

次元削減によって得られたエリートを用いた進化計算の高速化

裴, 岩
九州大学大学院芸術工学府

高木, 英行
九州大学大学院芸術工学研究院

<https://hdl.handle.net/2324/1434420>

出版情報：進化計算研究会・進化計算フロンティア研究会合同研究会．第1回/第7回，pp.25-31，2011-09．進化計算学会
バージョン：
権利関係：

次元削減によって得られたエリートを用いた進化計算の高速化

Accelerating Evolutionary Computation with Elite Obtained by Dimensionality Reduction

裴岩^{1*} 高木英行²
Yan Pei¹ Hideyuki Takagi²

¹九州大学大学院芸術工学府

¹ Graduate School of Design, Kyushu University

²九州大学大学院芸術工学研究院

² Faculty of Design, Kyushu University

Abstract: We propose an elitism collaborative optimization strategy for accelerating evolutionary computation (EC) searches using elite obtained in reduced dimension space. The method projects individuals onto n one-dimensional spaces corresponding to each of the n searching parameter axes, approximates each landscape using Lagrange polynomial interpolation or a linear function approximation by a least square method, finds the best coordinate for the approximated shape, obtains elite by combining the best n found coordinates, and uses the elite to accelerate EC in the next generation. The advantage of this method is that the elite may be easily obtained thanks to their projection onto each one-dimensional space and there is a higher possibility that the elite will be located near the global optimum. We conduct the experimental tests to compare our proposed approaches with previous acceleration approaches by differential evolution and ten benchmark functions, the results show that the proposed method accelerates EC convergence significantly, especially in early generations.

1 はじめに

進化計算の研究には、大きく分けて2つの方向がある。第1は、従来法では困難なタスクに適用することで進化計算の有効性を応用タスクの拡大につなげ、進化計算の実用性の恩恵を拡大することである。第2は、進化計算アルゴリズムの性能を向上させ、あるいは、より高性能の新しい進化計算アルゴリズムを開発することである。本論文は後者の進化計算の高速化に関するものである。高速化は多くの進化計算タスクでのより良い解を求めるためにも、実行時間で解を求めるためにも必要である。例えば、対話型進化計算の場合、ユーザ疲労が必要上の大きな問題になるので、特に収束の高速化が求められており、これまでも多くの高速化手法が提案されてきた [5]。

高速化の1つのアプローチは探索空間情報を直接利用することである。山登り法、勾配法、ヒューリスティッ

ク法を進化計算と組み合わせる方法が色々提案されている [3]。他の方法は探索空間の景観を色々な方法で単純な形状で近似し [2]、最適解近傍であることが期待できる単純な景観上の最適解を容易に得、本来の最適解探索に利用することである。高次非線形や微分不可点を含む空間での最適解探索は一般に困難であるが、単純化された形状であれば容易になり、単純化された形状での最適解は本来の最適解の位置とは異なるものの、近傍であることが期待できるので、探索に利用することで高速化が期待できる。

探索空間の近似アプローチの1つの具体的方法は、探索空間を単峰性関数で近似し、この頂点をエリート個体として最悪個体と入れ替え方法である [6]。もしエリート個体の適応度が悪くても多くの個体のうちの1つが悪いだけ（しかも最悪個体よりは良い）であり探索への悪影響は少ない。反対に、適応度が優れている場合は強力な親個体として次世代探索効率を向上させる、つまり、Low Risk, High Return 方式である。課題としては、探索空間が高次元になるほど関数近似の計算コストが大きくなる問題になる点である。この問題

*連絡先：九州大学大学院芸術工学府
815-8540 福岡市南区塩原4丁目9番1号
Email: peiyan@ieee.org

解決案の1つが本論文提案の探索次元数削減である。

筆者らは回帰空間でエリート解を得る方法を提案した [4]。この方法は探索空間の次元を下げて近似した回帰空間を考え、個々の低次元回帰空間で得られたエリート位置を合成することで本来の探索空間上のエリート個体を生成して次世代探索に用いる方式である。ここでいう低次元回帰空間とは、本来の n 次元の k 軸からなる k 次元空間で本来の探索空間を近似したものである。なお、本論文では $k = 1$ で実験評価を行う。すなわち、全個体はエリートを探し易くするために各 k 次元空間に射影される。低次元化はエリート探索だけに用いられるのであり、解探索空間近似によって本来の進化計算探索が妨げられるわけではなく、 n 個の k 次元空間 ($k=1$ の時) でエリート位置探索をすることで高速化が可能になる。

本手法の提案論文 [4] に続く本論文の目的は、最初に提案された単峰性関数による最小二乗法による関数近似によって得られたエリートによる高速化手法 [6] と本提案方式の簡易エリート選択法による高速化との収束性能と計算コストを比較評価することである。

本節に続き、第 2 節では、探索空間の低次元サブ空間でエリート位置を求めてエリート合成法する提案手法を述べる。第 3 節と第 4 節では、10 ベンチマーク関数 [1][7] による評価実験を行い考察し、第 5 節で提案手法の性能についてまとめ、今後の展開について述べる。

2 低次元回帰空間からのエリート

2.1 次元数削減法

従来の補間法や近似法では多次元空間の近似関数を正確に求めるためには時間がかかる。その代替法が本来の多次元空間の次元数を下げ低次元空間で曲線近似をすることである。

筆者らの次元数削減方法は、 n パラメータ軸全部を使う代わりに、 n 軸のうちの 1 軸のみを使い、 n 次元空間上の M 個体を 1 次元回帰空間に射影する方法である。

2.2 エリート選択のための回帰空間景観の単純化法

適応度関数を $y = f()$ とすると、 n 次元空間探索空間の m 番目の個体の適応度は $y_m = f(x_{1m}, x_{2m}, \dots, x_{nm})$ ($m = 1, 2, \dots, M$) で与えられる。 n 個の 1 次元回帰空間間々には M 個の射影された個体があり、射影された M 個体と適応度 (x_1, y) $(x_2, y) \dots (x_M, y)$ を用いて各 1

次元回帰空間を補間あるいは近似を行い、 n 個の 1 次元近似景観形状からエリートを選択する (図 1 参照)。

本論文では図 2 の 2 種類の 1 次元回帰空間近似法を用いる。第 1 の方法は Lagrange 補間による 2 次元多項式補間であり、第 2 の方法は最小二乗法による線形近似法である。エリートはこれら近似形状から求める。

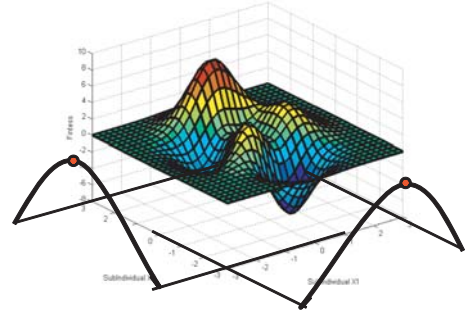


図 1: 本来の n 次元探索空間と 1 次元に次元数を削減した空間

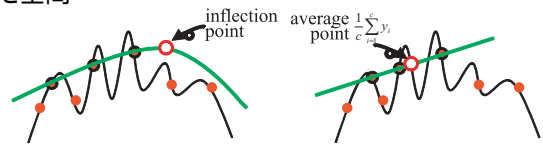


図 2: 回帰探索空間からのエリート選択法。(左) Lagrange 補間法 (右) 最小二乗法による直線近似法。

近似空間でのエリートを見つけるということは、本来の探索空間で比較的良さそうな個体の近傍の局所探索をすることに相当するので、大局的最適解がこの近傍にあると期待できれば、偶然よりも高い確率で探し出せるかもしれない [6]。したがって、提案する高速化手法では、エリート決定方法が重要なステップになる。異なるエリート選択法が異なるエリートを選び出すのであれば、高速化性能も異なると考えられる。さらに、探索空間の近似や補間で求めた回帰探索空間はそれぞれ異なる特性を持つので、その特性を解析した上で、その単純化した探索空間景観からエリートを探し出す効率的な方法を採用すべきである。

実際の補間多項式と直線の最小二乗近似は (1) 式と (2) 式で与えられる。

$$L(x) = \sum_{k=1}^3 \left\{ \prod_{i=1, i \neq k}^3 \frac{(x - x_i)}{(x_k - x_i)} \right\} y_k \quad (1)$$

$$\begin{pmatrix} (\varphi_0 \varphi_0) & (\varphi_0 \varphi_1) \\ (\varphi_1 \varphi_0) & (\varphi_1 \varphi_1) \end{pmatrix} \begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} y_0 \\ y_1 \end{pmatrix} \quad (2)$$

ここで、 x_i, \dots, x_k, y_i は、 i 番目の 1 次元回帰空間上に

射影された全個体と適応度 (x_{im}, y_m) 中の, 上位 k 個体と適応度である $(i = 1, 2, \dots, n), (m = 1, 2, \dots, M)$.

2次多項式による Lagrange 補間は非線形曲線で回帰曲線を近似するが, 変曲点を求めることは容易であり, この変曲点をエリート位置とする. 2番目の最小二乗近似では, 直線で回帰空間を近似するので勾配は正か負である. 両者を考慮する安全策を取って, 近似に用いる上位個体の平均適応度を求め, この近似直線上でこの平均適応度を取る変数 x をエリート位置とする (図2を参照).

両提案手法とも, 各世代の最悪個体をエリートで置き換える. 大局的最適解が最悪個体近傍にある可能性はないとは言えないけれど, 最悪個体が次世代の親として選択される可能性も同様に低いわけで, エリートと置き換えることは, リスクを最小にする妥当な方法であると考えられる.

2.3 エリート合成方法

前節の手法では n 個の 1次元回帰空間で n 個のエリート位置, $x_{1-elite}, x_{2-elite}, \dots,$ and $x_{n-elite}$ を得る. 次世代での進化計算高速化に用いる n 次元エリートは以下のように求める.

$$\text{新エリート} = (x_{1-elite}, x_{2-elite}, \dots, x_{n-elite}).$$

低次元空間でエリートを求めることは, 高次元空間で求めるよりも容易である. 本論文では 1次元の回帰空間を用いているが, 本提案手法は 1次元に限定するものではない.

得られた新エリートの扱いには 2つの方法がある. 1つは, このエリートの適応度を求めて本当に高速化に寄与しそうであるかどうかを判断してから高速化に利用する慎重な方法と, エリートを次世代探索にとにかく利用する直截簡明な方法である.

我々の提案手法は, 比較的適応度の高い個体を補間・近似して求めたエリートは同様に高い適応度を持つであろうとの仮説, n 個の 1次元回帰空間からの n 個のエリート位置を合成して得られた n 次元エリートの適応度も高かろうとの仮説, 大局的最適解の近傍に求めたエリートがある可能性が高いとの仮説, に基づいている. 一般的には, 本提案手法は進化計算高速化のための新しい局所探索に相当し, この点が本論文の貢献と言える.

3 比較評価実験

提案手法の性能を評価するため, ベンチマーク関数を用い, Lagrange 補間を用いる方法と直線の最小二乗法を用いる方法を, これらの高速化手法を組み込まない通常進化計算, および, 提案手法の原型 [6] と比較する. この高速化手法の原型は, n 次元の探索空間上の個体分布に二次形式の関数近似を行ってエリートを求める方法で, 文献と同様に, 適応度上位数個体を用いる場合, 最優良個体の近傍個体を用いる場合, 全個体を用いる場合, の 3つの場合に分けて関数近似を行う.

実験には差分進化 (DE/best/1/bin) を用いる. また, 各手法を表 1 の記号で区別する. これらの記号は図 4, 5, 6 でも用いる.

表 1: 実験に用いる手法を区別する記号の定義

DE-LR	(提案手法 1) Lagrange 補間によるエリート選択 + 差分進化
DE-LS	(提案手法 2) 最小二乗法の直線近似によるエリート選択 + 差分進化
DE-TB	(従来の高速化手法 1) 適応度上位個体を用いた n 次元二次形式関数近似によるエリート選択 + 差分進化
DE-TN	(従来の高速化手法 2) 最優良個体の近傍個体を用いた n 次元二次形式関数近似によるエリート選択 + 差分進化
DE-TA	(従来の高速化手法 3) 全個体を用いた n 次元二次形式関数近似によるエリート選択 + 差分進化
DE-N	通常の差分進化

3.1 実験条件

ベンチマーク関数として, De Jong の 5 関数 (F1 - F5), Rastrigin 関数 (F6), Schwefel 関数 2.26 (F7), Griewank 関数 (F8), Schaffer 第 1 関数 (F9), and Schaffer 第 2 関数 (F10) を用いる [1, 7]. 図 3 にこれらの景観を示す.

各ベンチマーク関数の次元数とパラメータの探索範囲は, 3次元 F1 [-5.12, 5.12], 2次元 F2 [-2.048, 2.048], 5次元 F3 [-5.12, 5.12], 30次元 F4 [-1.28, 1.28], 2次元 F5 [-65.536, 65.536], 5次元 F6 [-5.12, 5.12], 5次元 F7 [-512, 512], 5次元 F8 [-512, 512], 2次元 F9 [-100, 100], 2次元 F10 [-100, 100] で, すべて最小点探索問題としている.

これら関数の景観は, 連続 / 不連続, 凹 / 凸, 単峰 / 多峰, 低次元 / 高次元, 変数分離の可否など多彩である. これらの関数は大きく 4 つにグループ分けできる. F8 と F9 は多峰性で変数分離不可, F5, F10, F8 は

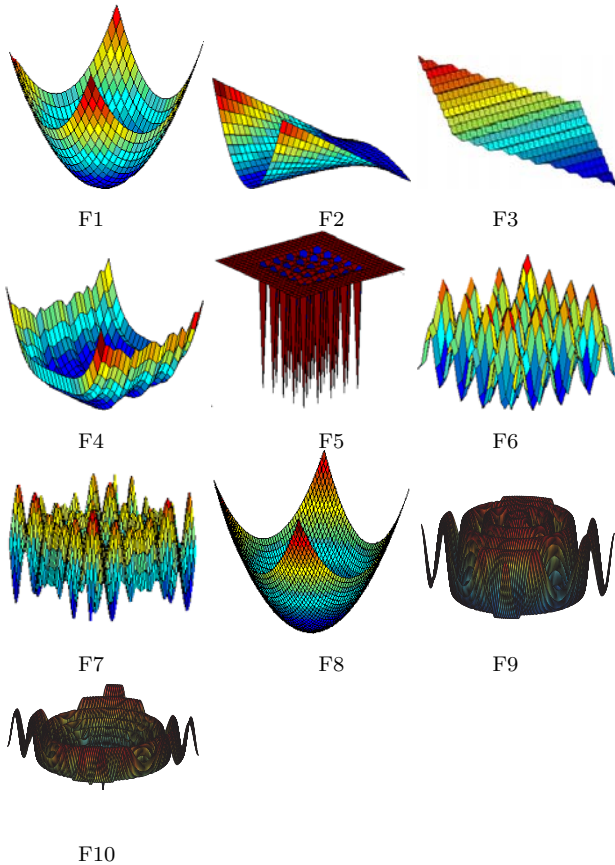


図 3: ベンチマーク関数の景観。(F1~F5) De Jong 第 1 関数~第 5 関数, (F6) Rastrigin 関数, (F7) Schwefel 関数, (F8) Griewank 関数, (F9) Schaffer 第 1 関数, (F10) Schaffer 第 2 関数。

多峰性で変数分離可能, F2 は単峰性で変数分離不可, F1, F3, F4, F6, F7 は単峰性で変数分離可能である。

差分進化の個体数を 30, F4 と F6 は 200 世代までを 50 試行, その他のベンチマーク関数は 50 世代まで 50 試行を行う。我々の提案手法とその他の手法の収束結果に対し符号検定で有意差検定を行う。

3.2 実験結果

図 4 は (DE-LR vs. DE-N), (DE-LR vs. DE-TB), (DE-LR vs. DE-TN), (DE-LR vs. DE-TA), (DE-LS vs. DE-N), (DE-LS vs. DE-TB), (DE-LS vs. DE-TN), (DE-LS vs. DE-TA) を 10 ベンチマーク関数に適用した時の, 最良適応度の 50 試行平均収束特性である。図 5 と図 6 は, 各世代毎の符号検定結果である。これらの結果から以下のことが言える。

1. 提案手法と通常の差分進化を比較すると, F3, F5,

F7 では両者の間に有意な収束の差は見られないが, 他の関数では提案手法が有意な収束の加速が確かめられた。

2. F3, F5, F7, F8 では, 提案手法と通常の差分進化との間に有意な収束の加速は見られない。
3. 提案手法と従来の加速法と比較すると, F8, F9, F10 の一部世代で, 従来の加速法には劣るものの, 全体的には両者に有意な違いは見られず, F1 と F2 に至っては, 全体的に提案手法の方が有意に加速されている。
4. 従来の加速法との比較では, Lagrange 補間よりも近似直線上の適応度重心点を用いるエリート選択手法の方が性能がよい。

4 考察

図 4, 6, 5 の実験結果から提案手法の性能を解析する。図 3 に各ベンチマーク関数の景観を示す。

4.1 提案手法と通常差分進化の比較

提案手法を組み込んだ差分進化は通常の差分進化より同等以上の収束高速化が得られた。有意に劣る場合はなかった。エリートを最悪個体と入れ替える Low Risk, High Return の技術的枠組みであることがうまく機能しているからであろう。

F3 のように探索空間の端に大局的最適解がある場合, 提案手法で得られるエリートは探索範囲外になることもある。この場合, 今回の実験ではエリートを採用しないため DE-LR は通常差分進化と同じになる。これが DE-N と DE-LR 間に有意な差がでなかった理由であろう。ただ DE-LS のエリートは近似に用いる個体の間に生成されるはずなので, 高速化効果がでなかった理由は別にあるのであろう。

F7 は傾き平面に sin 波形を重畳した関数であり F3 と類似した面がある。この関数の難しさは最小点(大局的最適解)近くに最大点があることである。局所最適解が大局的最適解近傍にある場合は, エリートが両者の間になりフィットネスの低いエリート, つまりエリートが働かない状態になる可能性がある。このことが, 提案手法を組み込んだ差分進化と通常の差分進化の性能に有意な差が見られない理由の 1 つかもしれない。

F4 はガウス雑音が重畳した 4 次関数で大局的最適解は探索空間の中心にある。多峰性でかつノイズの変動もあるが, これらの影響は比較的小さく全体的には F1

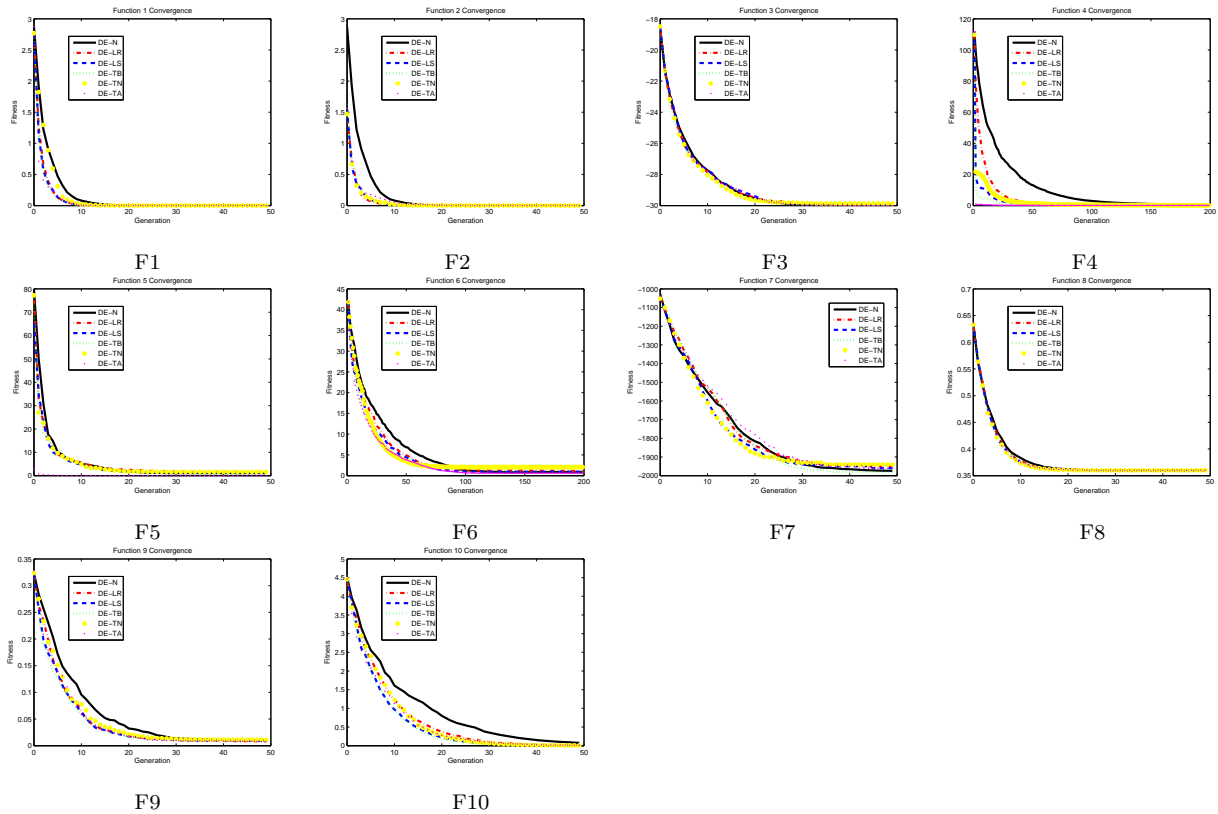


図 4: 50 試行の平均収束曲線 . F1 ~ F10 は図 4 を参照のこと .

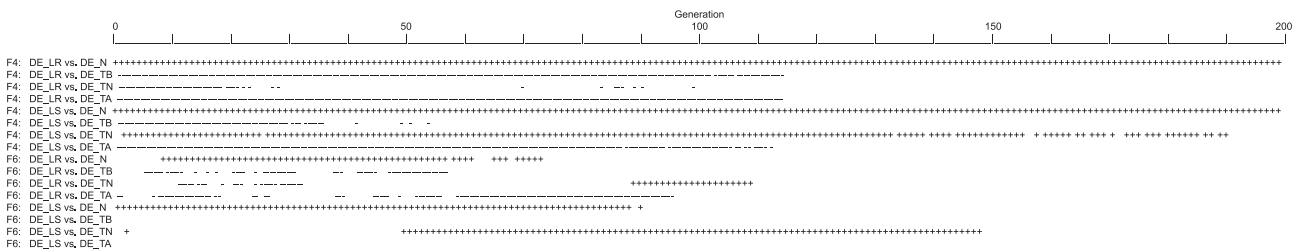


図 5: F4 と F6 の 200 世代までの提案手法と通常差分進化および従来の高速化手法の 50 試行の収束曲線の差に対して、各世代毎に行った符号検定結果 . 記号は表 1 を、ベンチマーク関数 F1 - F10 は図 4 を参照 .

のような 2 次形式である . 個々の個体位置は変動しているものの、これらからの近似関数から得られたエリートは中心に位置するため、F1 同様に提案手法が有効に働いた反面、この手法を組み込まない通常差分進化では F1 に比べての変動が大きいためこれらの変動のため F1 の場合よりも収束性能が劣化し、両者の差が大きくなったと言える .

4.2 提案手法と従来の高速化手法の比較

提案法 DE-(LR, LS) と従来の高速化手法 DE-(TB, TN, TA) の比較実験では、どちらかの一方的な優位性は見られないが、DE-TA の性能が比較的良く、今回の

ベンチマーク関数では、全個体を使って n 次元で関数近似することで正確なエリートが得られたと言える .

多峰性で変数分離型の関数の場合、DE-TB は提案手法の DE-LR と DE-LS よりも性能が良かった . タスクの複雑度が高くなり変数分離型でない場合、探索次元を削減する提案手法よりも正確にエリートを検出でき、その収束差が大きくなるかもしれない .

特殊な場合を除き、提案手法の DE-LR と DE-LS は DE-TN よりも収束性能がよい . 各世代のベスト個体が大局的最適解近傍にある保証がないので、ベスト個体近傍の個体だけで関数近似することが、有効なエリート個体を得ることに必ずしもつながらないと思われる .

4.3 計算コスト

本提案は比較実験で用いた従来の高速化手法 [6] と同じ技術的枠組みであるが、収束高速化のために 1 次元空間に個体を射影した後にエリート位置を求め本来の n 次元でのエリートを得ることで、計算時間増短縮を狙うことが本提案の最大の特徴である。代数演算をいくつか比較することでこの計算時間性能を比較する。第 3 節の実験と異なり、F4 は 3 次元関数として実験を行う。

$y = X\beta + \varepsilon$ の多変数回帰モデル (y, X, β, ε は各々、適応度、最適化パラメータ行列、係数ベクトル、誤差ベクトル) を使って我々の提案手法の計算コストを理論的に解析する。行列 X の行と列は探索空間の次元数と回帰モデルを得るために用いるデータ数で決まる。これらは 1 次元空間ではスカラーになる。最小二乗法で係数を求め、 m 個の (個体、適応度) サンプルデータとガウス消去法で連立方程式を解くと、代数演算数は、 n 次元空間では $O(2n^3 + 2(m-1)n^2)$ であるのに対し、1 次元空間での演算数を n 回繰り返す我々の方法では $O(n(2m+1))$ になる。この演算数の差は、我々の提案手法の計算コスト削減を歴然と示している。

DE-N, DE-LR, DE-LS, DE-TB, DE-TN, DE-TA の 50 試行平均を表 2 に示す。この表より、我々の提案手法は、理論通り従来の高速化手法よりも高速で、両提案手法の計算コストはほぼ同じであると言える。

表 2: 1 試行当たりの平均計算時間 (ms)。F は関数を、N, LR, LS, TB, TN, TA は各々 DE-N, DE-LR, DE-LS, DE-TB, DE-TN, DE-TA を意味する。

F	N	LR	LS	TB	TN	TA
F1	1.88	29.06	28.44	61.88	65.32	75.00
F2	1.56	30.00	28.76	72.80	65.64	68.12
F3	2.82	30.30	33.14	42.80	67.82	87.18
F4	30.32	60.30	60.94	110.94	174.06	204.70
F5	5.62	31.88	31.88	64.68	68.12	74.70
F6	4.68	31.88	31.88	75.94	68.74	85.64
F7	5.32	32.80	31.88	79.06	69.38	86.88
F8	5.94	34.38	32.80	80.00	71.88	83.12
F9	3.14	30.00	29.68	62.82	63.12	68.44
F10	2.82	30.00	30.30	73.44	70.00	71.88

5 まとめ

次元数を削減した探索空間で Lagrange 補間または最小二乗法での直線近似を用いて景観の近似を行い、近似関数から得られたエリート位置を合成して本来の探

索空間でのエリート探索点を得、これを探索に用いることで高速化させる手法を提案し評価した。特に探索空間が大谷構造をしている場合の収束の加速を実験的に確認した。また提案手法の性能と探索空間の景観との関係も解析した。

今回の研究から今後の 2 つの課題が明らかになった。第 1 は、どのようなモデルで線形・非線形の回帰空間を構成して本来に探索空間を近似するか、という点である。第 2 は計算コストの増加なしにこの次元削減法が本来の探索空間情報を維持したままどれだけ探索空間の次元が削減できるのかできるかの解析である。

謝辞

裴岩は吉田奨学会から博士研究のための奨学金を受け、本研究を遂行した。吉田奨学会に感謝する。

参考文献

- [1] De Jong, Kenneth A., “An Analysis of the Behavior of a Class of Genetic Adaptive System”, Ph.D Dissertation, University of Michigan, USA (1975).
- [2] Jin, Yaochu, “A Comprehensive Survey of Fitness Approximation in Evolutionary Computation”, *Soft Computing*, Springer, vol.9, no.1, pp.3–12 (2005).
- [3] Michalewicz, Zbigniew, “Heuristic Methods for Evolutionary Computation Techniques”, *J. of Heuristics*, vol. 1, no.2, pp.177–206 (1996).
- [4] Pei Y. and Takagi, H., “Accelerating Evolutionary Computation with Elite Obtained in Projected One-Dimensional Spaces”, 5th Int. Conf. on Genetic and Evolutionary Computing (ICGEC2011), Kimmen, Taiwan, pp.89–92 (Aug. 29–Spt. 1, 2011).
- [5] Takagi, Hideyuki, “Interactive Evolutionary Computation: Fusion of the Capabilities of EC Optimization and Human Evaluation”, *Proceedings of the IEEE*, vol. 89, no. 9, pp. 1275–1296 (2001).
- [6] 高木英行, 印具毅雄, 大西圭「単峰性関数当てはめによる GA 収束高速化」*知能と情報 (日本知能情報ファジィ学会誌)*, vol.15, no.2, pp.219–229 (2003) .
- [7] Whitley, D., Mathias, K., Rana, S., and Dzubera, J., “Building Better Test Functions”, 6th Int. Conf. on Genetic Algorithms (ICGA95), Pittsburgh, PA, USA, pp.239–246 (July, 1995).

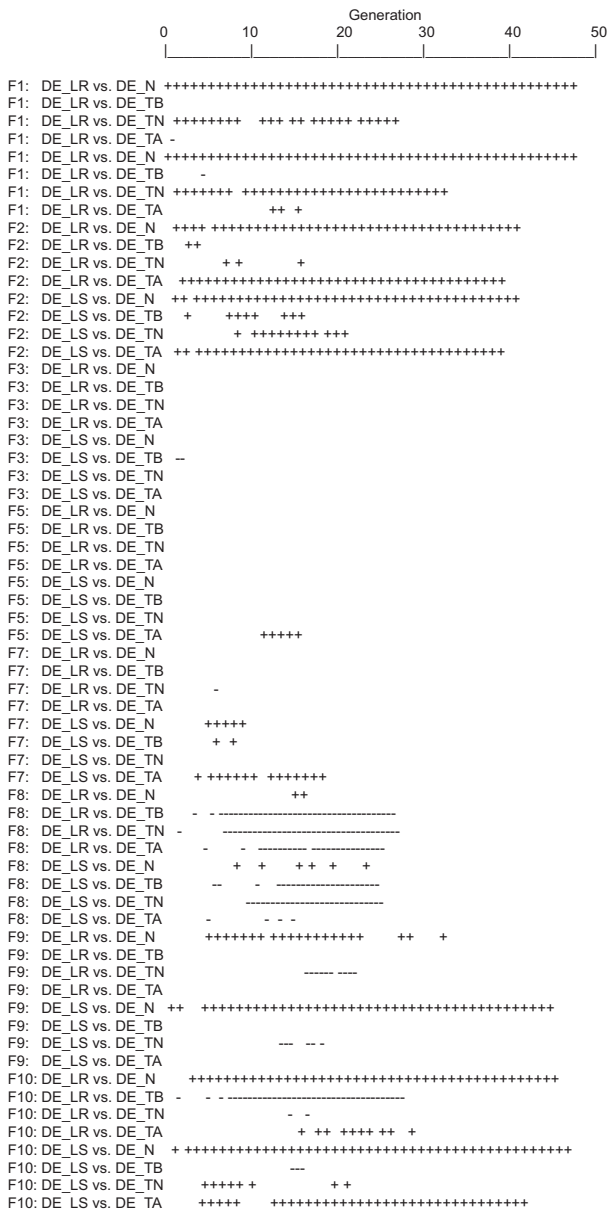


図 6: 提案手法と通常差分進化または従来的高速化手法との 50 試行の収束曲線の差に対して、各世代毎に行った符号検定結果。記号は表 1 を、ベンチマーク関数 F1 - F10 は図 4 を参照。+ 記号 (- 記号) は提案手法が比較対象に対して危険率 5% で有意に高速(劣る)ことを意味する。