数理解析研究所講究録
第1629巻 2009年 142-151



# マルチスタート単体法による多峰関数の最適化

外崎　真造 (Shinzo Tonosaki)* , 久野　誉人 (Takahito Kuno)
筑波大学大学院システム情報工学研究科
Graduate School of Systems and Information Engineering,
University of Tsukuba

**概 要**

非線形最適化のヒューリスティクスの一つであるNelder-Mead法は、対象とする問題の目的関数がブラックボックスであってもよいという利点があるが、多峰性を有するような問題の場合、得られる解は局所的最適解にすぎない。本稿では探索領域における目的関数の下界に関する情報が既知であることを仮定し、リプシッツ条件とNelder-Mead法に基づいた分枝限定法によって非線形計画問題の大域的な最適化を試みる。

## 1　はじめに

数理計画法のうち、線形計画問題や凸非線形計画問題に関しては十分に実用的なアルゴリズムが既に整備されている。しかし現実の応用に現れる最適化問題の多くは非線形かつ多峰性を有し、これらのアルゴリズムで大域的な最適解の得られる保証はない。本稿では、非線形計画問題のためのヒューリスティクスの1つであるNelder-Mead法 (単体法)[3] を用いて、多峰関数の大域的な最適化を試みる。一般に、Nelder-Mead法によって得られる解は局所的最適解である。そこで、複数の異なる初期点からNelder-Mead法を実行し、解を改善していくマルチスタート法を用いる。目的関数はブラックボックスであるが、探索領域における目的関数の下界に関する情報が分かっているような関数を想定し、リプシッツ条件とNelder-Mead法に基づいた分枝限定法を提案する。

第2節では、本研究で対象としている問題について述べる。第3節では、非線形最適化のヒューリスティクスであるNelder-Mead法とマルチスタート法について述べる。第4節では、マルチスタート単体法とリプシッツ条件に基づいた分枝限定法を提案する。第5節では、提案する方法と初期点をランダムに与える方法の比較実験を行う。第6節では、本稿での結果を総括する。

*tonosaki@syou.cs.tsukuba.ac.jp

## 2 多峰関数の最適化問題

関数 $f, g_i(i=1,\cdots,m), h_j(j=1,\cdots,l)$ を $n$ 次元空間 $\Re^n$ で定義された実数値関数とする。このとき、制約条件 $g_i(\boldsymbol{x}) \leq 0$ および $h_j(\boldsymbol{x}) = 0$ を満たすベクトル $\boldsymbol{x} = (x_1, x_2, \cdots, x_n)$ のなかで、目的関数 $f(\boldsymbol{x})$ を最小にする $\boldsymbol{x}$ を求める数理計画問題を考える。

本稿では関数 $f$ と $g_j, h_k$ に非線形な関数が含まれるものとし、実行可能集合を

$$D = \{\, \boldsymbol{x} \in \Re^n \mid g_i(\boldsymbol{x}) \leq 0,\ i = 1,\cdots,m\ ;\ h_j(\boldsymbol{x}) = 0,\ j = 1,\cdots,l \,\}$$

で表す。したがって、対象となる問題は次の非線形計画問題として定式化できる：

$$\left| \begin{array}{ll} \min. & f(\boldsymbol{x}) \\ \text{s.t.} & \boldsymbol{x} = (x_1,\cdots,x_n) \in D \end{array} \right. \tag{1}$$

この問題 (1) に対して、

$$f(\boldsymbol{x}^*) \leq f(\boldsymbol{x}) \qquad \forall \boldsymbol{x} \in D$$

を満足する $\boldsymbol{x}^* \in D$ を大域的最適解という [2]。問題 (1) には一般に $\boldsymbol{x}^*$ 以外の局所的最適解も存在する。例えば、$D$ が矩形の場合であっても $f$ が次の例のように複数の極値をもつ場合、図 1 のように 4 つの局所的最適解が存在する。

$$f(\boldsymbol{x}) = \sum_{i=1}^{n} (0.3{x_i}^4 + 0.4{x_i}^3 - 1.2{x_i}^2) + 10$$

この例では、大域的最適解は　$\boldsymbol{x}^* = (-2, -2)$, 大域的最適解 3.6 となる。

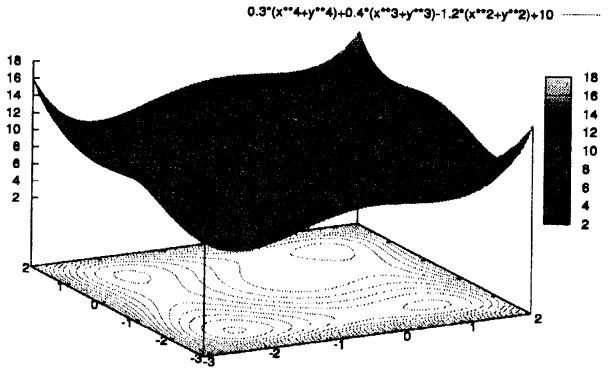


図 1: 2 次元多峰関数の例

# 3 非線形最適化のヒューリスティクス

## 3.1 Nelder-Mead法

非線形最適化の手法は、大まかに分けると関数の微分を用いる手法（勾配法）と用いない手法（直接探索法）がある。本研究では直接探索法であるNelder-Mead法を用いる。Nelder-Mead法は$D$上に与えられる$n$次元単体の各端点における目的関数$f$の値のみを用いて最適化を行う。与えられた初期単体に対して、終了条件が満たされるまで、Reflect, Expand, Contract, Shrinkの各基本操作を行い、単体を更新して、解を改善していく。

以下にNelder-Mead法のアルゴリズム[5]を示す。

**Step 0** 単体の各頂点を目的関数の値順に$f(\boldsymbol{x}_1) \leq f(\boldsymbol{x}_2) \leq \cdots \leq f(\boldsymbol{x}_{n+1})$とし、終了条件を満たしていなければStep 1へ、満たしていれば点$\boldsymbol{x}_1$を解として終了。

**Step 1** (Try to Reflect the simplex)

$\boldsymbol{x}_0$：点$\boldsymbol{x}_1 \cdots \boldsymbol{x}_n$による重心

$\boldsymbol{x}_r$：点$\boldsymbol{x}_{n+1}$の、重心$\boldsymbol{x}_0$に関して対称な方向へ移した点

$$\boldsymbol{x}_r := \boldsymbol{x}_0 + \alpha(\boldsymbol{x}_0 - \boldsymbol{x}_{n+1})$$

を求める。

**Step 2**

**Case 1** (Accept Reflect)

$f(\boldsymbol{x}_r) < f(\boldsymbol{x}_n)$であれば

$$\boldsymbol{x}_{n+1} := \boldsymbol{x}_r$$

とし、Step 0へ。

**Case 2** (Expand)

$f(\boldsymbol{x}_r) < f(\boldsymbol{x}_1)$であれば点$\boldsymbol{x}_{n+1}$と対称な方向に更に伸ばした点$\boldsymbol{x}_e$

$$\boldsymbol{x}_e := \boldsymbol{x}_0 + \gamma(\boldsymbol{x}_r - \boldsymbol{x}_0)$$

を求め、$f(\boldsymbol{x}_e) < f(\boldsymbol{x}_r)$であれば

$$\boldsymbol{x}_{n+1} = \boldsymbol{x}_e$$

とし、Step 0へ、それ以外は

$$\boldsymbol{x}_{n+1} := \boldsymbol{x}_r$$

とし、Step 0へ。

**Case 3** (Contract)

$f(\boldsymbol{x}_r) \geq f(\boldsymbol{x}_n)$のとき

**Case 3-1** (Outside Contract)
$f(\boldsymbol{x}_r) < f(\boldsymbol{x}_{n+1})$ であれば

$$\boldsymbol{x}_c := \boldsymbol{x}_0 + \beta(\boldsymbol{x}_r - \boldsymbol{x}_0)$$

を求め、Step 3へ。

**Case 3-2** (Inside Contract)
それ以外のとき、

$$\boldsymbol{x}_c := \boldsymbol{x}_0 + \beta(\boldsymbol{x}_{n+1} - \boldsymbol{x}_0)$$

を求め、Step 3へ。

**Step 3**

**Case 1** (Accept Contract)
$f(\boldsymbol{x}_c) < \min\{f(\boldsymbol{x}_r), f(\boldsymbol{x}_{n+1})\}$ であれば、

$$\boldsymbol{x}_{n+1} := \boldsymbol{x}_c$$

とし、Step 0へ。

**Case 2** (Shrink)
$f(\boldsymbol{x}_c) \geq \min\{f(\boldsymbol{x}_r), f(\boldsymbol{x}_{n+1})\}$ 単体の全ての辺を点 $\boldsymbol{x}_1$ 方向に縮める、すなわち

$$\boldsymbol{x}_i := \boldsymbol{x}_1 + \delta(\boldsymbol{x}_i - \boldsymbol{x}_1) \quad (i \neq 1)$$

とし、Step 0へ。 □

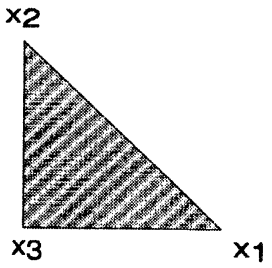


図 2: 初期単体

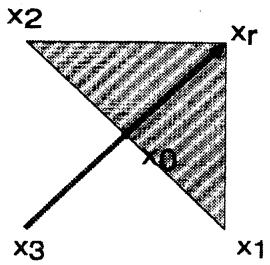


図 3: Reflect

Nelder-Mead法の終了条件には、目的関数値の評価回数や、単体の最良点と最悪点の目的関数値の差等が用いられる [5]。本研究では単体の体積によって終了判定を行う。Nelder-Mead法の各操作による単体への変形の種類で、変形後の単体 $S'$ の体積 $V(S')$ は以下のように決まる：

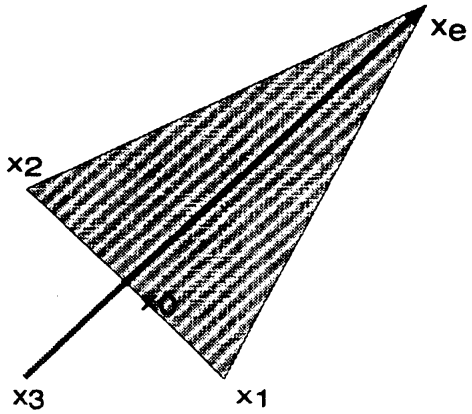


図 4: Expand

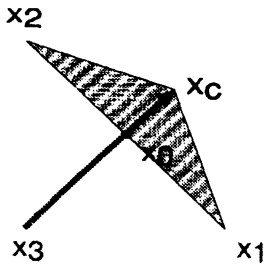


図 5: Outside Contract

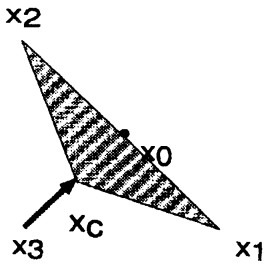


図 6: Inside Contract

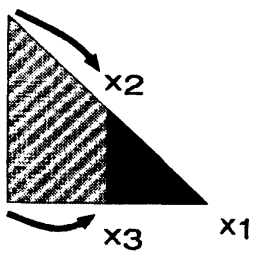


図 7: Shrink

$$\frac{V(S)}{V(S')} = \begin{cases} \alpha & \text{Reflect} \\ \beta & \text{Inside Contract} \\ \alpha \cdot \beta & \text{Outside Contract} \\ \alpha \cdot \gamma & \text{Expand} \\ \delta^n & \text{Shirink} \end{cases}$$

これを基に、初期単体の体積より十分小さくなったとき、具体的には許容誤差 $\epsilon > 0$ に対して $\frac{V(S)}{V(S')} < \epsilon$ が満たされればアルゴリズムを終了する。

### 3.2 マルチスタート法

Nelder-Mead 法によって得られる解は局所的最適解にすぎない。そこで本研究では、複数の異なる点を基準に初期単体を与えて、Nelder-Mead 法を複数回実行し、解を改善していくマルチスタート法を併用する。初期単体をランダムに与える通常のマルチスタート法では大域的最適解を得る保証はないので、Nelder-Mead 法を体系的に制御し、より精度の高い解を得る方法を提案する。

## 4 分枝限定法によるスタート制御

対象とする問題は、目的関数 $f$ に多峰性を想定し、実行可能領域は $D = \Re^n$ とする。解の探索領域は、基点 $s_0$ から各方向成分にある長さ isz だけ伸ばした単体上とし、この単体上で $f$ はリプシッツ連続であることを仮定する。以下に、マルチスタート単体法と、リプシッツ条件に基づいた分枝限定法で、この探索領域から $f$ の大域的最小点を求める方法を述べる。

### 4.1 リプシッツ条件

ある 2 点 $x, y$ において、関数値の差が、変数値の差に比例する値で抑えられるとき、すなわち

$$\exists L > 0 \quad \forall x, y \in \Re^n \quad \| f(x) - f(y) \| < L \cdot \| x - y \|$$

を満たすとき、$f$ をリプシッツ連続であるといい、$L > 0$ をリプシッツ定数 (Lipschitz constant) という。ある閉領域上で成り立つときは局所リプシッツ連続であるという。

探索領域である単体上で、目的関数がリプシッツ連続であれば、以下を単体上での $f$ の下界値とすることができる：

$$\max(l^{\max})(-L) + f(v_{\max})$$

ここで、$v_{\max}$ は単体の端点の中で、$f$ の値が最も大きい点を表し、$l^{\max}$ は $v_{\max}$ に接続する単体の辺の長さの集合を表す。

### 4.2 分枝限定法

提案するアルゴリズムは、Nelder-Mead 法とリプシッツ定数によって定まる下界値を用いて以下のように記述できる。

**Step 0.1** 基点 $s_0$ と単体の各辺の長さ isz から探索領域となる単体を作る。

**Step 0.2** 探索領域となる単体上で Nelder-Mead 法を実行し、得られる解における $f$ の値を暫定値 $V$ とする。

**Step 0.3** 単体を、最長の辺で二等分する。

**Step 0.4** 分割された単体それぞれについて、その単体上での $f$ の下界値を求め、下界値が $V$ よりも小さい値であれば探索候補に加える。

**Step 1** 探索候補の中で、下界値が最も小さいものを選択し、その単体を初期単体として Nelder-Mead 法を実行する。得られた解の値が $V$ よりも良い値であれば Step 2 へ、そうでなければ Step 3 へ。

**Step 2** 得られた解で暫定値 $V$ を更新し、探索候補の中で下界値が $V$ 以上の値になっているものを探索候補から外す。Step 3 へ。

**Step 3** Nelder-Mead 法を実行した単体を、最長の辺で二等分する。Step 4 へ。

**Step 4** 分割された単体それぞれについて、その単体内での $f$ の下界値を求め、下界値が $V$ 以下の値であれば探索候補に加える。Step 5 へ。

**Step 5** 探索候補が空でなければ Step 1 へ。空ならば終了。 □

## 5 数値実験

2次元のテスト問題1～3に対して提案する方法1と、探索領域内からランダムに初期点を与えて、その点を基点に探索領域と同じ大きさの単体を初期単体として Nelder-Mead 法を実行していく方法2を比較する。後者は、頂点評価回数が前者の実行結果と等しくなるまで繰り返し初期点を与えて実行した。Nelder-Mead 法の各基本操作に必要なパラメータは、$\alpha = 1, \beta = 0.5, \gamma = 2, \delta = 0.5$ とし、終了条件の許容誤差は方法1で $\epsilon = 2^{-3}$、方法2では $\epsilon = 2^{-10}$ とした。これは、方法1では反復が進むに従って Nelder-Mead 法に渡す初期単体が小さくなるのに対し、方法2では毎回の反復で同じ大きさの初期単体に Nelder-Mead 法を実行する点を考慮したものである。各問題のリプシッツ定数 $L$ は、探索領域で最大の導関数の絶対値とした [4]。実装には GNU Octave を用いた。

## 問題 1

$$f_1(\boldsymbol{x}) = \sum_{i=1}^{n}(0.3{x_i}^4 + 0.4{x_i}^3 - 1.2{x_i}^2) + 10$$

基点 $s_0 = (-3, -3)$, $\quad$ isz $= 5$, $\quad L = 28.8$

探索領域内での大域的最適解 3.6 $\quad \boldsymbol{x} = (-2, -2)$

## 問題 2

$$f_2(\boldsymbol{x}) = \sum_{i=1}^{n}({x_i}^3 - 3x_i) + 2$$

基点 $s_0 = (-1.5, -1, 5)$, $\quad$ isz $= 5$, $\quad L = 37.5$

探索領域内での大域的最適解 $-2$ $\quad \boldsymbol{x} = (1, 1)$

## 問題 3 [4]

$$\begin{aligned} f_3(\boldsymbol{x}) \quad = \quad & -25\exp\{-20(x_1 - 0.3)^2 - 18(x_2 - 0.7)^2\} \\ & -23\exp\{-17(x_1 - 0.65)^2 - 19(x_2 - 0.25)^2\} \end{aligned}$$

基点 $s_0 = (0, 0)$, $\quad$ isz $= 1$, $\quad L = 52.93$

探索領域内での大域的最適解 25.062 $\quad \boldsymbol{x} = (0.301, 0.699)$

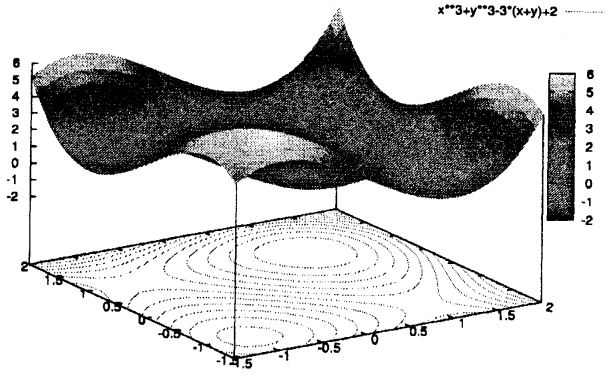


図 8: 問題 2

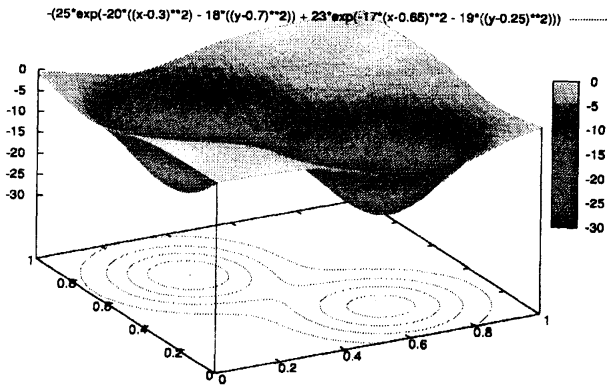


図 9: 問題 3

| 方法 | | 1 | 2 |
|---|---|---|---|
| 計算結果 | $x_1$ | -1.9999 | -2.0007 |
| | $x_2$ | -1.9997 | -1.9976 |
| | $f$ | 3.6000 | 3.6000 |
| 大域的最適解との差 | $x_1$ | 0.0001 | 0.0007 |
| | $x_2$ | 0.0003 | 0.0024 |
| | $f$ | 0.0000 | 0.0000 |
| CPU Time | (秒) | 84.454 | 75.116 |
| 分枝回数 | (回) | 13381 | - |
| 頂点評価回数 | (回) | 353121 | 353138 |

表 1: 問題 1 の実験結果

| 方法 | | 1 | 2 |
|---|---|---|---|
| 計算結果 | $x_1$ | 1 | 0.98598 |
| | $x_2$ | 1 | 1.00114 |
| | $f$ | -2 | -1.9994 |
| 大域的最適解との差 | $x_1$ | 0 | 0.0140191 |
| | $x_2$ | 0 | 0.0011416 |
| | $f$ | 0 | 0.0006 |
| CPU Time | (秒) | 68.023 | 59.067 |
| 分枝回数 | (回) | 12189 | - |
| 頂点評価回数 | (回) | 291083 | 291089 |

表 2: 問題 2 の実験結果

## 6 まとめと課題

本稿では、非線形で多峰性を有する問題に対し、Nelder-Mead 法を用いた二つの方法で大域的な最適化を試みた。問題 1 と 2 については大きな差は見られなかったが、問題 3 のように適切なリプシッツ定数が与えられた問題に対しては、分枝限定法が有効に働き、方法 2 よりも解の精度、実行時間の両方で勝っていることが確認できた。

課題としては、１０次元程度の問題に対して既に Nelder-Mead 法が有効に働かない場合があることと、方法 1 の終了条件である。問題 1 と 2 においても、分枝限定法が収束するよりも遥かに早い段階で良い解が得られているものと推測される。しかし、反復が進むにつれて、実際の下界値に対してリプシッツ定数が相対的に大きくなっていくせいで、探索領域を効果的に減らすことができていない。適当な反復回数で強制終了させるなど、新たなヒューリスティクスを用いた実用性の向上が期待される。

| 方法 | | 1 | 2 |
|---|---|---|---|
| 計算結果 | $x_1$ | 0.29932 | 0.28915 |
| | $x_2$ | 0.69873 | 0.62964 |
| | $f$ | -25.061 | -22.736 |
| 大域的最適解との差 | $x_1$ | 0.00168 | 0.011847 |
| | $x_2$ | 0.00027 | 0.017032 |
| | $f$ | 0.001 | 0.19376 |
| CPU Time | (秒) | 0.018697 | 0.16398 |
| 分枝回数 | (回) | 62 | - |
| 頂点評価回数 | (回) | 485 | 485 |

表 3: 問題 3 の実験結果

# 参考文献


[1] Nelder, J. A., R. Mead., "A simplex method for function minimization", *Computer Journal*, Vol.7, (1965), pp.308-313

[2] ORWiki (`http://www.orsj.or.jp/~wiki/wiki/index.php/`)

[3] Pedroso, J. P., "Simple meta-heuristics using the simplex algorithm for non-linear programming". (Departamento de Ciencia de Computadores, 2007).

[4] 正道寺勉, 「多変数多峰性関数に対する最適値探索法の研究」. 『日本オペレーションズ・リサーチ学会論文誌』, Vol.20, No.4, (1977), pp.311-320

[5] Singer, S., S. Singer., "Efficient Implementation of the Nelder-Mead Search Algoritm". *Applied Numerical Analysis & Computational Mathematics*, Vol.1, No.2, (2004), pp.524-534

[6] Storn, R., K. Price., "Differential evolution - a simple and efficient heuristic for global optimization over continious spaces". *Journal of Global Optimization*, Vol.11, (1997), pp.341-359

[7] Walters, F. H., L. R.Parker Jr, "*Sequential Simplex Optimization:A Technique for Improving Quality and Productivity*" in *Research,Development,and Manufacturing.* CRC Pr I Llc, (1991).