推定法によって大きく向上したと言える．

各問題における最良個体の関数値について，その平均値の変化を図3に示す．横軸は世代数，縦軸が関数値である．

6 おわりに

比較推定法では，低精度近似モデルを用い，良くないと推定された探索点の評価を省略することにより関数評価回数を削減し，探索効率を向上させる．本研究では，低精度近似モデルの精度を向上させるため，勾配ブースティングを適用する方法を提案した．代表的な単峰性，多峰性関数の最適化問題を解くことにより，勾配ブースティングによって探索効率が大きく向上することを示した．しかし，多峰性関数では，勾配ブースティングにおける学習器の数を増加させると探索効率が向上するが，探索に失敗する傾向があることが分かった．

今後は，このような問題が起こる理由について調査を行うとともに，余裕パラメータ δ の制御方法，学習器の数の選択方法について研究する予定である．

謝辞 この研究の一部は，本研究は JSPS 科研費 22510166, 24500177 および広島市立大学特定研究費（一般研究）の援助を受けた．

参考文献

[1] R. Storn and K. Price: "Minimizing the real functions of the ICEC'96 contest by differential evolution", Proc. of the International Conference on Evolutionary Computation, pp. 842–844 (1996).

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

表 2: 実験結果

function	algorithm	$< 1e^{-7}$	eval	ratio	success	fail	rate(%)
f_1	DE	30	73929.93 ± 1266.61	1	15010.63	58868.30	20.32
	DEpm	30	46152.27 ± 886.33	0.62	14713.47	31387.80	31.92
	DEgb($M = 1$)	30	46534.87 ± 955.59	0.63	14759.77	31724.10	31.75
	DEgb($M = 2$)	30	40296.67 ± 901.75	0.55	14463.27	25782.40	35.94
	DEgb($M = 5$)	30	35098.63 ± 894.01	0.47	13986.20	21061.43	39.91
	DEgb($M = 10$)	30	32149.93 ± 662.08	0.43	13744.73	18354.20	42.82
f_2	DE	30	104971.17 ± 1224.02	1	20471.17	84449.00	19.51
	DEpm	30	68199.80 ± 1180.19	0.65	20408.00	47740.80	29.95
	DEgb($M = 1$)	30	68447.37 ± 1025.92	0.65	20404.27	47992.10	29.83
	DEgb($M = 2$)	30	60335.63 ± 879.03	0.57	20102.47	40182.17	33.35
	DEgb($M = 5$)	30	53747.50 ± 874.39	0.51	19618.93	34077.57	36.54
	DEgb($M = 10$)	30	50043.80 ± 1384.08	0.48	19344.63	30648.17	38.69
f_3	DE	30	489104.70 ± 10218.82	1	17637.27	471416.43	3.61
	DEpm	30	307056.00 ± 8691.40	0.63	18025.70	288979.30	5.87
	DEgb($M = 1$)	30	306075.27 ± 7098.93	0.63	18063.97	287960.30	5.90
	DEgb($M = 2$)	30	266696.53 ± 7781.77	0.55	17875.17	248770.37	6.70
	DEgb($M = 5$)	30	229378.37 ± 7291.53	0.47	17137.13	212190.23	7.47
	DEgb($M = 10$)	30	213001.50 ± 6443.39	0.44	16696.60	196253.90	7.84
f_4	DE	30	160307.97 ± 4900.91	1	14886.90	145370.07	9.29
	DEpm	30	96893.07 ± 2720.62	0.60	14615.13	82226.93	15.09
	DEgb($M = 1$)	30	98205.00 ± 3942.98	0.61	14802.67	83351.33	15.08
	DEgb($M = 2$)	30	87384.27 ± 3665.27	0.55	14495.83	72837.43	16.60
	DEgb($M = 5$)	30	79453.03 ± 4502.92	0.50	14138.77	65263.27	17.81
	DEgb($M = 10$)	30	77058.60 ± 3088.06	0.48	13928.27	63079.33	18.09
f_5	DE	30	111605.03 ± 1058.69	1	22161.57	89392.47	19.87
	DEpm	30	69314.20 ± 1181.92	0.62	21795.67	47467.53	31.47
	DEgb($M = 1$)	30	69858.33 ± 1186.35	0.63	21880.80	47926.53	31.34
	DEgb($M = 2$)	30	60972.13 ± 912.31	0.55	21436.93	39484.20	35.19
	DEgb($M = 5$)	30	52677.87 ± 771.71	0.47	20721.37	31905.50	39.37
	DEgb($M = 10$)	30	48344.30 ± 993.75	0.43	20276.63	28016.67	41.99
f_6	DE	29	81966.34 ± 4805.21	1	15993.66	65921.69	19.52
	DEpm	29	54513.03 ± 5356.11	0.67	15701.45	38760.59	28.83
	DEgb($M = 1$)	28	54923.68 ± 5202.95	0.67	15774.57	39098.11	28.75
	DEgb($M = 2$)	30	46260.53 ± 4296.72	0.56	15223.73	30985.80	32.95
	DEgb($M = 5$)	28	40604.11 ± 3867.33	0.50	14729.21	25823.89	36.32
	DEgb($M = 10$)	27	38155.00 ± 4537.04	0.47	14544.89	23559.11	38.17

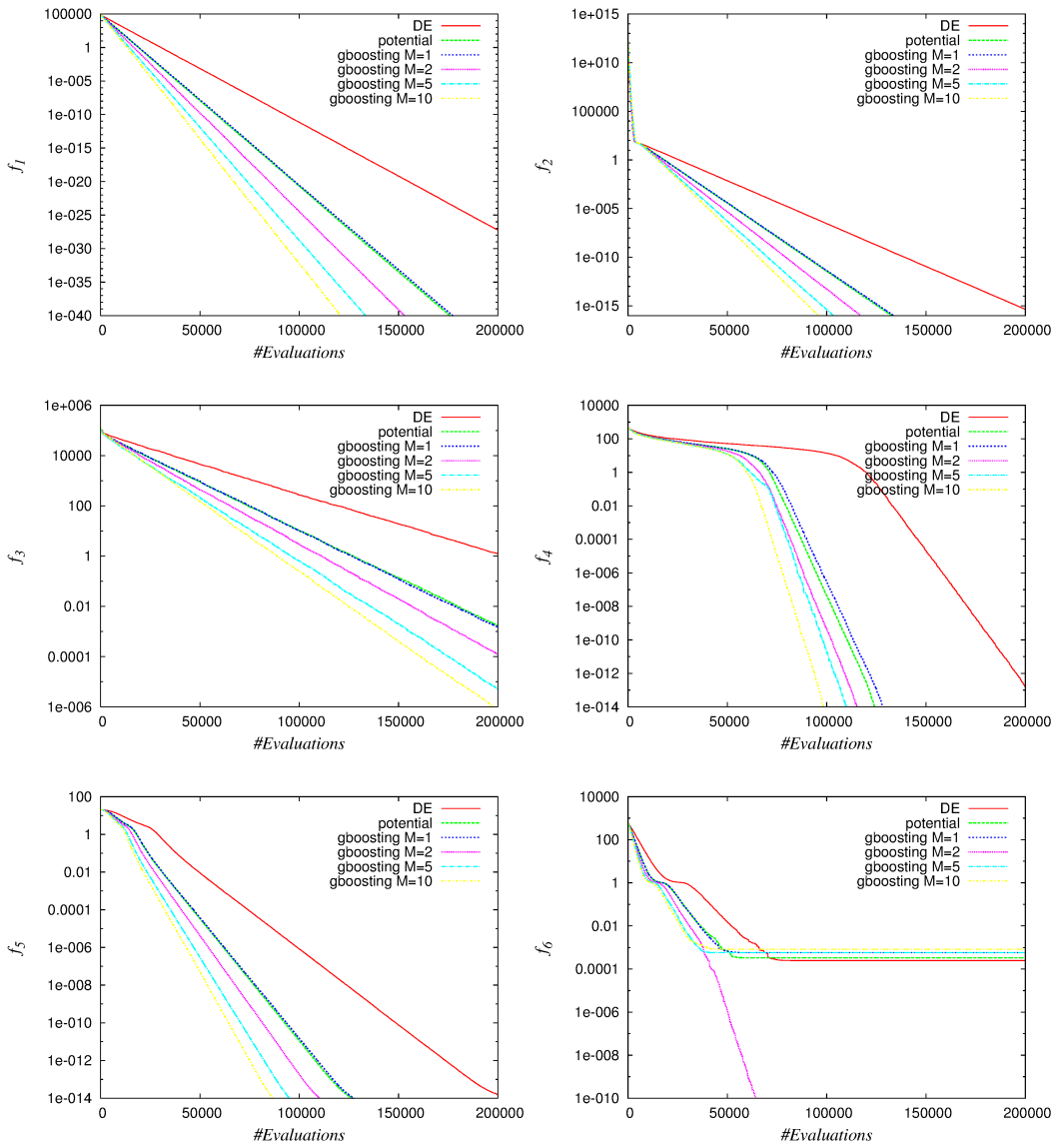

図 3: f_1〜f_6 における関数値の変化

[3] Y. Jin, M. Olhofer and B. Sendhoff: "On evolutionary optimization with approximate fitness functions", Proc. of the Genetic and Evolutionary Computation Conference, Morgan Kaufmann, pp. 786–792 (2000).

[4] 花木, 橋山, 大熊："適応度の推論による進化的アルゴリズムの計算時間の短縮", 電気学会論文誌 C, **120**, 1, pp. 123–129 (2000).

[5] 中山, 荒木, 佐々木："RBF ネットワークと遺伝的アルゴリズムによる未知目的の最適化", システム制御情報学会論文誌, **13**, 3, pp. 152–154 (2000).

[6] Y. Jin, M. Olhofer and B. Sendhoff: "A framework for evolutionary optimization with approximate fitness functions", IEEE Trans. on Evolutionary Computation, **6**, 5, pp. 481–494 (2002).

[7] Y. Jin and B. Sendhoff: "Reducing fitness evaluations using clustering techniques and neural networks ensembles", Genetic and Evolutionary Computation Conference, Vol. 3102 of LNCS, Springer, pp. 688–699 (2004).

[8] 田中, 溝口, 高見："適応度予測型遺伝的アルゴリズムにおける探索性能の向上", 電気学会論文誌 C, **124**, 9, pp. 1853–1860 (2004).

[9] D. Büche, N. N. Schraudolph and P. Koumoutsakos: "Accelerating evolutionary algorithms with gaussian process fitness function models", IEEE Trans. on Systems, Man, and Cybernetics, Part C: Applications and Reviews, **35**, 2, pp. 183–194 (2005).

[10] Y. S. Ong, Z. Zhou and D. Lim: "Curse and blessing of uncertainty in evolutionary algorithm using approximation", Proc. of the 2006 IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, pp. 9833–9840 (2006).

[11] F. G. Guimarães, E. F. Wanner, F. Campelo, R. H. Takahashi, H. Igarashi, D. A. Lowther and J. A. Ramírez: "Local learning and search in memetic algorithms", Proc. of the 2006 IEEE Congress on Evolutionary Computation, Vancouver, BC, Canada, pp. 9841–9848 (2006).

[12] 高濱, 阪井, 原："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, **J91-D**, 5, pp. 1275–1285 (2008).

[13] T. Takahama and S. Sakai: "Reducing function evaluations in differential evolution using rough approximation-based comparison", Proc. of the 2008 IEEE Congress on Evolutionary Computation, pp. 2307–2314 (2008).

[14] T. Takahama and S. Sakai: "Efficient optimization by differential evolution using rough approximation model with adaptive control of error margin", Proc. of the Joint 4th International Conference on Soft Computing and Intelligent Systems and 9th International Symposium on advanced Intelligent Systems, pp. 1412–1417 (2008).

[15] 高濱, 阪井："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, **24**, 1, pp. 34–45 (2009).

[16] T. Takahama and S. Sakai: "A comparative study on kernel smoothers in differential evolution with estimated comparison method for reducing function evaluations", Proc. of the 2009 IEEE Congress on Evolutionary Computation, pp. 1367–1374 (2009).

[17] S. Sakai and T. Takahama: "A parametric study on estimated comparison in differential evolution with rough approximation model", Social Systems Solution by Legal Informatics, Economic Sciences and Computer Sciences (Eds. by M.Kitahara and K.Morioka), Kyushu University Press, Fukuoka, pp. 112–134 (2010).

[18] T. Takahama and S. Sakai: "Reducing function evaluations using adaptively controlled differential evolution with rough approximation model", Computational Intelligence in Expensive Optimization Problems (Eds. by Y. Tenne and C.-K. Goh), Vol. 2 of Adaptation Learning and Optimization, Springer Berlin Heidelberg, pp. 111–129 (2010).

[19] J. H. Friedman: "Greedy function approximation: a gradient boosting machine", Annals of Statistics, pp. 1189–1232 (2001).

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