多峰性最適化問題での局所最適解推定高度化のための補正法 : 局所最適解が2個の場合

余, 俊
九州大学大学院芸術工学府

髙木, 英行
九州大学大学院芸術工学研究院

出版情報：進化計算研究会, pp.92-97, 2015-09-07
バージョン：
権利関係：

多峰性最適化問題での局所最適解推定高度化のための補正法

——局所最適解が 2 個の場合——

余俊 †，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

進化計算の共通特徴は，複数の探索点（個体）がfitnessに基づいて逐次的に最適解に収束することである．多くの場合，親個体から子個体を生成するアルゴリズム開発・改良に注力されているが，進化方向にも有用な情報がある．一親個体から一子個体を生成する多くの進化計算では，親個体から子個体への世代間の移動ベクトル方向，さらには，数世代に渡る進化パスが得られ，最適解に向かう情報を含んだこれら移動ベクトルの向きに着目する進化計算の性能向上法があり得る．

進化計算の逐次探索を行わなくても，親個体群から子個体群への複数の移動ベクトルがあれば，それらの移動ベクトルが向かう収束先が数学的に計算できる[2, 3]．この収束先は最適解近傍であることが期待できるので，エリート個体として探索に利用することで進化計算の探索加速が期待できる．

本論文の目的は，移動ベクトルから収束点を計算するこの手法を，多峰性のタスクに拡張することである．単峰性の場合，移動ベクトルは一つの最適解に向かうので問題はないが，多峰性の場合，移動ベクトルは各局所最適解に向かっているので，異なる収束点に向かう複数の移動ベクトル全部から得られる一つの収束点の信頼性は低くなる．このための解決策は，局所最適解毎に移動ベクトルをクラスタリングしてから収束点計算，すなわち，局所最適解の推定を行うことである．

この目的のために本論文では，第2節で移動ベクトルから収束点を計算する方法を簡単に概観した後，第3節で局所最適解毎に移動ベクトルをクラスタリングする方法を提案する．次に，第4節でこのクラスタリング精度を向上させる4つの改良案を提案し，第5節でこの評価を行う．

2　進化計算の収束点推定法

初めに，基本となる複数個体の世代間の移動ベクトルから収束点を数学的に決定する手法[2, 3]を簡単に説明する．まず記号の説明であるが，Fig. 1の$\boldsymbol{a}_i$と$\boldsymbol{c}_i$はi番目の親個体とその子個体を表す．すると，移動ベクトル$\boldsymbol{b}_i = \boldsymbol{c}_i - \boldsymbol{a}_i$が定義できる．$\boldsymbol{b}_i$の単位ベクトルは$\boldsymbol{b}_{0i} = \boldsymbol{b}_i/||\boldsymbol{b}_i||$（すなわち，$\boldsymbol{b}_{0i}^T\boldsymbol{b}_{0i} = 1$）である．

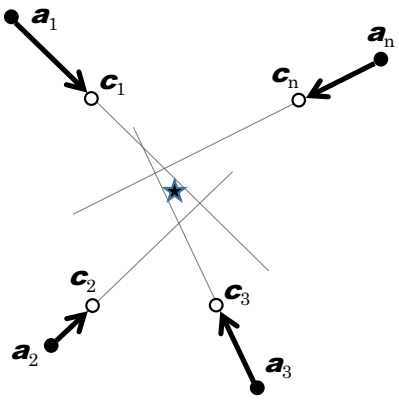


Fig. 1　d次元空間上の親個体 $\boldsymbol{a}_i$ $(1 \le i \le n)$と対応する子個体$\boldsymbol{c}_i$から世代間の移動ベクトル$\boldsymbol{b}_i$ (= $\boldsymbol{c}_i$ - $\boldsymbol{a}_i$)が得られる．★が求める収束点．

これら移動ベクトル$\boldsymbol{b}_i$ $(i = 1, 2, ..., n)$を延長した有向線分$\boldsymbol{a}_i + t_i\boldsymbol{b}_i$ $(t_i \in R)$への距離の総和$\mathrm{J}(\boldsymbol{x}, \{t_i\})$（式(1)）を最少とする点$\boldsymbol{x}$が求める収束点である．

ここで求める収束点$\boldsymbol{x}$から有向線分への最短距離の線分は有向線分と直交するので，直交条件の式(2)を式(1)に代入してt_iを削除する．

$$\mathrm{J}(\boldsymbol{x}, \{t_i\}) = \sum_{i=1}^{n} \|\boldsymbol{a}_i + t_i\boldsymbol{b}_i - \boldsymbol{x}\|^2 \qquad (1)$$

Correction methods for improving estimated convergence points for multi-modal optimization

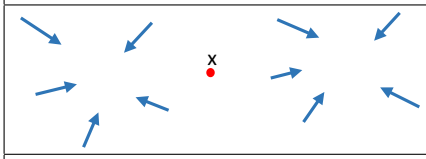
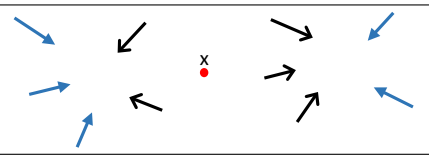
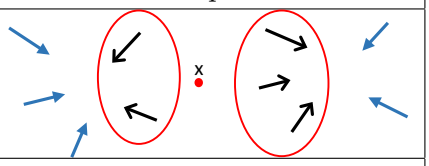
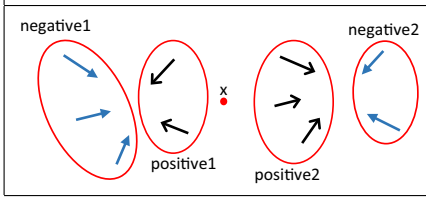
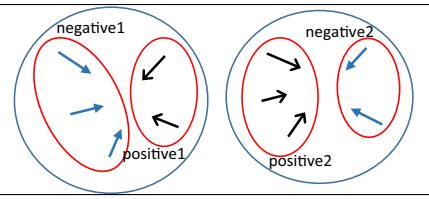
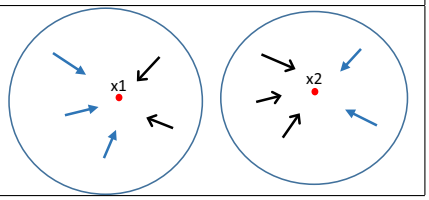
† Jun Yu (yujun901101@yahoo.co.jp)

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

Graduate School of Design, Kyushu University (†)

Faculty of Design, Kyushu University (†)

Table 1　双極性問題の移動ベクトルを各局所最適解に向かう移動ベクトルをクラスタリングする方法.

Step 1	Step 2	Step 3
		
Step 4	**Step 5**	**Step 6**
		

$$\boldsymbol{b}_i^{\mathrm{T}}(\boldsymbol{a}_i + t_i\boldsymbol{b}_i - \boldsymbol{x}) = 0 \quad \text{(直交条件)} \qquad (2)$$

式(1)の総和距離を最小とする$\hat{\boldsymbol{x}}$は，$\boldsymbol{x}$の各要素で偏微分し，0とおけばよい．これより収束点$\hat{\boldsymbol{x}}$は式(3)で与えられる．ここでI_dは単位行列である．

$$\hat{\boldsymbol{x}} = \left\{\sum_{i=1}^{n}\left(I_d - \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^{\mathrm{T}}\right)\right\}^{-1}\left\{\sum_{i=1}^{n}\left(I_d - \boldsymbol{b}_{0i}\boldsymbol{b}_{0i}^{\mathrm{T}}\right)\boldsymbol{a}_i\right\} \qquad (3)$$

なお，文献[2, 3]では，行列演算が容易ではないプログラミング言語利用者のために，式(3)の近似式と反復計算式も提示している．

3　多峰性タスクの局所最適解推定のためのクラスタリング

第2節で述べた収束点計算法は，単峰性であれば有効である[2, 3]が，多峰性の問題に適用するには，各局所最適解を目指している移動ベクトルをクラスタリングし，クラス毎に収束点を推定する必要がある．これは，複数の局所最適解を探索する新しいニッチ法になり得る．

本論文は，収束点計算法を多峰性タスクへ拡張する取組の第一歩として，局所最適解が2個の関数を題材にして，移動ベクトルをクラスタリングするアルゴリズム開発を目指す．双極性問題の提案クラスタリング手法をTable 1の6ステップで説明する．

Step 1: 全移動ベクトルを使って，第2節の方法で収束点$\boldsymbol{x}$を求める．

Step 2: 得られた収束点$\boldsymbol{x}$に向かって来る移動ベクトルと離れていく移動ベクトルに二分する．判定には，移動ベクトルと推定収束点から移動ベクトル先端に向かうベクトルの内積の符号判定をすればよい．$\boldsymbol{b}_i^{\mathrm{T}}\cdot(\boldsymbol{c}_i - \boldsymbol{x})$が正であれば同じ向きを向いている（すなわち推定収束点から離れていく方向）し，負であれば移動ベクトルは推定収束点方向に向かっている．これら2クラスをpositiveクラスとnegativeクラスと呼ぼう．

Step 3: 全移動ベクトル終点の中で最遠にある2点を探し，$\boldsymbol{c}^1$と$\boldsymbol{c}^2$としよう．他のpositiveクラス内の移動ベクトルが$\boldsymbol{c}^1$と$\boldsymbol{c}^2$のどちらに向いているかでこれらpositiveクラスの移動ベクトルを二分する．判定には，$d1 = ||\boldsymbol{c}^1 - \boldsymbol{a}_i||/||\boldsymbol{c}^1 - \boldsymbol{c}_i||$と$d2 = ||\boldsymbol{c}^2 - \boldsymbol{a}_i||/||\boldsymbol{c}^2 - \boldsymbol{c}_i||$を用いる．$d1$が1より大きく$d2$が1より小さければ，$i$番目の移動ベクトル$\boldsymbol{b}_i$は$\boldsymbol{c}^1$方向を向いており，その逆であれば$\boldsymbol{c}^2$方向を向いている．$d1$, $d2$共に1より大きい，あるいは，小さい場合が生じた時は，$d1$と$d2$の大きい方に向いていると判断する．あるいは，そのような曖昧な方向の移動ベクトルは収束点計算に用いない，という選択肢もある．こうして，どちらの方向に向かっているかで，positiveクラス内の移動ベクトルを二分できる．

Step 4: negativeクラスの移動ベクトルはpositiveクラスの移動ベクトルよりも離れて二分できるので，クラスタリングアルゴリズムを適用する．例えば，最遠の移動ベクトル終点を2点探して初期値としてk-means++法[1]で2クラスに分ける．これらをnegative1サブクラスとnegative 2サブクラスと呼ぼう．

Step 5: Step 3とStep 4で四つのサブクラスができる．Step 3の$\boldsymbol{c}^1$と$\boldsymbol{c}^2$に向かっている移

動ベクトル群と遠ざかっていく移動ベクトル群がStep 3で分かっており，さらに，Step 4で$\boldsymbol{c}^1$と$\boldsymbol{c}^2$がどのサブグループに属しているかが分かるので，四つのサブグループの中でペアになる二つのサブグループが分かる．Table 1では，(positive1とnegative1)，および，(positive2とnegative2)がペアになることが分かる．

Step 6: Step 5で同じ局所最適解に向かう移動ベクトル群が二つ得られたので，第2節の方法でそれぞれの収束点（$\boldsymbol{x}_1$と$\boldsymbol{x}_2$）を推定する．

4 クラスタリング性能向上のための移動ベクトル方向の補正

進化計算の探索では，移動ベクトルが真っ直ぐ局所最適解に向かうことは保障できない．そのような移動ベクトルを含めて求める収束点は局所最適解から離れてしまう．さらに，前節のクラスタリングでも分類誤りが増え，同様に局所最適解の推定精度が悪くなる．

本節ではこのために4種類の対策を提案する．

改良1 微小領域内で，ランダムに2個体を生成して移動ベクトル生成．

改良2 fitness上位個体のみの移動ベクトルを使ったクラスタリング，および，クラス毎の収束点の計算

改良3 直交ベクトルを用いた移動ベクトルの向きの補正．

改良4 クラス毎の収束点から同じクラス内の移動ベクトルの向きをチェックし，クラスタリング誤りを補正

以下，現在取り組んでいるこれらの改良案を説明する．

（改良1）微小領域での移動ベクトル生成

進化計算で得られた親子個体の移動距離が局所最適解の山谷を越えてしまうことは多々起こる．この場合の移動ベクトルは，偶然に別の局所最適解を目指すことはあるかもしれないが，親個体近傍の局所最適解に向かう情報を持たないので，収束点推定には不適である．

親個体近傍の局所最適解に向かう移動ベクトルを生成するためには，平面近似が可能な微小領域で子個体を生成することが解決策になる．具体的な実現例としては，ランダムに生成した$\boldsymbol{a}_i$の近傍で次元毎に探索範囲の$1/p$内で一様乱数を発生させ$\boldsymbol{c}_i$とする．

$$c_i[j] = a_i[j] + rand(-1,1) \times \frac{(\max\{v_i[j]\} - \min\{v_i[j]\})}{p} \tag{4}$$

ここで，$a_i[j]$と$c_i[j]$は，それぞれ探索点$\boldsymbol{a}_i$と$\boldsymbol{c}_i$のj番目次元の値（$1 \leq j \leq d$, dは探索空間の次元数）である．

両者のfitnessを比較し，fitnessの高い個体を$\boldsymbol{c}_i$，低い個体を$\boldsymbol{a}_i$と名前を置き換え，移動ベクトル$\boldsymbol{b}_i$を決定する．進化計算で子個体のfitnessが親個体よりも良くなる保証はないので，約半数の（親個体→子個体）は局所最適解推定の推定に使えない．しかし，fitness景観は超平面で近似できる微小領域の2個体でランダムに生成する2点の良い方を$\boldsymbol{c}_i$とする本改良は，必ずすべての個体に対してより良い方向を向く移動ベクトルが生成できる．推定に使えない無効移動ベクトルがなくなるので，局所最適解推定精度が向上する．

子個体を親個体の極近傍に得るということは進化計算の探索から考えれば探索が遅くなるが，推定精度が上がった収束点をエリート個体とすると進化計算の加速につながる．進化計算の探索と，移動ベクトル生成と局所最適解推定とを別プロセスとして分離するという探索戦略も選択肢の一つである．すなわち，進化計算で得られる子個体を移動ベクトルに使わずに，ランダムに選択した微小領域の点を$\boldsymbol{c}_i$として移動ベクトルを求め，局所最適解近傍の収束点を計算する方法である．

（改良2）優良移動ベクトルのみによるクラスタリングと収束点推定

多峰性の場合，局所最適解から遠いfitnessの低い領域の移動ベクトルは複数の局所最適解の山谷の影響を受け，必ずしも局所最適解方向に向いているわけではない．正しく局所最適解方向を向いていない移動ベクトルで収束点を計算すると誤差が大きくなるであろうことは容易に推察される．しかも多峰性の局所最適解対策として移動ベクトルのクラスタリングを行うので，このクラスタリング誤りも起こしかねず，更に得られた収束点の信頼性が下がる．

この改良として，局所最適解に近い個体程（優良個体程）移動ベクトルの向きは局所最適解方向

Table 2　式(6)のパラメータ

次元数	2-D	6-D	10-D
個体数	100	150	200
探索範囲	2変数とも[-4,4]	6変数とも[-4,4]	10変数とも[-4,4]
a_1とa_2	(2.1, 3.4)	(2.1, 3.4)	(2.1, 3.4)
$\boldsymbol{\sigma}_1$と$\boldsymbol{\sigma}_2$	{(1.5,1.5), (2.0,2.0)}	{(1.5,1.5,1.5,1.5,1.5,1.5), (2.0,2.0,2.0,2.0,2.0,2.0)}	{(1.5,1.5,1.5,1.5,1.5,1.5,1.5,1.5,1.5,1.5), (2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0,2.0)}
$\boldsymbol{\mu}_1$と$\boldsymbol{\mu}_2$	{(-2.0,2.0), (2.0,-2.0)}	{(-2.0,2.0,-2.0,2.0,-2.0,2.0), (2.0,-2.0,2.0,-2.0,2.0,-2.0)}	{(-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0), (2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0,2.0,-2.0)}

を向くであろうと期待して，ftness上位個体の移動ベクトルのみを用い，局所最適解別に移動ベクトルを分類する第3節のクラスタリング，および，(改良3)以降で用いる移動ベクトルを限定する．次節の評価実験では，子個体$\boldsymbol{c}_i$のfitnessが平均fitnessよりも良い場合の移動ベクトルを採用する．

（改良3）直交ベクトルを用いた移動ベクトルの向きの補正

（改良1）で得た$\boldsymbol{b}_i$はfitnessの良い方向を指しているが，局所最適解方向を指している保証はない．そこで直交ベクトルとそのfitnessを用いて少しでも局所最適解方向に向くように向きの補正をする．

（改良1）で得た$\boldsymbol{c}_i$以外に$\boldsymbol{a}_i$近傍に$(d-1)$個のランダムに生成し，$\boldsymbol{a}_i$からこれらの点への線形独立な方向ベクトルを$\boldsymbol{b}_i^2, \boldsymbol{b}_i^3, ..., \boldsymbol{b}_i^d$としよう．(改良1）で得た$\boldsymbol{b}_i$を$\boldsymbol{b}_i^1$とすれば，これら$d$個のベクトルから，Gram-Schmidtの直交化法で，正規直交ベクトル$\boldsymbol{e}_1, \boldsymbol{e}_2, ..., \boldsymbol{e}_d$を得る．これらのベクトルの長さは1なので，微小領域でのfitness計算には長すぎることもある．そこで,(改良1）で得た$\boldsymbol{b}_i$の長さを掛け補正する．長さを補正したこれらの直交化ベクトルを$\boldsymbol{e}_1^{'}, \boldsymbol{e}_2^{'}, ..., \boldsymbol{e}_d^{'}$としよう（$\boldsymbol{e}_1^{'} = \boldsymbol{b}_i$）．

次にこれらの長さ補正をした直交ベクトルをfitness増加分で重み付き平均し，fitnessが最も高くなる方向への合成ベクトルを求め，これを新たに$\boldsymbol{b}_i$とする．fitness関数を$f(\,)$とすると，長さを補正したこれらの直交化ベクトルのfitness増加分は$\Delta_j = f(\boldsymbol{e}_j^{'}) - f(\boldsymbol{a}_i)$で表される．もし，$f(\boldsymbol{e}_j^{'}) < f(\boldsymbol{a}_i)$の場合は，$\boldsymbol{a}_i$に対する鏡像点を$\boldsymbol{e}_j^{'}$ $(= \boldsymbol{a}_i + (\boldsymbol{a}_i - \boldsymbol{e}_j^{'}))$とすればよい．式(5)に示すように，fitness増加分で重み付平均を取って局所最適解方向に向くよう向きを補正した合成移動ベクトルを新たな$\boldsymbol{b}_i$とする．

$$\boldsymbol{b}_i = \frac{\sum_{j=1}^{d} \Delta_j (\boldsymbol{e}_j^{'} - \boldsymbol{a}_i)}{\sum_{j=1}^{d} \Delta_j} \tag{5}$$

（改良4）クラスタリング誤りを補正

以上の（改良1）～（改良3）で得られたn本の移動ベクトルを第3節の方法でクラスタリングし，クラス毎に収束点を求める．しかし，このクラスタリング結果に誤りがあることもある．この分類誤りを補正する．

正しく移動ベクトルが推定されれば，得られた収束点$\boldsymbol{x}_k$から移動ベクトル終点$\boldsymbol{c}_i$への距離は移動ベクトルの始点$\boldsymbol{a}_i$への距離よりも近いはずである．$||\boldsymbol{c}_i - \boldsymbol{x}_k|| > ||\boldsymbol{a}_i - \boldsymbol{x}_k||$であればクラスタリング誤りと判断し，他のクラスの収束点から同様に移動ベクトルの終点と始点への距離を求め，正しく収束点に向かうクラスを探してクラスタリング誤り訂正をする．その後最終的に，各クラス毎に収束点を計算し，局所最適解の推定とする．

5　評価実験

2つのd次元Gauss関数（$d = 2, 6, 10$）を組み合わせた式(6)の双極関数を用いて，提案手法が正しく局所最適解を推定するように移動ベクトルをクラスタリングできるかどうか，および，より局所最適解近傍に収束点が計算できるかを評価する．

$$f^{(d)}(x) = -\sum_{j=1}^{2} \left\{ \boldsymbol{a}_j^{(d)} \exp\left(-\sum_{i=1}^{d} \frac{(x_{ij} - \mu_{ij}^{(d)})^2}{2\sigma_{ij}^{(d)}} \right) \right\} \tag{6}$$

実験に用いた式(6)のパラメータをTable 2に示す．式(4)のpは50とした．

第4節で提案した改良手法を，Table 3の2次元，6次元，10次元の双極関数で求めた収束点と局所最適解のEuclidian距離で評価する．10回試行平均をTable 3，および，Fig. 2に示す．提案改良を進めるにつれて二つの局所最適解の推定がどんどん良くなっていることが判る．

また，改良によるクラスタリング性能と局所最適解の推定精度をTable 4に示す.（改良1）の微小領域では移動ベクトルが見えないので，式(4)

Table 3　求めた2つの収束点から局所最適解への10回試行平均のEuclidian距離で示した提案改良手法の効果．距離1は2つの収束点から各局所最適解へのEuclidian距離の10回平均，距離2は両者の平均．

改良手法の組合せ	2-D		6-D		10-D	
	距離1	距離2	距離1	距離2	距離1	距離2
改良1	1.686239 1.869849	1.77804435	4.639765 4.733906	4.68683515	6.181468 6.371355	6.27641105
改良1+2	0.90755 0.614865	0.7612071	2.892427 2.150131	2.5212789	4.901894 3.374397	4.13814545
改良1+2+3	0.706034 0.298248	0.50214115	1.435035 0.840024	1.1375294	2.819972 1.377682	2.09882705
改良1+2+3+4	0.362957 0.394462	0.3787096	0.739636 0.899609	0.81962245	2.050443 1.422954	1.73669845

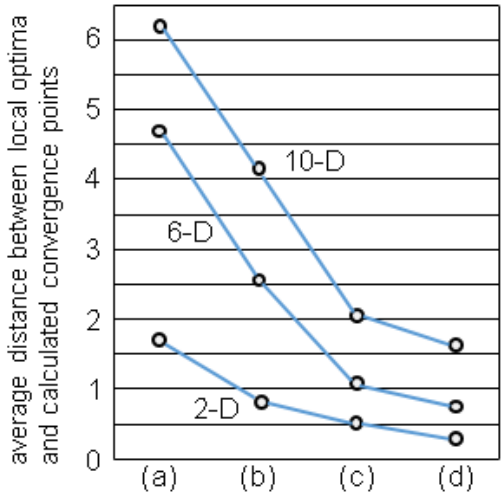


Fig. 2　2次元，6次元，10次元の双極性関数を用いた改良案による収束計算点と局所最適解との距離．(a) 改良1，(b) 改良1+2，(c) 改良1+2+3，(d) 改良1+2+3+4.

のpを15として移動ベクトルを求め， Table 4(a) としている．

6　結論

本論文は，進化計算個体群の収束点を求める方法を多峰性に拡張するための第一歩として，局所最適解が2個ある双極タスクを取り上げ，多峰性に展開するための個体群の動き（移動ベクトル）を各局所最適解への向きでクラスタリングした．第1に，移動ベクトルの向きと収束点計算から，2つの局所最適解に向かう2つの移動ベクトル群に分類する方法を提案した．第2に，このクラスタリング精度向上のための4つの改良案を提案し，そのクラスタリング性能の向上と，各局所最適解の推定精度向上を確認した．

今後は，局所最適解が3個以上ある一般的な多峰性でのクラスタリングに提案アルゴリズムを拡張する予定である．

謝辞

本研究はJSPS科学研究費（課題番号 15K00340, および，26540145）の助成を受けたものである．

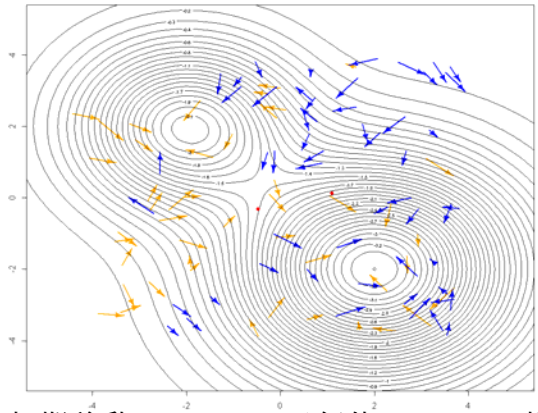
参考文献

1) Arthur, D. and Vassilvitskii, S., “k-means++: the advantages of careful seeding,” 18th ACM-SIAM symposium on discrete algorithms (SODA2007), PA, USA, pp.1027–1035 (2007).

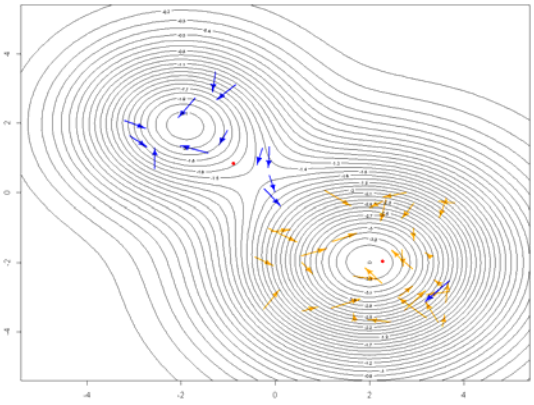
2) 村田昇，西井龍映，高木英行，裴岩「世代間移動ベクトル群の収束点推定法」2014進化計算シンポジウム，廿日市市, pp.210–215 (2014年12月20-21日).

3) Murata, N., Nishii, R., Takagi, H., and Y. Pei, “ Analytical Estimation of the Convergence Point of Populations,” 2015 IEEE Congress on Evolutionary Computation (CEC2015), Sendai, Japan, pp. 2619–2624 (May 25-28, 2015).

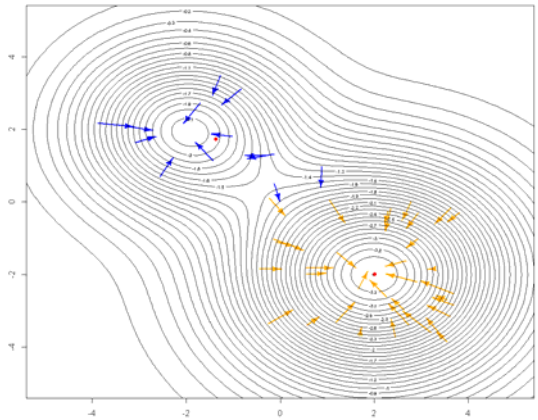
Table 4　提案改良によるクラスタリング性能と局所最適解の推定精度改善の視覚化．赤点はクラスタリングした移動ベクトルで求めた収束点．



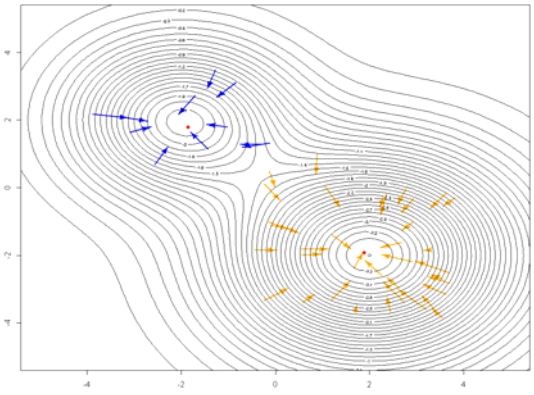
(a) 初期移動ベクトル．子個体のfitnessが親個体よりも悪い場合は，子個体→親個体への移動ベクトルに変換してある．



(b) 改良2



(c) 改良2＋3



(d) 改良2＋3＋4