DEIM Forum 2023 1a-2-5

# 悪条件な目的関数に動的に適応するCMA-ES

清水　洸希†　　豊田　正史††

† 東京大学大学院 情報理工学系研究科 〒113–8654 東京都文京区本郷 7–3–1
†† 東京大学 生産技術研究所 〒153–8505 東京都目黒区駒場 4–6–1
E-mail: †{shimizu,toyoda}@tkl.iis.u-tokyo.ac.jp

**あらまし**　CMA-ES (共分散適応進化戦略) はブラックボックス最適化において非常に優れたアルゴリズムとして知られている．一方で，目的関数が高次元かつ悪条件な場合では，解へと到達するまでに要する反復数が大幅に増加したり，解への到達に失敗する問題が報告されている．座標選択は，そのような場合に，探索の各反復で更新する座標 (変数) を選択することで，目的関数の条件数を低減し，探索性能を大幅に向上させる手法である．本論文では，この座標選択手法における，目的関数が良条件な場合にも座標の選択を行うことで発生する性能低下の問題を解決する．本手法では，最適化の過程において目的関数の条件数を推定し，それに基づいた座標の選択方法を提案する．これにより，目的関数が良条件な場合には座標の分割数を減らすことができ，従来の座標選択法よりも効率的な探索が可能となる．ベンチマーク関数を用いた実験では，良条件な場合には従来の座標選択手法から大幅な改善が見られ，悪条件な場合には同程度の性能が得られることが確認された．

**キーワード**　学習理論，汎用機械学習技術，ブラックボックス最適化，進化計算

## 1. はじめに

ブラックボックス最適化は，新幹線車両の設計や機械学習モデルにおけるハイパーパラメタの探索など，我々の身の回りの様々な場面に現れる重要な課題である．最適解問題とは，与えられた目的関数を最小化 (あるいは最大化) する入力変数を探索することであるが，ブラックボックス最適化は，その中でも目的関数の導関数といった情報が明に与えられないものを指す．一般に目的関数が持ちうる性質として，悪条件性，多峰性，変数間依存性が挙げられるが，ブラックボックス最適化においては，目的関数がこのような性質を持つかどうかを事前に知ることはできないため，最適化の過程においてこれらの性質に適応可能なアルゴリズムを設計することが必要となる [1] [2].

CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [3] は，連続関数を目的関数とするブラックボックス最適化において，効率よく解を探索可能なアルゴリズムとして知られている．CMA-ES は，進化戦略 (ES) に共分散行列を導入した手法で，多変量正規分布 $\mathcal{N}(\boldsymbol{m}, \sigma^2\boldsymbol{C})$ に基づいた個体の生成を行い，多変量正規分布の変数 (平均 $\boldsymbol{m}$ 及び共分散行列 $\boldsymbol{C}$) とステップサイズ $\sigma$ の最適化を通して解の探索を行う．ここでステップサイズは，探索地点と最適解との距離に応じて増減するスカラー値で，分散共分散行列の補正を行い探索速度を高める役割を持つパラメタである．CMA-ES の特徴として，共分散行列の非対角成分が変数間の依存性に，対角成分が各変数軸のスケールに適応することで，前述の悪条件性や変数間依存性を考慮した解候補の生成を行うことができる点が挙げられる．

CMA-ES の課題としては，目的関数の次元数 $d$ に対し，時間計算量が $O(d^3)$，空間計算量が $O(d^2)$ であることから，高次元な最適化問題への適用が困難な点が挙げられるが，これについては，共分散行列を対角成分に限定することで双方の計算量を $O(d)$ に低減した sep-CMA [4] をはじめとして，共分散行列を数本のベクトルで近似する VD-CMA [5], VkD-CMA [6] や LM-MA [7] など多くの手法が提案されている．また，目的関数が超高次元なだけではなく，強い悪条件性を持つ場合に，前述の共分散行列の悪条件性への適応が妨げられ，解へ到達するまでに要する時間の大幅な増加や，局所解を持たない単峰性関数にも関わらず解に到達できないといった事例が報告されている [8]．このような問題への解決法として，確率的次元選択 CMA-ES [9] が提案されている．この手法では，最適化の各反復において，全ての変数を更新するのではなく，ランダムにいくつかの軸を選んで更新を行うことで，選択された座標空間における条件数を低減することが可能となる．また，軸を選択する際に，目的関数の各軸の曲率を推定し，曲率の値が近い軸同士を選ぶことで，ランダムな選択よりも性能を向上させる手法も提案されている [10]．これらの手法により，高次元かつ悪条件な問題における性能の大幅な向上が実現したが，一方で，各反復中に選択する軸の数をハイパーパラメタとして与える必要があるため，良条件な問題においても不必要な選択が行われてしまい，従来の手法よりも性能が悪化するという課題が残った．

本論文では，この座標選択を改良し，最適化の過程で目的関数の悪条件性に対して選択数を適応可能な手法を提案する．本手法では，目的関数の各軸の二階偏微分を有限差分法を用いて推定し，選択された座標空間における条件数が予め設定した許容条件数を上回らないという制約のもとにクラスタリングを行う．これにより，悪条件関数では座標軸の選択数が増大し，良条件関数では減少することで，良条件関数の最適化における不必要な座標の選択を防ぐことが可能となる．また，入力変数によって条件数が変化するような目的関数の場合にも，探索地点における条件数に応じた座標の選択を行うことができる．ベンチマーク関数を用いた実験では，提案手法を適用することで，

目的関数が悪条件な場合には性能が向上し，良条件な場合においても適用前と同等の性能が得られることを確認した．

以下，2 章において CMA-ES の概要と確率的次元選択 CMA-ES が悪条件関数に対して優れる理由について説明する．3 章では，提案手法である悪条件性適応座標選択のアルゴリズムについて，有限差分法を用いた条件数の推定方法，条件数の推定頻度の設計，許容条件数を満たすクラスタリング法について述べる．そして，4 章においてベンチマーク関数を用いた実験を通して，悪条件な場合に性能が向上し，良条件な場合に適用前と同等の性能を得られることを確認する．最後に，5 章でまとめと今後の展望について記す．

## 2. 関 連 研 究

本章では，CMA-ES のアルゴリズムの概要と，本手法のベースとなる確率的次元選択 CMA-ES について述べる．

### 2.1 $(\mu/\mu_w, \lambda)$-CMA-ES

本節では，CMA-ES のうち最も一般的とされる $(\mu/\mu_w, \lambda)$-CMA-ES について概要を述べる．CMA-ES は，一般の進化戦略とは異なり，解候補を直接保持せず，多変量正規分布 $\mathcal{N}(\boldsymbol{m}^{(t)}, {\sigma^{(t)}}^2\boldsymbol{C}^{(t)})$ により解候補の集団を生成し，目的関数を用いた評価で高評価を得た候補から平均 $\boldsymbol{m}^{(t)}$ と分散共分散行列 $\boldsymbol{C}^{(t)}$ を更新することで解の探索を行う．ここで $t$ はステップ数とする．

目的関数 $f(\boldsymbol{x}) \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^d$ を最小化する場合の，具体的なアルゴリズムについて以下に記す．はじめに，変数の初期化を行う．平均 $\boldsymbol{m}^{(0)} \in \mathbb{R}^d$，分散共分散行列 $\boldsymbol{C}^{(0)} \in \mathbb{R}^{d\times d}$，ステップサイズ $\sigma^{(0)} \in \mathbb{R}$ をそれぞれ探索領域に応じて決める．また，分散共分散行列とステップサイズのそれぞれの進化パス $\boldsymbol{p}_{\boldsymbol{c}}^{(0)} \in \mathbb{R}^d$ および $\boldsymbol{p}_{\boldsymbol{\sigma}}^{(0)} \in \mathbb{R}^d$ を $\mathbf{0}$ とする．そして，あらかじめ定めた終了条件を満たすまで以下のステップを繰り返す．

[Step 1.] 式 (1) の通り，正規分布 $\mathcal{N}(\mathbf{0} \in \mathbb{R}^d, \boldsymbol{I} \in \mathbb{R}^{d\times d})$ から，$\lambda$ 個の個体 $\boldsymbol{z} \in \mathbb{R}^{\lambda\times d}$ を生成する．次に式 (2) の通り，$\boldsymbol{C}^{(t)}$ を平方分解し，$\boldsymbol{z}^{(t)}$ との行列積から $\boldsymbol{y}^{(t)} \in \mathbb{R}^{\lambda\times d}$ を求める．そして，式 (3) の通り，$\sigma^{(t)}$ を $\boldsymbol{y}^{(t)}$ の各要素にかけ，平均 $\boldsymbol{m}^{(t)}$ との和から解集団 $\boldsymbol{x}^{(t)} \in \mathbb{R}^{\lambda\times d}$ を生成する．

$$\boldsymbol{z}^{(t)} \sim \mathcal{N}(\mathbf{0}, \boldsymbol{I}), \tag{1}$$

$$\boldsymbol{y}^{(t)} = \boldsymbol{z}^{(t)}\sqrt{\boldsymbol{C}^{(t)}}, \tag{2}$$

$$\boldsymbol{x}^{(t)} = \boldsymbol{m}^{(t)} + \sigma^{(t)}\boldsymbol{y}^{(t)}. \tag{3}$$

[Step 2.] Step 1. で生成した解集団 $\boldsymbol{x}^{(t)}$ の各個体について $f(\boldsymbol{x}_i), (i = 1, ..., \lambda)$ から評価値を計算し，昇順に $\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}, \boldsymbol{z}^{(t)}$ を個体 $(\lambda)$ の軸について並び替える．

[Step 3.] 各個体の順位についての重み $\boldsymbol{w} \in \mathbb{R}^\lambda$ を用いて，式 (4), (5) の通り $\boldsymbol{w}$ と $\boldsymbol{x}^{(t)}, \boldsymbol{y}^{(t)}$ の個体の軸についての内積を取り，$\boldsymbol{dy}^{(t)}, \boldsymbol{dz}^{(t)} \in \mathbb{R}^d$ を求める．ここで，重み $\boldsymbol{w}$ は，$1 < \mu < \lambda$ として，$w_1 \geqq \cdots \geqq w_\mu > 0, w_{\mu+1}, \cdots, w_\lambda = 0, \|\boldsymbol{w}\|_1 = 0$ を満たすものとする．

$$\boldsymbol{dy}^{(t)} = \sum_i^\lambda w_i \boldsymbol{y}_{\boldsymbol{i}}^{(t)}, \tag{4}$$

$$\boldsymbol{dz}^{(t)} = \sum_i^\lambda w_i \boldsymbol{z}_{\boldsymbol{i}}^{(t)}. \tag{5}$$

[Step 4.] Step 3. で求めた $\boldsymbol{dy}^{(t)}$, $\boldsymbol{dz}^{(t)}$ を用いて式 (6), (7), (8) の通り進化パスを計算する．ここで，$\mu_w = \frac{1}{\|\boldsymbol{w}\|}$, $\chi_d = \mathbb{E}[\|\mathcal{N}(\mathbf{0}, \boldsymbol{I})\|] \simeq \sqrt{d}(1 - \frac{1}{4d} + \frac{1}{21d^2}) \in \mathbb{R}$ とし，$c_\sigma, c_c \in \mathbb{R}$ を各進化パスの学習率とする．

$$h_\sigma^{(t+1)} = \begin{cases} 1 & \|\boldsymbol{p}_{\boldsymbol{\sigma}}^{(t+1)}\| < (1.4 + \frac{2}{n+1})\chi_d \\ 0 & else \end{cases}, \tag{6}$$

$$\boldsymbol{p}_{\boldsymbol{\sigma}}^{(t+1)} = (1 - c_\sigma)\boldsymbol{p}_{\boldsymbol{\sigma}}^{(t)} + \sqrt{c_\sigma(2 - c_\sigma)\mu_w}\boldsymbol{dz}, \tag{7}$$

$$\boldsymbol{p}_{\boldsymbol{c}}^{(t+1)} = (1 - c_c)\boldsymbol{p}_{\boldsymbol{c}}^{(t)} + h_\sigma^{(t+1)}\sqrt{c_c(2 - c_c)\mu_w}\boldsymbol{dy}. \tag{8}$$

[Step 5.] Step 4. で求めた進化パスを用いて式 (9), (10), (11) の通り多変量正規分布の変数の更新を行う．ここで，$OP(\cdot) \in \mathbb{R}^{n\times n}$ は入力ベクトルとそれ自身の直積行列とする．また，$\eta_m \in R$ を平均 $\boldsymbol{m}$ の学習率，$\eta_{c_1}, \eta_{c_\mu} \in R$ をそれぞれ分散共分散行列 $\boldsymbol{C}$ の rank-one update (式 (11) 右辺二項目), rank-$\mu$ update (式 (11) 右辺三項目) の学習率とし，$d_\sigma \in R$ をステップサイズの減衰率とする．

$$\boldsymbol{m}^{(t+1)} = \boldsymbol{m}^{(t)} + \eta_m\sigma^{(t)}\boldsymbol{dy}^{(t)}, \tag{9}$$

$$\sigma^{(t+1)} = \sigma^{(t)} \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_{\boldsymbol{\sigma}}^{(t+1)}\|}{\chi_d} - 1\right)\right), \tag{10}$$

$$\boldsymbol{C}^{(t+1)} = \boldsymbol{C}^{(t)} + \eta_{c_1}\left(OP(\boldsymbol{p}_{\boldsymbol{c}}^{(t+1)}) - \boldsymbol{C}^{(t)}\right) + \eta_{c_\mu}\sum_i^\lambda w_i\left(OP(\boldsymbol{y}_{\boldsymbol{i}}^{(\boldsymbol{t})}) - \boldsymbol{C}^{(t)}\right). \tag{11}$$

以上の Step 1.∼Step 5. を繰り返すことで，平均ベクトル $\boldsymbol{m}$ が解に，ステップサイズ $\sigma$ が 0 に，分散共分散行列 $\boldsymbol{C}$ が $\mathbf{0}$ に収束することで，多変量正規分布がデルタ関数に収束し，最適解が求まる．

### 2.2 確率的次元選択 CMA-ES

本節では，本手法のベースとなる確率的次元選択 CMA-ES について述べる．同アルゴリズムは目的関数が高次元かつ悪条件な場合に，更新する座標軸をランダムに選択することで，条件数の低減とステップサイズのベクトル化を達成し，より高速な解の探索を実現したアルゴリズムである．以下に具体的なアルゴリズムを記す．

最小化したい目的関数を $f(\boldsymbol{x}) \in \mathbb{R}, \boldsymbol{x} \in \mathbb{R}^d$ とする．はじめに，平均 $\boldsymbol{m}^{(0)} \in \mathbb{R}^d$，分散共分散行列 $\boldsymbol{C}^{(0)} \in \mathbb{R}^{d\times d}$，ベクトル化されたステップサイズ $\boldsymbol{\sigma}^{(0)} \in \mathbb{R}^n$ をそれぞれ探索領域に応じて決め，分散共分散行列とステップサイズのそれぞれの進化パスを $\boldsymbol{p}_{\boldsymbol{c}}^{(0)} \in \mathbb{R}^d$, $\boldsymbol{p}_{\boldsymbol{\sigma}}^{(0)} \in \mathbb{R}^d = \mathbf{0}$ と初期化する．次に，目的関数の入力次元長のインデックスベクトル $\boldsymbol{\iota} \in \mathbb{R}^d = (0, 1, ..., d-1)$

を新たに導入する．インデックスベクトルは，予め要素をランダムに並び替え，インデックスの参照を終えた要素の終端地点を $e \in \mathbb{R} = 0$ と初期化し，各反復で選択する次元数を $s \in \mathbb{R}$ とする．そして，予め定めた終了条件を満たすまで以下のステップを繰り返す．

[Step 1.] $\boldsymbol{\iota}$ から，前世代での終了地点 $e$ から $e+s$ までを取り出し，当世代でのインデックスベクトル $\boldsymbol{\iota'} \in \mathbb{R}^s$ とする．ただし，$e+s > d$ の場合，$e$ から $d$ までのベクトル $\boldsymbol{\iota'} \in \mathbb{R}^{d-e}$ とし，$\boldsymbol{\iota}$ の要素を全てランダムに並び替え，$e = 0$ とする．

[Step 2.] 生成した当世代でのインデックスベクトル $\boldsymbol{\iota'}$ に基づいて，各変数から部分ベクトルおよび部分行列を生成する．具体的には，平均 $\boldsymbol{m'} \in \mathbb{R}^s$，分散共分散行列 $\boldsymbol{C'} \in \mathbb{R}^{s\times s}$，ステップサイズ $\boldsymbol{\sigma'} \in \mathbb{R}^s$，進化パス $\boldsymbol{p'_c} \in \mathbb{R}^s, \boldsymbol{p'_\sigma} \in \mathbb{R}^s$ となる．分散共分散行列については，指定されたインデックスの対角成分およびそれらの共起成分からなる対称行列を抽出する．

[Step 3.] 正規分布 $\mathcal{N}(\boldsymbol{0} \in \mathbb{R}^s, \boldsymbol{I} \in \mathbb{R}^{s\times s})$ から，$\lambda$ 個の個体 $\boldsymbol{z'}^{(t)} \in \mathbb{R}^{\lambda\times s}$ を生成する．$\boldsymbol{C'}$ を平方分解し，$\boldsymbol{z'}^{(t)}$ との行列積から $\boldsymbol{y'}^{(t)} \in \mathbb{R}^{\lambda\times s} = \boldsymbol{z'}^{(t)}\sqrt{\boldsymbol{C'}}$ を求める．そして，$\boldsymbol{\sigma'}^{(t)}$ を $\boldsymbol{y'}^{(t)}$ の各要素にかけ，平均 $\boldsymbol{m}^{(t)}$ の $\boldsymbol{\iota'}$ の要素との和から解集団 $\boldsymbol{x}^{(t)} \in \mathbb{R}^{\lambda\times d} = \boldsymbol{m}^{(t)} + \boldsymbol{\sigma'}^{(t)} \circ \boldsymbol{y'}^{(t)}$ を生成する (ここで $\circ$ は二つのベクトル間の要素積を取る演算子とする)．このとき，$\boldsymbol{x}^{(t)}$ は通常の CMA-ES と同じ次元の行列となるが，$\boldsymbol{\iota'}$ 以外のインデックスの次元については，各個体全て同じ値 ($\boldsymbol{m}^{(t)}$ の値) を取る．

[Step 4.] 通常の CMA-ES と同様に，目的関数を用いて各個体を評価・順位づけし，昇順に並び替える．また，同様に重み $\boldsymbol{w} \in \mathbb{R}^\lambda$ との個体の軸 ($\lambda$ の軸) についての内積 $\boldsymbol{dy'}^{(t)}$, $\boldsymbol{dz'}^{(t)} \in \mathbb{R}^s$ を求める．

[Step 5.] Step 4. で求めた $\boldsymbol{dy'}^{(t)}$, $\boldsymbol{dz'}^{(t)}$ をもとに，進化パスを更新する．このとき，$\boldsymbol{\iota'}$ で指定された要素のみ更新を行い，その他の要素については元の値を維持する．

[Step 6.] 進化パスをもとに平均 $\boldsymbol{m}^{(t)}$，分散共分散行列 $\boldsymbol{C}^{(t)}$，ステップサイズベクトル $\boldsymbol{\sigma}^{(t)}$ の更新を行う．ここでも，Step 4. と同様に $\boldsymbol{\iota'}$ で指定された要素のみ更新を行い，その他の要素については元の値を維持する．ステップサイズの更新値は，2 章式 (10) にある通りスカラー値であり，提案手法においても，同一反復中での更新値は $\boldsymbol{\iota'}$ の次元について同一となる．しかし，提案手法では，更新に用いられる $\boldsymbol{p'_\sigma}$ の次元の組が毎回異なることから，反復を重ねることでステップサイズベクトルの各要素は異なる値を取る．

以上の Step 1.∼Step 6. を通常の CMA-ES と同様に終了条件を満たすまで繰り返す．sep-CMA-ES についても分散共分散行列の対角成分から $\boldsymbol{\iota'}$ に対応する次元を抽出することで CMA-ES の場合と同様に適用することができる．

## 3. 提 案 手 法

確率的次元選択 CMA-ES や推定曲率に基づいて座標軸を選択する手法は，悪条件な目的関数の場合には高い性能を発揮するが，良条件な場合には適用前よりも性能が悪化する．これは，座標選択により一回の反復で更新される変数の数が制限されることに起因している．悪条件な場合では条件数の低減の影響が大きいため性能が改善されるが，良条件な場合にはすでに条件数が小さいためその恩恵を受けることができず，変数全体を更新するためにより多くの反復回数が必要となるためである．

そこで，本手法では，従来の手法のようにあらかじめ決められた選択数で座標軸を分割するのではなく，有限差分法により求めた各座標軸の二階偏微分値に基づき，あらかじめ設定した許容条件数を超えない範囲で座標軸を分割する．これにより，条件数の分散が大きい悪条件な目的関数の場合には，選択数が増大し，良条件な場合には減少させることができる．以下にアルゴリズムの詳細を記す．

### 3.1 有限差分法を用いた二階偏微分値の推定

目的関数の条件数を推定するにあたり，目的関数の各座標軸の二階偏微分値を有限差分法により求める．入力 $\boldsymbol{x} \in \mathbb{R}^d$ に対して $x_j, 1 \leqq j \leqq d$ における目的関数 $f(x)$ の二階微分を 3 点公式を用い，$h$ を微小な刻み幅として，

$$\frac{\partial^2 f}{\partial x_i^2} \approx \frac{f(x_1,...,x_i+h,...,x_d)-2f(\boldsymbol{x})+f(x_1,...,x_i-h,...,x_d)}{h^2}, \quad (12)$$

と算出する．この際に要する目的関数の評価回数は入力次元 $d$ に対して 3 点ずつ評価を行うため，$3d$ となる．

### 3.2 二階偏微分値に基づいた座標選択

本手法では，前節で求めた各軸の二階偏微分の値に対して，分割された各クラスタに含まれる二階偏微分値の絶対値の最大値と最小値の比がハイパーパラメタ $\alpha$ を用いて表現する許容条件数 $1+\alpha$ を超えないという制約のもとに座標の分割を行う．

### 3.3 二階偏微分値を求める頻度

上述のように，目的関数の各軸の二階偏微分を求めるには一回につき $3d$ の関数評価回数が必要となり，最適化の各反復のたびに算出する場合その計算コストは無視できないものとなる．変数間に依存関係を持たない二次関数のように二階偏微分をした段階で全ての変数が消去され定数となるような場合には，最初の一度のみ二階偏微分を求めればよく，計算コストの大幅な削減が可能となる．そこで，本手法では，二階偏微分値の変化量が元の値の 50%以上となった場合は，次回の反復でも二階偏微分を計算し，そうでない場合には，反復回数が入力次元数の倍数となった場合にのみ計算を行う．

以下に，提案手法のアルゴリズムの具体的な動作を示す．

[Step 1.] 有限差分法により目的関数の各軸の二階偏微分の絶対値を推定する．ここで，現在の最適化における反復数が入力次元数の正数倍となっている場合には，全ての軸について二階偏微分を求める．それ以外の場合には，前回と前々回に求めた二階偏微分の値が 50% 以上変化した軸についてのみ求め，それ以外の軸については，前回の二階偏微分の値をそのまま用いる．このステップにおいて，d 次元の二階偏微分の絶対値を要素とするベクトル $\boldsymbol{a} = (a_1, ..., a_d)$ を得る．

[Step 2.] Step 1. で得られた二階偏微分ベクトルについて，最小の要素 $a_{min} = min(\boldsymbol{a})$ を求め，$a_i \leqq a_{min}(1+\alpha)$ を満たす要素を $\boldsymbol{a}$ から抽出する．$\boldsymbol{a}$ から抽出した要素を取り除いたベクトルを新たな $\boldsymbol{a}$ とする．これにより，条件数が $1+\alpha$ 以下となる座標軸のクラスタが形成される．

表 1: 実験に用いたベンチマーク関数．$R$ はランダムに生成した $n \times n$ の直交行列である．

| 関数名 | 定義 | 変数間依存性 | 悪条件性 | 多峰性 |
|---|---|---|---|---|
| Sphere | $f_{sph}(\boldsymbol{x}) = \sum_{i=1}^{d} x_i^2$ | | | |
| Ellipsoid | $f_{ell}(\boldsymbol{x}) = \sum_{i=1}^{d} (1000^{\frac{i-1}{d-1}} x_i)^2$ | | ✓ | |
| Rot Ellipsoid | $f_{ellrot}(\boldsymbol{x}) = f_{ell}(\boldsymbol{R}\boldsymbol{x})$ | ✓ | ✓ | |
| Chain-Rosenbrock | $f_{c-ros}(\boldsymbol{x}) = \sum_{i=1}^{d-1} [10^2 (x_i^2 - x_{i+1})^2 + (x_i - 1)^2]$ | ✓ | | ✓ |
| Star-Rosenbrock | $f_{s-ros}(\boldsymbol{x}) = \sum_{i=2}^{d} \left(100(x_1 - x_i^2)^2 + (1 - x_i)^2\right)$ | ✓ | ✓ | ✓ |

表 2: 適用元のアルゴリズム (sep-CMA, LM-MA, VD-CMA) 毎の比較．解に到達するまでに要した関数の評価回数を示した．解に到達しなかったものについては最適化の過程で最も小さい目的関数値を記載した．いずれの場合も小さい値を持つ手法が優れているとみなすことができる (目標値に到達した場合は，目標値未到達の場合よりも優れているとみなす)．各関数毎に最も良い結果が得られた数字を太字で示した．

| | sep-CMA | sep-CMA +ランダム座標選択 | sep-CMA + 動的座標選択 |
|---|---|---|---|
| | ベースライン | | 提案手法 |
| $10^5$-D Sphere | $\mathbf{1.45 \times 10^7}$ | $2.22 \times 10^7$ | $1.60 \times 10^7$ |
| $10^5$-D Ellipsoid | $6.57 \times 10^7$ | $4.45 \times 10^7$ | $\mathbf{2.05 \times 10^7}$ |
| $10^4$-D Rot Ellipsoid | 8623479.1 (目標値未到達) | **24458.8** (目標値未到達) | 9125044.1 (目標値未到達) |
| $10^4$-D Chain-Rosenbrock | 6813.8 (目標値未到達) | 9311.2 (目標値未到達) | $\mathbf{2.04 \times 10^8}$ |
| $10^4$-D Star-Rosenbrock | 2384.8 (目標値未到達) | **35.9** (目標値未到達) | 2290.9 (目標値未到達) |
| | LM-MA | LM-MA +ランダム座標選択 | LM-MA + 動的座標選択 |
| | ベースライン | | 提案手法 |
| $10^5$-D Sphere | $\mathbf{1.03 \times 10^7}$ | $1.87 \times 10^7$ | $1.15 \times 10^7$ |
| $10^5$-D Ellipsoid | 2335.5 (目標値未到達) | 571053.2 (目標値未到達) | $\mathbf{1.57 \times 10^7}$ |
| $10^4$-D Rot Ellipsoid | $\mathbf{1.87 \times 10^8}$ | 566.2 (目標値未到達) | $2.05 \times 10^8$ |
| $10^4$-D Chain-Rosenbrock | $1.90 \times 10^8$ | 9895.4 (目標値未到達) | $\mathbf{1.06 \times 10^8}$ |
| $10^4$-D Star-Rosenbrock | 1343.1 (目標値未到達) | **106.1** (目標値未到達) | 1415.9 (目標値未到達) |
| | VD-CMA | VD-CMA +ランダム座標選択 | VD-CMA + 動的座標選択 |
| | ベースライン | | 提案手法 |
| $10^5$-D Sphere | $\mathbf{1.04 \times 10^7}$ | $2.32 \times 10^7$ | $1.45 \times 10^7$ |
| $10^5$-D Ellipsoid | $1.69 \times 10^8$ | $6.21 \times 10^7$ | $\mathbf{1.57 \times 10^7}$ |
| $10^4$-D Rot Ellipsoid | 1327940.9 (目標値未到達) | **6948.4** (目標値未到達) | 1342470.9 (目標値未到達) |
| $10^4$-D Chain-Rosenbrock | 8830.1 (目標値未到達) | 8985.3 (目標値未到達) | $\mathbf{1.71 \times 10^8}$ |
| $10^4$-D Star-Rosenbrock | 926.3 (目標値未到達) | **67.7** (目標値未到達) | 823.9 (目標値未到達) |

[Step 3.] Step 2. において $\boldsymbol{a}$ から要素を抽出したことで，$\boldsymbol{a}$ の要素数が 0 となった場合は，クラスタリングを終了し，そうでない場合は，Step 2. を繰り返す．

## 4. 評価実験

本章では，提案手法のベンチマーク関数における性能を確認する．評価実験に用いたベンチマーク関数を表 1 に示した．いずれの関数の場合も初期値は，$m^{(0)} = U(-5, 5)$, $C^{(0)} = I$, $\sigma^{(0)} = 1.0$, $\lambda = 4 + 3\lfloor \ln(d) \rfloor$ とし，目的関数の目標値は $10^{-10}$，最大評価回数は $\lambda \times 10^7$ と設定した．

### 4.1 ベンチマーク関数を用いた各アルゴリズムの比較

本節では，悪条件適応座標選択を sep-CMA-ES, LM-MA, VD-CMA に適用し，従来のアルゴリズム，従来のアルゴリズムにランダム座標選択を適用したものとの比較を行う．各アルゴリズムにおけるハイパーパラメータは，提案手法における許容条件数を $\alpha = 0.5$ とし，ランダム座標選択における座標選択数を 100 とした．それ以外については，各論文で推奨される値を用いた．目的関数毎の関数評価回数に対する目的関数値の推移を図 1 に示した．実線 (solid line) が従来手法，破線 (dashed line) が従来手法に提案手法 (適応座標選択) を適用したもの，点線 (dotted line) が従来手法にランダム座標選択を適用したものを表している．赤線が sep-CMA, 黒線が LM-MA, 水色線が VD-CMA に関する各アルゴリズムに対応している．また，表 2 に，各適用元アルゴリズム毎に，目標値に到達するまでに要した関数の評価回数，目標値に到達しなかった場合は最大評価回数内で得られた最も小さな目的関数値を示した．各ベンチマーク関数に対して最も良い結果が得られた数字を太字で表している．

**Sphere 関数を用いた比較**

Sphere 関数は表 1 の通り，いずれの性質も持たない最も解の探索が容易な関数である．図 1 (a) より，いずれの適用元アルゴリズムの場合にも，座標選択を適用しない手法において最も良い結果が得られ，ランダム座標選択が最も悪い結果となり，提案手法を適用した場合はその中間の性能が得られた．Sphere

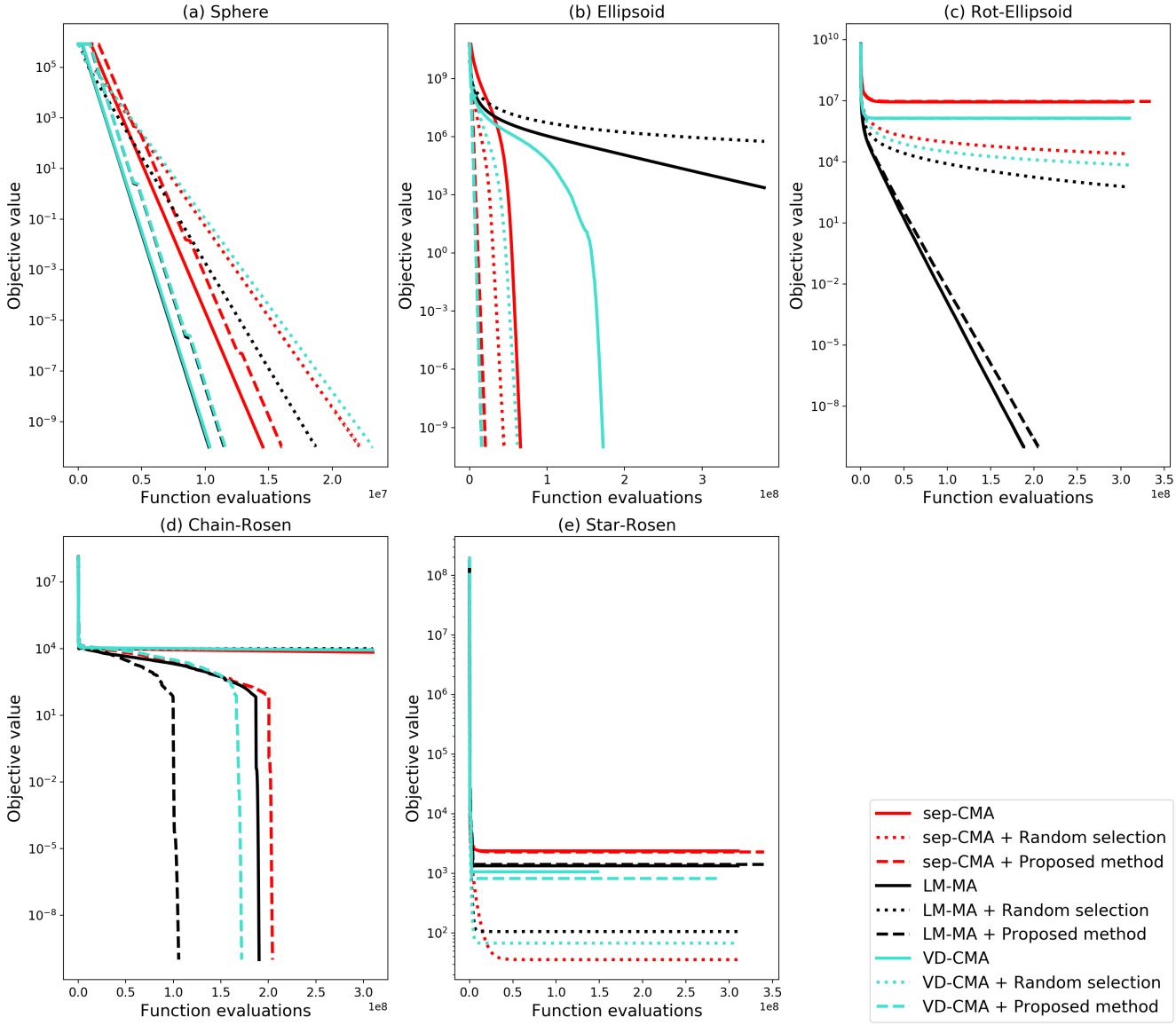


図 1: 目的関数毎の関数評価回数に対する目的関数値の推移．より左下を推移するアルゴリズムが優れているとみなすことができる．実線 (solid line) が従来手法，破線 (dashed line) が従来手法に提案手法 (適応座標選択) を適用したもの，点線 (dotted line) が従来手法にランダム座標選択を適用したものを表している．赤線が sep-CMA, 黒線が LM-MA, 水色線が VD-CMA に関する各アルゴリズムに対応している．Sphere 関数においては，黒線の実線 (LM-MA) と水色線の実線 (VD-CMA) 及び黒破線 (LM-MA+ 提案手法) と水色破線 (VD-CMA+ 提案手法) が同様の推移をするためグラフ上では線が重なっている．Ellipsoid 関数においては，黒破線 (LM-MA+ 提案手法) と水色破線 (VD-CMA+ 提案手法) が同様の推移をするためグラフ上では線が重なっている．

関数は表 1 からも明らかなように，条件数は 1 の完全に良条件な関数であるため，座標選択により条件数を低減させることができない関数である．ランダム座標選択では，ハイパーパラメータとして与えた座標選択数に基づいて必ず座標軸を分割するため，適用前と比べて約 50% 関数評価回数が増大しており，性能が大幅に悪化している．一方で，提案手法では，二階偏微分を推定したことにより，座標を分割することなく探索を行った結果，二階偏微分の推定に必要なコストのみが加算され，性能の悪化を適用前と比べて約 10% に抑えることができている．

**Ellipsoid 関数を用いた比較**

Ellipsoid 関数は表 1 の通り，悪条件性のみを持つ関数である．図 1 (b) より，いずれの適用元アルゴリズムの場合にも，提案手法を適用した際に最も良い結果が得られた．ランダム座標選択では，LM-MA に適用した場合に，適用前よりも性能が悪化することが明らかとなった．また，LM-MA では適用前では目標値に到達することができていないが，提案手法を適用することで，表 1 の通り VD-CMA に提案手法を適用した場合とほぼ等しい関数評価回数で解に到達することができており，提案手法の優位性を示している (図 1 (b) では黒破線 (LM-MA+ 提案手法) と水色破線 (VD-CMA+提案手法) が重なってプロットされている)．

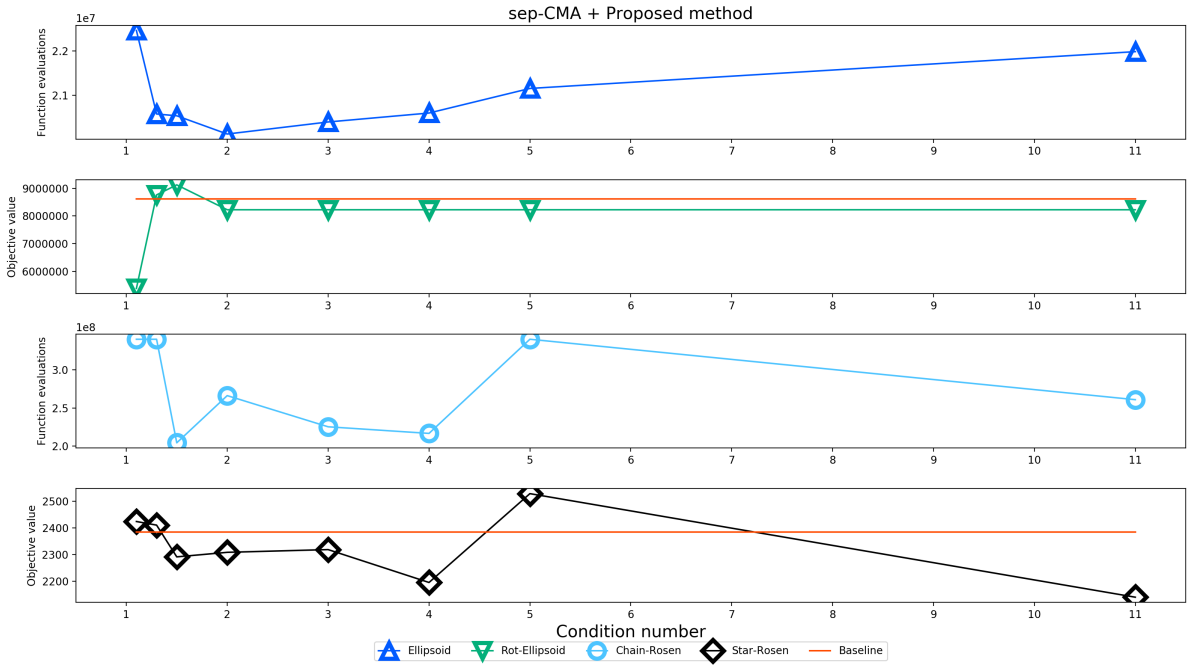


図 2: sep-CMA における許容条件数と性能の関係．Rot-Ellipsoid 関数では，許容条件数 1.3 以上で座標選択が行われずクラスタ数 1 となるため，値の差は誤差である．Rot-Ellipsoid 関数の許容条件数 3 以降は許容条件数 2 の結果をプロットした．

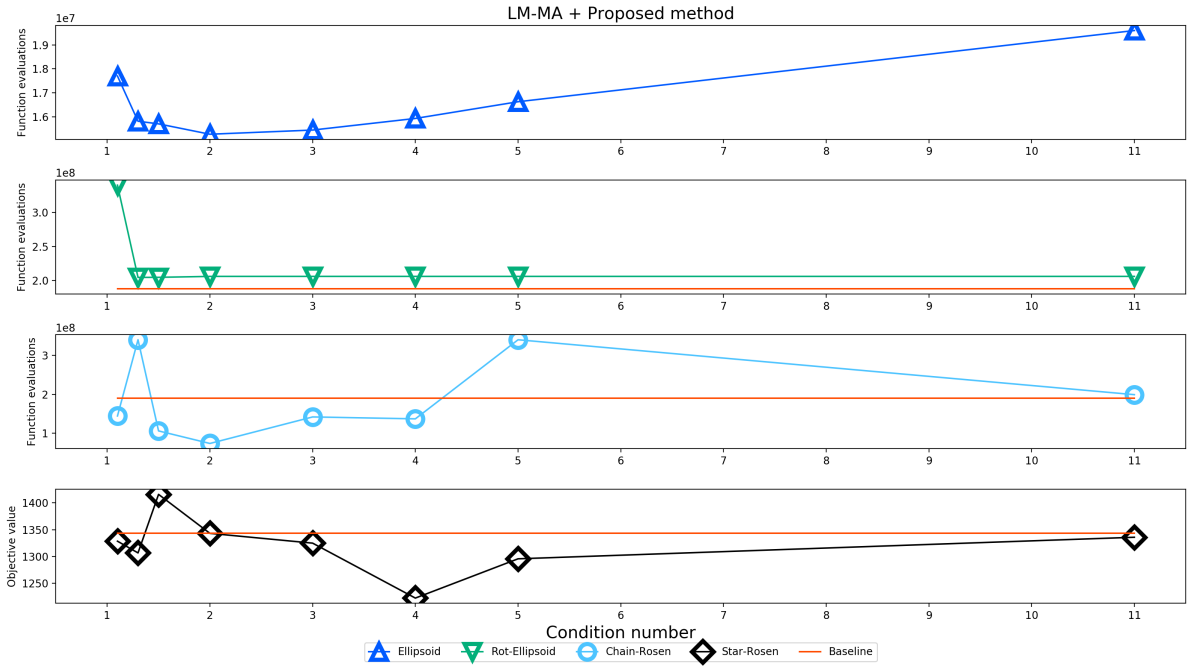


図 3: LM-MA における許容条件数と性能の関係．Rot-Ellipsoid 関数では，許容条件数 1.3 以上で座標選択が行われずクラスタ数 1 となるため，値の差は誤差である．Rot-Ellipsoid 関数の許容条件数 3 以降は許容条件数 2 の結果をプロットした．

**Rot Ellipsoid 関数を用いた比較**

Rot Ellipsoid 関数は表 1 の通り，ランダムに生成した直行行列を用いて回転させた入力ベクトルを Ellipsoid 関数で評価する関数であり，悪条件性と変数間依存性を持つ．図 1 (c) より，このケースでは，提案手法は優位性を示すことはできなかった．Rot Ellipsoid 関数は，入力ベクトルを直行行列で回転させているが，この写像により，入力の各軸全体に均一に依存関係が発生するため，二階偏微分の値だけでは条件数を十分に推定することができない．実際には二階偏微分の値は各軸についてほぼ等しい値を取り，それに基づいて条件数を推定すると良条件な関数であると誤認してしまい，今回のケースでは提案手法では座標の分割が行われず，適用前の手法と同様の結果が得られた (Sphere 関数の場合と同様に，二階偏微分の推定コストの分だけ性能が僅かに悪化している)．

**Chain-Rosenbrock 関数を用いた比較**

Chain-Rosenbrock 関数は表 1 の通り，変数間依存性と多峰

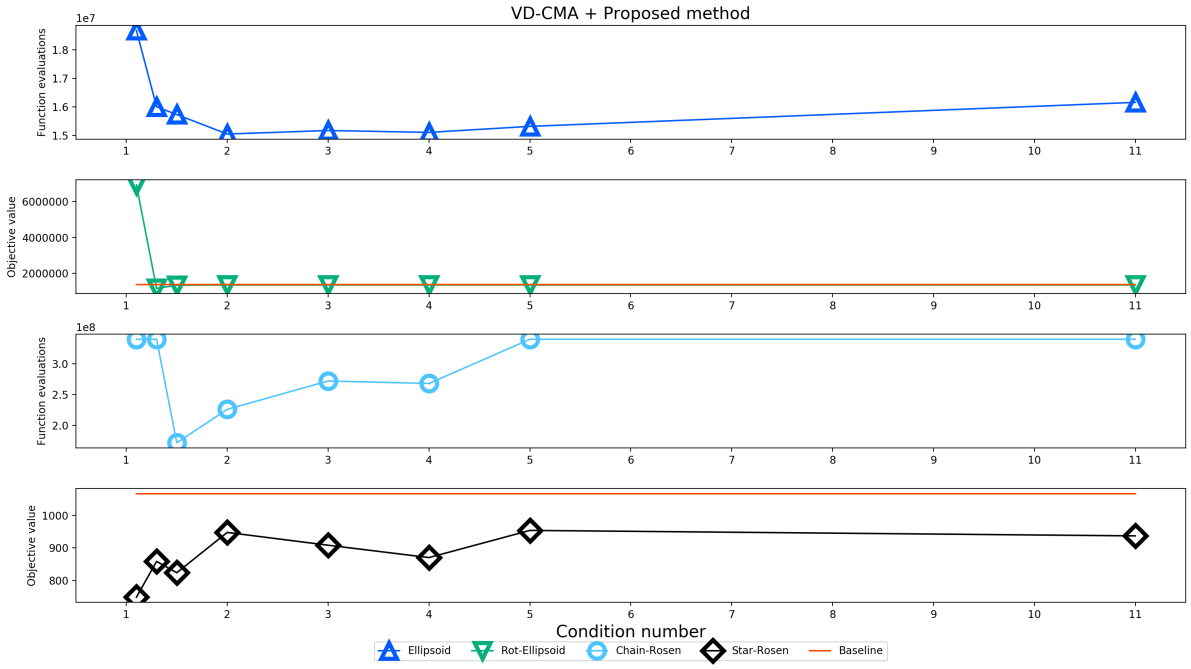


図 4: LM-MA における許容条件数と性能の関係．Rot-Ellipsoid 関数では，許容条件数 1.3 以上で座標選択が行われずクラスタ数 1 となるため，値の差は誤差である．Rot-Ellipsoid 関数の許容条件数 3 以降は許容条件数 2 の結果をプロットした．

性を持つ関数である．図 1 (d) よりこの場合には，いずれの適用元アルゴリズムについても提案手法を適用することで大幅な性能の良化がみられた．Chain-Rosenbrock 関数は，良条件な関数であると一般的に見なされており，条件数を軽減することで性能を向上させる手法である座標分割によって性能が向上することは，直感に反する現象であると言える．実際にランダム座標分割では，いずれの場合にも性能が悪化している．今回得られた結果は，非常に興味深い事象であり，次節でその分析を行う．

**Star-Rosenbrock 関数を用いた比較**

Star-Rosenbrock 関数は表 1 の通り，悪条件性，変数間依存性，多峰性を持つ関数である．一つ目の変数とそれ以降の変数との間にのみ依存関係を持つことが大きな特徴である．図 1 (e) よりこの場合には，いずれの適用元アルゴリズムについても，ランダム座標選択を行った場合に最良の結果が得られ，提案手法を用いた場合には適用前とほぼ同等の結果となった．本関数は，前述の通り，一つ目の変数がそれ以外の全ての変数に対して依存関係を持つため，座標を分割する際に一つ目の変数が均等な頻度で他の軸のクラスタに割り当てられるランダム座標分割が最も適していたと考えられる．

### 4.2 許容条件数と性能の関係

提案手法では，許容条件数をハイパーパラメタとして事前に設定する必要がある．そこで本節では，許容条件数の値をどの範囲に設定するべきであるかを確認するために，各アルゴリズムとベンチマーク関数毎に許容条件数と性能の関係をシミュレーションを通して分析する．以下では，許容条件数を [1.1, 1.3, 1.5, 2.0, 3.0, 4.0, 5.0, 11.0] の範囲で実験を行った．

図 2 に sep-CMA に提案手法を適用した場合の許容条件数と性能の関係を図示した．Ellipsoid 関数，Chain-Rosenbrock 関数については関数の評価回数を，Rot-Ellipsoid 関数，Star-Rosenbrock 関数については最大評価回数内における目的関数の最小値を示した．Rot-Ellipsoid 関数では，許容条件数が 1.3 以上の場合でクラスタ数が 1 となり座標分割が行われないため，値の差は試行毎の誤差である．また，Rot-Ellipsoid 関数における許容条件数 3 以上の結果は，許容条件数 2 で行ったシミュレーションの結果をプロットした．Chain-Rosenbrock 関数については，許容条件数 1.1, 1.3, 5.0 では，目的関数値が目標値に到達しなかったため，最大評価回数の値をプロットした．

図 3 に LM-MA に提案手法を適用した場合の許容条件数と性能の関係を図示した．Ellipsoid 関数，Chain-Rosenbrock 関数，Rot-Ellipsoid 関数については関数の評価回数を，Star-Rosenbrock 関数については最大評価回数内における目的関数の最小値を示した．Rot-Ellipsoid 関数では，許容条件数が 1.3 以上の場合でクラスタ数が 1 となり座標分割が行われないため，値の差は試行毎の誤差である．また，Rot-Ellipsoid 関数における許容条件数 3 以上の結果は，許容条件数 2 で行ったシミュレーションの結果をプロットした．Rot-Ellipsoid 関数の許容条件数 1.1 では，目的関数値が目標値に到達しなかったため，最大評価回数の値をプロットした．Chain-Rosenbrock 関数については，許容条件数 1.3, 5.0 では，目的関数値が目標値に到達しなかったため，最大評価回数の値をプロットした．

図 4 に VD-CMA に提案手法を適用した場合の許容条件数と性能の関係を図示した．Ellipsoid 関数，Chain-Rosenbrock 関数については関数の評価回数を，Rot-Ellipsoid 関数，Star-Rosenbrock 関数については最大評価回数内における目的関数の最小値を示した．Rot-Ellipsoid 関数では，許容条件数が 1.3

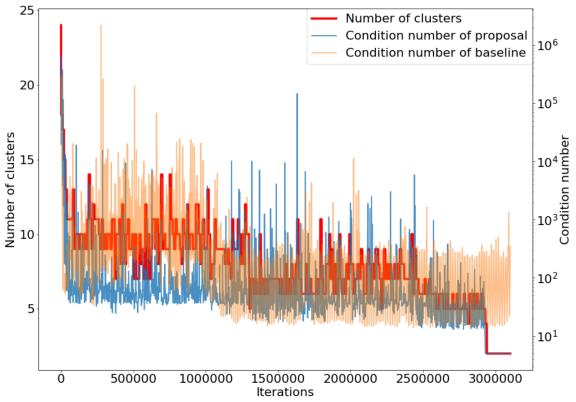


図 5: LM-MA に提案手法を適用した場合のクラスタ数 (赤線) と条件数 (青線) の変化．黄線は通常の LM-MA の条件数の変化を表している．縦軸左が各反復におけるクラスタ数，縦軸右が各反復における条件数を示している．

以上の場合でクラスタ数が 1 となり座標分割が行われないため，値の差は試行毎の誤差である．また，Rot-Ellipsoid 関数における許容条件数 3 以上の結果は，許容条件数 2 で行ったシミュレーションの結果をプロットした．Chain-Rosenbrock 関数については，許容条件数 1.1, 1.3, 5.0, 11.0 では，目的関数値が目標値に到達しなかったため，最大評価回数の値をプロットした．

以上の実験から，いずれの場合でも許容条件数を 1.5 4.0 の範囲に設定することで，プロットの曲線がベースラインの線を概ね下回り，良好な性能を示すことが明らかとなった．厳密に最も優れる許容条件数は，目的関数とアルゴリズムの組み合わせによって異なるため，更なる分析が必要であると言える．

### 4.3 Chain-Rosenbrock 関数の最適化における提案手法の動作の分析

前節で述べた通り，一般的に良条件とされる Rosenbrock 関数の最適化に提案手法を用いることで，直感に反し優れた結果が得られた．本節では，最適化中における提案手法のクラスタリングの振る舞いや条件数の変化から本事象の分析を行う．

図 5 に，LM-MA に提案手法を適用した場合のクラスタ数 (赤線) 及び推定条件数の絶対値 (青線) の変化を示した．また，黄線は通常の LM-MA の条件数の変化を表している．本図より，Chain-Rosenbrock 関数では最適化の反復中に大きく条件数が変化していることが読み取れる．実際に，表 1 に示す Chain-Rosenbrock 関数の式を二階偏微分すると，

$$\frac{\partial^2 f}{\partial x_i^2} = \begin{cases} 202 + 400(3x_i^2 - x_{i+1}) & , 1 \leqq i \leqq d-1 \\ 200 & , d = n \end{cases}, \quad (13)$$

となり，入力の値 (各反復における探索の中心点) によって，値が大きく変化することが分かる．図 5 より，提案手法におけるクラスタ数は，条件数の変化に応じて同様の軌跡を描いており，条件数が増大した場合にはクラスタ数を増やすことで座標軸をより分割し，条件数が減少した場合には座標の分割を抑制していることが分かる．Chain-Rosenbrock 関数のように多峰性を持ち，条件数が大きく変化するような目的関数について，提案手法のように条件数を考慮した座標分割を行うことで局所解を回避する性能が向上する可能性が示唆されているが，本事象に関しては更なる分析が必要であると言える．

## 5. おわりに

本論文では，CMA-ES において座標選択を行う場合に，選択する座標数を目的関数の悪条件性に適応させる悪条件性適応座標選択手法を提案した．本手法により，目的関数が悪条件な場合には選択する座標数が増大し，良条件な場合では減少することで，従来の座標選択が抱えていた良条件関数における性能の低下という課題を解決した．また，Chain-Rosenbrock 関数のように，探索地点によって条件数が変化するような目的関数においても提案手法により性能が向上することを確認した．

今後の展望として，設定した許容条件数と得られた座標軸のクラスタから，最適化問題における目的関数の景観分析や，大規模機械学習モデルにおけるパラメタの分布の分析に応用することが期待される．

## 謝　辞

本研究は，JST CREST JPMJCR19A4 及び JSPS 科研費 JP21H03445 の支援を受けたものである．

## 文　献

[1] 坂井節子, 高濱徹行. 最適化手法における関数評価回数の削減手法 -ポテンシャルモデルに基づく比較推定法の提案-. 数理解析研究所講究録, Vol. 1548, No. 7, pp. 61–70, 2007.

[2] 秋本洋平. Evolution Strategies による連続最適化 CMA-ES の設計原理と理論的基盤. 進化計算の時代 特集号, Vol. 60, pp. 292–297, 2016.

[3] Nikolaus Hansen. The CMA evolution strategy: A tutorial. 2016.

[4] Raymond Ros and Nikolaus Hansen. A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. In *Parallel Problem Solving from Nature – PPSN X*, Vol. 5199, pp. 296–305. Springer Berlin Heidelberg, 2008.

[5] Youhei Akimoto, Anne Auger, and Nikolaus Hansen. Comparison-based natural gradient optimization in high dimension. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation - GECCO '14*, pp. 373–380. ACM Press, 2014.

[6] Youhei Akimoto and Nikolaus Hansen. Projection-Based restricted covariance matrix adaptation for high dimension. In *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, GECCO '16, pp. 197–204, New York, NY, USA, July 2016. Association for Computing Machinery.

[7] Ilya Loshchilov, Tobias Glasmachers, and Hans-Georg Beyer. Large scale Black-Box optimization by Limited-Memory matrix adaptation. *IEEE Trans. Evol. Comput.*, Vol. 23, No. 2, pp. 353–358, April 2019.

[8] Ilya Loshchilov. LM-CMA: An Alternative to L-BFGS for Large-Scale Black Box Optimization. *Evolutionary Computation*, Vol. 25, No. 1, pp. 143–171, 2017.

[9] 清水洸希, 小宮山純平, 豊田正史. CMA-ES における高次元・悪条件最適化のための確率的次元選択手法. 電子情報通信学会論文誌 D, Vol. J103-D, No. 05, 2020.

[10] Hiroki Shimizu and Masashi Toyoda. CMA-ES with Coordinate Selection for High-Dimensional and Ill-Conditioned Functions. *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, p. 209–210, 2021.