CH-010

RE アルゴリズム：非凸最小二乗法問題の大域的最適化手法

RE Algorithm: Global Optimization Method for Nonconvex Least Squares Problems

伊神　大貴[†]　　山崎 俊彦[†]　　相澤 清晴[†]
Daiki Ikami　Toshihiko Yamasaki　Kiyoharu Aizawa

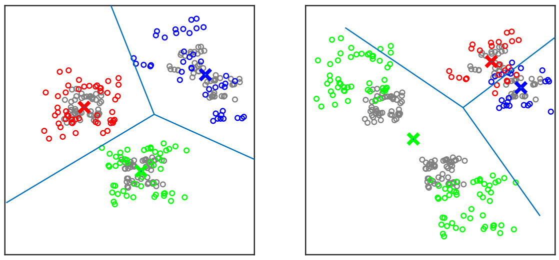
1. はじめに

多くのコンピュータビジョンや機械学習の問題は最適化問題を解くことに帰着する．問題を凸最適化問題で定式化可能な場合，局所解はただ一つ存在し大域的最適解と一致するため，勾配法などにより大域的最適解を求めることが可能である．しかし，局所解を複数持つような非凸最適化問題で定式化された場合，凸最適化手法では局所解を発見することしかできない．

非凸最適化問題の大域的最適化手法は非常に多く提案されている．一つは良い初期解を利用することで，k-means クラスタリングや Iterative Closest Point（ICP）アルゴリズムなどでは良い初期解を求める手法が提案されている [2, 20]．しかし，これらの初期解が効果的でない場合や，そもそも良い初期解を与えるのが困難な場合も多い．その他の大域的最適化手法として，確率的探索や多点探索に代表される，ヒューリスティックな最適化手法が数多く存在する．具体的なアルゴリズムとしては遺伝的アルゴリズム [17]，群粒子最適化 [10]，焼きなまし法 [12] などが挙げられる．実際にこれらの手法は k-means クラスタリングや ICP アルゴリズムで用いられており [5, 13, 16, 24, 21]，特に低次元パラメータの最適化問題では有効である．しかし高次元パラメータの最適化問題では有効に働かないことが多く，また一般に良い解を得るためには，計算コストが増大する傾向にある．

本稿では，最小二乗法の非凸最適化問題に限定し，効率の良い（準）大域的最適化アルゴリズムを提案する．アルゴリズムの基となる重要な概念として，本稿では局所最適解の大域的最適化についての指標である Residual Expansion（RE）収束を提案する．RE 収束は局所最適解であるという条件の下で，データをどの程度残差方向に移動させることが可能であるかを表す．図 1 は k-means クラスタリングの結果と，残差方向に移動したデータ点である．図 1a の解は移動したデータ点に対しても収束しているが，図 1b では解は収束していない．我々は最小二乗法問題において，データを最も大きく移動しても局所最適解を保つ解が大域的最適解に近いという仮説を立て，実際に一次元四次関数の最小化ではこの仮説が成り立っていることを証明した．

また，この仮説に基づき，できるだけデータを大きく残差方向に移動しても収束する解を高速に見つけるためのヒューリスティックなアルゴリズムを提案する（Residual Expansion Algorithm, RE アルゴリズム）．RE アルゴリズムは多点探索や確率的探索に基づかない決定的な最適化手法であり，多くの問題で高速かつ高精度な大域的最適化を実現する．



(a) 大域的最適解であるクラスタリング結果　(b) 局所解であるクラスタリング結果

図 1: 異なる RE 定数を持つ K-means クラスタリングの結果．グレーの点は元のデータ点を示し，赤，緑，青の点はそれぞれのクラスタ中心から RE により広げられたデータ点である．図 1a は $\alpha = 1$ で RE を行っても収束しているが，図 1b では収束していない．この例では，大きな RE 定数を持つ解が大域的最適解となっている．RE や RE 定数の詳しい説明は第 2 章で述べる．

我々の貢献は以下のとおりである．

1. 我々は新しい収束の概念として，RE 収束を提案する．これは局所最適解の大域的最適性と関連があると考えられ，一部の最適化問題について理論的な説明を与えた．
2. 非凸最小二乗法問題に適用可能な大域的最適化手法である RE アルゴリズムを提案する．RE アルゴリズムは高速，高精度かつ，非常に簡易な実装が可能である．
3. 多くの問題で RE アルゴリズムが高精度な大域的最適化を実現することを実験結果により示す．具体的には k-means クラスタリング，ICP アルゴリズム，最適直積量子化（Optimized Product Quantization, OPQ），非負値行列分解（Nonnegative Matrix Factorization, NMF） の四つの問題での実験結果を示す．

2.Residual expansion (RE)

ここでは残差拡大（Residual Expansion, RE）という，観測データを残差方向へ移動させる操作と，RE 収束という局所解の性質について定義する．更に RE 収束と大域的最適解の関係性について説明する．

本稿では以下の形で定式化される非凸最小二乗問題

[†]東京大学情報理工学系研究科電子情報学専攻

を考える．

$$E(\theta) = \frac{1}{2}\|\mathbf{y} - \mathbf{f}(\theta)\|_2^2. \tag{1}$$

RE と RE 収束の定義は以下の通りである．

定義 2.1 (RE)**.** θ^* を式 *(1)* のある局所最適解とし，以下のような目的関数 $E_\alpha(\theta)$ を考える*:*

$$E_\alpha(\theta) = \frac{1}{2}\|\hat{\mathbf{y}} - \mathbf{f}(\theta)\|_2^2. \tag{2}$$

$\hat{\mathbf{y}}$ は観測データ $\mathbf{y}$ を残差方向に大きさ α で移動させることで以下のように得られる．

$$\hat{\mathbf{y}} = \mathbf{y} + \alpha(\mathbf{y} - \mathbf{f}(\theta^*)), \tag{3}$$

この観測点を残差方向に移動させ，$E_\alpha(\theta)$ を作る操作を *RE* とする*.*

定義 2.2 (α-RE 収束)**.** θ^* が $E_\alpha(\theta)$ の局所解であるようなある定数 $\alpha \geq 0$ が存在するとき，θ^* は *α-RE* 収束であるとする．特に，そのような最大の定数 α を *RE* 定数とする[1]．

2.1. 制約無し微分可能な問題

ここでは最も単純な，制約無しの微分可能な最小二乗問題を考える．ある局所最適解 θ^* に対して，$E_\alpha(\theta)$ の一次微分と二次微分を考える．

$$\nabla E_\alpha(\theta^*) = (1+\alpha)\mathbf{J}^\top(\theta^*)(\mathbf{y} - \mathbf{f}(\theta^*)), \tag{4}$$

$$\nabla^2 E_\alpha(\theta^*) = \mathbf{J}^\top(\theta^*)\mathbf{J}(\theta^*) + (1+\alpha)\mathbf{S}(\theta^*). \tag{5}$$

ここで $\mathbf{J}$ はヤコビ行列であり，$\mathbf{S}(\theta)$ は

$$\mathbf{S}(\theta^*) = \sum_i \nabla^2 f_i(\theta^*)(y_i - f_i(\theta^*)) \tag{6}$$

で表される．式 (4) は θ^* が局所解であることから $\mathbf{0}$ であり，θ^* は常に $E_\alpha(\theta)$ の停留点であることを示している．よって，θ^* が局所最適解であるための必要十分条件は $\nabla^2 E_\alpha(\theta)$ が半正定値行列ということである．もし $\mathbf{S}$ が半正定値行列でなければ，$\nabla^2 E_\alpha(\theta)$ が半正定値行列でなくなる $\alpha \leq 0$ が存在し，その最大の α が RE 定数となる．

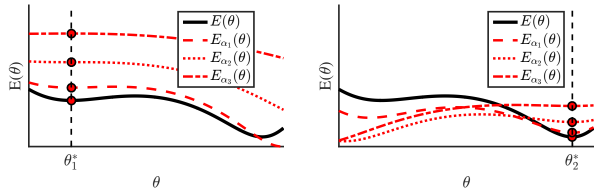
図 2 は $E_\alpha(\theta)$ の例を示している．RE は目的関数を θ^* の周りで押し上げる効果があり，α が十分に大きければ θ^* 局所最適解でなくなる．

一次元四次関数の最小化: ここでは一次元四次関数の最小化問題，特に以下のように最小二乗法で定式化される問題を考える．

$$E(\theta) = \frac{1}{2}\left(\left(y_1 - \theta^2\right)^2 + \left(y_2 - \theta\right)^2\right). \tag{7}$$

式 (7) が二つの局所最適解 θ_1, θ_2 を持つとき，以下の定理が成り立つ．

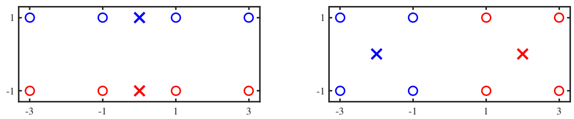
[1] θ^* が $\alpha \geq 0$ に対して常に $E_\alpha(\theta^*)$ の局所解である場合，RE 定数は ∞ とする．



(a) 局所解 θ_1^* と RE により得られる関数 $E_\alpha(\theta)$.

(b) 局所解 θ_2^* と RE により得られる関数 $E_\alpha(\theta)$.

図 2: 異なる局所解 θ_1^*，θ_2^* における，RE により得られる $E_\alpha(\theta)$．赤い破線はそれぞれ $\alpha_1 < \alpha_2 < \alpha_3$ を用いた RE により得られる目的関数である．θ_1^* における $E_{\alpha_3}(\theta)$ のように，十分に大きな α を用いて RE を行うと，元関数の局所解が $E_\alpha(\theta)$ では局所解でなくなることがある．



(a) 悪い局所解である k-means クラスタリング結果

(b) 大域的最適解である k-means クラスタリング結果

図 3: $k = 2$ による異なる k-means クラスタリング結果．図 3a では $\alpha = \infty$ の RE 定数を持つが，悪い局所解となっている．一方，図 3b の RE 定数は 1 であるが，大域的最適解である．

定理 1. θ_1 と θ_2 をそれぞれ式 *(7)* での局所最適解とし，それぞれの *RE* 定数を α_1，α_2 とする．もし $\alpha_1 > \alpha_2$ ならば θ_1 が大域的最適解であり，そうでなければ θ_2 が大域的最適解となる．

証明．付録を参照. □

2.2.α-RE 収束と大域的最適解の関係

我々の仮説である，大きい RE 定数を持つ解が大域的最適解に近いという仮説がどのような場合に成り立つかは十分に明らかでなく，また成り立たない問題も存在する．例えば k-means クラスタリングでは図 3 に示すような反例が存在する．しかし，大きな RE 定数を持つ解を発見するための RE アルゴリズムは実問題で非常に有効に働く．

3.RE アルゴリズム

ここでは大きな RE 定数を持つ解を発見するためのアルゴリズムである，RE アルゴリズムについて説明する．厳密に最も多きな RE 定数を持つ解を発見することは困難であるため，我々はヒューリスティックなアルゴリズムを提案する．

RE アルゴリズムはパラメータ更新ステップと RE ステップの二つのステップからなる．RE ステップではデータを残差方向に移動させ，これは目的関数を現在の解の周りで押し上げることとも言える．パラメータ更新ステップでは，元の目的関数式 (1) を最小化する

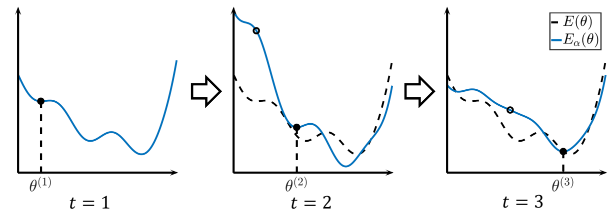


図 4: RE アルゴリズムの振る舞いの概念図．RE アルゴリズムはパラメータ更新ステップと RE を交互に繰り返し最適化を行う．

Algorithm 1 RE アルゴリズム．

Initialize: $t = 1, \hat{\mathbf{y}}^{(1)} = \mathbf{y}, \mathbf{r}^{(1)} = \mathbf{0}$

while not converged **do**
 θ の更新（式 (8) による）
 $\mathbf{r}^{(t+1)} = p^{(t)}\left(\mathbf{y} - \mathbf{f}\left(\theta^{(t+1)}\right)\right) + \left(1 - p^{(t)}\right)\mathbf{r}^{(t)}$
 $\hat{\mathbf{y}}^{(t+1)} = \mathbf{y} + \alpha^{(t)}\mathbf{r}^{(t+1)}$
 $t = t + 1$
end while

Output: θ

代わりに，我々は以下の関数 $E_t(\theta)$ を最小化する．

$$E_t(\theta) = \frac{1}{2}\|\hat{\mathbf{y}}^{(t)} - \mathbf{f}(\theta)\|_2^2, \tag{8}$$

$\hat{\mathbf{y}}^{(t)}$ は以下で与えられる．

$$\hat{\mathbf{y}}^{(t)} = \mathbf{y} + \alpha^{(t)}\mathbf{r}^{(t)}, \tag{9}$$
$$\mathbf{r}^{(t)} = p^{(t)}(\mathbf{y} - \mathbf{f}(\theta^{(t)})) + (1 - p^{(t)})\mathbf{r}^{(t-1)}. \tag{10}$$

ここで α と $0 < p \leq 1$ はそれぞれ拡大率とモーメンタムのパラメータである．モーメンタムは高性能な大域的最適化のために必要であり，また後述するようにアルゴリズムが発散することを防ぐためにも必要である．

RE アルゴリズムは式 (8) の最小化によるパラメータ更新ステップと，式 (9)，(10) による RE ステップを交互に繰り返す． その際，最初に大きな $\alpha^{(1)}$ を用い，徐々に $\alpha^{(t)}$ を小さくしていくことで，大きな RE 定数を持つ解を発見することを期待する．提案手法のアルゴリズムを Algorithm 1 に示す．

RE アルゴリズムは大きな RE 定数を発見するためのアルゴリズムであるが，別の視点から大域的最適化が可能であることを説明できる．RE ステップは目的関数を現在の解の周りで押し上げることになるため，RE アルゴリズムの振る舞いは図 4 に示すようになり，目的関数の押し上げにより悪い局所解を抜け出すことが可能になると期待できる．

3.1. 収束のための RE アルゴリズムのパラメータ設定

RE アルゴリズムは残差方向への拡大率 α とモーメンタムに関する p の二つのパラメータを持つ．$\alpha^{(t)}$ は大きな値から徐々に減らしていき，最終的に 0 にする．$\alpha = 0$ の場合，残差拡大は行われず，元の最小二乗問題を最小化することになる．したがって RE アルゴリズムは元の最小二乗問題が収束するアルゴリズムを持っている場合，収束が保証される．

RE アルゴリズムの収束は保証されているが，適切な α と p を用いないとベクトル $\mathbf{r}$ が非常に大きくなりアルゴリズムが不安定になる．$\mathbf{r}^{(t+1)}$ について，以下の式が得られる．

$$\begin{aligned}\left\|\mathbf{r}^{(t+1)}\right\|_2^2 &= \left\|p\left(\mathbf{y} - \mathbf{f}\left(\theta^{(t+1)}\right)\right) + (1-p)\,\mathbf{r}^{(t)}\right\|_2^2 \\ &\sim (1 - p - \alpha p)^2 \left\|\mathbf{r}^{(t)}\right\|_2^2.\end{aligned} \tag{11}$$

ここで $\mathbf{f}\left(\theta^{(t+1)}\right) \sim \hat{\mathbf{y}}^{(t)}$ を近似として用いた．式 (11) から $(1 - p - \alpha p)^2 \leq 1$ となる α と p を用いることで，$\mathbf{r}$ の発散を防ぐことができることが示唆される．

3.2.RE アルゴリズムの特色

RE アルゴリズムはパラメータ更新ステップと RE ステップから成るが，パラメータ更新は単純な最小二乗法である．そのため元問題を解くソースコードが存在する場合，RE アルゴリズムの実装は RE ステップについて，数行追加するだけで実現できる．

また，RE ステップの計算量は一般にパラメータ更新ステップと比べて小さいため，RE アルゴリズムの 1 イテレーションの計算量は元問題を解く際の 1 イテレーションの計算量とほとんど変わらない．したがって SA や GA のような多点探索や確率的探索手法と比べて，RE アルゴリズムはパラメータ設定によっては高速な大域的最適化が実現可能である．

RE アルゴリズムは局所解を見つけるアルゴリズムが存在する最小二乗問題であれば適用可能であるため，非常に多くの問題に適用可能である．また，実験結果で示すように，パラメータ数の多い最適化問題でも良い大域的最適化を実現することができる．

4. 実装の詳細

RE アルゴリズム（Algorithm 1）のパラメータ α と p について，以下の式を用いた．

$$\alpha^{(t)} = (1 - \mu^{(t)})/\mu^{(t)}, \tag{12}$$
$$p^{(t)} = \mu^{(t)}/(1 + \mu^{(t)}). \tag{13}$$

これは第 3.1 章で述べたアルゴリズムの安定に必要な条件 $(1 - p - \alpha p)^2 \leq 1$ を常に満たし，また実験的によい大域的最適化を実現する．また，パラメータ μ に関しては，$\mu^{(t+1)} = \min(\rho\mu^{(t)}, 1)$ を用いた．ここで $\rho = \exp(-\log(\mu^{(1)})/T)$ を用いた．これにより T イテレーション時に $\mu^{(T)} = 1$ となり，RE が行われなくなる．最終的な提案手法のパラメータは $\mu^{(1)}$ と T の二つのみである．

パラメータ θ の更新では，1 イテレーションの交互最適化を用いた．例えば k-means クラスタリングでは，クラスタ中心と割り当てを収束するまで更新するのではなく，一回のみ更新を行った．今回実験で扱う全ての最適化問題は交互最適化が可能である．

表 1: 合成データでの k-means クラスタリングによる実験結果．異なる $\mu^{(0)}$ と T を用いた RE アルゴリズムの平均相対エラーを示す．

(a) 合成データ A（$k = 100$）．

	$\mu^{(0)} = 0.5$	$\mu^{(0)} = 0.2$	$\mu^{(0)} = 0.1$	$\mu^{(0)} = 0.01$
$T = 30$	0.905	0.894	0.902	0.921
$T = 100$	0.854	0.856	0.862	0.876
$T = 300$	0.843	0.844	0.846	0.854
$T = 1,000$	0.837	0.839	0.840	0.843

(b) 合成データ B（$k = 10$）．

	$\mu^{(0)} = 0.5$	$\mu^{(0)} = 0.2$	$\mu^{(0)} = 0.1$	$\mu^{(0)} = 0.01$
$T = 30$	2.699	1.758	1.493	0.998
$T = 100$	2.784	1.209	0.789	0.630
$T = 300$	2.722	1.036	0.708	0.552
$T = 1,000$	2.749	0.994	0.630	0.552

5. 実験

ここでは RE アルゴリズムの性能を評価するため，k-means クラスタリング，点群の位置合わせ，OPQ，非負値行列分解の 4 つの最小二乗法で定式化される非凸最適化問題での評価実験を行う．点群の位置合わせで用いる Go-ICP [25] とそこでの比較実験を除き，全てのソースコードは MATLAB により実装されている．Go-ICP のソースコードは C++で公開されているものを用いた[2]．

5.1.K-means クラスタリング

背景

K-means クラスタリングは最も代表的なクラスタリング手法の一つであり，量子化 [9] や教師無し学習 [6]，またセグメンテーション [1] などの分野で使われる基礎的な技術である．K-means クラスタリングはデータベクトル $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$ を，代表する k 個のクラスタ中心のいずれかに割り当てる最適化問題で，以下のように定式化される．

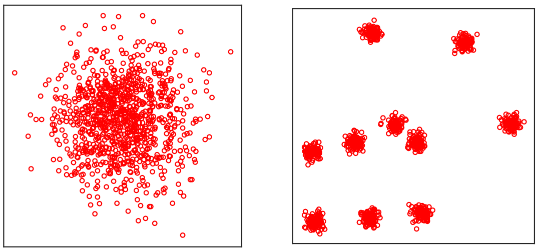
$$\min_{\mathbf{C},\mathbf{Z}} \frac{1}{2}\|\mathbf{X} - \mathbf{CZ}\|_F^2 \quad (14)$$
$$\text{s.t.} \quad z_{ij} = \{0, 1\}, \|\mathbf{z}_i\|_1 = 1,$$

ここで $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d\times n}$ はデータ行列，$\mathbf{C} = [\mathbf{c}_1, \ldots, \mathbf{c}_k] \in \mathbb{R}^{d\times k}$ はクラスタ中心を表す行列，$\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_n] \in \mathbb{R}^{k\times n}$ は割り当てを表す行列である．

最も代表的な最適化手法は Lloyd のアルゴリズム [15] で，これはクラスタ中心の更新と割り当ての更新を交互に行う手法，つまり $\mathbf{Z}$ と $\mathbf{C}$ について片方の変数を固定しもう片方の変数を更新する交互最適化を行うものである．別の手法として，Hartigan のアルゴリズム [8] が挙げられる．この手法で得られる局所解集合は Lloyd のアルゴリズムで得られる局所解集合の部分集合であり，Lloyd のアルゴリズムよりもよいクラスタリングを得られる手法である [23, 22]．また，初期解を求める手法としてはランダムにデータからサンプリングする手法の他に，k-means++ [2] がよい初期解を得られるとしてよく利用されている．

実験の詳細

比較手法として，ランダムな初期解と k-means++による初期解それぞれに対して Lloyd のアルゴリズムを用いた．また，Hartigan のアルゴリズムには k-means++を初期解として Lloyd のアルゴリズムにより得られた解を初期解として与えた．計算時間は Lloyd のアルゴリズムによる解を得るものも含めて報告している．RE アルゴリズムには Lloyd のアルゴリズムとランダムな初期解を用いた．エラーとして式 (14) による目的関数の値を用い，k-means++で得られた値を 1 とする相対エラーを載せる．



(a) 合成データ A.　(b) 合成データ B.

図 5: 1000 点，2 次元からなる合成データ．

合成データによる実験結果

まず RE アルゴリズムのパラメータ $\mu^{(1)}$ と T の影響を調べるために，図 5 にある二つの合成データでの実験を行った．k-means クラスタリングを異なる初期解から 50 回行い，その平均エラーを表 1 に載せる．大きな T を用いるほど，どのケースでも最適化性能が向上する．$\mu^{(1)}$ については，データセット B では小さい $\mu^{(1)}$ がよい性能であるが，データセット A では大きな $\mu^{(1)}$ のほうが性能が高い．これはデータセット B は局所解から抜け出すのに大きな RE が必要であるのに対し，データセット A では小さな RE でも局所解から抜け出すことが可能であるからだと考えられる．これらの結果から，同じ最適化問題であっても，データの分布により最適なパラメータ $\mu^{(1)}$ が異なることが分かる．

また，比較実験の結果を表 2 に載せる．K-means++はデータセット B では非常に効果的だが，データセット A では十分によいクラスタリングができていない．一方，Hartigan のアルゴリズムはデータセット A では最適化性能を向上させているが，データセット B では有効に働いていない．RE アルゴリズムは大きな T を用いた場合，どちらのデータセットでも最適化性能を大幅に向上させている．特にデータセット A では $T = 30$ でも k-means++や Hartigan のアルゴリズムを上回る最適化性能を，ほぼ同等の計算時間で達成している．

5.1.1. 実データでの実験結果

実データの実験では，cloud データ[3]と COIL20[18] を用いた．第 5.1 章と同様に 50 回の異なる初期解か

[2]http://iitlab.bit.edu.cn/mcislab/~yangjiaolong/go-icp/

[3]https://archive.ics.uci.edu/ml/datasets/Cloud

表 2: 合成データでの k-means クラスタリングの比較実験結果．各手法での平均/最小/最大の相対エラーと平均計算時間を示す．

(a) 合成データ A（$k = 100$）．

		相対エラー			計算時間
		Mean	Min	Max	[sec]
Random seeding		1.246	1.102	1.473	0.058
k-means++ [2]		1.000	0.944	1.081	0.244
Hartigan のアルゴリズム [8]		0.925	0.881	0.981	0.359
RE アルゴリズム（$\mu^{(0)} = 0.1$）	$T = 30$	0.902	0.875	0.942	0.258
	$T = 100$	0.862	0.846	0.873	0.780
	$T = 300$	0.846	0.836	0.856	2.29
	$T = 1,000$	0.840	0.831	0.850	7.61

(b) 合成データ B（$k = 10$）．

		相対エラー			計算時間
		Mean	Min	Max	[sec]
Random seeding		4.277	0.552	22.680	0.0194
k-means++ [2]		1.000	0.552	8.743	0.0271
Hartigan のアルゴリズム [8]		1.000	0.552	8.743	0.0429
RE アルゴリズム（$\mu^{(0)} = 0.1$）	$T = 30$	1.493	0.552	6.777	0.0699
	$T = 100$	0.789	0.552	2.500	0.176
	$T = 300$	0.708	0.552	2.500	0.473
	$T = 1,000$	0.630	0.552	2.500	1.51

表 3: 実データでの k-means クラスタリングの比較実験結果．各手法での平均/最小/最大の相対エラーと平均計算時間を示す．

(a) Cloud データ（$\mathbf{X} \in \mathbf{R}^{10 \times 1024}$，$k = 10$）．

		相対エラー			計算時間
		Mean	Min	Max	[sec]
Random seeding		1.255	1.003	1.438	0.0444
k-means++ [2]		1.000	0.920	1.097	0.0395
Hartigan のアルゴリズム [8]		0.994	0.920	1.093	0.0593
RE アルゴリズム（$\mu^{(0)} = 0.1$）	$T = 30$	0.980	0.920	1.031	0.0719
	$T = 100$	0.941	0.920	0.986	0.183
	$T = 300$	0.926	0.920	0.983	0.516
	$T = 1,000$	0.920	0.920	0.921	1.63

(b) COIL20（$\mathbf{X} \in \mathbf{R}^{1024 \times 1440}$，$k = 20$）．

		相対エラー			計算時間
		Mean	Min	Max	[sec]
Random seeding		0.999	0.953	1.076	2.22
k-means++ [2]		1.000	0.962	1.038	3.52
Hartigan のアルゴリズム [8]		0.990	0.960	1.021	8.07
RE アルゴリズム（$\mu^{(0)} = 0.1$）	$T = 30$	0.951	0.939	0.977	4.37
	$T = 100$	0.945	0.938	0.960	12.6
	$T = 300$	0.942	0.938	0.950	36.6
	$T = 1,000$	0.941	0.938	0.956	119

ら各手法でクラスタリングを行い，実験結果を表 3 に示す．Hartigan のアルゴリズムはどちらのデータセットでも大きな改善をしておらず，また k-means++は cloud データでは有効だが COIL20 ではランダムな初期解とほぼ同じ性能である．RE アルゴリズムはどちらのデータセットでも最適化性能を改善しており，実データでは非常に有効な手法だと考えられる．

5.2. 点群の位置合わせ

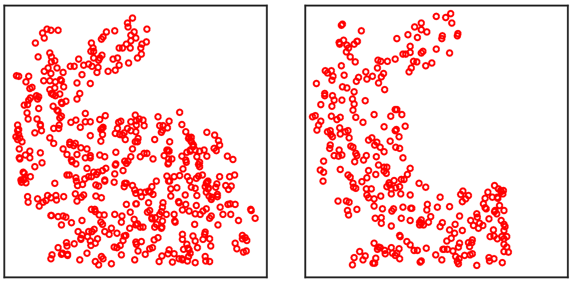
背景

点群の位置合わせはコンピュータビジョンにおける重要な基礎技術である．ここでは 3 次元の剛体変換による点群の位置合わせを考える．つまり，ソースの点群 $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{3 \times n}$ とターゲットの点群 $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_m] \in \mathbb{R}^{3 \times m}$ が与えられたとき，3 次元の回転と並進のパラメータを推定する問題である．ここでは特に point-to-point なコスト関数を用いた以下の最適化問題を考える．

$$\min_{\mathbf{R},\mathbf{t},\mathbf{Z}} \frac{1}{2} \|\mathbf{RX} + \mathbf{t}\mathbf{1}^\top - \mathbf{YZ}\|_F^2 \qquad (15)$$
$$\text{s.t.} \quad z_{ij} = \{0, 1\}, \|\mathbf{z}_i\|_1 = 1, \mathbf{R}^\top \mathbf{R} = \mathbf{I},$$

ここで $\mathbf{R} \in \mathrm{SO}(3)$ は回転行列，$\mathbf{t} \in \mathbb{R}^3$ は並進ベクトルであり，$\mathbf{Z} = [\mathbf{z}_1, \ldots, \mathbf{z}_n] \in \mathbb{R}^{k \times n}$ は割り当ての行列である．また，$\mathbf{I}$ と $\mathbf{1}$ はそれぞれ単位行列と，すべての要素が 1 であるベクトルを表す．

式 (15) の最適化手法として一般的に Iterative Closest Point (ICP) アルゴリズム [4] が用いられる．これは交互最適化手法であり，$\mathbf{Z}$ を固定して $\mathbf{R}$ と $\mathbf{t}$ の更新を行うステップと，$\mathbf{R}$ と $\mathbf{t}$ を固定して $\mathbf{Z}$ の更新を行う二つのステップを繰り返すことで局所最適解を求める手法である．大域的最適化を実現するために，確率的探索を用いる手法が提案されている（GA [21], PSO [24], and SA [16]）．また，最近では Yang らが大域的最適化手法として Go-ICP [25] を提案した．これは branch-and-bound アルゴリズムを用いることで理論的に大域的最適化を保証するが，計算コストが非常に大きいという欠点がある．



(a) ソース点群（500 点）．　(b) ターゲット点群（313 点）．

図 6: 点群位置合わせに用いたデータ．

実験の詳細と結果

回転角 ϕ とランダムな回転軸により回転行列を生成し，ソース点群からターゲット点群を生成し，標準偏差 $\sigma = 0.03$ によるガウスノイズを加える．回転角には $\phi = \pi/3, 5\pi/12, \pi/2$ を用いた．エラーとしては式 (15) による目的関数の値を用い，異なる点群に対して 50 回の平均エラーと成功回数（目的関数値が 1 より小さい

表 4: 3 次元点群の位置合わせの実験結果．異なる回転角から生成された 50 個の回転行列に対する，推定の成功回数と，合計 150 回の試行でのアルゴリズムの平均イテレーション数を示す．

		成功回数			イテレーション数
		$\phi = \pi/3$	$\phi = 5\pi/12$	$\phi = \pi/2$	
ICP アルゴリズム		26	4	1	46.7
RE アルゴリズム（$\mu^{(0)} = 0.1$）	$T = 30$	46	25	2	47.4
	$T = 100$	49	31	5	110.1
	$T = 300$	49	33	6	310.2
	$T = 1,000$	49	36	6	1008

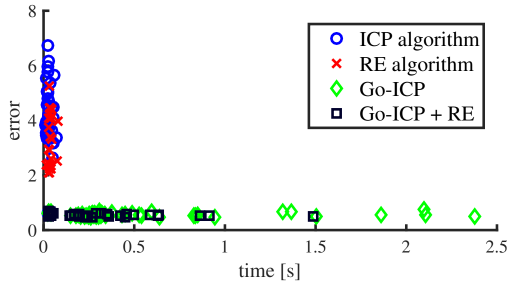


図 7: 点群の位置合わせの比較実験（$\phi = 5\pi/12$）．50 回の試行での目的関数値と計算時間をプロットしている．RE アルゴリズムでは $T = 30$ を用いた．ICP アルゴリズム，RE アルゴリズム，Go-ICP，Go-ICP + RE の平均計算時間はそれぞれ 0.0304，0.0347，0.551，0.231 秒である．

回数）を報告する．

比較手法として，通常の ICP アルゴリズムと Go-ICP[25] を用いた．Go-ICP は ICP アルゴリズムによる局所解探索と branch-and-bound アルゴリズムによる探索空間の枝狩りを交互に繰り返し，大域的最適解を発見するアルゴリズムである．ここでは通常の Go-ICP に加え，Go-ICP の局所解探索を RE アルゴリズムに変更した Go-ICP + RE の実験も行う．データセットには図 6 に示す Stanford3D データセットの bunny[4]を用いた．ターゲット点群は図 6b に示すようにソース点群の一部を用い，点群の各座標は全て $[-1, 1]^3$ の範囲内に正規化した．

まず，ICP アルゴリズムと RE アルゴリズムと比較実験結果を表 4 に示す．RE アルゴリズム（$T = 30$）はほぼ同じ計算時間でよりよい最適化性能を示していることが分かる．k-means クラスタリングと同じく T を大きくすることで最適化性能は良くなるが，大域的最適化の成功率は大きく変わらない．

図 7 に Go-ICP との比較を示す．ここでは $\phi = 5\pi/12$ の回転角から生成された 50 個のランダムな回転行列に対する結果を全てプロットしている．Go-ICP は常に大域的最適解を発見することができているが，通常の ICP や RE アルゴリズムと比べ計算量が非常に大きい．Go-ICP + RE は Go-ICP と同じく大域的最適解を保証しつつ，Go-ICP と比べて計算量を減らしている．

[4]http://graphics.stanford.edu/data/3Dscanrep/

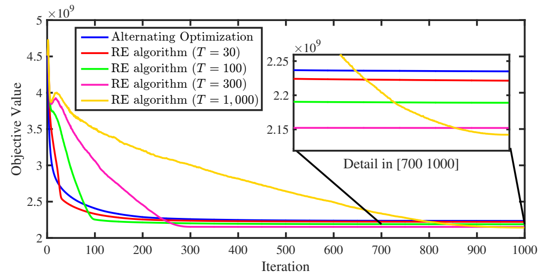


図 8: OPQ の比較実験．異なる初期解から 5 回の試行での平均目的関数値をプロットしている．

5.3. 最適直交量子化（OPQ）

背景

OPQ[7, 19] は近年提案された直積量子化 [9] の拡張手法であり，効率的な近似最近棒探索手法のための量子化手法の一種である．OPQ の最適化問題は以下のようになる．

$$\min_{\mathbf{R},\mathbf{C},\mathbf{Z}} \frac{1}{2}\sum_{i=1}^{N}\left\|\mathbf{x}_i - \mathbf{R}\begin{bmatrix}\mathbf{C}^{(1)}\mathbf{z}_i^{(1)} \\ \vdots \\ \mathbf{C}^{(M)}\mathbf{z}_i^{(M)}\end{bmatrix}\right\|_2^2 \tag{16}$$

$$\text{s.t. } z_{ij}^{(m)} = \{0, 1\}, \left\|\mathbf{z}_i^{(m)}\right\|_1 = 1, \mathbf{R}^\top\mathbf{R} = \mathbf{I},$$

ここで $\mathbf{X}, \mathbf{C}, \mathbf{Z}$ は k-means クラスタリングと同様それぞれデータ，クラスタ中心，割り当ての行列であり，$\mathbf{R}$ は回転行列である．

式 (16) の最適化は $\mathbf{R}$，$\mathbf{C}$，$\mathbf{Z}$ に関する交互最適化により実現される [7, 19]．Ge らはデータ分布がガウス分布に従うという仮定を用いたパラメトリックな最適化手法も提案している [7]．

実験の詳細と結果

ここでは 100,000 個の 128 次元 SIFT 特徴量からなる SIFT 1M [9] を用いて OPQ の実験を行う．式 (16) の分割数パラメータ $M = 8$ と，クラスタ数 $k = 256$ を用いた．比較手法は交互最適化 [7, 19] である．

異なる初期解から各手法で 5 回最適化を行い，目的関数の平均値をプロットした結果を図 8 に示す． RE アルゴリズムは大域的最適化性能を向上させるのみならず，$T = 30$ と $T = 100$ の場合は収束を早めている．RE アルゴリズムは目的関数を現在の解の周りで押し上げるため，勾配を急峻にしていることが収束が早くなる理由として考えられる．

5.4. 非負値行列分解（NMF）

背景

NMF はデータ行列を非負値の行列で分解する手法であり，以下のように定式化される．

$$\min_{\mathbf{U},\mathbf{V}} \frac{1}{2}\|\mathbf{X} - \mathbf{UV}\|_F^2 \tag{17}$$

$$\text{s.t. } u_{ij} \geq 0, v_{ij} \geq 0 \text{ for all } i, j$$

表 5: 合成データでの NMF の実験結果．各手法での MSE と，MSE<0.0015 を実現した回数を示す．

	MSE			成功回数
	Mean	Min	Max	
ALS	0.2128	0.0109	0.7333	0
ALS + RE	0.0683	0.0184	0.1925	0
MU	0.0016	0.0015	0.0017	2
MU + RE	0.0015	0.0014	0.0016	9
ANLS-BPP	0.0032	0.0014	0.0038	1
ANLS-BPP + RE	0.0017	0.0014	0.0021	4

ここで $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}^{d \times n}$ はデータ行列である．NMF は音声処理や画像処理の文脈で広く使われている基礎的な技術である．

NMF の最適化手法は様々な手法が提案されているが，本稿では交互最小二乗法（ALS）[3]，Multiplicative update（MU）[14]，block principal pivoting を用いた交互最適化（ANLS-BPP）[11] の三つを用いて実験を行う．これらの手法は基本的に局所解を発見する手法であるが，ALS のみ各イテレーションでエネルギーが下がる保証がないため，局所解を発見する保証はない．

合成データでの実験

ランダムに行列 $\mathbf{U}_{gt} \subset \mathbb{R}^{n \times r}$ と $\mathbf{V}_{gt} \subset \mathbb{R}^{r \times m}$ を生成し，データ行列を $\mathbf{X}$ を $\mathbf{X} = \mathbf{U}_{gt}\mathbf{V}_{gt} + \mathbf{N}$ により計算する．ここで $\mathbf{N}$ の各要素は分散 $\sigma^2 = 0.01$ のガウス分布からなるノイズである．このデータ行列 $\mathbf{X}$ からパラメータ $\mathbf{U}$ と $\mathbf{V}$ を推定し，エラーの指標として MSE（$\|\mathbf{X} - \mathbf{UV}\|_F^2/nm$）用いた．

異なる 10 個の初期解から，ALS, MU と ANLS-BPP の手法と，それぞれのアルゴリズムに RE アルゴリズムを組み合わせて推定した結果を表 5 に示す．推定の成功回数は MSE が 0.0015 を下回った回数である．RE アルゴリズムのパラメータには $\mu^{(1)} = 1.0 \times 10^{-6}$，$T = 300$ を用い，それぞれのアルゴリズムは 10,000 イテレーションで打ち切った．RE アルゴリズムはどの手法と組み合わせてもエラーを小さくすることに成功していることが確認できる．しかし ALS はエラーが非常に大きく，RE アルゴリズムと組み合わせても他の手法と比べて良い結果を得ることができていない．

実データでの実験

実データとして AT&T Laboratories Cambridge の顔画像データベース[5]を用いた．これは異なる 40 人の顔画像，計 400 枚からなるデータセットで，各画像の大きさは 92×112 ピクセルである．この画像群から成るデータ行列 $\mathbf{X} \in \mathbb{R}^{10304 \times 400}$ を用い，合成データの実験と同様に各手法で異なる 5 個の初期解から推定を行いエラーを評価した．

各イテレーション時点での平均のエラーをプロットした実験結果を図 9 に示す．RE アルゴリズムのパラメータは $\mu^{(1)} = 0.01$ と $T = 50$ を用いた．ALS の手法は厳密にエネルギーを下げる手法ではないため，RE を使っても良い解を求めることが出来ていない．MU は収束にかなり時間がかかるが，RE と組み合わせることで OPQ と同様に収束を早めていることが確認できる．ANLS-BPP は収束が早く，エラーも他のアルゴリズムと比べて低いが，RE と組み合わせることで最終的なエラーを下げることに成功している．

[5] http://www.cl.cam.ac.uk/research/dtg/attarchive/facedatabase.html

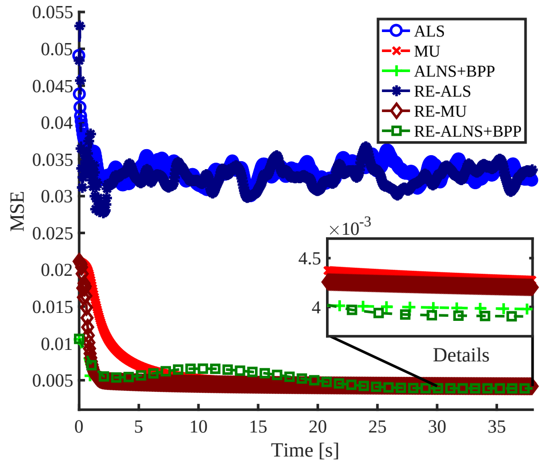


図 9: 実データでの NMF の実験結果．各イテレーションでの MSE をプロットしている．

6. まとめ

本稿では非凸最小二乗法の最適化問題に対する大域的最適化手法として，RE アルゴリズムを提案した．RE アルゴリズムは RE 収束という新しい大域的最適化に関する収束の指標に基づき，RE 定数の大きな最適解を発見するようなアルゴリズムとなっている．我々は RE 収束と大域的最適解の関係に対する理論的な説明と，多くの最適化問題（k-means クラスタリング，点群の位置合わせ，OPQ，NMF）での高い最適化性能を実験的に示した．

今後の課題として，まず理論の面では大きな RE 定数を持つことと大域的最適解の関係については十分に明らかにされていない．今回実験で扱った問題で，最大の RE 定数を持つ解が大域的最適解となるのかは不明であり，実際に k-means クラスタリングでは反例も存在する．これらの問題で，どのような条件であれば大域的最適解となるかなどの説明を与えることは重要な課題である．また，RE アルゴリズムは最小二乗法の問題にしか適用できないため，これを最小二乗法以外の問題に適用できるように拡張することを考えている．

謝辞

本研究の一部は，JST CREST JPMJCR1686 と科研費 15K12025 の支援を受けた．

参考文献

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk. SLIC superpixels compared to state-of-the-art superpixel methods. *TPAMI*, 34(11):2274–2282, 2012.

[2] D. Arthur and S. Vassilvitskii. k-means++: The advantages of careful seeding. In *SODA*, pages 1027–1035, 2007.

[3] M. W. Berry, M. Browne, A. N. Langville, V. P. Pauca, and R. J. Plemmons. Algorithms and applications for approximate nonnegative matrix factorization. *Computational statistics & data analysis*, 52(1):155–173, 2007.

[4] P. J. Besl and H. D. McKay. A method for registration of 3-D shapes. *TPAMI*, 14(2):239–256, 1992.

[5] G. Blais and M. D. Levine. Registering multiview range data to create 3D computer objects. *TPAMI*, 17(8):820–824, 1995.

[6] G. Csurka, C. Dance, L. Fan, J. Willamowski, and C. Bray. Visual categorization with bags of keypoints. In *In Workshop on Statistical Learning in Computer Vision, ECCV*, pages 1–22, 2004.

[7] T. Ge, K. He, Q. Ke, and J. Sun. Optimized product quantization for approximate nearest neighbor search. In *CVPR*, pages 2946–2953, 2013.

[8] J. A. Hartigan. *Clustering algorithms*. John Wiley & Sons, Inc., 1975.

[9] H. Jegou, M. Douze, and C. Schmid. Product quantization for nearest neighbor search. *TPAMI*, 33(1):117–128, 2011.

[10] J. Kennedy and R. Eberhart. Particle swarm optimization. In *ICNN*, volume 4, pages 1942–1948, 1995.

[11] J. Kim and H. Park. Toward faster nonnegative matrix factorization: A new algorithm and comparisons. In *ICDM*, pages 353–362, 2008.

[12] S. Kirkpatrick, M. P. Vecchi, et al. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[13] K. Krishna and M. N. Murty. Genetic K-means algorithm. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 29(3):433–439, 1999.

[14] D. D. Lee and H. S. Seung. Algorithms for non-negative matrix factorization. In *NIPS*, pages 556–562, 2001.

[15] S. Lloyd. Least squares quantization in PCM. *TIT*, 28(2):129–137, 1982.

[16] J. Luck, C. Little, and W. Hoff. Registration of range data using a hybrid simulated annealing and iterative closest point algorithm. In *ICRA*, volume 4, pages 3739–3744, 2000.

[17] M. Mitchell. *An introduction to genetic algorithms*. MIT press, 1998.

[18] S. A. Nene, S. K. Nayar, H. Murase, et al. Columbia object image library (COIL-20). Technical report.

[19] M. Norouzi and D. J. Fleet. Cartesian k-means. In *CVPR*, pages 3017–3024, 2013.

[20] R. B. Rusu, N. Blodow, and M. Beetz. Fast point feature histograms (fpfh) for 3d registration. In *ICRA*, pages 3212–3217. IEEE, 2009.

[21] L. Silva, O. R. P. Bellon, and K. L. Boyer. Precision range image registration using a robust surface interpenetration measure and enhanced genetic algorithms. *TPAMI*, 27(5):762–776, 2005.

[22] N. Slonim, E. Aharoni, and K. Crammer. Hartigan's k-means versus Lloyd's k-means: Is it time for a change? In *IJCAI*, pages 1677–1684, 2013.

[23] M. Telgarsky and A. Vattani. Hartigan's method: k-means clustering without Voronoi. In *AISTATS*, pages 820–827, 2010.

[24] M. P. Wachowiak, R. Smolíková, Y. Zheng, J. M. Zurada, and A. S. Elmaghraby. An approach to multimodal biomedical image registration utilizing particle swarm optimization. *TEVC*, 8(3):289–301, 2004.

[25] J. Yang, H. Li, and Y. Jia. Go-ICP: Solving 3D registration efficiently and globally optimally. In *ICCV*, pages 1457–1464, 2013.

付録：Theorem 1 の証明

まず，$E(\theta)$ を θ について微分することで以下の式を得る．

$$\nabla E(\theta) = 2\theta^3 + (-2y_1 + 1)\theta - y_2. \tag{18}$$

式 (7) において，θ_3 を局所最大値を与える解とすると，$\theta_1, \theta_2, \theta_3$ はそれぞれ式 (18)$= 0$ の根である．解と係数の関係より，以下の関係式を得る．

$$\theta_1 + \theta_2 + \theta_3 = 0, \tag{19}$$

$$\theta_1\theta_2 + \theta_2\theta_3 + \theta_3\theta_1 = \frac{-2y_1 + 1}{2}, \tag{20}$$

$$\theta_1\theta_2\theta_3 = \frac{y_2}{2}. \tag{21}$$

式 (19)-(21) を整理することで以下の二つの式を得る．

$$y_1 = \frac{1}{2}\left((\theta_1 + \theta_2)^2 + \theta_1^2 + \theta_2^2 + 1\right) \geq \frac{1}{2}\left(\theta_1^2 + \theta_2^2 + 1\right). \tag{22}$$

$$y_2 = -\theta_1\theta_2(\theta_1 + \theta_2) \tag{23}$$

$E(\theta_1)$ と $E(\theta_2)$ の大小関係を調べるため，その差を考える．

$$E(\theta_1) - E(\theta_2) = -\frac{1}{2}(\theta_1 - \theta_2)^2(\theta_1^2 - \theta_2^2). \tag{24}$$

よって，もし $\theta_1^2 > \theta_2^2$ ならば $E(\theta_1) < E(\theta_2)$ であり，そうでなければ $E(\theta_1) > E(\theta_2)$ であることが分かる．

ある局所解 θ_* での RE された関数 $E_\alpha(\theta)$ は以下のようになる．

$$E_\alpha(\theta) = \frac{1}{2}\begin{pmatrix} \left(y_1 - \theta^2 + \alpha\left(y_1 - \theta_*^2\right)\right)^2 \\ + \left(y_2 - \theta + \alpha\left(y_2 - \theta_*\right)\right)^2 \end{pmatrix}. \tag{25}$$

$E_\alpha(\theta)$ の二階微分は以下で与えられる．

$$\nabla^2 E_\alpha(\theta) = 2(\theta^2 - y_1) + 2\alpha(\theta_*^2 - y_1) + 4\theta^2 + 1. \tag{26}$$

したがって θ_* の RE 定数は以下で与えられる．

$$\alpha_* = \begin{cases} \infty & (\theta_*^2 \geq y_1) \\ -1 + \frac{(4\theta_*^2 + 1)}{y_1 - \theta_*^2} & (\text{otherwise}) \end{cases} \tag{27}$$

式 (22) より，局所解 θ_1 と θ_2 について以下の三つのケースを考えればよい．

(i) $\theta_1^2 \geq y_1$ かつ $\theta_2^2 < y_1$ の場合：この場合 θ_1 の RE 定数 $\alpha_1 = \infty$ かつ θ_2 の RE 定数 α_2 は有限である．従って $\alpha_1 > \alpha_2$ であり $\theta_1^2 > \theta_2^2$ となる．

(ii) $\theta_1^2 < y_1$ かつ $\theta_2^2 \geq y_1$ の場合：(i) と同様にして $\alpha_1 < \alpha_2$ かつ $\theta_1^2 < \theta_2^2$ となる．

(iii) $\theta_1^2 < y_1$ かつ $\theta_2^2 < y_1$ の場合：以下の関係式を得る．

$$\alpha_1 - \alpha_2 = \frac{(4y_1 + 1)\left(\theta_1^2 - \theta_2^2\right)}{2\left(y_1 - \theta_1^2\right)\left(y_1 - \theta_2^2\right)}. \tag{28}$$

従って α_1，α_2 間の大小関係は θ_1^2，θ_2^2 間の大小関係と一致する．

(i)，(ii)，(iii) より，α_1，α_2 間の大小関係は常に θ_1^2，θ_2^2 間の大小関係と一致することが言える．よって式 (24) より，もし $\alpha_1 > \alpha_2$ ならば θ_1 が大域的最適解であり，そうでなければ θ_2 が大域的最適解である．