平成 30 年度 電気通信大学大学院情報理工学研究科 修士論文

適応的個体間距離に基づく複数解探索型 Bat Algorithm

所属	情報学専攻メディア情報学プログラム
学籍番号	1730022
氏名	岩瀬拓哉
主任指導教員	高玉圭樹教授
指導教員	佐藤寛之准教授
提出日	平成 30 年 1 月 29 日

概要

本論文では, 大域探索と局所探索を調節可能な Bat Algorithm と個体間距離に基づく動的変化を考慮した Niche Radius を組み合わせることによる多峰性最適化問題における，複数解探索アルゴリズムを構築し，その手法の有効性を検証する．従来の多点探索アルゴリズムは一つの最適解に収束する傾向にあるが，実問題への適用を考慮した時に複数の最適解及び局所解を探索する必要がある．多点探索アルゴリズムの中でも大域と局所の探索バランスに優れた Bat Algortihm は探索する上で全個体の最良解を参照して移動するため，最終的に最適解に収束することから複数解を同時に探索することは困難である．本研究では各個体の探索領域を分割させる Niche Radius を用いることで最適解だけでなく，局所解も同時に探索可能な Bat Algorithm の構築をする．従来手法と提案手法の性能を比較するため，最適解と局所解の数が異なるパターンの多峰性関数を用いてシミュレーション実験を行った結果，全ての関数に対して従来手法は一つの最適解に収束していたが，提案手法では全最適解及び局所解を探索することができ，従来に対する変更点が有効であることを示した．

目 次

1 はじめに **1**
1.1 背景と目的 1
1.2 関連研究 2
1.3 論文構成 3

2 Metaheuristic Algorithms **4**
2.1 Genetic Algorithm (GA) 4
2.2 Particle Swarm Optimization (PSO) 5
2.3 Differential Evolution (DE) 6
2.4 Bat Algorithm (BA) 8
2.5 Evolution Strategies with Covariance Matrix Adaptation (CMA-ES) . . . 10

3 Niching Scheme **13**
3.1 Crowding 13
3.2 Fitness Sharing 13
3.3 クラスタリング 14
3.3.1 階層的クラスタリング 14
3.3.2 非階層的クラスタリング 14
3.4 Niche Radius 14
3.5 Dynamic Niche Radius 15

4 Niching Methods **15**
4.1 Crowding DE (CDE) 15
4.2 A Dynamic Archive Niching Differential Evolution (dADE) 15
4.3 Niching the CMA-ES via Nearest-Better Clustering (NEA) 15
4.4 Covariance Matrix Self Adaptation Evolution Strategy with Repelling Subpopulations (RS-CMSA-ES) 15

5 Novelty Search-based Bat Algorithm (NSBA) **16**
5.1 概要 16
5.2 アルゴリズム 16

6 Niche Radius-based Bat Algorithm (NRBA) **17**
6.1 概要 17
6.2 アルゴリズム 17

7 Dynamic Niche Radius-based Bat Algorithm (DNRBA) **18**
7.1 概要 18
7.2 アルゴリズム 18

8 多峰性最適化問題 **19**

9 実験 **20**
9.0.1 評価基準とハイパーパラメーター 20
9.0.2 結果 20
9.0.3 結果 20
9.1 考察 20

10 おわりに **21**
10.1 まとめ 21
10.2 今後の課題 21

謝辞 **22**

参考文献 **23**

付録 **24**

1　はじめに

1.1　背景と目的

近年，多点探索アルゴリズムは最適化問題において，一般的なメタヒューリスティック手法として用いられるようになった．多点探索アルゴリズムは特に非線形な問題に対しても適用することが可能であり，魚や鳥の群れをモデルにしたParticle Swarm Optimization(PSO)[6]や，ホタルの光強度により互いのホタルが引き寄せられるFirefly Algorithm(FA)[**?**]は高次元な最適化問題に対して有効であることを示している．中でもBat Algorithm(BA)は大域探索と局所探索の性能を自動で切り替えるという点で優れたアルゴリズムである[9]．しかし多峰性最適化問題における，従来の多点探索アルゴリズムは全個体の中の最良解を参照して一点へ移動するため，探索終了時に一つの最適解に収束する傾向にあるが，実環境への適用を考慮した時に最適解だけでなく局所解を探索し，保持しておくことは非常に重要な意味を持つ．応用先の一例として，災害時における被災者を解，救助ロボットを個体と見立てた時に，不特定多数の被災者を探索することは人間にとって困難であり，複数の解を同時に探索しなければならない．

そこで本研究では，探索範囲の自動調整可能なBAを用い，各個体の探索領域を分割させることで個体の分散化を図る．探索空間のスケールと解の数から算出されるNiche Radius[**?**]を用いることで，予め各個体の探索領域を決定し，その探索領域内の最良個体から遠ざかる方向へ移動することで，個体同士が同じ解に留まらず，散らばるように改良する．従来手法の探索アルゴリズムに対して3つの変更をした．(i) 大域探索: 分割探索領域内の最良解を参照;(ii) 局所探索: 分割探索領域内の最良解付近に解候補を生成; (iii) ランダム探索: 選択した個体の分割領域内にランダムで解候補を生成．これらの変更により，従来手法と提案手法の探索性能を比較するため，発見した最適解及び局所解の数を評価尺度として，複数の異なる峰を持つ多峰性関数を用いてシミュレーション上で実験を行う．

1.2　関連研究

1.3　論文構成

本論文の構成は次の通りである．**??**章で最適化問題において一般的に用いられる EAs について説明し，3 章で実問題を模擬的に表した多峰性最適化問題における複数解探索手法の機構である Niching Scheme について説明する．4 章では，EAs と Niching Scheme を組み合わせた複数解探索手法を説明し，5 章から 7 章までは提案手法である NSBA, NRBA 及び DNRBA の説明をする．

2 Metaheuristic Algorithms

現実問題は非常に複雑であり，その複雑さを多峰性関数として表現した問題をを最適化するために Evolutionary Algorithms (EAs) が用いられるようになった．EAs のメカニズムとして単一の最適解を探索することを目的として設計されており，生物の生殖や突然変異，交叉，適者生存といった過程をモデルとしている．その代表的なアルゴリズムを 2.1 節から 2.5 節で紹介する．

2.1 Genetic Algorithm (GA)

遺伝的アルゴリズム (Genetic Algorithm:GA) [5] は生物の進化の過程を模擬したアルゴリズムであり，最適化問題において最も基本的なヒューリスティック手法である．生物は次の世代により良い遺伝子を持った個体を残すため，まずは親集合の中から個体を選択し，選択した個体同士を交叉させる．この時，ある一定の確率で突然変異させる．次に，交叉させて新たに生成された個体を子集合の解候補とし，親と子の個体を評価して，評価値の高い個体は次世代の個体として保存され，評価値の低い個体は淘汰される．各個体に対し，この「選択」,「交叉」,「突然変異」,「評価 (淘汰)」を繰り返すことで，環境に対する適合度が高くなっていく (評価値の高い遺伝子が残る)．アルゴリズムの疑似コードは，以下の Algorithm1 に記す．

- **STEP1: 個体の初期化**
  N 個の個体 x_i を初期集合として生成し，ランダムな評価値を割り当てる．また交叉率と突然変異率を定義する．(1-2 行目)
- **STEP2: 選択と交叉**
  選択した親同士 x_i を交叉率 P_c により交叉させ，子 (解候補) x_i^{new} を生成する．(5-7 行目)
- **STEP4: 突然変異**
  突然変異率 P_m により，子 (解候補) の評価値を変化させる．(8-10 行目)
- **STEP5: 評価**
  親個体 x_i と子個体 x_i^{new} の評価値を比較し，評価値の高い個体を次世代に残す．(11-13 行目)

Algorithm 1 Genetic Algorithm

Require: Objective Function $F(x), x = (x_1, x_2, ..., x_d)$

Initialize Population $x_i (i = 1, 2, ..., N)$
Initialize P_m, P_c
while t < Max Generation **do**
  for i=1 to N **do**
    if $rand(0,1) < P_c$ **then**
      Generate an offspring x_i^{new}
    end if
    if $rand(0,1) < P_m$ **then**
      Replace mutated offspring x_i^{new}
    end if
    if $F(x_i) < F(x_i^{new})$ **then**
      Replace x_i with x_i^{new}
    end if
  end for
  t=t+1
end while

2.2 Particle Swarm Optimization (PSO)

最適化手法の一つとして粒子群最適化 (Particle Swarm Optimization:PSO) がある．PSO は，ランダムに個体を初期化するという点において，遺伝的アルゴリズム (GA) に似た性質を持つ．魚や鳥の群れの動きをモデルにしたアルゴリズムであり，個体間のユークリッド距離を速度として新たに解候補を生成する．個体の速度及び生成式は以下の通りである．

$$\boldsymbol{v_i^{t+1}} = w\boldsymbol{v_i^t} + c_1 r_1 \cdot (\boldsymbol{x_{pbest}^t} - \boldsymbol{x_j^t}) + c_2 r_2 \cdot (\boldsymbol{x_{gbest}^t} - \boldsymbol{x_i^t}) \tag{2-2-1}$$

$$\boldsymbol{x_i^{t+1}} = \boldsymbol{x_i^t} + \boldsymbol{v_i^{t+1}} \tag{2-2-2}$$

$\boldsymbol{x_i}$，$\boldsymbol{v_i}$ は時刻 t における各個体の現在位置と速度を表し，w, c_1, c_2 は係数，r_1, r_2 は一様乱数を表す．これらの係数が速度を制限するパラメータとなっており，w が時刻 t での速度を調整し，c_1, c_2 は値が 1 より小さいほど局所探索を行い，1 より大きくなると最良個体を含む広い範囲を大域探索するようになる．PSO のアルゴリズムの疑似コードである Algorithm2 を以下に記す．

Algorithm 2 Particle Swarm Optimization

Require: Objective Function $F(x), x = (x_1, x_2, ..., x_d)$
Initialize Population $x_i (i = 1, 2, ..., N)$ and Velocity v_i
$F(x_{gbest}) = \max F(x_i)$
Initialize c_1, c_2, w
while t < Max Iteration **do**
  for i=1 to N **do**
    Generate a new solution x_i^{t+1} and update v_i [Eqs.(2-2-1),(2-2-2)]
    if $F(x_i^{t+1}) > F(x_{pbest})$ **then**
      Replace the individual x_i^{t+1} as x_{pbest}
    end if
    if $F(x_{gbest}) < F(x_{pbest})$ **then**
      $x_{gbest} = x_{pbest}$
    end if
  end for
  t=t+1
end while

2.3 Differential Evolution (DE)

差分進化 (Differential Evolution:DE) [7] は進化的計算手法の一つであり，問題に応じて個体間同士の相対距離に基づいた探索戦略を用いることのできる，他のアルゴリズムとは異なる特徴を持つ．DE は以下の手順により探索を繰り返す．

- **STEP1: 個体の初期化**
  探索領域内にランダムで個体を生成する．
- **STEP2: 突然変異による解候補の生成**
  DE は以下，いずれかの探索戦略を用いて解候補を生成する．

  1. DE/rand/1
  $$\boldsymbol{v_{i,j}^{t+1}} = \boldsymbol{x_{r1,j}^{t}} + F \cdot (\boldsymbol{x_{r2,j}^{t}} - \boldsymbol{x_{r3,j}^{t}}) \tag{2-3-1}$$
  2. DE/best/1
  $$\boldsymbol{v_{i,j}^{t+1}} = \boldsymbol{x_{gbest,j}^{t}} + F \cdot (\boldsymbol{x_{r1,j}^{t}} - \boldsymbol{x_{r2,j}^{t}}) \tag{2-3-2}$$
  3. DE/current-to-best/1
  $$\boldsymbol{v_{i,j}^{t+1}} = \boldsymbol{x_{i,j}^{t}} + F \cdot (\boldsymbol{x_{gbest,j}^{t}} - \boldsymbol{x_{i,j}^{t}}) + F \cdot (\boldsymbol{x_{r1,j}^{t}} - \boldsymbol{x_{r2,j}^{t}}) \tag{2-3-3}$$
  4. DE/rand/2
  $$\boldsymbol{v_{i,j}^{t+1}} = \boldsymbol{x_{r1,j}^{t}} + F \cdot (\boldsymbol{x_{r2,j}^{t}} - \boldsymbol{x_{r3,j}^{t}}) + F \cdot (\boldsymbol{x_{r4,j}^{t}} - \boldsymbol{x_{r5,j}^{t}}) \tag{2-3-4}$$

5. DE/best/2

$$\boldsymbol{v_{i,j}^{t+1}} = \boldsymbol{x_{gbest,j}^{t}} + F \cdot (\boldsymbol{x_{r1,j}^{t}} - \boldsymbol{x_{r2,j}^{t}}) + F \cdot (\boldsymbol{x_{r3,j}^{t}} - \boldsymbol{x_{r4,j}^{t}}) \tag{2-3-5}$$

この時，i, j は個体番号と次元数を表し，$r_1, ..., r_4$ は 1 から N までの一様乱数の整数を表す.「DE/rand/1」は，個体の親集団の中から 3 つの個体をランダムに選択し，その相対距離を用いて新たに解候補 $v_{i,j}^{t+1}$ を生成する.「DE/best/1」は，個体の親集団の中の最良個体と個体を 2 つランダムに選択し，その最良個体付近に新しい解候補 $v_{i,j}^{t+1}$ を生成する.「DE/current-to-best/1」は，ランダムに選択した 2 つの個体と，最良個体と個体自身の相対距離と用い，最良個体方向へ新たに解候補 $v_{i,j}^{t+1}$ を生成する.「DE/rand/2」では，親集合から 5 つの個体をランダムに選択し，式 (2-3-1) よりも広い探索領域内に新しく解候補 $v_{i,j}^{t+1}$ を生成する.「DE/best/2」も同様，式 (2-3-2) よりも広い探索領域内に新しく解候補 $v_{i,j}^{t+1}$ を生成する.

- **STEP3: 交叉**
  解に多様性を持たせるため，ここでは STEP2 で生成した解候補 $v_{i,j}^{t+1}$ と個体 x_i を確率的に交叉させ，新たに解 $u_{i,j}^{t+1}$ を生成する．生成式は次式の通りである．

$$u_{i,j}^{t+1} = \begin{cases} v_{i,j}^{t+1} & \text{if}(rand(0,1) \leq C_r) \ or \ j = D \\ x_{i,j}^{t+1} & \text{if}(rand(0,1) > C_r) \ and \ j \neq D \end{cases} \tag{2-3-6}$$

  ここで C_r は $[0, 1]$ の範囲内での交叉係数を表す．この交叉係数 C_r が一様乱数以上であれば STEP2 で生成した解候補 $v_{i,j}^{t+1}$ が採用され，一様乱数未満であれば個体 $x_{i,j}$ が採用される．

- **STEP4: 評価**
  最良個体と生成した解 $u_{i,j}^{t+1}$ を比較し，評価値の高い方を次世代に引き継ぐ.

- **STEP5: 終了条件を満たすまで STEP2 へ戻る**

ここでは，一般的に用いられる DE/rand/1 のアルゴリズムの疑似コードを以下の Algorithm3 に記す.

Algorithm 3 Differential Evolution (DE/rand/1)

Require: Objective Function $F(x), x = (x_1, x_2, ..., x_d)$
Initialize Population $x_i (i = 1, 2, ..., N)$ and Velocity v_i
Initialize C_r, F
while t < Max Iteration **do**
  for i=1 to N **do**
    Select random integer $r_1, r_2 \in \{1, 2, ..., N | r_1 \neq r_2 \neq i\}$
    Generate a candidate $v_{i,j}$ [Eq. (2-3-1)]
    for $j = 1$ to D **do**
      if $rand(0,1) \leq C_r$ *or* $j = D$ **then**
        offspring $u_{i,j} = v_{i,j}^{t+1}$
      else $\{rand(0,1) > C_r$ *and* $j \neq D\}$
        offspring $u_{i,j} = x_{i,j}^{t}$
      end if
    end for
    if $F(u_{i,j}^{t+1}) > F(x_{pbest})$ **then**
      Replace the offspring $u_{i,j}^{t+1}$ as x_{pbest}
    end if
  end for
  t=t+1
end while

2.4 Bat Algorithm (BA)

Bat Algorithm(BA) [9] は群知能アルゴリズムの一つで，対象物までの方向や距離を知るコウモリの特性（エコロケーション）を利用して周囲の状況を認知し，大域探索と局所探索が進むにつれて探索速度を徐々に落とし，探索性能を自動調節することが可能なアルゴリズムである．各個体の周波数 f_i，速度 v_i，位置 x_i は以下の式で定義し，更新される．ラウドネス A は，コウモリが対象物に近づくと値が減少し，移動距離も比例して短くなる．コウモリの行動は以下 3 つで構成される．

- 大域探索: 各コウモリは位置 x_i において，自身が発する周波数 f_i の反響によって対象物との距離を測り，対象物に向かって速度 v_i で移動する．
- 局所探索: 対象物近辺にコウモリを移動させる．
- ランダム探索: 探索領域内にコウモリをランダムで移動させる．

BA で扱う各個体の周波数 f_i，速度 v_i，位置 x_i は以下の式で定義される．

$$f_i = f_{min} + (f_{max} - f_{min})\beta \tag{2-4-1}$$

$$\boldsymbol{v_i^{t+1}} = \boldsymbol{v_i^t} + \boldsymbol{(x_* - x_i^t)} * f_i \tag{2-4-2}$$

$$\boldsymbol{x_i^{t+1}} = \boldsymbol{x_i^t} + \boldsymbol{v_i^{t+1}} \tag{2-4-3}$$

個体番号を i とし，各個体の周波数 f_i は個体の速度を制限するパラメータであり，[0, 1] の区間で表される．ここでは $f_{min} = 0$，$f_{max} = 1$ として設定し，β は 0 から 1 の乱数が割り当てられる．局所探索では，全個体の最良解（グローバルベスト）x_* の周辺に新しい解候 x_{loc} を生成する．生成式は次の通りである．

$$\boldsymbol{x_{loc}} = \boldsymbol{x_*} + \epsilon A_i^t \tag{2-4-4}$$

パラメータ ϵ は $1 \times D$ 次元の配列で $[-1, 1]$ 区間のランダムな値が割り当てられる． ランダム探索では解探索空間にランダムで新たに解候補を生成する．生成式は以下の通りである．

$$\boldsymbol{x_{rnd}} = \boldsymbol{x_{lb}} + (\boldsymbol{x_{ub}} - \boldsymbol{x_{lb}}) * rand(1, D) \tag{2-4-5}$$

解探索空間の上限と下限をそれぞれ $\boldsymbol{x_{ub}}, \boldsymbol{x_{lb}}$ とし，$rand$ は 0 から 1 までの乱数が入る．以上より各個体の解候補 x_i^{t+1}，x_{loc}，あるいは x_{rnd} の評価値が各個体の最良解（パーソナルベスト）x_{i*} より良ければ更新され，同時にラウドネス A とその反射波であるパルスレート r も以下の式に基づいて更新される．

$$A_i^{t+1} = \alpha A_i^t \tag{2-4-6}$$

$$r_i^{t+1} = r_i^t[1 - exp(-\gamma t)] \tag{2-4-7}$$

解を更新する度にラウドネス A_i は徐々に減少し，それに比例して評価頻度を下げていく．対してパルスレート r_i は増加していき，探索が進むにつれて局所探索頻度が減少する．

従来の BA の疑似コードは以下の Algorithm 1 に記す．

Algorithm 4 Bat Algorithm

Require: 評価関数 $F(x)$ の設定

1: 各個体 $x_i(i = 1, 2, ..., N)$ と速度 v_i の初期化
2: 周波数 f_i の定義　[eq.(2-4-1)]
3: パルスレート r_i とラウドネス A_i の初期化
4: **while** t < Max Iteration **do**
5: **for** i=1 to N **do**
6: 大域探索: 新しい解候補 x_i^{t+1} の生成と速度 v_i の更新 [eqs.(2-4-2),(2-4-3)]
7: **if** $rand > r_i$ **then**
8: 局所探索: グローバルベスト x_* 近辺に新しい解候補 x_{loc} を生成 [eq.(2-4-4)]
9: **end if**
10: ランダム探索: 解空間に解候補の生成 [eq.(2-4-5)]
11: **if** $(rand < A_i \ \& \ F(x_i), F(x_{loc}), F(x_{rnd}) < F(x_{i*})$ **then**
12: 新しい解の評価と更新
13: パルスレート r_i の増加とラウドネス A_i の減少 [eqs.(2-4-6),(2-4-7)]
14: **end if**
15: **end for**
16: t=t+1
17: **end while**

2.5 Evolution Strategies with Covariance Matrix Adaptation (CMA-ES)

Evolution Strategy with Covariance Matrix Adaptation (CMA-ES) [3][4] は多変量正規分布を用いて解候補を生成し，それらの評価値を基に算出する共分散行列から探索範囲を決定する．CMA-ES の大きな特徴として，変数間の依存度を考慮している (パラメータチューニングを必要としない) という点と，単調に増減する線形的な問題に依存しないという点が挙げられる．CMA-ES には様々な手法がある中で，ここでは一般的な $(\mu|\mu_w, \lambda)$-CMA-ES [4] を紹介する．具体的なアルゴリズムについては以下の Algorithm 5 に示し，終了条件を満たすまで以下の手順を繰り返す．

- **STEP1: 個体の生成**
  まず，正規分布 $N(0, I)$ の範囲で解候補 $z_i (i = 1, 2, ..., \lambda)$ を独立に生成し，それを基に設計変数 $x_i (i = 1, 2, ..., \lambda)$ を生成する．生成式は以下の通りである．

$$\boldsymbol{y_i^t} = \sqrt{\boldsymbol{C^t}}\boldsymbol{z_i} \tag{2-5-1}$$

$$\boldsymbol{x_i^{t+1}} = \boldsymbol{m^t} + \sigma^t \boldsymbol{y_i} \tag{2-5-2}$$

  設計変数 x_i は共分散行列 $N(0, (\sigma^t)^2 C^t)$ の範囲から生成される．生成後，x_i を評価値で降順 (評価値の高い順) にソートして順位付けし，y_i 及び z_i も同様に順位付けする．C^t は，その共分散行列の範囲の広がり度合いを $d \times d$ 次元で表し，σ^t はステップサイズを，m^t は平均値ベクトル (探索範囲の中心) を表す．初期個体生成時，$C^0 = I, \sigma^0 = 0$ とする．また，$\lambda = 4 + [3In(n)]$, $\mu = [\frac{\lambda}{2}]$ とする．

- **STEP2: μ 個体の荷重和を算出し，平均ベクトル $\boldsymbol{m_i^t}$ を更新**
  評価値で降順ソートした設計変数 $\boldsymbol{x_i}$ のうち，上位 μ 個の荷重和を算出し，次式に従って平均値ベクトル m を更新する．

$$\boldsymbol{m^{t+1}} = \boldsymbol{m^t} + \sum_{i=1}^{\mu} w_i (\boldsymbol{x_i^t} - \boldsymbol{m^t}) \tag{2-5-3}$$

  この時，重み w は次式で表される．

$$w_i = In(\frac{\lambda + 1}{2}) - In(i) \tag{2-5-4}$$

  重み w は $w_1 \geq w_2 \geq ... \geq w_\mu > 0$ であり，$\sum_{i=1}^{\mu} w_i = 1$ を満たす．

- **STEP3: ステップサイズ σ の更新**
  次式に従って進化パス p_σ 及びステップサイズ σ を更新する．

$$p_\sigma^{t+1} = (1 - c_\sigma) p_\sigma^t + \sqrt{c_\sigma (2 - c_\sigma)} \sqrt{\mu_{eff}} \sum_{i=1}^{\mu} w_i \boldsymbol{z_i} \tag{2-5-5}$$

$$\sigma^{t+1} = \sigma^t exp(\frac{c_\sigma}{d_\sigma}(\frac{||p_\sigma^{t+1}||}{\hat{\chi_n}} - 1)) \tag{2-5-6}$$

この時，c_σ は進化パス p_σ の前世代との重みを表し，μ_{eff} は設計変数の上位 μ 個の加重平均の補正値，d_σ はステップサイズの減衰係数，$\hat{\chi_n}$ は n 変量正規分布のノルムの期待値を表す．これらのパラメータは次式で表される．

$$c_\sigma = \frac{4}{n+4} \tag{2-5-7}$$

$$\mu_{eff} = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \tag{2-5-8}$$

$$d_\sigma = \frac{1}{c_\sigma} + 1 \tag{2-5-9}$$

$$\hat{\chi_n} = E[||N(0,1)||] \approx \sqrt{n}(1 - \frac{1}{4n} + \frac{1}{21n^2}) \tag{2-5-10}$$

ここで n は最適化する数を表す．

- **STEP4: 共分散行列 C の更新**
  共分散行列 C 及びその進化パス p_c は次式で更新される．

$$p_c^{t+1} = (1 - c_c)p_c^t + \sqrt{c_c(2 - c_c)} \sum_{i=1}^{\mu} w_i \boldsymbol{y_i^t} \tag{2-5-11}$$

$$\boldsymbol{C^{t+1}} = \boldsymbol{C^t} + c_1[p_c^{t+1}(p_c^{t+1})^T + (1 - c_c(2 - c_c))\boldsymbol{C^t}] + c_\mu \sum_{i=1}^{\mu} w_i(\boldsymbol{y_i}(\boldsymbol{y_i})^T - \boldsymbol{C^t}) \tag{2-5-12}$$

c_σ は共分散行列 C の進化パス p_c の前世代との重みであり，c_1 及び c_μ は共分散行列 C の更新に用いられる学習率を表す．これらのパラメータは次式で表される．

$$c_1 = \frac{2}{(n + \sqrt{2})^2} \tag{2-5-13}$$

$$c_\mu = 1 - c_1 \tag{2-5-14}$$

$$c_c = \frac{4}{n+4} \tag{2-5-15}$$

Algorithm 5 CMA-ES

Require: Objective Function $F(x), x = (x_1, x_2, ..., x_d)$
Calculate parameters [Eqs. (2-5-7) to (2-5-10), (2-5-13) to (2-5-15)]
while t < Max Iteration **do**
for i=1 to N **do**
Generate solution $x_i (i = 1, 2, ..., \lambda)$ within $N(0, \sigma^2 C)$ [Eqs. (2-5-1),(2-5-2)]
end for
Evaluate and sort solution x_i
for i=1 to N **do**
Update m_i by w_i [Eqs. (2-5-4),(2-5-3)]
Update step size σ and covariance matrix C by the evolution path p_σ and p_c [Eqs. (2-5-5),(2-5-6),(2-5-11),(2-5-12)]
end for
t=t+1
end while

3 Niching Scheme

この章では，多峰性最適化問題において，EAs を拡張させて複数の最適解及び局所解探索を行うための機構である Niching Scheme について説明する．EAs は一つの最適解を探索することを目的とした設計であるため，複数の局所解を探索し，保持するには限界がある．この問題を解決するために導入した Niching Scheme について 3.1 節から 3.5 節で紹介する．

3.1 Crowding

Crowding [1] は評価値が類似する個体が同じ場所付近に位置しているとき，その最近傍個体同士を比較し，評価値の高い方を残す．この時，比較する個体は親集合と子集合を含めた全個体における最近傍個体同士が比較対象となる．この動作を毎世代繰り返すことによって同じ局所解に個体が陥らないことを目的とした機構である．

3.2 Fitness Sharing

Fitness Sharing [2] は Crowding と同様，類似個体の評価値が低い方を淘汰させるための機構として用いられる．ここでは，その類似度を定義することで，同じ類似度を持つ個体同士が評価値を比較する．一般的に，類似度の算出方法は次式で表される．

$$sh(d_{ij}) = \begin{cases} 1 - (\frac{d_{ij}}{\sigma})^{\alpha} & (\text{if } d_{ij} < \sigma) \\ 0 & (\text{otherwise}) \end{cases} \tag{3-2-1}$$

ここで d_{ij} は個体 i, j の間の距離を表し，α は係数で σ はある恣意的な閾値 (あるいは Niche radius[1]) を表す．個体間距離が近いほど Sharing function $sh(d_{ij})$ の値は大きくなり，この数値を基に Niche count m_i を算出する．

$$m_i = \sum_{j=1}^{N} sh(d_{ij}) \tag{3-2-2}$$

Niche count m_i は i 番目の個体に対する全個体の密度を表している．この Niche count より，Shared function f_i' が次式のように算出される．

$$f_i' = \frac{f_i}{m_i} \tag{3-2-3}$$

ここで，f_i は i 番目の個体の評価値を表す．Shared function f_i' は，その個体密度である Niche count m_i によって値が大きく変動する．f_i' の値が小さいほど i 番目の個体密度は高く，値が大きければ個体密度は低いということになる．

[1] 次節で説明する．

3.3 クラスタリング

クラスタリングは対象となる集合の全個体に対して部分集合に分割する手法である．クラスタリングは，**階層的クラスタリング**と**非階層的クラスタリング**の 2 つに分類される．

3.3.1 階層的クラスタリング

階層的クラスタリング (Hierarchical Clustering) は，一般的に N 個のデータ (個体) が与えられたときに初期状態として N 個のクラスタを生成する．次にクラスタ間の距離に基づいてクラスタ同士を併合させる．一つのクラスタになるまでこの併合を繰り返す．代表的なクラスタ分類に関して以下の手法がある．

- **最短距離法 (Nearest neighbor method)**
  最近傍クラスタ同士を併合していく手法 [8] である．クラスタ間の距離 $d(C_i, C_j)$ は次の式で定義される．

$$d(C_1, C_2) = \min_{x_i \in C_i, x_j \in C_j} (d(x_i, x_j)) \tag{3-3-1}$$

- **最長距離法 (furthest neighbor method)**
  最も遠いクラスタ同士を併合する手法．クラスタ間の距離 $d(C_i, C_j)$ は次の式で定義される．

$$d(C_1, C_2) = \max_{x_i \in C_1, x_j \in C_2} (d(x_i, x_j)) \tag{3-3-2}$$

- **群平均法 (group average method)**

$$d(C_1, C_2) = \frac{1}{|C_1||C_2|} \sum_{x_i \in C_1} \sum_{x_j \in C_2} d(x_i, x_j) \tag{3-3-3}$$

- **ウォード法 (Ward's method)**

$$d(C_1, C_2) = E(C_1 \cup C_2) - E(C_1) - E(C_2) \tag{3-3-4}$$

- **重心法 (Centroid method)**
  重心法は

$$d(C_1, C_2) = d(\bar{x_i}, \bar{x_j}) \tag{3-3-5}$$

3.3.2 非階層的クラスタリング

非階層的クラスタリング (Non-hierarchical Clustering) は

3.4 Niche Radius

Niche Radius は探索空間の

3.5 Dynamic Niche Radius

4 Niching Methods

4.1 Crowding DE (CDE)

4.2 A Dynamic Archive Niching Differential Evolution (dADE)

4.3 Niching the CMA-ES via Nearest-Better Clustering (NEA)

4.4 Covariance Matrix Self Adaptation Evolution Strategy with Repelling Subpopulations (RS-CMSA-ES)

5 Novelty Search-based Bat Algorithm (NSBA)

5.1 概要

5.2 アルゴリズム

6 Niche Radius-based Bat Algorithm (NRBA)

6.1 概要

6.2 アルゴリズム

7 Dynamic Niche Radius-based Bat Algorithm (DNRBA)

7.1 概要

7.2 アルゴリズム

8　多峰性最適化問題

9 実験

目的

設定

9.0.1 評価基準とハイパーパラメーター

9.0.2 結果

9.0.3 結果

9.1 考察

10　おわりに

この章では本論文のまとめを 10.1 節で行い, 今後の課題を 10.2 節で行う.

10.1　まとめ

10.2　今後の課題

謝辞

本論文の執筆並びに研究を進める上で御指導頂いた高玉圭樹教授に感謝の意を表します. また, 論文執筆において校正・校閲をして頂いた博士課程の高野諒さん, 日々の研究テーマに関して助言をして頂いた佐藤寛之准教授, 並びに日々研究を共にしている研究室の皆様, 研究を支えて頂いている皆様にこの場を借りて感謝致します.

参考文献

[1] A. Kenneth De John. *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*, Vol. 36. University of Michigan Ann Arbor, MI, USA, 1975.

[2] D. E. Goldberg and J. Richardson. Genetic algorithms with sharing for multimodal function optimization. *in Proc. of the Second International Conference on Genetic Algorithms*, pp. 41–49, 1987.

[3] N. Hansen and A. Ostermeier. Convergence properties of evolution strategies with the derandomized covariance matrix adaptation: The $(\mu/\mu_i, \lambda)$-cma-es. *EUFIT'97, 5th Europ. Congr. on Intelligent Techniques and Soft Computing, Proceedings*, pp. 650–654, 1997.

[4] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, Vol. 9(2), pp. 159–195, 2001.

[5] J. H. Holland. Adaptation in natural and artificial systems. *University of Michigan Press, Ann Arbor, MI*, 1975.

[6] J. Kennedy and R. Eberhart. A new optimizer using particle swarm theory. *Proc. Sixth International Symposium on Micro Machine and Human Science*, pp. 39–43, 1995.

[7] K. Price R. Storn. Differential evolution a simple and efficient heuristic for global optimization over continuous spacesy. *International computer science institute*, Vol. 46, No. 2, 1995.

[8] R. Sibson. Slink: An optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, Vol. 16, pp. 30–34, 1973.

[9] X.S. Yang. A metaheuristic bat-inspired algorithm. *in: Nature Inspired Cooperative Strategies for Optimization*, Vol. 284, pp. 65–74, 2010.

付録