JAIST Repository
https://dspace.jaist.ac.jp/

Title	GAを用いた粘性流れ場における翼型の多目的形状最適設計
Author(s)	高橋，真嘉
Citation	
Issue Date	1999-03
Type	Thesis or Dissertation
Text version	author
URL	http://hdl.handle.net/10119/1246
Rights	
Description	Supervisor:松澤 照男，情報科学研究科，修士

JAIST
JAPAN
ADVANCED INSTITUTE OF
SCIENCE AND TECHNOLOGY

Japan Advanced Institute of Science and Technology

修 士 論 文

GA を用いた粘性流れ場における翼型の多目的形状最適設計

指導教官　松澤照男 教授

北陸先端科学技術大学院大学
情報科学研究科情報システム学専攻

高橋真嘉

1999 年 2 月 15 日

Copyright © 1999 by Masayoshi Takahashi

要 旨

本稿では、最適化問題を解くための一手法である遺伝的アルゴリズム（GA) を、実数変数に対応させた実数領域適応型 GA を用いて翼型の空力最適化を構築する。

目 次

1 緒言 **1**

2 最適化手法 **3**

2.1 遺伝的アルゴリズム 3

2.1.1 コード化 5

2.1.2 評価 5

2.1.3 選択 5

2.1.4 交差 7

2.1.5 突然変異 7

2.2 実数領域適応型 GA 8

2.2.1 染色体の実数変換 8

2.2.2 側面制約 9

2.2.3 最良値の保証 10

2.2.4 特徴 11

2.3 多目的最適化問題への対応 12

2.3.1 多目的最適化問題と GA 12

2.3.2 パレート最適解 13

2.3.3 パレートランキング 14

2.3.4 選択 14

3 アルゴリズムの検証 **16**

3.1 単一目的の最適化 16

3.1.1 システムパラメータの影響 17

3.1.2 エリート保存 21

3.1.3 染色体のビット数 21

3.2 多目的の最適化 21

4 翼型の空力最適化 **25**

4.1 平面翼型の形状定義 25

4.2 流れ場の設定 26

4.3 単一目的の最適化 26

4.4 多目的の最適化 31

5 結言 **37**

第 1 章

緒言

翼型の設計は従来、過去のデータや設計者の経験等に基づいて行われていたが、近年の計算機の発達により CFD と組み合わせて数値的な最適化問題として取り扱うことが可能となった。

しかし最適化問題は設計支援の手法として注目され盛んに研究されているにも関わらず、普及するまでには至っていない。その理由として実際の最適化問題は、最適化の目的意外にも非常に多くの制約条件を考慮する必要があり、その中には重要であることがわかっていても、定式化できないものも多く、また定式化できても最適解を求めることが困難な場合も多い。つまり最適化の問題を数理的に処理するには次のような問題点がある。

- 設計問題を厳密に定式化することの困難性。
- 定式化された問題から最適な解を求めることの困難性。

前者は何を評価すれば良いか決めにくい場合などを指摘している。後者は定式化できた問題が有っても、その目的関数などが複雑な多峰性を有するものであれば、従来から知られている数理的手法で、その解を求めることは極めて難しいということを指摘している。このように問題の定式化を困難にする一つの要因として多目的関数の規定がある。しかし設計の目的が一つの性能を満たせば良いという事は少なく、翼型の設計においても空力 (揚力、抗力)、構造などの様々な要素が絡み合っている複合最適化問題となる。以上のような問題点を一部改善した手法として遺伝的アルゴリズム (GA) を多目的問題に拡張した多目的 GA がある。多目的 GA は GA の集合による探索という特徴からパレート最適解の集合を同時に求めることができ、設計者に各目的間のトレードオフ情報を明確に提供できる。

GA の問題点の一つとして個体の評価を繰り返し行うため、計算時間が大きくなってしまうことがあげられる。そのため、GA の各オペレータを工夫していかに効率良く最適解を探索するかが重要課題である。また GA は本来、離散的な探索空間を持つ組み合わせ最適化等に向いているが、そのロバスト性の高さから、連続な設計空間を持つ問題にも適用されている。しかし GA による実数の扱いには工夫が必要であり、その検討は十分になされていない。

そこで本研究では連続な設計空間を持つ翼型の多目的最適化問題に GA を適用したアルゴリズムを構築する。

第 2 章

最適化手法

本章では、本研究で用いた最適化手法である遺伝的アルゴリズムについて説明する。

2.1 遺伝的アルゴリズム

メンデルの法則とダーウィンの理論で知られているように、あらゆる生物は生殖と淘汰および突然変異によって環境に適応しつつ進化する。その生物進化のメカニズムは、生物を構成している細胞中の核が、染色体の形でそれら生物固有の遺伝プログラムを有し、これが生殖、淘汰および突然変異によって種々変更され、次世代に引き継がれて行くからと言われている。遺伝的アルゴリズム (GA) は、このような生物進化の過程を、人工的最適化問題の解法へと応用した計算機用アルゴリズムである。具体的には「集団」と呼ばれる解の集団をつくり、これを構成する「個体」と呼ばれる解候補群が「選択」、「交差」、「突然変異」と言う過程を繰り返しながら、最適な解へと収束していくものである (図 2.1)。そしてこの手法は、これまで知られている数理的な最適化手法と以下の点で異なる特徴を持っている。

- 解の探索には、変数を２進数などにコード化した「染色体」と呼ばれる遺伝子を用いる。
- 設計空間内の１点からの探索ではなく、多数の点からの同時探索を行う。
- 解の評価には目的関数値のみを用い、その微分値は用いない。
- 確定的ではなく、確率的な方法である。

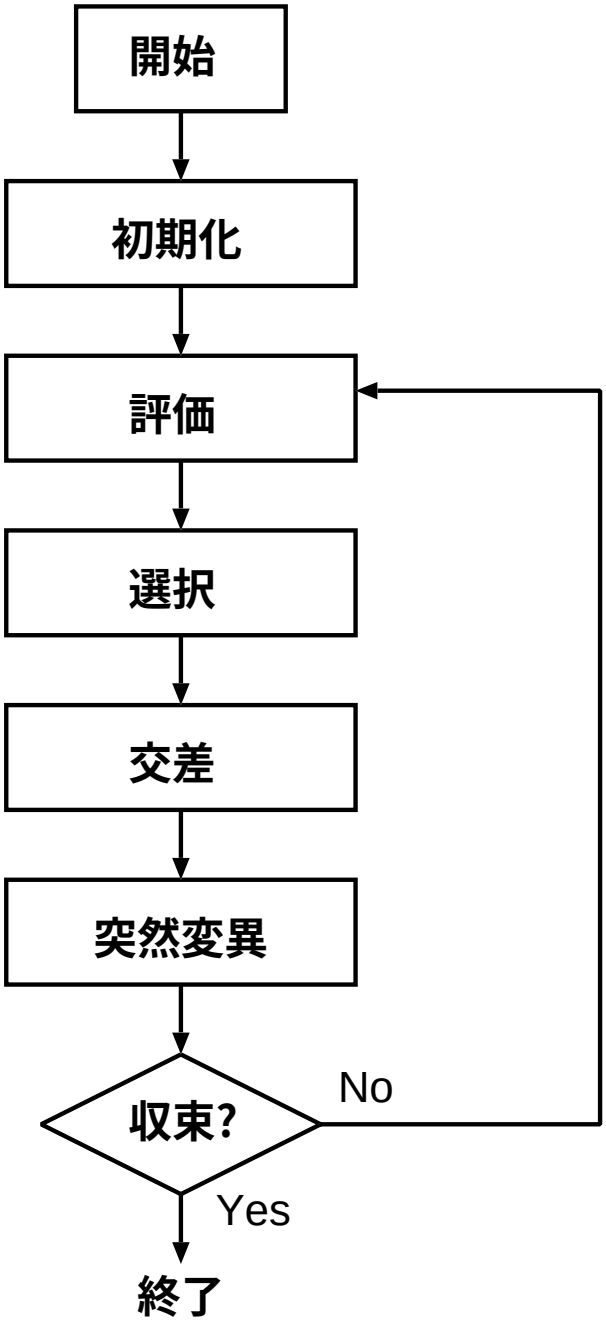

図 2.1: GA の処理手順

以上の特徴から明らかなように、GA は与えられた問題が数理的に明確に記述できないものであっても解くことができ、また多数の点での同時探索であることから、多峰性の強い問題であっても、その大局的最適解か、あるいはそれに近い候補解をいくつか見出すことができる。そして、この手法を用いる場合、まず与えられた問題をその変数のコーディングを中心とした GA 問題へ変換すること重要である。次に GA を用いる際の基本的な動作について説明する。

2.1.1 コード化

GA を用いて最適化問題を解くとき、初めに設計変数を染色体の形で表現しなければならない。この実際の設計変数を染色体型に変換することをコード化と言う。どのようにコード化するかは問題によって異なる。というのは、交差の部分では実際の個体ではなく、コード化された染色体を入れ替えることになるので、コード化が不適切だと解が収束しない、もしくは収束するまでに莫大な時間がかかると言う危険性が生じるからである。従って、設計が成功するかどうかは設計者がどのようなコード化手法を選ぶかと言うことに依存する。通常、染色体を表現するときはビット列を用いるが、設計変数に実数を用いたい場合精度が粗いなどの問題が起こる。本研究においても設計変数に実数値を用いるためコード化には実数領域適応型 GA を用いる。実数領域適応型 GA については次節において説明する。

2.1.2 評価

与えられた目的関数と制約条件のもとで、集団中の各個体の優劣（適応度）を決める。この評価の高い個体ほど、優れた個体として自分の遺伝子を次世代に残すことができる可能性が高くなる。本研究においては翼型の形状最適化と言うことで、流れ場を評価する必要がある。そのため CFD から計算される空力性能を評価値に用いる。よってアルゴリズムが終了する全計算のうち、CFD の計算が大半を占める事になる。

2.1.3 選択

集団中から次世代に残す個体を選び出す過程である。この選び出される確立の計算には適応度 F_iが用いられる。選択には種々の方法が知られているが、最も基本的で簡単な方法はルーレット選択である。まずある個体の適応度 F_iが全体の中で相対的にどのくらい

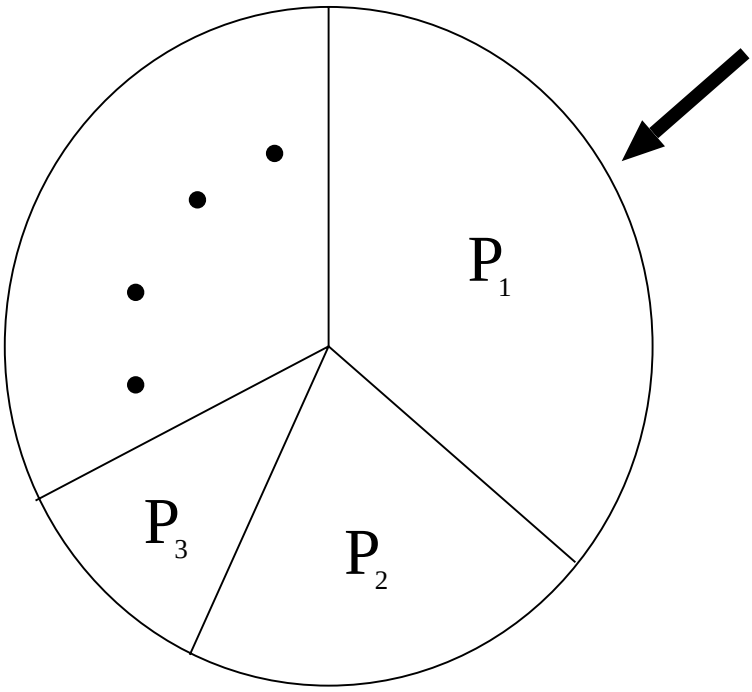

図 2.2: 選択確率の設定

の割合 P_iを占めているかを式 (2.1) によって計算する。

$$P_i = \frac{F_i}{\sum F} \tag{2.1}$$

その割合に応じたルーレットホイールを作る (図 2.2)。次に矢に見立てた乱数を発生させ、矢の当たった個体を次世代へ生き残らせるものである。よって、この方法ではルーレット上の占有面積の広いものほど次世代に生き残れることになる。このとき淘汰圧を適度に保つために、目的関数値をそのまま適応度に用いてルーレット選択をするのではなく、ランキング選択やトーナメント選択を行うことが望ましい。本研究ではランキング選択を用いた。ランキング選択では個体集団中における各個体の評価値の大きな個体から順に１、２、３、$\cdots$ とランク付けをし、このランクに応じて各個体の選択確率を決める。つまり、ランク i の個体の適応度を以下の式で再定義する。

$$f_i = c \cdot (1 - c)^{i-1} \tag{2.2}$$

そして式 (2.2) によって与えられる適応度 f_iを用いて式 (2.1) より選択確率を決定し、ルーレット選択を用いて次世代に残る個体を選択する。ここで c はランク１の適応度を表している。この値は大き過ぎると、ランク１の個体が選択される確率が高くなり過ぎるため、

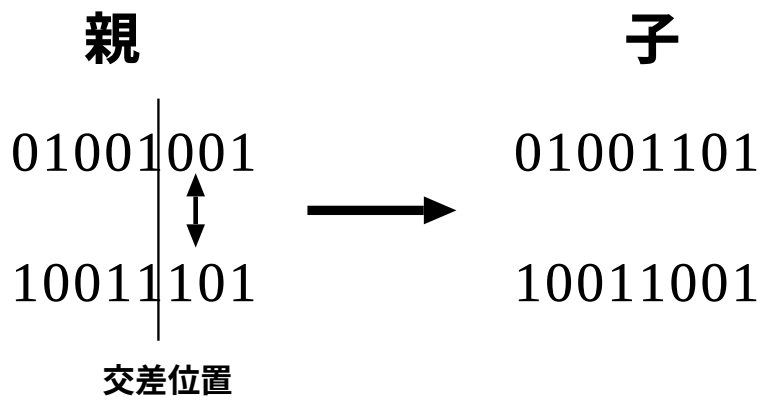

図 2.3: 交差

その影響で全体の個体が同じものばかりになってしまい、逆に小さ過ぎるとランク１の個体の選択確率も小さくなるため、いつまでたっても良い個体が増えない。このように、ランキングによる選択は直接目的関数の評価値によって決まるのではなく、設計者の意図に基づいて決定されるので選択確率を自由に設定することができる。ここでは $c = 0.1$ と設定した。

2.1.4 交差

乱数により選択された個体間（親）で遺伝子の入れ換えを行い、新たな個体を形成し、この個体を子孫として次の世代に残す (図 2.3)。ここで、入れ換える遺伝子の数が多すぎたり少なすぎたりすると、優れた子孫を残すことができなくなり、最適値の探索能力が衰えるので、どの程度遺伝子を入れ換えたら良いか適当な確率を決める必要がある。この交差する確率を交差率と言う。

2.1.5 突然変異

交差とは異なり、図 2.4のように個体中の遺伝子を強制的に操作することによって、交差だけでは得られないような解を探索するために用いられる。突然変異が起こる確率を突然変異率と言う。突然変異率が高すぎると、ランダムサーチのように解空間の中をランダムに探索するアルゴリズム程の収束性しか示さなくなる。このため、突然変異率は一般的に小さく設定されている。

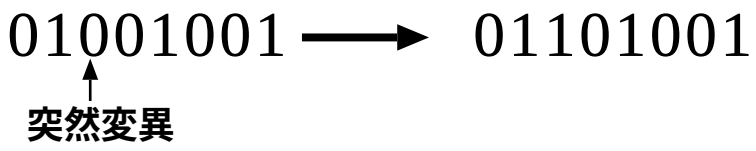

図 2.4: 突然変異

2.2 実数領域適応型 GA

翼の性能は少しの形状変化でも大きく変ってしまう。そのため設計変数には実数をそのまま用いることが望ましい。そこで本研究では荒川らにより提唱された実数領域適応型遺伝的アルゴリズム [4] を取り入れる。これは遺伝的アルゴリズムを実数表現に対応させた手法である。この手法では、染色体に実数表現を用いず、バイナリ表現を基調としている。例えば多次元ベクトルとしての変数が与えられた場合、各変数ごとに適当な数のビットを割り当て一つの染色体とする。

2.2.1 染色体の実数変換

各世代の選択、交差、突然変異の後に得られたその世代 i での各設計変数の平均値μ_i と標準偏差値σ_iを基にして、式 (2.3) により与えられる分布 (図 2.5) の縦軸を LBと UBの間で等分割することにより m ビットのバイナリ表現の染色体から実数値への変換を行う。

$$N(x) = exp\left(\frac{-(x-\mu_i)^2}{2\sigma_i^2}\right) \tag{2.3}$$

よって実数値は以下のようになる。

$$R(p_i) = \begin{cases} \mu_i - \sqrt{-2{\sigma_i}^2 \ln\left(LB + \frac{(UB-LB)C(p_i)}{2^{m-1}-1}\right)}, \\ \quad \text{for } C(p_i) < 2^{m-1} \\ \mu_i + \sqrt{-2{\sigma_i}^2 \ln\left(UB - \frac{(UB-LB)(C(p_i)-2^{m-1})}{2^{m-1}-1}\right)}, \\ \quad \text{for } C(p_i) \geq 2^{m-1} \end{cases} \tag{2.4}$$

ここでUB, LBはシステムパラメータである。σ_iは設計変数の領域の幅を決定するもので上下限値 $(\sigma_{max}, \sigma_{min})$ をシステムパラメータとして設定する。σ_{max}は計算の初期段階において個体のばらつきが大きいときに探索領域が不必要に拡大してしまうのを防ぐ目的があり、σ_{min}はある程度収束したときに、探索領域が狭くなりすぎ、有効な探索領域を確保

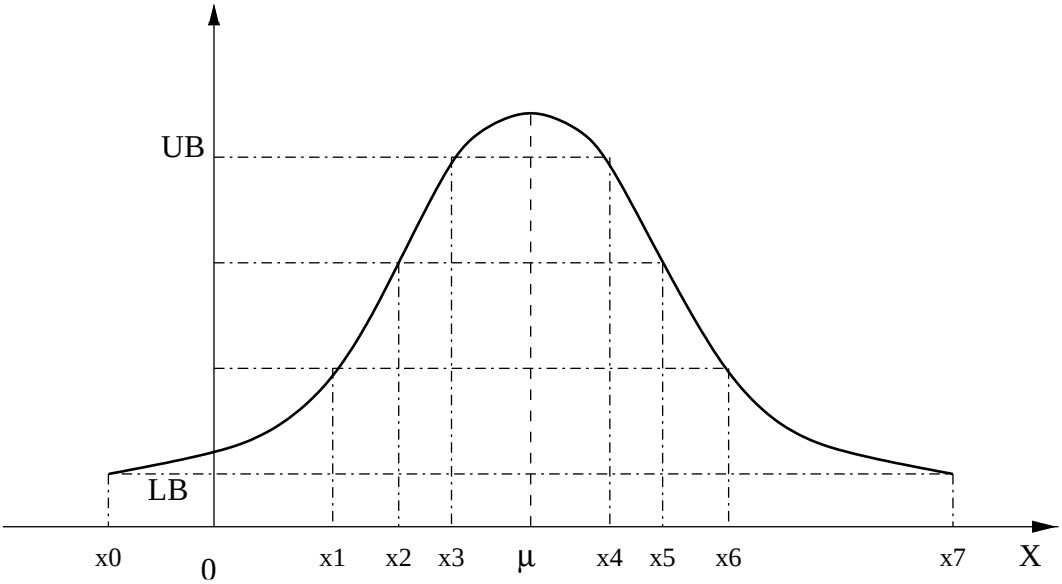

図 2.5: 実数への変換 $(m = 3)$

できなくなるのを防ぐ役割を持っている。また $C(p_i)$ はバイナリ表現 p_i を整数化したものである。例えば $m = 3$ の場合、ある染色体を整数化したとき 5 であったとすると、得られる実数値は図 2.5の x_5 になる。また染色体はビット列とそれを実数値に変換した表現型の 2 つの情報を持つことになる。

2.2.2 側面制約

従来までの GA では最適化の際に側面制約 (上限 $upper_i$、下限 $lower_i$) を意識する必要はなかった。しかしこの手法では探索領域が世代によって変化するため探索領域が設計変数の定義域を越える可能性がある。そのため側面制約を守るように LB, σ_i を設計変数が平均値 μ より小さい領域と大きい領域とに別けて以下のように操作する。ここで LB と σ の左側を LB^L, σ^L、右側を LB^R, σ^R とすると下限は

$$\begin{aligned} LB^L_{i,new} &= \exp\left(-\frac{(lower_i-\mu_i)^2}{2\sigma_i^2}\right) \\ &\quad \text{if } LB^L_{i,new} > UB - margin \\ &\quad \text{then } LB^L_{i,new} = UB - margin \text{ and} \\ \sigma^L_{i,new} &= \frac{lower_i-\mu_i}{\sqrt{-2\ln(LB^L_{i,new})}} \end{aligned} \tag{2.5}$$

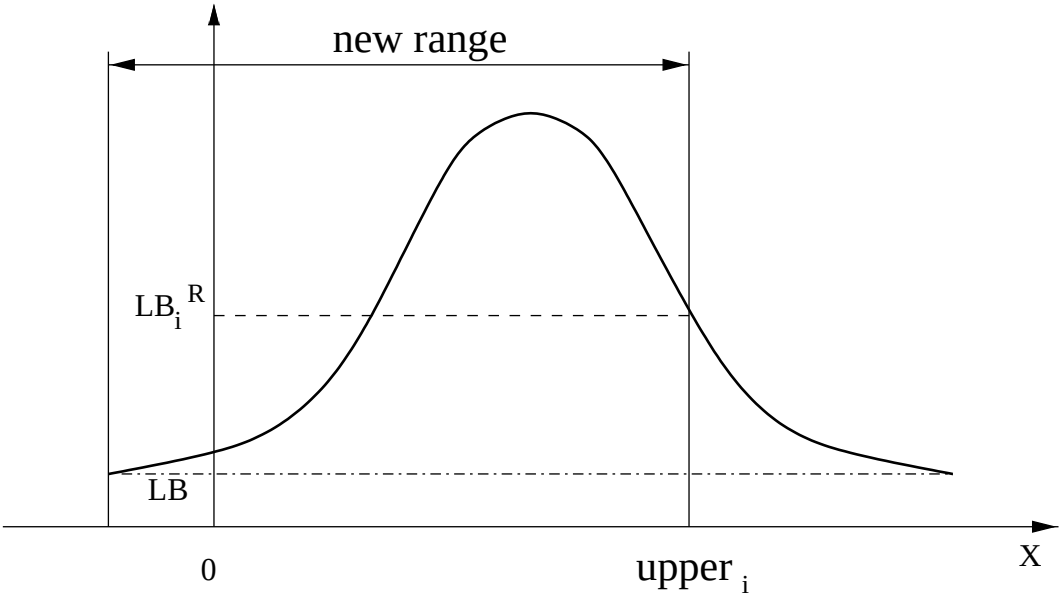

図 2.6: 側面制約

となり、上限は

$$
\begin{aligned}
&LB^R_{i,new} = \exp\left(-\frac{(upper_i-\mu_i)^2}{2\sigma_i^2}\right) \\
&\quad \text{if } LB^R_{i,new} > UB - margin \\
&\quad \text{then } LB^R_{i,new} = UB - margin \text{ and} \\
&\sigma^R_{i,new} = \frac{upper_i-\mu_i}{\sqrt{-2\ln(LB^R_{i,new})}}
\end{aligned}
\tag{2.6}
$$

となる。ここで $margin$ とは $[LB, UB]$ 間の任意の値であり、LBだけの操作では$LB^L_{i,new}, LB^R_{i,new}$ が UBを越えてしまう可能性があるので、それを回避するために設定する。また $LB_{i,new}$ が $margin$ の値を超えたときは、σ_iの値も同時に操作することで前世代の平均値μ_i と上限の側面制約 $upper_i$あるいは下限の側面制約 $lower_i$との間で分割ができるように設定の変更を行っている。図 2.6に示すようにこの操作によって領域が制約条件を守った new range に変更される。

2.2.3 最良値の保証

この手法では世代が進み探索領域が変化していく過程で、領域内での個体の位置に偏りが出てくる。そのため最適値近傍の探索領域に入っても、1 世代の GA 操作では、その偏りを解消できずに通り過ぎてしまう (オーバーシュート) 可能性がある。これを繰り返す

うちに、最適値が探索領域の外に出てしまい、収束性の向上が望めなくなる。その対策として全世代を通して最も良い個体が探索領域からはずれないようにパラメータσ_iを以下のように左右別に定義し直す (図 2.7)。

$$\begin{aligned} &\text{for } Maxvalue_i > \mu_i, \text{if } Maxvalue_i > \mu_i + \sqrt{-2\sigma_i ln(LB)} \text{ then} \\ &\sigma^R_{i,new} = \frac{Maxvalue_i - \mu_i}{\sqrt{-2\ln(LB)}} \\ &\text{for } Maxvalue_i < \mu_i, \text{if } Maxvalue_i < \mu_i - \sqrt{-2\sigma_i ln(LB)} \text{ then} \\ &\sigma^L_{i,new} = \frac{-Maxvalue_i + \mu_i}{\sqrt{-2\ln(LB)}} \end{aligned} \tag{2.7}$$

ここで $Maxvalue_i$とは、初期世代から前世代までの間で最も適応度の値が良かった個体のその変数であり、$\sigma^R_{i,new}, \sigma^L_{i,new}$はそれぞれ平均値の右側、左側に新しく与えられたパラメータである。

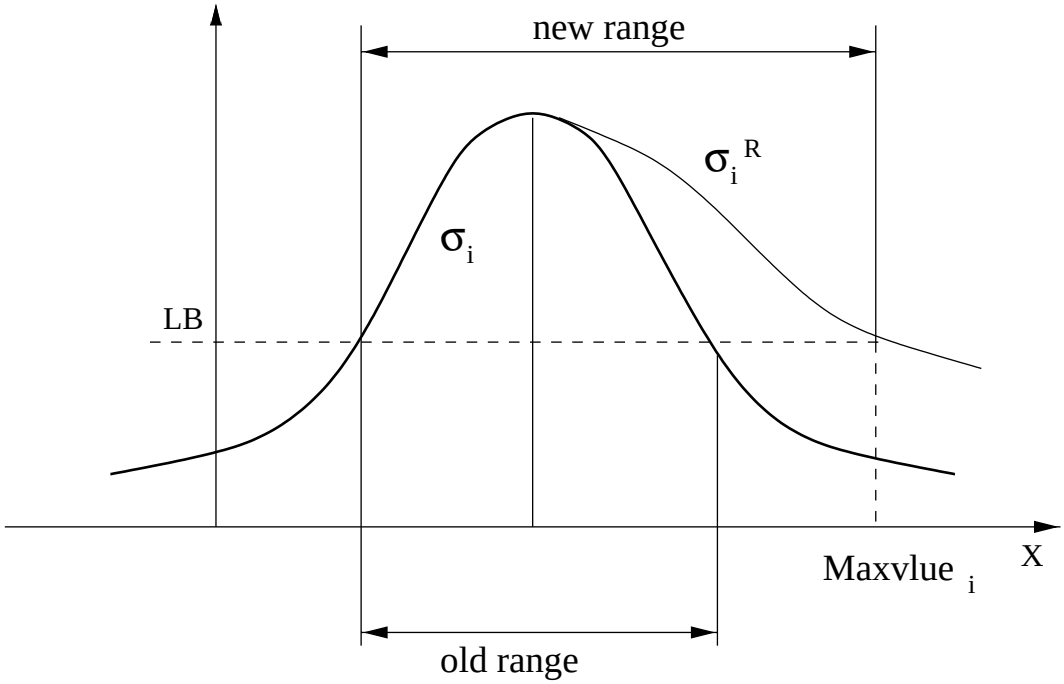

図 2.7: 最良値の保証

2.2.4 特徴

以上の実数領域適応型 GA は以下の特徴を持つ。

- 実数探索領域が前世代の平均値、標準偏差により変動する。そのため染色体が、ある 1 つの値に対応するのではなく、同じ染色体でも世代によって値が異なる。染色体は探索領域の移動の方向を決定する。
- 世代が進むに連れて探索領域が狭まっていき、探索の精度を向上させていく。
- ある程度少ないビット数でも実数の変数を持つ関数の最良値またはその近傍を求めることができる。

2.3 多目的最適化問題への対応

2.3.1 多目的最適化問題と GA

一般に、設計において実現したい性能や機能が一つであると言うことは稀である。そのため単一の目的関数による最適化問題でカバーできないような多目的最適化問題の重要性が高まってきている。

従来の線形計画法や非線形計画法は広い意味で効率の良さを求めるための数学的手段であるが、ほとんどの場合、解空間内で一つの目的関数を最適化することが論じられている。多目的最適化問題に対しても、二つ以上の目的関数に重みづけをして和をとり、一つの目的関数にするなどして、従来の数理的手法を応用する方法が用いられている。しかし、多目的最適化の本質は複数の目的関数間でのトレードオフをいかにバランスさせるかという点にあり、従来の方法論では不十分である。

一方、多目的最適化の理論について見ると、目的関数間でのトレードオフをバランスさせ得る解に関して、「パレート最適性」が重要な概念である。このパレート最適性を満足する解は一般に複数個あり、これを集合として求めることが、効率的かつ適切に設計における意志決定を行なう上で重要となる。多目的最適化問題を取り扱う立場から言えば、何らかの過程を経て解を一つに絞ることが必要であると言える。これは意思決定者の意志を定量化することである。しかし意思決定は最終的には人間の選考によるものであり、これを定量化することは非常に難しいと言える。そのため選考に関する情報をなるべく引き出し、意思決定過程に合理性を与えることが多目的最適化手法の役割である。

多目的 GA はその集合による探索という特徴からパレート最適解の集合を直接的に求めることができる。そのため従来法に比べて意思決定のための情報をより合理的に得ることができる。多目的 GA は単一目的 GA とくらべて選択の方法が異なる。

2.3.2 パレート最適解

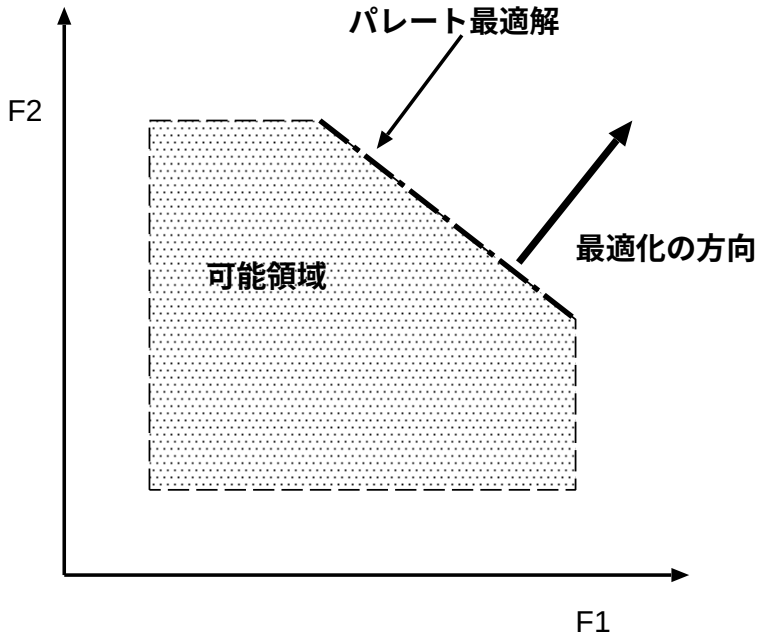

図 2.8: パレート最適解

[定義]:集団中に $X_i = (x_i, y_i), X_j = (x_j, y_j)$ が存在し、また最大化すべき評価関数の組を$\hat{F} = (F_1, F_2)$ としたとき、以下の式を満たすとき、X_iは X_jに支配されている (X_iの劣解) という。

$$\begin{gathered} F_1(X_i) \leq F_1(X_j) \quad \text{かつ} \quad F_2(X_i) \leq F_2(X_j), \\ \text{ただし } F_1(X_i) \neq F_1(X_j) \end{gathered} \tag{2.8}$$

上式を満たす X_iが存在しない、すなわち

$$F_1(X_i) > F_1(X_j) \quad \text{または} \quad F_2(X_i) > F_2(X_j) \tag{2.9}$$

のとき、X_iは X_jに支配されない (X_jの非劣解) という。つまり、評価関数空間の実行可能領域内にある非劣解の集合がパレート最適となる。図 2.8は 2 目的で両目的関数を最大化したい場合である。目的関数間の可能領域の境界線上に存在する、どちらか一方の値を大きくするためにはもう一方を小さくしなければならない部分 (図中太線) がパレート最適解である。

2.3.3 パレートランキング

個体 X_iがある世代において、n_i個の個体で支配されているとき、個体 X_iのランクは次式で決定する。

$$rank(X_i) = 1 + n_i \tag{2.10}$$

非劣解の個体はすべてランク 1 である。例えば図 2.9で示されるような各個体のランクが決まる。

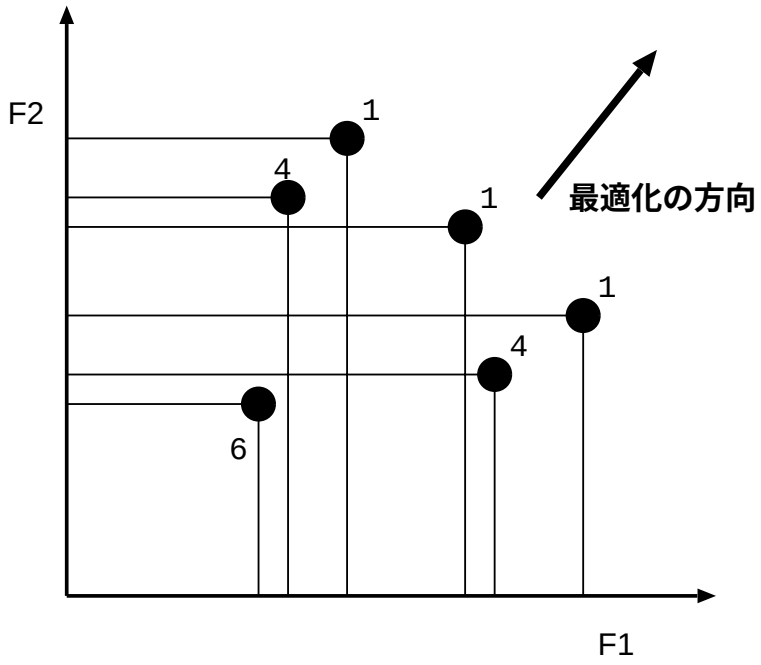

図 2.9: パレートランキング

2.3.4 選択

個体群を目的関数の種類に等しい部分個体群に分割し、各目的関数値に応じて独立に個体を選択し、それぞれの部分個体群を生成する。交差および突然変異は、生成された部分個体群をすべて合わせて一つの個体群としたものに対して適用した。またある世代 t で得られた非劣解はすべての世代を通して得られた非劣解と比較し、パレート最適な個体は別の個体群に保存した (図 2.10)。

選択に用いられる適応度は探索領域がパレート最適解から離れてしまうのを防ぐ目的で、各目的関数値のランキングより求められる適応度にパレートランキングより求められ

る適応度を掛け合わせ以下のように設定した。

$$
\begin{aligned}
&\text{評価値ランキングより求められる適応度}: F^e \\
&\text{パレートランキングより求められる適応度}: F^p \\
&\text{選択に用いられる適応度}: F^e \cdot F^p
\end{aligned}
\tag{2.11}
$$

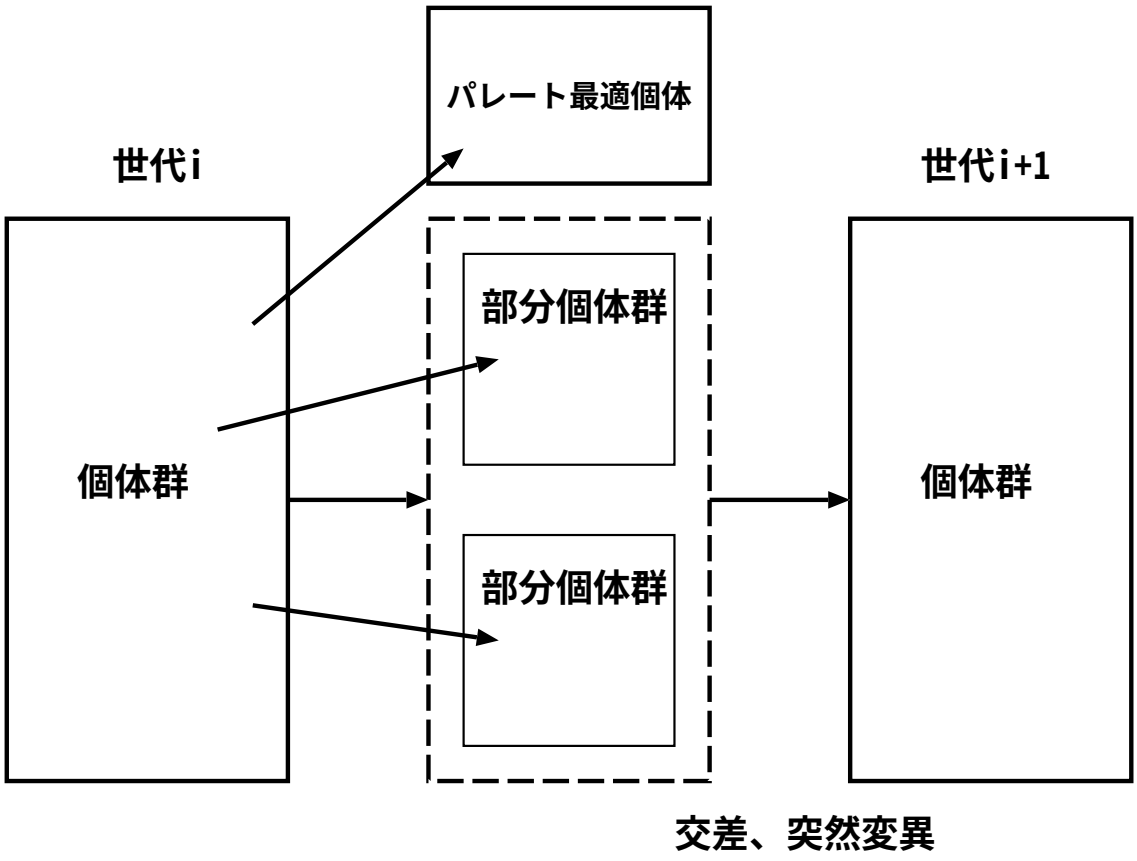

図 2.10: 選択の方法

第 3 章

アルゴリズムの検証

3.1 単一目的の最適化

翼型の設計に適用する前に簡単な数学モデルに実数領域適応型 GA を適用した。適用する問題は以下のように設定した。

$$f = 10\exp(-0.01(x_1 - 10)^2 - 0.01(x_2 - 15)^2)\sin(x_1) + 10 \tag{3.1}$$

$$f \rightarrow \text{最大化} \tag{3.2}$$

関数 fをプロットしたものを図 3.1に示す。fは多峰性の関数であることが分かり、これ

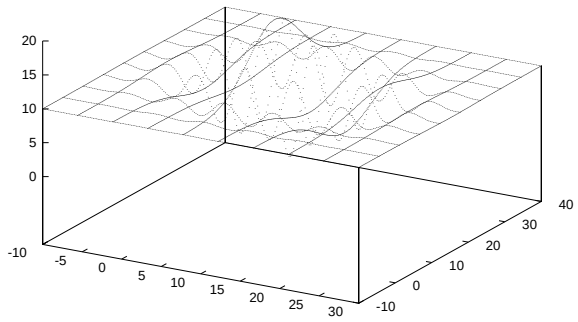

図 3.1: 適用した関数

は一般的に数理的な最適化手法によって最適解を探索するのが難しいとされる問題である。fの最大値は約 19.55853 である。

3.1.1 システムパラメータの影響

実数領域適応型 GA ではシステムパラメータの設定により探索の収束具合が変化する。システムパラメータを適切に設定することで探索の効率と精度を上げることができる。そのため各パラメータの設定について検討する必要がある。

UBはなるべく 1 に近いのが理想である. σ_{max}は探索領域の幅を決定するもので、不必要に探索領域が拡大しないように、設計領域の広さに応じて決める必要がある。特に探索の収束に影響があるパラメータはLB, σ_{min}である。実数領域適応型 GA では世代が進むに連れて探索領域が狭まっていき、探索の精度を向上させていく事が一つの特徴であった。その理由は個体が収束するに連れて設計変数の標準偏差が小さくなっていき、探索領域が狭まるためである。その様子を図 3.2に示す。適応度の最大値が大きくなるにつれて標準偏差が小さくなっていくのがわかる。標準偏差は最終的には 0 になる事が期待されるわけだが、そうなると探索がそこで止まってしまうため標準偏差の最小値σ_{min}を設定しておく必要がある。

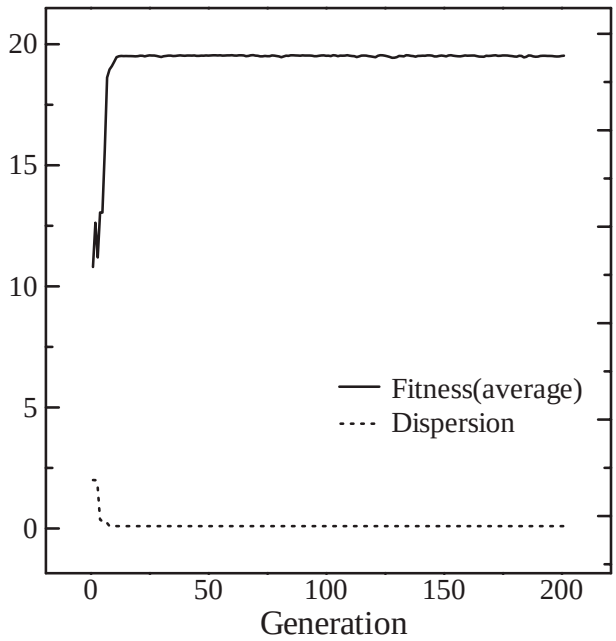

図 3.2: 収束と標準偏差の変化

表 3.1: LBの影響

LB	0.1	0.2	0.3
最大値の平均	19.555670	19.556674	19.557091

そこで $UB = 0.95, \sigma_{max} = 2.0$ と固定し、LB, σ_{min}の設定を変えて収束に与える影響を調べた。また GA の各パラメータを以下のように設定した。

- 個体数 50
- 最大世代数 200
- 交差率 0.3
- 突然変異率 0.02
- 初期集団 $2 < x_1, x_2 < 10$ においてランダムに生成

図 3.3に LBを 0.1〜0.9 まで 0.2 刻みで 変えて、その時の適応度の最大値の変化を示す。また、同様の計算を 1000 回繰り返し、得られた最大値の平均を表 3.1 に示す。これはσ_{min}を 0.1 としたときの結果である。図 3.3によると LBが大きくなるに連れて収束が遅くなることがわかる。また表 3.1によると LBが大きくなるに連れて得られる最大値が最適値に近い値である可能性が大きくなることがわかる。これは LBが 0.3 より大きくなっても同様の傾向になる。LBが大きくなると、1 世代で探索できる領域が狭くなるため、領域の移動が小さくなる。そのため目的関数の勾配に沿って領域の移動が行われ、オーバーシュートが起こりにくくなり、探索の確実性が上がるためだと考えられる。しかし収束は遅くなり、極所解に陥りやすくなると考えられる。以上より非常に多峰性が強い問題においては多少 LBを大きくすることは有効な手段となる。結論として LBは 0.1〜0.3 の範囲で選ぶと良いと考えられる。

図 3.4にσ_{min}を 0.1,0.2 としたときの適応度の平均の変化を示す。また、同様の計算を 1000 回繰り返し、得られた最大値の平均を表 3.2 に示す。これは LBを 0.1 としたときの結果である。表 3.2によるとσ_{min}が 0.15 の場合が最も収束の確実性が高いと言える。しかしσ_{min}が 0.15 から離れると最大値の平均が下がってしまう。これはσ_{min}が大きいと交差、突然変異による個体の変化が大雑把に行われるためで、σ_{min}が小さい場合は逆に交差、突然変異による個体の変化が小さくなるからと考えられる。図 3.4を見るとσ_{min}が 0.2 のと

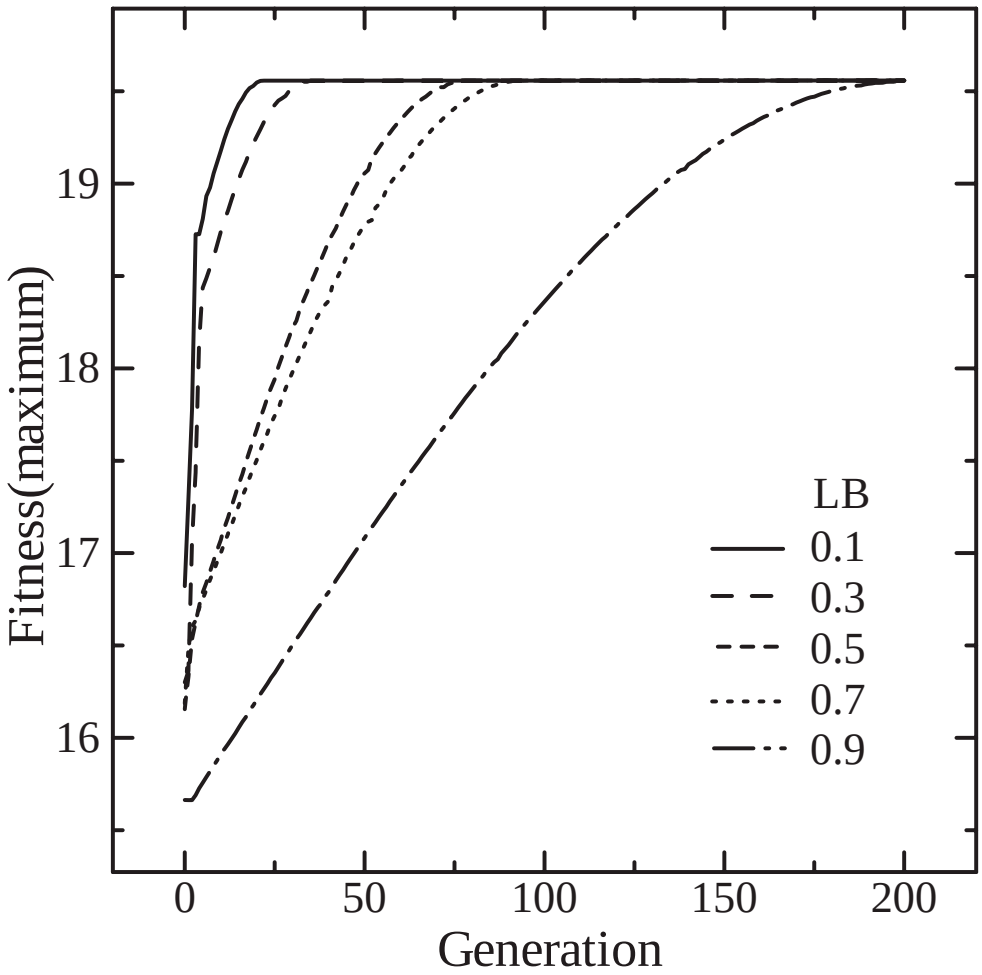
Fitness(maximum)
19
18
17
16
0
50
100
150
200
Generation
LB
0.1
0.3
0.5
0.7
0.9

図 3.3: LBの影響

表 3.2: σ_{min}の影響

σ_{min}	0.05	0.10	0.15	0.20	0.25
最大値の平均	19.554379	19.555670	19.556012	19.555423	19.555390

きは集団が最適値付近に集中してから、適応度の平均の変動が大きく、0.1 のときは小さい。以上よりσ_{min}は 0.15 程度が良いと考えられる。

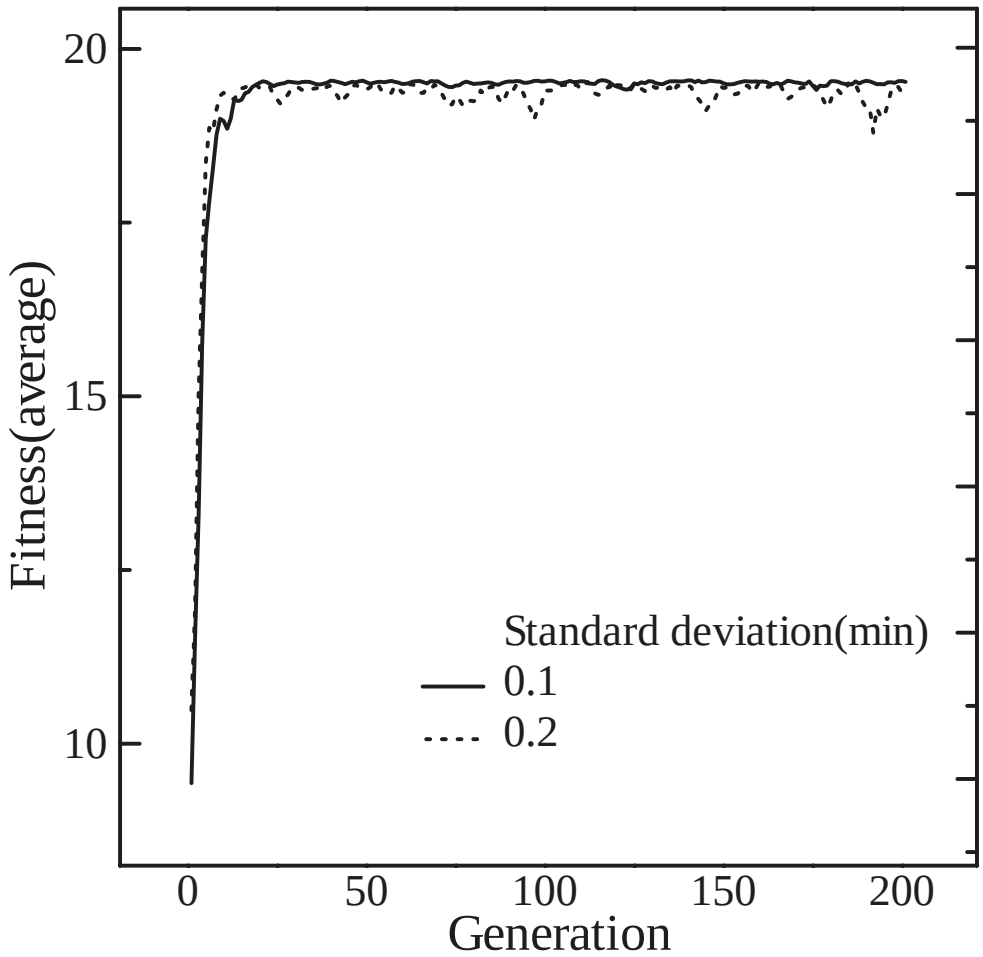

図 3.4: σ_{min}の影響

3.1.2 エリート保存

GA では交差、突然変異により、次世代に残すべき子孫を形成する。しかしそれが親よりも優れた評価を持っているとは必ずしもいえない。逆に形成された子孫の評価が悪くなり、悪い解に収束する可能性もありうる。それを防ぐため、各世代の集団中で最も優れた個体は優先的にそのまま次世代に残すという手段をとる。しかし実数領域適応型 GA では同じ染色体であっても、世代が変ると異なった個体となる特徴があるため染色体をエリート保存しても、そのビット列の表す探索領域内での個体の位置と表現型が表す設計空間内での位置が世代が進むに連れて無関係のものとなってしまう。そのため図 3.5に示すようにエリート保存を行った。全体を通して最も良い個体は表現型のみを保存することで

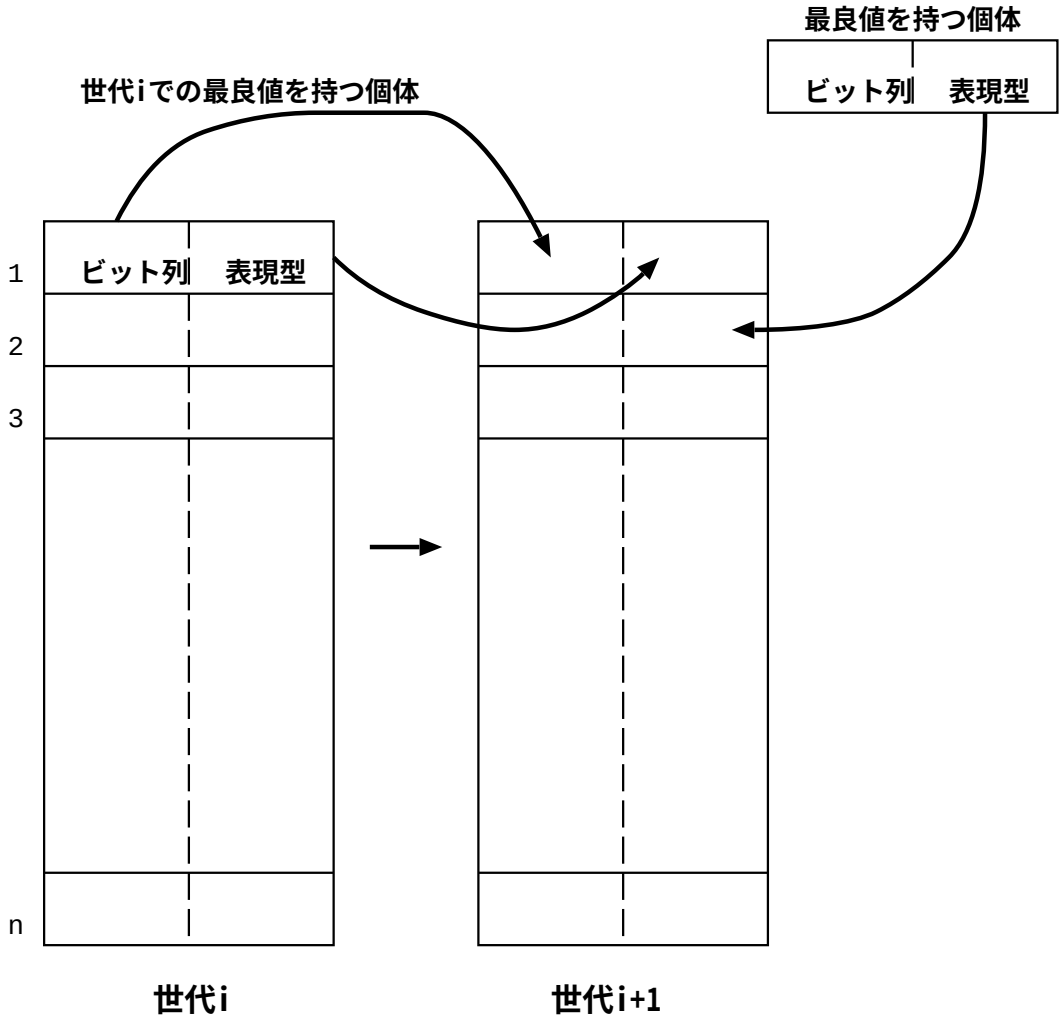

図 3.5: エリート保存

変数の平均値が最良値の変数に近づき、オーバーシュートを弱め、収束の安定を図った。またビット列はある世代の領域内での個体の分布を表すため、その世代で最も良い個体は表現型、ビット列共に保存した。また計算を 1000 回繰り返しその時得られた最大値の平均をエリート保存を行った場合とそうでない場合で比較した。この時の LBは 0.1 でσ_{min}は 0.15 である。

- エリート保存有り 19.556012
- エリート保存無し 19.549197

これよりエリート保存が有効に働いていることが確認された。

3.1.3 染色体のビット数

実数領域適応型 GA では染色体のビット数が少なくても最良値を得られる事が特徴であったが、あまり少ないビット数であると領域の移動が極端に大きく行われてしまい解の収束を妨げると考えられる。そのため 3 ビット以上で染色体を生成するのが適当であると思われる。ここでは 4 ビットとした。

3.2 多目的の最適化

単一目的の場合と同様に、以下に示す簡単な関数最適化問題に 2 章で提示した多目的最適化の手法を適用した。

$$f_1(x) = 2\sqrt{x_1} \tag{3.3}$$

$$f_2(x) = x_1(1 - x_2) + 5 \tag{3.4}$$

$$1 < x_1 < 4 \tag{3.5}$$

$$1 < x_2 < 2 \tag{3.6}$$

$$f_1(x) \rightarrow \text{最小化} \tag{3.7}$$

$$f_2(x) \rightarrow \text{最小化} \tag{3.8}$$

各パラメータを以下のように設定した。

- 個体数 30
- 最大世代数 50

- 交差率 0.3
- 突然変異率 0.02
- UB 0.95
- LB 0.3
- σ_{max} 1.0
- σ_{min} 0.15

図 3.6に得られたパレート最適解と目的関数空間での可能領域を示した。実数領域適応型 GA を用いた多目的最適化が有効に働きパレート最適解の集合を求めることができた。ここで用いたような並列選択を行う多目的 GA はパレート最適解のいくつかを求めることができるが、一つの目的関数に対して極端に良い解が選られることが多く、妥協解が選られにくいと言う問題点が指摘されている [3]。しかし実数領域適応型 GA を用いることで、図 3.6のように妥協解も多く得られた。ここで求められたパレート最適解の集合が目的関数間のトレードオフ情報を提供する。実際の設計においては、設計者がその情報に基づいて最終的な設計を得られた集合の中から一点に決定することになる。

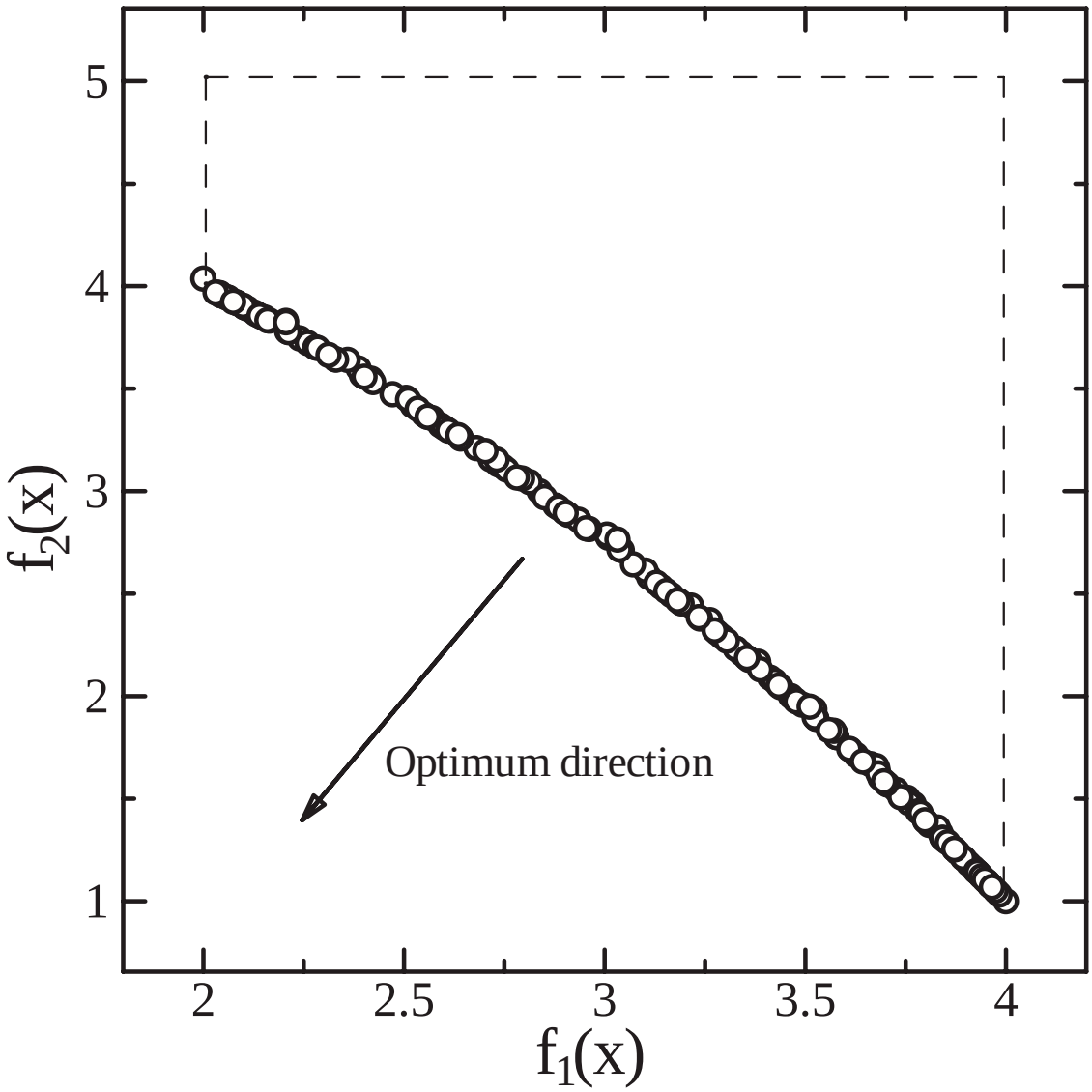

図 3.6: 得られたパレート最適解

第 4 章

翼型の空力最適化

4.1 平面翼型の形状定義

遺伝的アルゴリズムでは個体の適応度を調べるため繰り返し評価を行う。ここで厳密に翼型の形状を定義してしまうと設計変数が多くなり過ぎ解空間が大きくなるため個体の最適解への収束が遅くなってしまう。そのため個体の評価に CFD を必要とする場合はそれに費やす時間は膨大になる。そこで本研究では大林らが適用した様に複数の既知の翼型の線形結合、式 (4.1) で形状を定義する [5]。この方法を用いることで、直接形状を定義する場合に比べてはるかに設計変数を減らすことができるため GA を適用できる。

$$\hat{Y} = a_1\hat{Y}_1 + a_2\hat{Y}_2 + \cdots + a_n\hat{Y}_n \tag{4.1}$$

ここで係数 $a_1 \sim a_n$は設計変数であり、$Y_1 \sim Y_n$は基本となる翼型の与えられた x 座標での上下面における y座標の値である (図 4.1)。ここでは

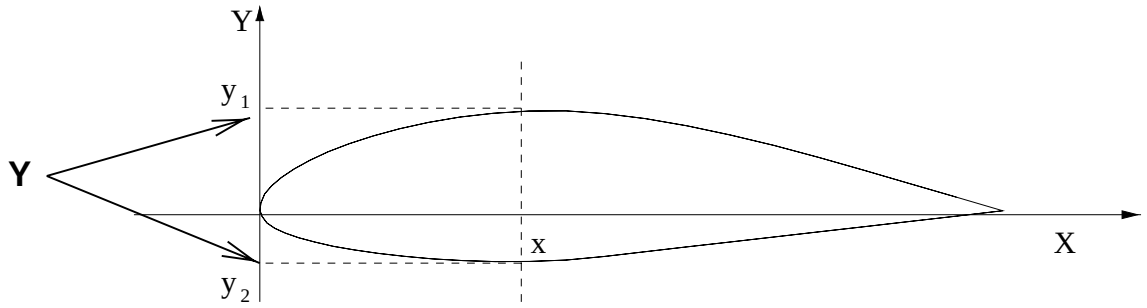

図 4.1: 翼形状の座標

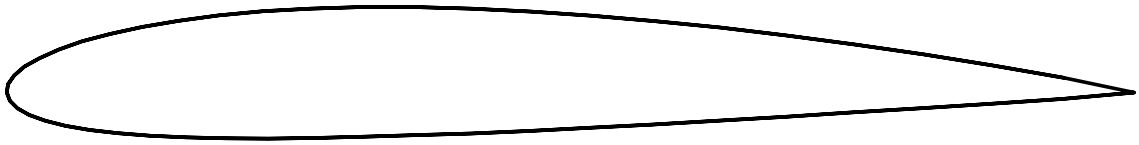

図 4.2: NACA2412

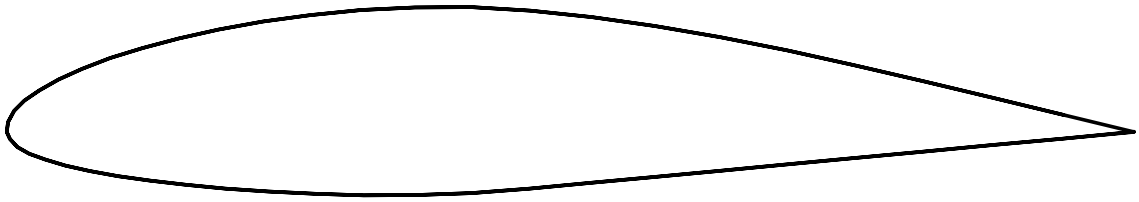

図 4.3: NACA64A516

$$\hat{Y} = a_1\hat{Y}_1 + a_2\hat{Y}_2 \tag{4.2}$$

$$-3.5 < a_1, a_2 < 3.5 \tag{4.3}$$

とし、$\hat{Y}^1, \hat{Y}^2$ として NACA2412、NACA64A516 (図 4.2,4.3) を選択した。NACA2412、NACA64A516 は最大翼厚比がそれぞれ 0.12,0.16 の翼である。式 (4.3) では係数に負の領域も含めることによって、設計空間を広い空間にしている。また翼厚が極端に薄い場合や、設計変数がすべて負になる時は翼形状がうまく定義できないため、最大翼厚比が 0.05 以下の時、つまり式 (4.4) を満たす個体は強制的に次世代に残らないように操作した。

$$0.12a_1 + 0.16a_2 < 0.05 \tag{4.4}$$

4.2 流れ場の設定

流れ場は二次元非圧縮性 Navier-Stokes 方程式に従うものとする。レイノルズ数は 1.0×10^7 とした。また流れ場の評価には PAM-FLOW を用いた。図 4.4のようなモデルで迎角を 6 度に設定し a_1, a_2を 0.2 刻みで変化させ揚力、抗力をプロットした (図 4.5)。揚力、抗力は単調な変化ではなく多峰性を示していることがわかる。

4.3 単一目的の最適化

単一目的の最適化として、

$$\text{揚力} \rightarrow \text{最大化} \tag{4.5}$$

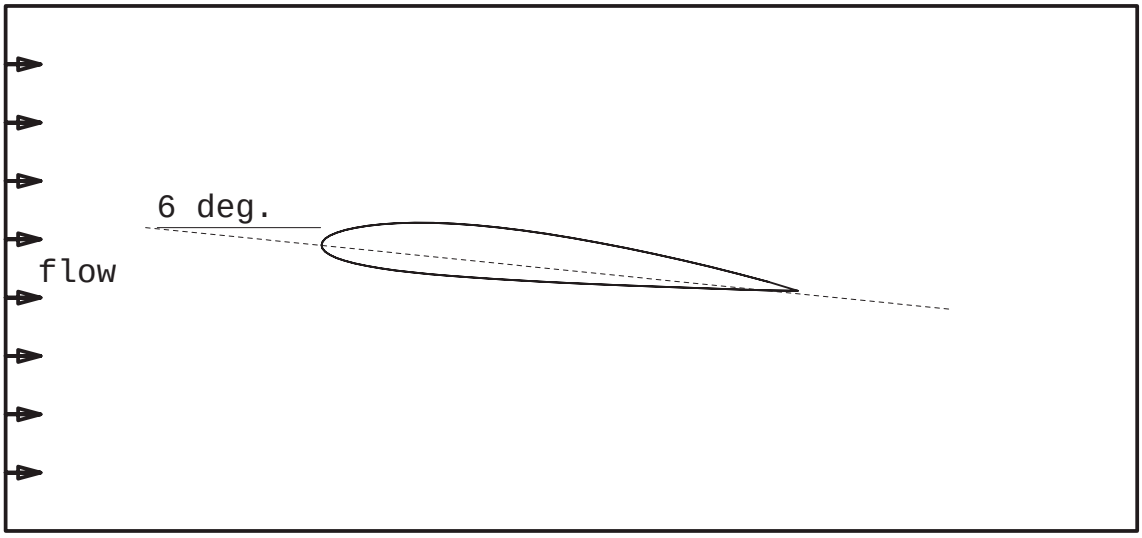

図 4.4: 計算モデル

を行なった。ここでは各パラメータを以下のように設定した。

- 個体数 30
- 最大世代数 20
- 交差率 0.3
- 突然変異率 0.02
- UB 0.95
- LB 0.1
- σ_{max} 1.5
- σ_{min} 0.15

図 4.6に最適化の履歴を示す。集団の標準偏差の大きな世代の初めの方で急に最大値を大きくしていき、標準偏差の小さくなり細かい探索が可能となった 13 世代目で最も大きな揚力係数を得た。また表 4.1より基本となった翼型より計算で得られた翼型の方が揚力が高くなっていることが確認された。得られた翼型は図 4.7のようになり、非常に厚い翼となった。このような翼は迎角をこれ以上大きくするとすぐに失速してしまうことが予想でき、揚力は高くなったが同時に抗力も高くなってしまい、翼の性能は低くなってしまった

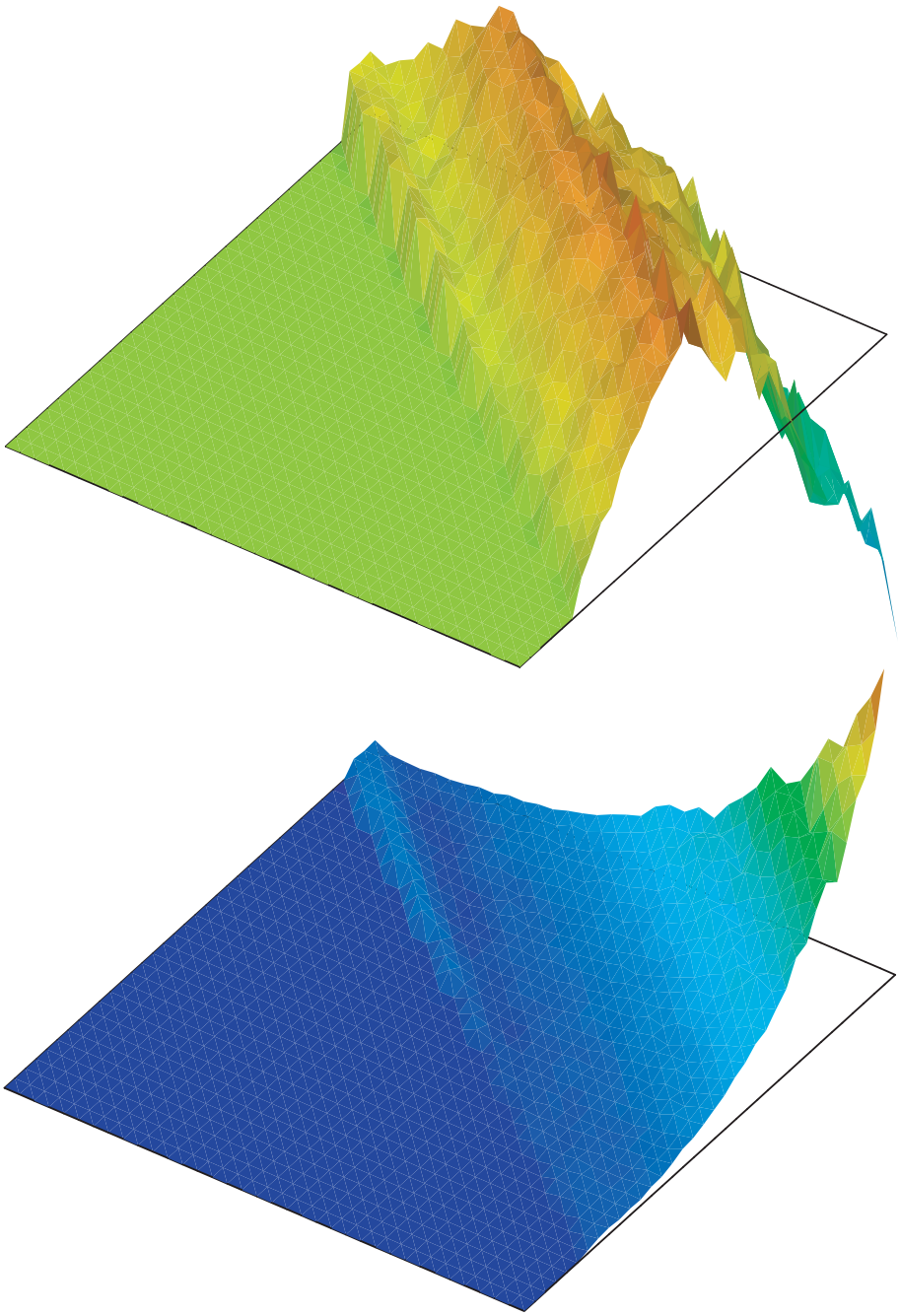

図 4.5: 設計空間内での揚力 (上)、抗力 (下)

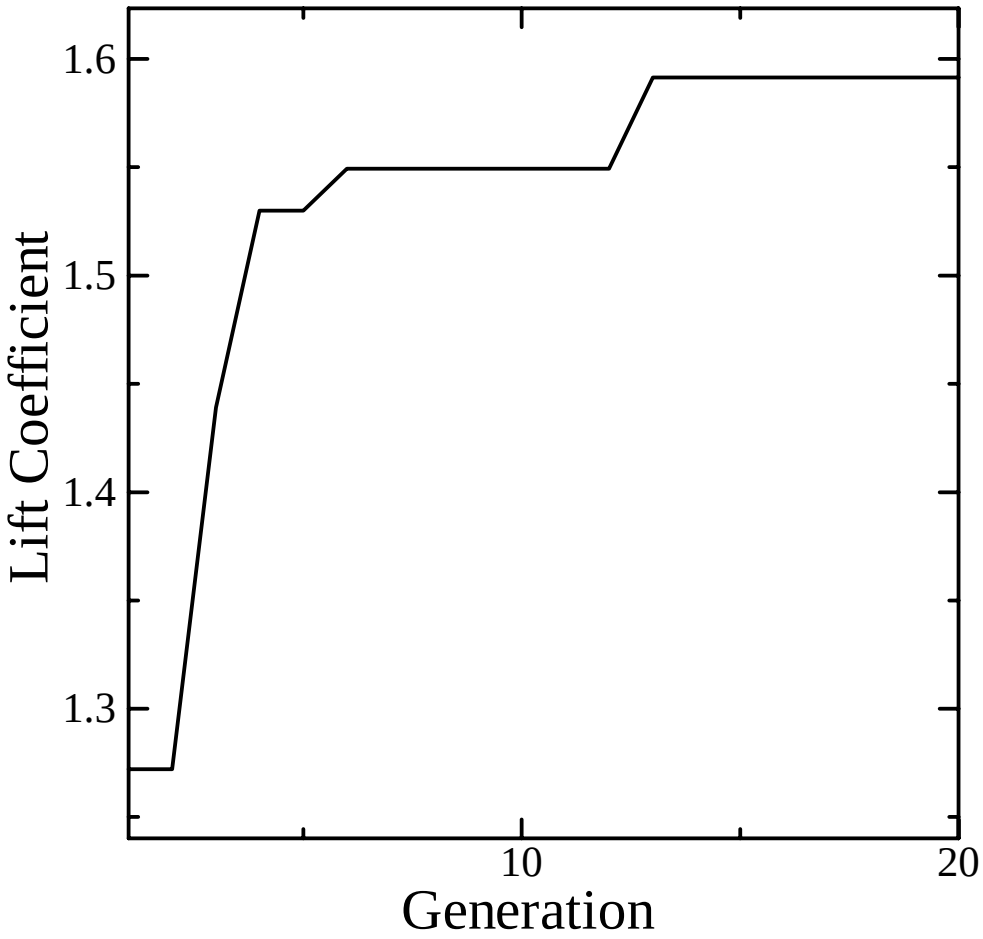

図 4.6: 最適化の履歴 (最大値)

と言える。そのため翼の本質的な性能を追求するためには、より多くの目的を同時に考慮する必要があると言える。

表 4.1: 揚力、抗力係数の比較

	揚力係数	抗力係数
naca2412	0.728	0.0436
naca64A516	0.562	0.0253
得られた翼型	1.591	0.0548

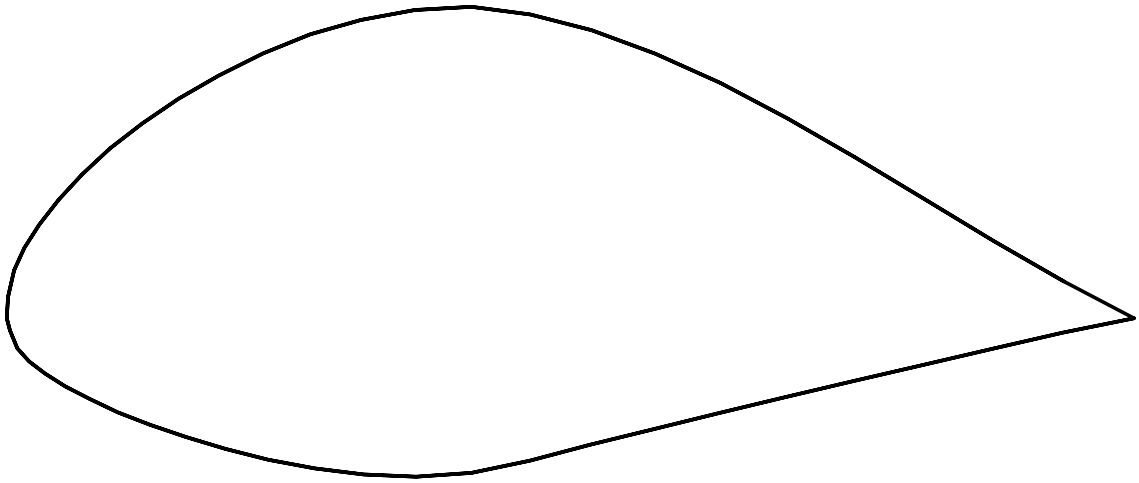

図 4.7: 得られた翼型

4.4 多目的の最適化

前節の結果より揚力の最大化をするだけでは、抗力、翼厚も大きくなってしまい翼の性能が上がらないことがわかった。そこで多目的の最適化として

$$\text{揚力} \rightarrow \text{最大化} \tag{4.6}$$

$$\text{抗力} \rightarrow \text{最小化} \tag{4.7}$$

を行なった。ここでは各パラメータを以下のように設定した。

- 個体数 30
- 最大世代数 30
- 交差率 0.3
- 突然変異率 0.02
- UB 0.95
- LB 0.3
- σ_{max} 1.5
- σ_{min} 0.15

図 4.8に得られたパレート最適解を示す。設計可能空間にパレート最適面が存在し、揚力と抗力のトレードオフ情報を得ることができた。しかしこれだけでは形状決定の重要な要素である翼の形状を推し量ることができないため、図 4.9,4.10にパレート最適解の最大翼厚比と揚力係数、抗力係数の関係を示した。これにより揚力最大化で得たような厚い翼型を避けることができる。ここで揚力、抗力のバランス、翼厚を考慮して最終的に選択した翼型を図 4.11 に示す。またその空力性能は以下のようになった。

図 4.9では最大翼厚比の小さな領域では、直線的に変化している。これは迎角を大きくしていった時の揚力の変化と同様の変化を示していると考えられる。同様に図 4.10では迎角を大きくしていった時の抗力の変化と同様の変化を示している。このようになるのは翼厚の小さな領域では、それほど多峰性が強くないからだと考えられる。逆に多峰性の強い最大翼厚比の大きな領域では規則的な変化が見られなくなっている。

- 揚力係数 0.767

- 抗力係数 0.0265
- 最大翼厚比 0.191

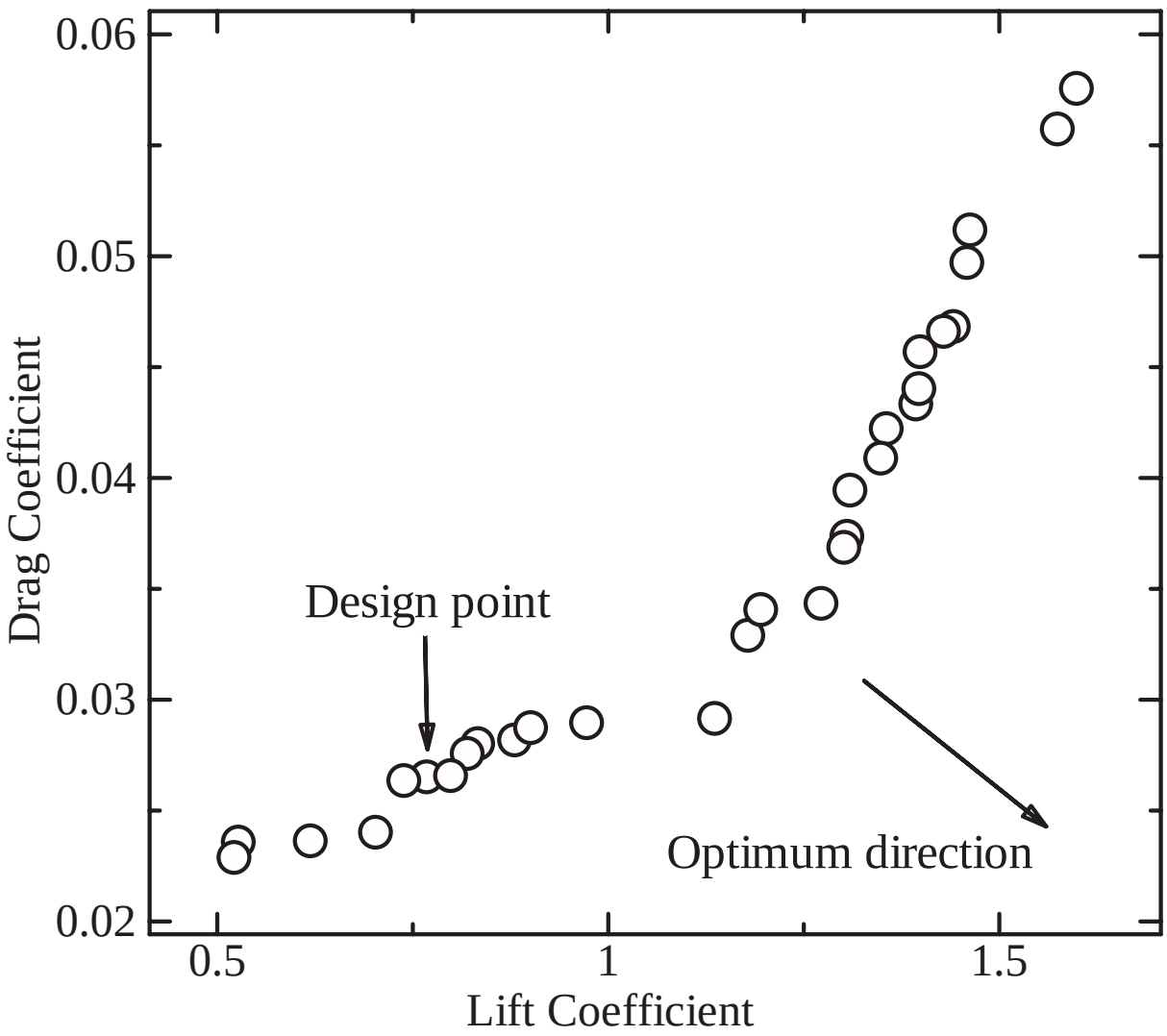

図 4.8: 得られたパレート最適解

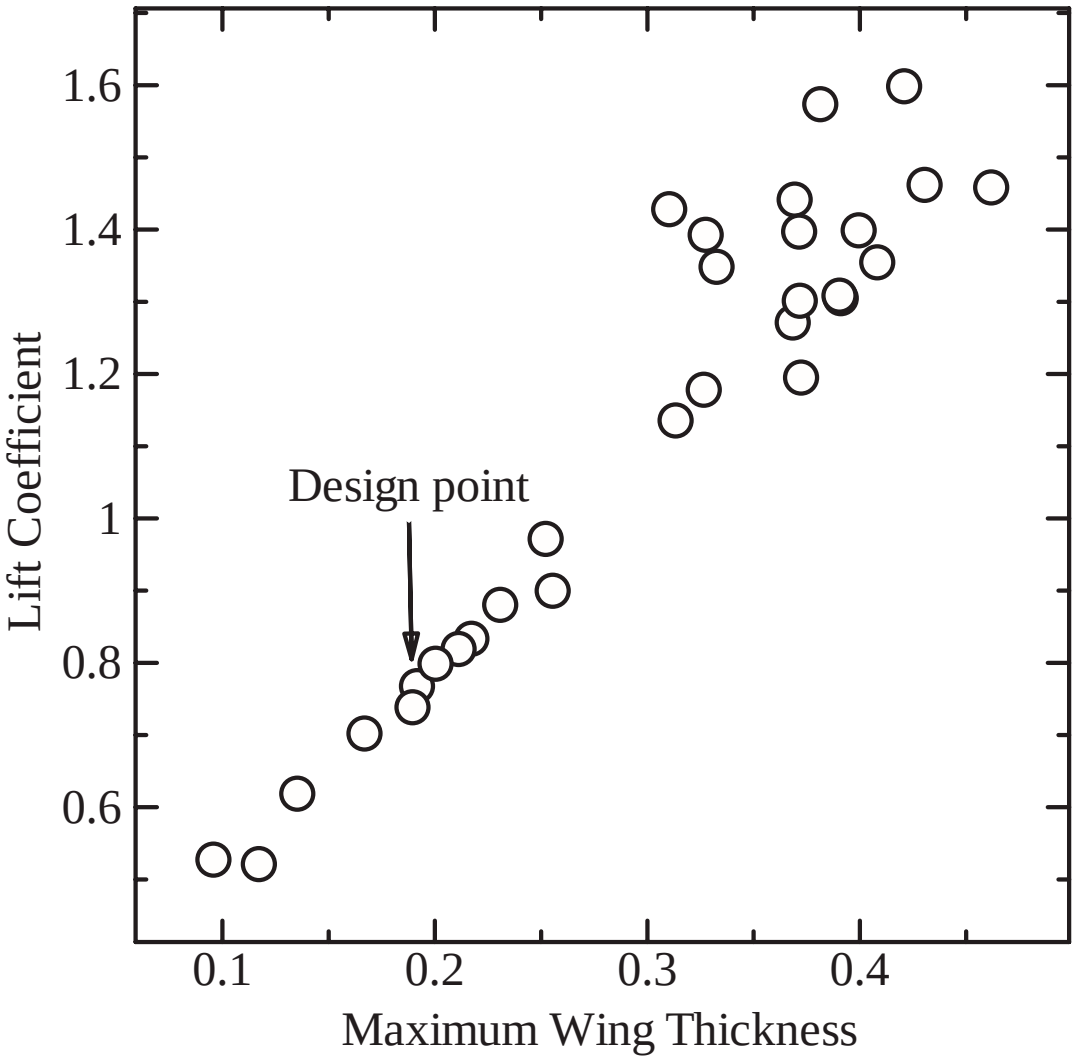

図 4.9: 翼厚と揚力の関係

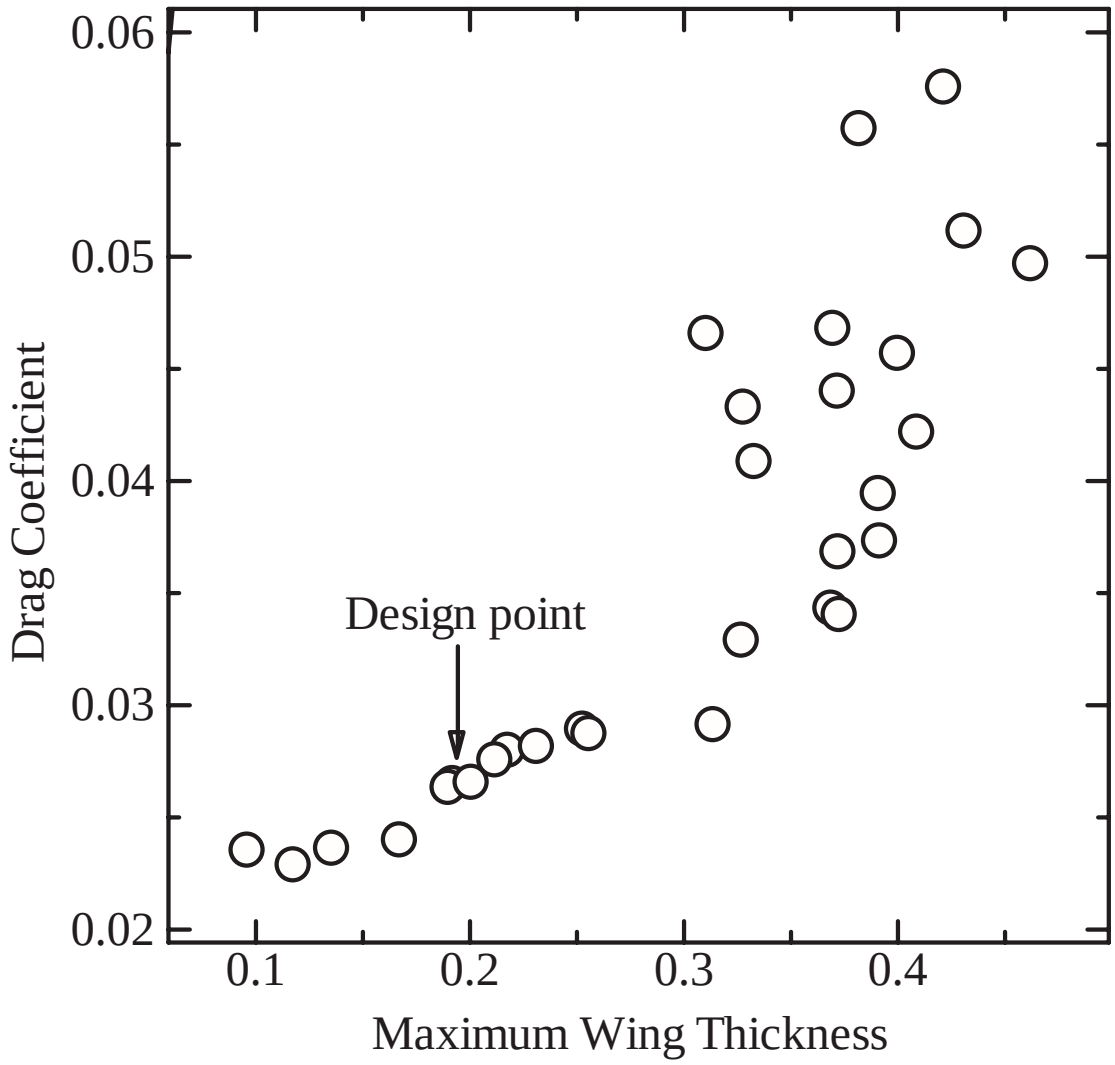

図 4.10: 翼厚と抗力の関係

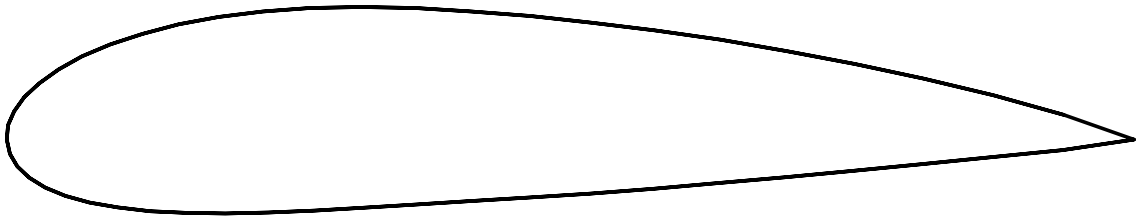

図 4.11: 得られた翼型

第 5 章

結言

本研究において以下のことを行った。

- 最適化の手法としてビット表現法に基づいて実数を取り扱う実数領域適応型 GA を用いた。
- 実数領域適応型 GA が有効に働く様なシステムパラメータの値と、エリート保存の方法を実験によって設定した。
- 実数領域適応型 GA を多目的最適化問題に対応させた。
- 翼型の多目的形状最適化問題を構築した。
- 揚力、抗力、翼厚の情報から良好な性能を持つ翼型を決定した。

考察

GA は元々、設定しなければならないパラメータが多く、その設定によっては適切に探索を行わなくなることもあった。実数領域適応型 GA では設定しなければならないパラメータが更に多くなり、設計者にとって負担となってしまうのが問題である。しかし実験により設定したパラメータを用いて翼型の最適化を行ったところ、有効な結果が得られたため、本研究で設定したパラメータにある程度汎用性があると言えるだろう。また、多目的最適化で得られた最も揚力の高い翼と揚力最大化で得られた翼の性能は近い値をもっているため、単一目的の最適化は多目的のパレート最適解の一点を求めていると考えられる。計算量としては単一目的と多目的ではそれほど違わないため、より多くの意味ある解を求められる多目的 GA は単一目的 GA に比べて効率が良いと言える。

謝辞

本研究を行うにあたり、数多くのご指導、ご助言を賜わりました松澤照男教授ならびに松澤研究室の皆様に深く感謝いたします。

参考文献

[1] 日本機会学会編, 工学問題を解決する適応化・知能化・最適化法技報堂出版,1996.

[2] 安居院 猛, 長尾 智晴, ジェネティックアルゴリズム, 昭晃堂,1993.

[3] 北野 宏明, 遺伝的アルゴリズム 2 産業図書,1995.

[4] 荒川 雅生, 荻原 一郎, 実数領域適応型 (ARRange) 遺伝的アルゴリズムの開発, 日本機会学会論文集 (C 編)63 巻 616 号,pp4216-4217,1997.

[5] 大林 茂, 塚原 孝則, 進化アルゴリズムによる空力最適化 I, 数値流体力学, 第 6 巻 2 号,pp59-73,1998.

[6] 大林 茂, 塚原 孝則, 進化アルゴリズムによる空力最適化 II, 数値流体力学, 第 6 巻 3 号,pp83-97,1998.