DEIM Forum 2019 A4-3

高次元悪条件最適化問題のための確率的次元選択CMA-ES

清水　洸希†　　小宮山純平††　　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,jkomiyama,toyoda}@tkl.iis.u-tokyo.ac.jp

あらまし　本論文では，高次元かつ悪条件な目的関数の最適化のために，CMA-ES において目的関数の次元を任意の次元の組に確率的に分割し，一組ずつ更新するアルゴリズムを提案する．CMA-ES は，進化戦略 (ES) に分散共分散行列を導入することで，変数間の依存関係を考慮した最適化を行うことを可能としたアルゴリズムで，ブラックボックス最適化問題において優れた性能を発揮している．一方で，100,000 次元の Ellipsoid 関数のような非常に高次元かつ悪条件な最適化問題においては十分な最適化が行えないことが報告されている [1]．本研究では，CMA-ES の変数の一つであるステップサイズが，目的関数が悪条件である場合に，分散共分散行列の適応の妨げとなっていることが，原因の一つとして考えられることを実験を通して確認した．この問題の解決のため，提案手法では，各世代で目的関数の次元 d を次元 s $(0 < s \leqq d)$ の組に確率的に分割し，その次元の組ごとに分散共分散行列の適応とステップサイズの更新を行う．実験では，Ellipsoid 関数や Star 型の Rosenbrock 関数といったベンチマーク関数を用いて，提案手法と既存の CMA-ES および L-BFGS 法との比較を行い，その有効性を確認した．

キーワード　進化計算, 最適化, 機械学習

1. はじめに

最適化問題とは，与えられた目的関数に対して，それを最小化あるいは最大化する変数の組を求める問題である．特に目的関数が明示的に与えられておらず，入力に対する出力のみが得られるような場合をブラックボックス最適化と呼ぶ．

ブラックボックス最適化は実世界においても，鉄道のダイヤグラム作成や航空機の翼の設計等多数の応用が存在する．実世界のブラックボックス最適化において想定される目的関数の特徴として，悪条件性，多峰性，変数間依存性が指摘されている [2] [3]．また，近年では，ニューラルネットワークのように数十万次元に及ぶ非常に高次元なモデルも主流であり，このようなモデルでは前述の特徴がより現れやすいと考えられる．そのため，これらの特徴に対して有効かつ高次元においても適用可能なアルゴリズムの開発は社会的に非常に大きな意義が存在すると言える．

CMA-ES (Covariance Matrix Adaptation Evolution Strategy) [4] はブラックボックス最適化における最も優れたアルゴリズムの 1 つとして知られている．CMA-ES は，進化戦略 (ES) に分散共分散行列を導入したアルゴリズムで，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 $\boldsymbol{m}$ 及び分散共分散行列 $\boldsymbol{C}$) とステップサイズ σ 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 の持つ大きな特徴として，分散共分散行列の適応による変数間の依存性を考慮した個体生成，大域ステップサイズによる解探索の効率の向上が挙げられ，rank-one update と rank-μ update の組み合わせによる個体数の増加への対応, CSA (Cumulative Step-size Adaptation) を用いた変数のスケール性を考慮したステップサイズの更新により，一般に前述の悪条件性，多峰性，変数間依存性を持つ目的関数に対して有効であるとされている [5].

CMA-ES の欠点として，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，分散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-ES [6] をはじめとして，さまざまな手法が提案されている．特に sep-CMA-ES はその計算量の小ささにも関わらず，目的関数に変数間の依存性がない場合は，CMA-ES より高速に解を発見することができ，非常に優れたアルゴリズムである．これらの計算量を改善したアルゴリズムの貢献により 10,000 次元程度までの最適化問題への拡張を可能としたが，それ以上の超高次元な場合では依然として課題が存在する．Loshchilov は，原因については言及していないものの，実験を通して sep-CMA-ES は 100,000 次元の Ellipsoid 関数について，十分な最適化を行えないことを指摘している [1]．本論文では，実験を通して，この問題の原因が，目的関数が超高次元なだけではなく，非常に強い悪条件性を持つ場合に，ステップサイズの収束と分散共分散行列の収束が大きく乖離し，探索を妨げているためであることを確かめた．

この問題を解決するため，本論文では確率的に次元を選択する CMA-ES を提案する．提案手法では，目的関数の次元を任意の次元の組に確率的に分割する．そして世代毎に組を選択し，その選択された次元について，平均，分散共分散行列，ステップサイズといった変数の更新を行う．このとき，ステップサイズを従来のようなスカラー値ではなく，ベクトルを用いて表現する．このように，各世代で一部の次元のみを用いて変

数を更新することで，次元間での極端なスケールの違いに適応することが可能となり，超高次元の Ellipsoid 関数や Star 型 Rosenbrock 関数といった悪条件性が非常に強い関数の最適化において，従来の CMA-ES よりも探索の速さや解の優劣の観点から優れた結果を得た．

以下，2 章で関連研究について説明し，次に 3 章で既存の CMA-ES を用いた実験を通して前述の問題の原因について検討を行う．そして，4 章でその問題を解決するため，確率的に次元を選択する CMA-ES を提案し，5 章で実験を通してその評価を行う．最後に 6 章でまとめと今後の展望について述べる．

2. 関連研究

本章では，CMA-ES のアルゴリズムの概要と，目的関数の特徴に応じた CMA-ES の改良の研究について述べる．

2.1 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，次に sep-CMA-ES について述べる．最後に CMA-ES のハイパーパラメタの設定について述べる．

2.1.1 $(\mu/\mu_w, \lambda)$-CMA-ES

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 $\boldsymbol{p_c}^{(0)} \in \mathbb{R}^d$ および $\boldsymbol{p_\sigma}^{(0)} \in \mathbb{R}^d$ を $\boldsymbol{0}$ 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 $\mathcal{N}(\boldsymbol{0} \in \mathbb{R}^d, \boldsymbol{I} \in \mathbb{R}^{d\times d})$ から，λ 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．

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

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

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

[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 (λ) 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．

$$\boldsymbol{dy}^{(t)} = \sum_i^\lambda w_i \boldsymbol{y_i}^{(t)} \tag{4}$$

$$\boldsymbol{dz}^{(t)} = \sum_i^\lambda w_i \boldsymbol{z_i}^{(t)} \tag{5}$$

[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，$\mu_w = \frac{1}{\|\boldsymbol{w}\|}$, $\chi_d = \mathbb{E}[\|\mathcal{N}(\boldsymbol{0}, \boldsymbol{I})\|] \simeq \sqrt{d}(1 - \frac{1}{4d} + \frac{1}{21d^2}) \in \mathbb{R}$ とし，$c_\sigma, c_c \in \mathbb{R}$ を各進化パスの学習率とする．

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

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

$$\begin{aligned} \boldsymbol{p_c}^{(t+1)} &= (1 - c_c)\boldsymbol{p_c}^{(t)} \\ &+ h_\sigma^{(t+1)} \sqrt{c_c(2 - c_c)\mu_w}\boldsymbol{dy} \end{aligned} \tag{8}$$

[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-μ 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．

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

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

$$\begin{aligned} \boldsymbol{C}^{(t+1)} &= \boldsymbol{C}^{(t)} + \eta_{c_1}\left(OP(\boldsymbol{p_c}^{(t+1)}) - \boldsymbol{C}^{(t)}\right) \\ &+ \eta_{c_\mu} \sum_i^\lambda w_i \left(OP(\boldsymbol{y_i}^{(t)}) - \boldsymbol{C}^{(t)}\right) \end{aligned} \tag{11}$$

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 σ が 0 に，分散共分散行列 $\boldsymbol{C}$ が $\boldsymbol{0}$ 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．

2.1.2 sep-CMA-ES

sep-CMA-ES は目的関数が高次元である場合に，時間・空間計算量の点から CMA-ES の適用が難しいことから，CMA-ES の分散共分散行列を対角成分に限定することで，その問題の解消を図ったアルゴリズムである．sep-CMA-ES では時間計算量・空間計算量ともに $O(d)$ となる．

また，sep-CMA-ES では分散共分散行列の学習率を増加させており，目的関数の入力次元を d として，rank-one update, rank-μ update ともに $(d + 2)/3$ を係数にかける．

2.1.3 ハイパーパラメタ

CMA-ES はアルゴリズム中のハイパーパラメタがいくつか存在するが，それらは目的関数の入力次元によって定めている．本論文中では，文献 [7] に示されている推奨値を用いる．以下目的関数の入力次元を d とし，各値について示す．

$$\begin{aligned}
&\lambda = 4 + 3\lfloor \ln(d) \rfloor, \quad \mu = \lfloor \frac{\lambda}{2} \rfloor \\
&w_i = \ln(\frac{\lambda+1}{2}) - \ln(i), \quad \boldsymbol{w} = \frac{[w_1, w_2, \cdots, w_\lambda]}{\|[w_1, w_2, \cdots, w_\lambda]\|} \\
&c_\sigma = \frac{\mu_w + 2}{d + \mu_w + 5}, \quad c_c = \frac{4 + \frac{\mu_w}{d}}{d + 4 + \frac{2\mu_w}{d}} \\
&d_\sigma = 1 + c_\sigma + 2\max\left(0, \sqrt{\frac{\mu_w - 1}{d+1}} - 1\right) \\
&\eta_m = 1, \quad \eta_{c_1} = \frac{2}{(d+1.3)^2 + \mu_w} \\
&\eta_{c_\mu} = \min\left(1 - \eta_{c_1}, \frac{2(\mu_w - 2 + \frac{1}{\mu_w})}{(d+2)^2 + \mu_w}\right)
\end{aligned} \tag{12}$$

2.2 ステップサイズの役割

CMA-ES では，多変量正規分布における分散をステップサイズ σ と分散共分散行列 $\boldsymbol{C}$ に分割して表現している．ステップサイズの更新には，式 (10) の通り，ステップサイズの進化パスのノルム $\|\boldsymbol{p_\sigma}\|$ と，多変量正規分布のノルムの期待値 χ_d の比が用いられ，$\|\boldsymbol{p_\sigma}\| > \chi_d$ の場合，ステップサイズは増大し，$\|\boldsymbol{p_\sigma}\| < \chi_d$ の場合，減少する．ステップサイズの進化パス $\boldsymbol{p_\sigma}$ は，式 (7) の通り，良い評価を得た解候補の生成の基になった乱数 $z^{(t)}$ の重み付き和 $dz^{(t)}$ を累積した値である．つまり，現在の探索域が最適解から離れている場合，$\|\boldsymbol{p_\sigma}\|$ は増大し，最適解に近い場合は減少する傾向にある．従って，ステップサイズは探索地点と最適解との間の距離に応じて χ_d を閾値として増減することで，探索速度の向上を図る役割を持っていると言える．

2.3 CMA-ES の改良

CMA-ES の改良は多く存在し，目的関数の特徴に応じて設計される．目的関数の特徴は主に，変数間依存性 (Separability), 悪条件性 (Ill-Conditioning), 多峰性･大谷構造 (Multimodality) に分類される [2] [3]．

変数間依存性とは，目的関数の変数が分離可能であるか否かを指す．全ての変数について分離可能であるならば，全ての変数を独立に最適化しても最適値に到達することができるが，そうでなければ変数間の依存関係を考慮する必要性がある．CMA-ES は分散共分散行列行列を用いることで解決を図っているが，高次元 (large scale) の場合には時間・空間計算量が重大な問題となる．前述の sep-CMA-ES や，LM-CMA [1], VD-CMA [8] といった研究では，分散共分散行列のうち保持する値を制限することで，時間・空間計算量を削減しつつ，変数間の共起を考慮した最適化を可能としている．

悪条件性とは，目的関数における各次元のスケールの差が非常に大きい状態を指し，目的関数のヘッセ行列の最大固有値と最小固有値の比で定義される条件数によって定量的に測ることができる．

表 1 Benchmark functions

Function	Definition
Sphere	$\sum_{i=1}^{n} x_i^2$
Ellipsoid	$\sum_{i=1}^{n} (1000^{\frac{i-1}{n-1}} x_i)^2$
Rosenbrock (Star)	$\sum_{i=2}^{n} \left(100(x_1 - x_i^2)^2 + (1 - x_i)^2\right)$

現在の CMA-ES ステップサイズ更新手法の主流であり，前述の $(\mu/\mu_w, \lambda)$-CMA-ES にも採用されている Cumulative Step-size Adaptation (CSA) [9] [10] は，ステップサイズを進化パスのマハラノビス距離を用いて更新することで，悪条件性に対する頑健性の向上を図っている．

多峰性・大谷構造とは，目的関数が局所最適解を持つ状態 (non-convex な状態) を指す．個体数の増加 [11]，再スタート毎に個体数を変化させる再スタート戦略 [12]，最適化の度合いに応じて世代毎に個体数を変化させる Population Size Adaptation (PSA) [13] が提案されている．

本研究の提案手法では，従来の手法がいずれも，一反復中に全ての次元を同時に更新するのに対し，一反復中に更新する次元の選択を行う．

3. 予備実験: 高次元悪条件関数に対する CMA-ES の挙動

本章では，高次元かつ悪条件な目的関数における CMA-ES の挙動，即ち，良条件な目的関数の場合と比較してどのような性能低下やパラメタの推移の変化が見られるかを確認する．はじめに，良条件かつ変数間依存性のない Sphere 関数を用いた実験を行い，その結果を他の目的関数に対してのベースラインとする．次に，悪条件な関数として，Ellipsoid 関数及び Star 型 Rosenbrock 関数を用い，Sphere 関数の結果との比較を行う．一般的によく知られる Chain 型の Rosenbrock 関数は隣り合う変数間の依存性が強い関数として知られているが，Star 型の場合は，一つ目の変数とそれ以降の変数との間に強い依存性を持つ [14] ことから，一つ目とそれ以降の変数について依存の度合いが異なり，高次元な場合については悪条件な関数とみなすことができる．また，本論文中で用いるベンチマーク関数を表 1 に示した．

3.1 Sphere 関数を用いた実験

はじめに，sep-CMA-ES を用いて 100,000 次元の Sphere 関数の最適化を行う．初期値については，$\boldsymbol{m}^{(0)} = U(-5, 5)$, $\boldsymbol{C}^{(0)} = \boldsymbol{I}$, $\sigma^{(0)} = 1.0$, $\lambda = 4 + 3\lfloor \ln(d) \rfloor$ と設定し，以降の全ての実験で同一の設定を用いた．. また，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．

評価回数 (Func evals) に対する目的関数の値 (Objective), ステップサイズ (Sigma)，分散共分散行列 (Covariance) の変化を，図 1 に示した．分散共分散行列については，全ての次元の平均の値 (Covariance (mean)), 1 次元目の値 (Covariance (dim=1)), 100,000 次元目の値 (Covariance (dim=100,000)) を示した．図より，約 1.5×10^7 回の評価回数で目標値に到達し，ステップサイズ及び分散共分散行列の値は単調に減少して

いることが伺える．また，分散共分散行列が次元別に示したものと平均を示したものとでおおよそ同じ値を取りつつ推移していることから，Sphere 関数の良条件性を確認することができる．

3.2 Ellipsoid 関数を用いた実験

本節では，sep-CMA-ES を用いて，Loshchilov による実験 [1] の再現を行った．目的関数以外の実験設定については，前節と同様に扱った．

評価回数 (Func evals) に対する目的関数の値 (Objective)，ステップサイズ (Sigma)，分散 (Covariance) の変化を図 2 に示した．また，前節と同様に，分散共分散行列について各次元の値の推移と平均値の推移を示した．図より，約 6.5×10^7 回の評価回数で目標値に到達しており，前節の Sphere 関数の結果と比較して，約 5.0×10^7 回程度多くなった．目的関数の値の減少は，Sphere 関数の場合と Ellipsoid 関数の場合のいずれも約 1.0×10^6 回付近から始まっており，Ellipsoid 関数の方が評価回数あたりの最適化速度が遅いことがわかる．また，Sphere 関数の場合では，ステップサイズと分散共分散行列が共に単調に減少しているのに対し，Ellipsoid 関数の場合では，分散共分散行列の平均 (Covariance (mean)) については，スケールの大きな次元の分散 (Covariance (dim=100,000)) への適応に応じて上昇したのち減少に転じており，ステップサイズについては単調減少ではあるものの，分散共分散行列の平均の上昇中は緩やかであり，スケールの大きな次元の適応後は急速に減少している．スケールの大小に関わらず単一のステップサイズを適用しているため，スケールの小さな次元についてはステップサイズが過大となり，スケールの大きな次元については過小となることで，結果として分散共分散行列とステップサイズの間の乖離が Sphere 関数の場合と比較して拡大していると考えられる．この乖離が Sphere 関数の場合と比較して Ellipsoid 関数の最適化速度を下げていると推測され，ステップサイズをより各次元のスケールの大きさに適応させ，この乖離を抑制することでこの差を軽減することができると考えられる．

3.3 Star 型 Rosenbrock 関数を用いた実験

本節では，10,000 次元の Star 型 Rosenbrock 関数を用いた実験を行う．Star 型 Rosenbrock 関数では，一つ目の変数はそれ以降の全ての変数との間に依存関係を持つのに対して，二つ目以降の変数は一つ目の変数との間のみ依存関係を持つため，一つ目とそれ以降の変数との間でスケールの差が生じ，高次元になるほど悪条件性が高まる．

前節と同様に結果を，図 3 に示した．目標値には到達せず，最大評価回数時の目的関数の値は約 2252.2 となった．ステップサイズに注目すると，1.0×10^6 回の評価回数付近で，一次元目の変数の分散と共に収束に向かってしまっている．その後，一次元目の変数の分散も上昇に転じ，全ての次元の分散の平均が大きく上昇しているのに対して，ステップサイズは収束してしまい，最適化が緩やかになっていることがわかる．Star 型 Rosenbrock 関数は一次元目の変数の影響が非常に大きく，一次元目の変数の収束がステップサイズに大きな影響を及ぼしたと考えられる．

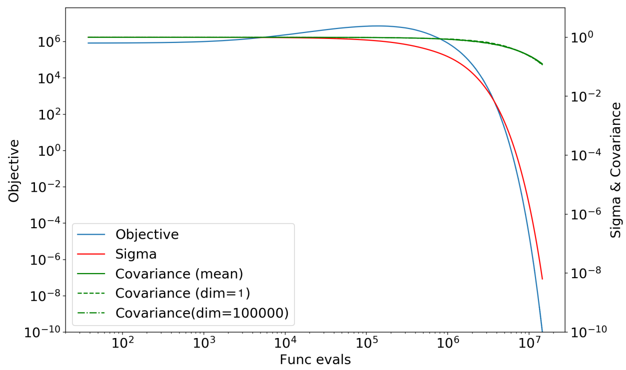

図 1　100,000 次元の Sphere 関数における各パラメタの推移 (sep-CMA-ES)

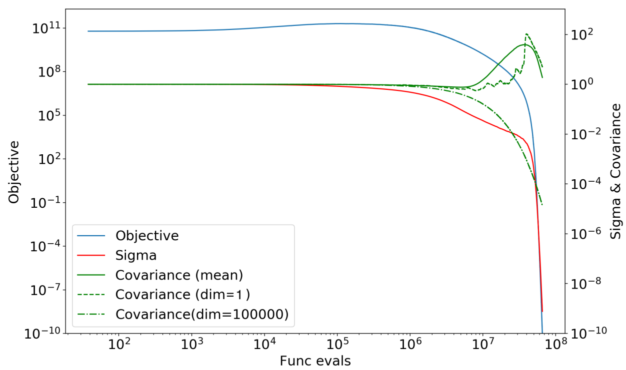

図 2　100,000 次元の Ellipsoid 関数における各パラメタの推移 (sep-CMA-ES)

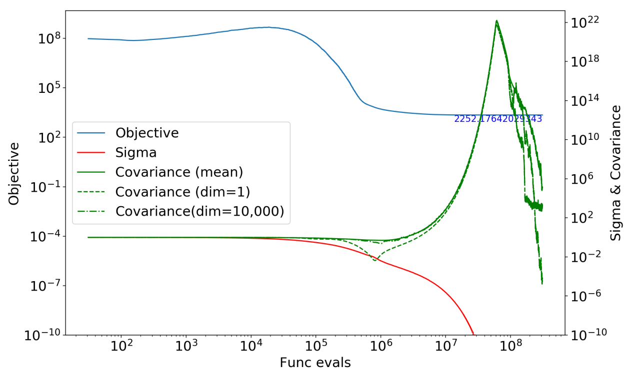

図 3　10,000 次元の Star 型 Rosenbrock 関数における各パラメタの推移 (sep-CMA-ES)

4. 提 案 手 法

本章では，予備実験の結果を受け，ステップサイズの大域的な収束性を維持しつつ，分散共分散行列の局所的な収束を考慮した変数の更新を目的とするアルゴリズムの提案を行う．

4.1 更新次元の確率的選択

本節では，CMA-ES において各世代で更新する次元を確率的に選択するアルゴリズムを提案する．提案手法では，目的関数の入力次元をインデックス化し，そのインデックスの要素をランダムに並び替え，予め指定した次元長の組に分割する．そして，分割された組のインデックスに指定された次元について，CMA-ES の各パラメタから区分行列及びベクトルを抽出し，解

候補を生成する．パラメタの更新についても，指定された次元についてのみ行う．インデックスの最後の要素まで参照を終えたら，再びインデックスをランダムに並び替える．この手法は CMA-ES と sep-CMA-ES の双方に適用でき，アルゴリズムの変更点は同様なため，CMA-ES への適用について以下の項で述べる．

4.1.1 CMA-ES への適用

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 $\boldsymbol{p_c}^{(0)} \in \mathbb{R}^d, \boldsymbol{p_\sigma}^{(0)} \in \mathbb{R}^d = \boldsymbol{0}$ 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 $\boldsymbol{p}'_{\boldsymbol{c}} \in \mathbb{R}^s, \boldsymbol{p}'_{\boldsymbol{\sigma}} \in \mathbb{R}^s$ 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，λ 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 (λ 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 $\boldsymbol{p}'_{\boldsymbol{\sigma}}$ 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．

4.1.2 ハイパーパラメタ

提案手法ではハイパーパラメタについて，通常の CMA-ES と同様の式を用いる．ただし，ハイパーパラメタの引数となる次元の値については，目的関数の入力次元数 d ではなく，各世代ごとに選択された次元数 s を用いる．また，sep-CMA-ES への適用の場合は，sep-CMA-ES と同様に分散共分散行列の rank-one update, rank-μ update 双方の学習率に $(s+2)/3$ の係数をかける．

4.2 手法の特徴

進化計算において，最適化を行う変数を制限する手法は，Zhang らの OpenAI ES [15] でも用いられており，目的関数が高次元である場合に，集団サイズを維持した状態で更新する次元を減らすことができ，分割された次元の更新のタイミングを揃えることで並列化も可能なため，有効であるとされている．

本研究では，次元のインデックスを最後まで参照した際に，インデックスベクトルの要素をランダムに並び替えることで，制限された次元に含まれる次元の組が毎回変わるように設計を行なった．これは Rosenbrock 関数のように次元間に依存性がある場合に有効であると考えられる．また，インデックスベクトルを先頭から順番に参照していくことで，全ての次元が均一に選択される．これは，選択される回数にばらつきが発生した場合に，選択されなかった次元の共起成分が維持され続け，世代を重ねるごとに対角成分に対して大きな値をとって最適化を妨げることを防ぐためである．

また，ステップサイズをベクトルに変更し，選択された次元を用いて更新を行なっている点が，従来の CMA-ES との最大の相違点である．ステップサイズは，式 (10) の通り，従来の CMA-ES ではスカラーで表すことしかできないが，次元の選択を行うことでベクトルで表すことが可能となり，ステップサイズの表現力を向上させることが可能となる．従来の CMA-ES では，目的関数の各次元のスケールが大きく異なっていた場合に，分散共分散行列の局所的な収束との乖離が大きいという問題が生じていたが，提案手法では，大域的な収束を維持しつつ分散共分散行列の局所的な収束にも対応できると考えられる．

最後に，時間・空間計算量について考える．空間計算量については元の分散共分散行列の値を保持するため $O(d^2)$ (sep-CMA-ES の場合は $O(d)$) で変わらない．一方で，時間計算量については，分散共分散行列の固有値分解について，選択した次元のオーダーで行うことができる．

表 2　計算機環境

CPU	Intel Xeon E5-2680 v4 2.40GHz
RAM	256GB
GPU	NVIDIA Quadro P6000
GPU 環境	Cuda9.0 Cudnn7.2 CuPy4.3.0
OS	Ubuntu 16.04.3 LTS

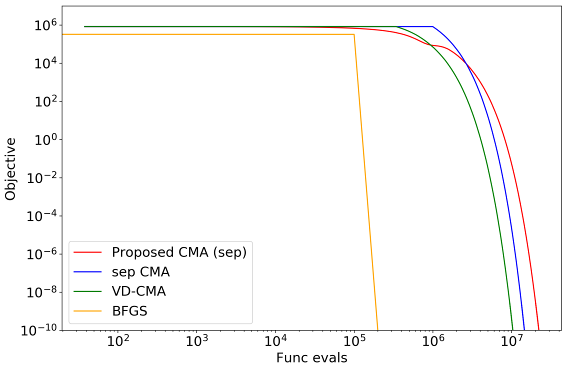

図 4　100,000 次元の Sphere 関数における各アルゴリズムの比較

5. 評価実験

本章では，提案手法の高次元悪条件問題における性能を確認する．目的関数は予備実験と同様の関数を用いた．本実験で用いた計算機環境を表 2 に示した．

5.1　高次元悪条件関数における各アルゴリズムの比較

本節では，提案手法，既存の CMA-ES，L-BFGS 法との性能の比較を行う．既存の CMA-ES として，sep-CMA-ES 及び VD-CMA を用いた．VD-CMA は，分散共分散行列のベクトルを用いた近似を提案したアルゴリズムで，計算量を低減しつつ変数間の依存性を考慮した最適化を行うことができる．L-BFGS 法は，勾配法をベースとしたアルゴリズムの中で最も優れたものの一つとして知られている．本実験では，L-BFGS 法について目的関数の一階導関数を既知の情報として与え，一反復あたりの評価回数については，目的関数の評価回数に各次元における導関数の計算回数を加えた回数とした．また，初期値，目標値，最大評価回数等の設定については予備実験と同様とし，提案手法における各反復の選択次元数は 100 とした．

各アルゴリズムについて，評価回数 (Func evals) に対する目的関数の値 (Objective) の推移を，Sphere 関数について図 4, Ellipsoid 関数について図 5, Star 型 Rosenbrock 関数について図 6 に示した．また，各関数について提案手法における，評価回数に対するステップサイズと分散の推移を，Sphere 関数について図 7, Ellipsoid 関数について図 8, Star 型 Rosenbrock 関数について図 9 に示した．

はじめに，Sphere 関数では，L-BFGS が最も少ない評価回数で目標値に到達し，提案手法は約 2.5×10^7 回で最も多くの評価回数を必要とした．提案手法は，一反復あたりに更新を行う次元数を制限するため，良条件な関数では優位性を示すことはできなかった．

次に，Ellipsoid 関数では，約 4.5×10^7 回と最も少ない評価回数で目標値に到達し，sep-CMA-ES よりも約 2.0×10^7 回程度評価回数が少なくなった．各パラメタについて見ると，予備実験における sep-CMA-ES の場合と比較して，分散共分散行列とステップサイズの推移間の乖離が小さく抑えられており，乖離の抑制が結果に影響したと考えられる．

最後に，Star 型 Rosenbrock 関数では，いずれのアルゴリズムも目標値には到達しなかったが，提案手法が最大評価回数時に目的関数の値が約 59.76 となり最小の値に到達した．こちらの場合でも，ステップサイズと分散の乖離が，sep-CMA-ES よりも小さくなったことが原因であると考えられる．

5.2　選択次元の大きさについての比較

本節では，提案手法の選択次元の大きさの違いついての性能の比較を行う．100,000 次元の Ellipsoid 関数と 10,000 次元の Star 型 Rosenbrock 関数に対して，選択次元を 10, 100, 1,000

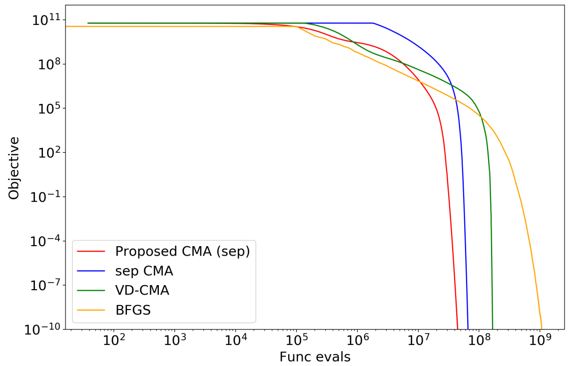

図 5　100,000 次元の Ellipsoid 関数における各アルゴリズムの比較

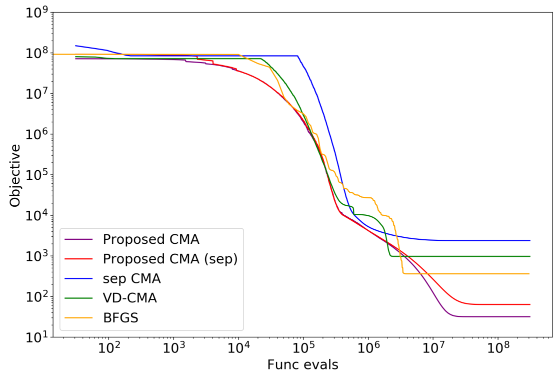

図 6　10,000 次元の Star 型 Rosenbrock 関数における各アルゴリズムの比較

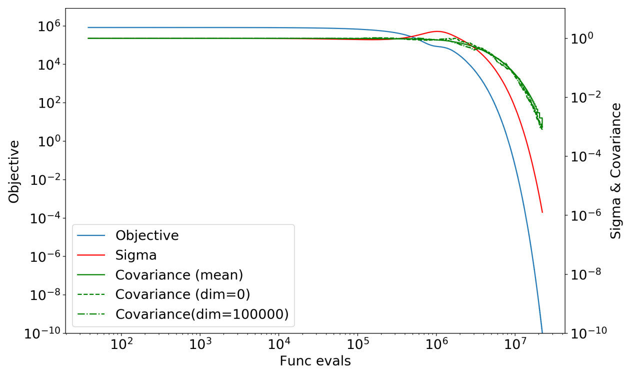

図 7　100,000 次元の Sphere 関数における各パラメタの推移 (提案手法)

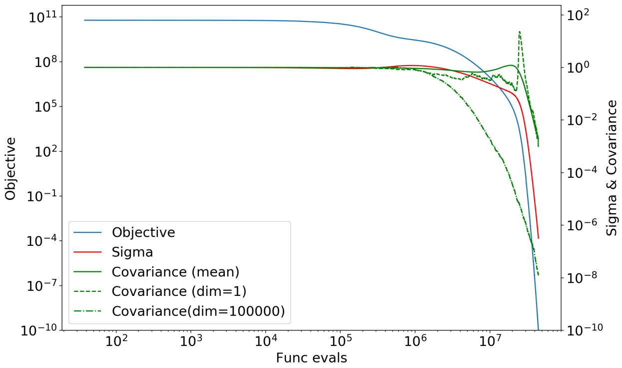

図 8　100,000 次元の Ellipsoid 関数における各パラメタの推移 (提案手法)

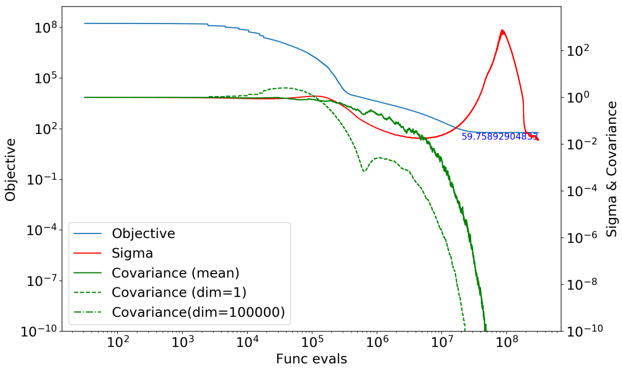

図 9　10,000 次元の Star 型 Rosenbrock 関数における各パラメタの推移 (提案手法)

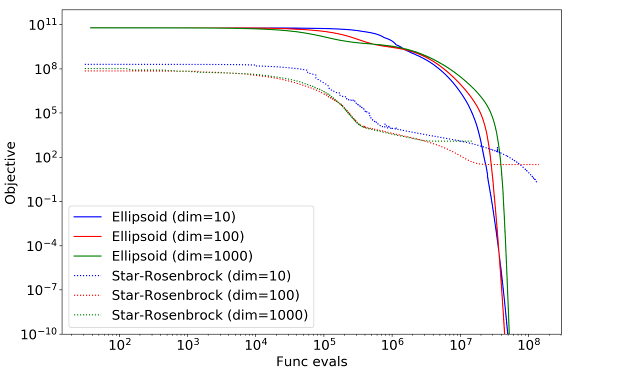

図 10　選択次元数を変化させた場合の評価回数毎の目的関数の値の推移

と変えて実験を行った．Ellipsoid 関数に対しては sep 型の提案手法を用い，Star 型 Rosenbrock 関数については通常の (分散共分散行列の全ての要素を考慮する) 提案手法を用いた．目標値は 10^{-10} とした．

評価回数あたりの目的関数の値を Fig. 10 に示す．Ellipsoid 関数では，選択次元を 100 としたものが最も少ない評価回数で目標値に到達した．一方で，Star 型 Rosenbrock 関数では，目標値に到達したものはなかったが，選択次元を 10 としたものが最もよい値を得た．以上の結果より，さらなる検証が必要ではあるが，目的関数の入力次元数に対して 0.1%程度の選択次元が適しているという推測が得られた．

5.3　次元の制限方法についての比較

本節では，提案手法における次元の制限方法について，ランダムに制限する場合と，制限する次元の組を固定する場合とで比較を行う．後者については，各次元について先頭から 100 ずつ制限を行った．

はじめに，Ellipsoid 関数について，評価回数に対する目的関数値，分散共分散行列，ステップサイズの推移を，図 11 に示した．図より，ランダムに制限する場合が約 4.5×10^7 回で目標値に到達したのに対し，組を固定した場合では，約 2.0×10^7 回で到達した．Ellipsoid 関数では，各次元について単調にスケールが増大するため，順番に次元を固定した場合，スケールが近い次元同士が同じ組に含まれることが影響したと考えられる．

次に，各次元について係数の順番をランダムに入れ替えた Ellipsoid 関数について，評価回数に対する目的関数値，分散共分散行列，ステップサイズの推移を，図 12 に示した．前述の通り，従来の Ellipsoid 関数は各次元について単調にスケールが変化するため，順番を入れ替えることでその影響の確認を行う．図より，どちらの制限方法の場合でもほぼ同様の結果が得られた．Ellipsoid 関数の係数をランダムに入れ替えたことにより，次元の組を固定した場合でも，ランダムに制限する場合と同等の状態になったと考えられる．

最後に，Star 型 Rosenbrock 関数について，評価回数に対する目的関数値，分散共分散行列，ステップサイズの推移を，図 13 に示した．図より，どちらの場合も目標値に到達できなかったが，ランダムに制限を行ったものが，最大評価回数において，より良い値に到達することができた．Star 型 Rosenbrock 関数では，一つ目の変数とそれ以降の変数との間に依存関係があるため，変数の組にそれらが同時に含まれることが望ましいと考えられる．ランダムに制限を行う場合は，確率的に依存関係のある変数同士が含まれる次元の選択が実現されるのに対し，固定した場合は，はじめの選択に含まれなかった変数については実現されないため，この点が結果に影響を与えたと考えられる．

以上より，変数の制限方法について，目的関数における変数間のスケールの近さや依存関係が既知の場合は，それらを考慮することで最適化性能を高めることが可能であると言える．一方で，目的関数がブラックボックスである場合では，現状ではランダムな制限が妥当であると考えられる．

6.　お わ り に

本論文では，CMA-ES が最適化を行う次元を各世代でランダムに制限し，制限された次元からステップサイズをはじめとする CMA-ES の変数を更新するアルゴリズムを提案し，その評価を行った．実験を通して，提案手法は，制限された次元の局所性を考慮したステップサイズの更新を行うことで，Ellipsoid 関数や Star 型 Rosenbrock 関数のような悪条件な目的関数については，従来の CMA-ES より少ない評価回数で優れた解に到達できることを明らかにした．一方で，Sphere 関数のように良条件な関数に対しては，世代毎に更新する変数を制限していることから，目標値への到達までの評価回数で従来の CMA-ES を下回った．また，次元の制限方法については，目的関数がブ

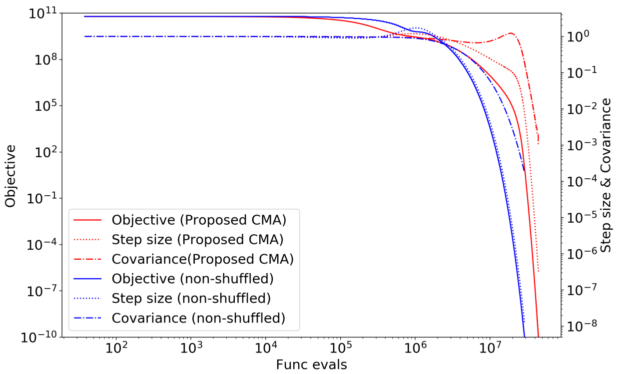

図 11　100,000 次元の Ellipsoid 関数における次元の選択方法による比較

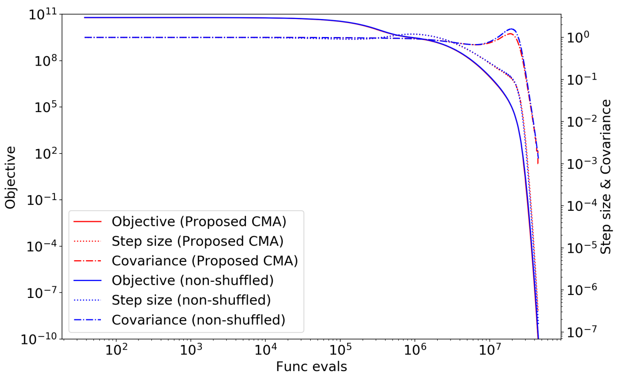

図 12　係数を入れ替えた 100,000 次元の Ellipsoid 関数における次元の選択方法による比較

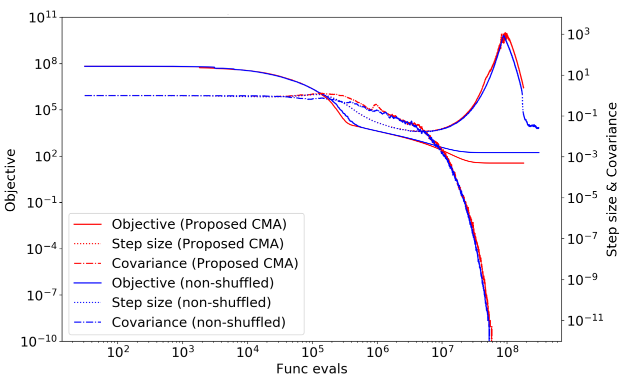

図 13　10,000 次元の Star 型 Rosenbrock 関数における選択方法による比較

ラックボックスである場合はランダムな制限が最も妥当であることを示した．

今後は，提案手法の性質についての更なる検証，本手法のニューラルネットワークの学習への適用，各反復における次元の制限について次元毎のスケールの近さや共起関係に最適化中に適応可能な選択方法の提案，世代毎の変数を選択することなくステップサイズをベクトル化する手法の提案等を行いたい．

謝　辞

本研究は JSPS 科研費 16H02905，17K12736 の助成を受けたものです．

文　献

[1] Ilya Loshchilov. A computationally efficient limited memory CMA-ES for large scale optimization. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation - GECCO '14*, pages 397–404. ACM Press, 2014.

[2] Nikolaus Hansen, Steffen Finck, Raymond Ros, and Anne Auger. Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. 37(1):55–57, 2009.

[3] 阪井 節子 and 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**, 2007.

[4] Nikolaus Hansen. The CMA Evolution Strategy: A Tutorial. `http://arxiv.org/abs/1604.00772`, 2018-11-18.

[5] 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**, 2016.

[6] Raymond Ros and Nikolaus Hansen. A Simple Modification in CMA-ES Achieving Linear Time and Space Complexity. In Günter Rudolph, Thomas Jansen, Nicola Beume, Simon Lucas, and Carlo Poloni, editors, *Parallel Problem Solving from Nature – PPSN X*, volume 5199, pages 296–305. Springer Berlin Heidelberg, 2008.

[7] Nikolaus Hansen and Anne Auger. Principled Design of Continuous Stochastic Search: From Theory to Practice. In Yossi Borenstein and Alberto Moraglio, editors, *Theory and Principled Methods for the Design of Metaheuristics*, pages 145–180. Springer Berlin Heidelberg, 2014.

[8] 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*, pages 373–380. ACM Press, 2014.

[9] Andreas Ostermeier, Andreas Gawelczyk, and Nikolaus Hansen. Step-size adaptation based on non-local use of selection information. In Yuval Davidor, Hans-Paul Schwefel, and Reinhard Männer, editors, *Parallel Problem Solving from Nature — PPSN III*, volume 866, pages 189–198. Springer Berlin Heidelberg, 1994.

[10] Nikolaus Hansen and Andreas Ostermeier. Completely Derandomized Self-Adaptation in Evolution Strategies. *Evolutionary Computation*, 9(2):159–195, 2001.

[11] Nikolaus Hansen and Stefan Kern. Evaluating the CMA Evolution Strategy on Multimodal Test Functions. In Xin Yao, Edmund K. Burke, José A. Lozano, Jim Smith, Juan Julián Merelo-Guervós, John A. Bullinaria, Jonathan E. Rowe, Peter Tiňo, Ata Kabán, and Hans-Paul Schwefel, editors, *Parallel Problem Solving from Nature - PPSN VIII*, volume 3242, pages 282–291. Springer Berlin Heidelberg, 2004.

[12] A. Auger and N. Hansen. A Restart CMA Evolution Strategy With Increasing Population Size. In *2005 IEEE Congress on Evolutionary Computation*, volume 2, pages 1769–1776. IEEE, 2005.

[13] Kouhei Nishida and Youhei Akimoto. PSA-CMA-ES: CMA-ES with population size adaptation. In *Proceedings of the Genetic and Evolutionary Computation Conference on - GECCO '18*, pages 865–872. ACM Press, 2018.

[14] Shigenobu Kobayashi. The frontiers of real-coded genetic algorithms. *Transactions of the Japanese Society for Artificial Intelligence*, 24(1):147–162, 2009.

[15] Xingwen Zhang, Jeff Clune, and Kenneth O. Stanley. On the Relationship Between the OpenAI Evolution Strategy and Stochastic Gradient Descent. `http://arxiv.org/abs/1712.06564`, 2018-11-18.