数理解析研究所講究録 第2126巻 2019年 53-62



# 集団的降下法に対するペナルティ係数の適応的調整法の提案

# A Proposal of Adaptive Control of Penalty Coefficient for Population-Based Optimization Algorithms

広島修道大学商学部　　阪井　節子 (Setsuko Sakai)
Faculty of Commercial Sciences, Hiroshima Shudo University
広島市立大学大学院 情報科学研究科　高濱　徹行 (Tetsuyuki Takahama)
Graduate School of Information Sciences, Hiroshima City University

## 1　はじめに

制約付き最適化問題は与えられた制約の下で目的関数を最適化する問題である．この中でも特に制約付き非線形最適化問題は実世界に頻繁に出現する重要な最適化問題である．制約付き非線形最適法問題の解法としては，逐次２次計画法，射影法，一般縮小勾配法などの効率的な方法が存在する．しかし，これらの方法は目的関数の微分可能性や制約領域の凸性など問題に対して幾つかの条件を仮定している．したがって，これらの方法を様々な分野における問題に広く適用することは困難である．

これに対して，目的関数の値だけを利用して制約のない非線形最適化問題を解決する方法が提案されている．この方法は直接探索法 (direct search method) と呼ばれ，様々な問題に適用することができるが，制約付き非線形最適法問題を直接解くことはできない．本研究では，直接探索法を制約付き非線形最適法問題に適用する方法について考察する．

制約付き最適化問題を直接探索法によって解く際には，目的関数だけではなく，一般に複数の制約も最適化する必要がある．制約付き最適化のためには、目的関数の最適化と制約の最適化という 2 種類の最適化が含まれるため，2 つの最適化のバランスを適切に取ることが必要となる．制約を扱う方法は，同時に最適化する目的の数に基づき，以下のように分類できる．

(1) 目的関数のみを最適化する方法

制約を満足する一つ以上の探索点を初期点として準備し，制約を満足する探索点のみを考慮してゆくことにより，制約の最適化を省略する方法であり，death penalty 法とも呼ばれる．探索の過程で得られた点が制約を満足しない場合には，単純に無視されるか，制約を満足するように修正される．例えば，遺伝的アルゴリズム (Genetic Algorithm, GA) において，制約を満足した探索点を参照して制約を満足しない探索点を修正する方法が提案されている [1, 2, 3]．また，パーティクルスォーム最適化 (Particle Swarm Optimization, PSO)[4, 5, 6] において，制約を満足しない探索点を無視し，既知の制約を満足する探索点に置換する方法も提案されている [7]．これらの方法は制約領域が比較的広い場合には有効である．しかし，実際には制約の厳しい問題も多く，特に等式制約を含む問題では，初期点を準備したり探索点を修正することは不可能に近い．

(2) 目的関数と制約逸脱度 (constraint violation) の荷重和を最適化する方法

複数の制約条件を組み合わせて制約逸脱度を定義し，目的関数と制約逸脱度の荷重和を求め，その荷重和の一目的最適化問題として解く方法である．制約逸脱度は目的関数に対するペナルティと考えられるため，この方法は一般にペナルティ関数法 (penalty function method) と呼ばれ，目的関数と制約逸脱度の荷重和は拡張目的関数 (extended objective function) と呼ばれている．ペナルティ関数法では，制約逸脱度の強さ

を調整するための荷重であるペナルティ係数 (penalty coefficient) を適切に選択することが困難であるという問題点がある．ペナルティ係数が大きいと，制約を満足する解は得られるが，目的関数の最適化が不十分になり，質の高い解を得ることが困難になる．逆にペナルティ係数が小さいと，目的関数は最適化されるが，制約の最適化が不十分になり，実行可能解を得ることが困難になる．ペナルティ係数を動的に調整する方法もあるが，適切な調整方法は問題に依存するため，一般的な調整方法を実現するのは困難であるという問題がある．

(3) 目的関数と制約逸脱度を辞書式比較により最適化する方法
　目的関数と制約逸脱度を分離して扱い，制約逸脱度を優先する辞書式比較により一目的化して解く方法である．例えば，GA において辞書式比較を実現する拡張目的関数を利用する方法が提案されている [8]．これは，制約を満足しない探索点の拡張目的関数値を，集団中の制約を満足する探索点における最悪の目的関数値と制約逸脱度との和として与える方法である．進化的戦略 (Evolutionary Strategy) において，単に制約逸脱度を優先するのではなく，ある確率で制約逸脱度を無視し目的関数のみで比較を行うという拡張された辞書式比較により最適化を行う方法が提案されている [9]．これにより，制約条件が確率的に緩和され，質の高い実行可能解が得られることが示されている．より一般的な方法として，直接探索法全般に対して，制約条件を緩和することができる辞書式比較である $\alpha$ レベル比較を使用する $\alpha$ 制約法 [10, 11, 12, 13, 14, 15, 16, 17, 18] および $\varepsilon$ 比較を使用する $\varepsilon$ 制約法 [19, 20] が提案されている．$\alpha$ 制約法および $\varepsilon$ 制約法は，直接探索法における比較演算子を $\alpha$ レベル比較および $\varepsilon$ レベル比較に置換することにより，制約のない問題に対する最適化アルゴリズムを制約付きの最適化アルゴリズムに変換する方法，すなわちアルゴリズム変換法である．$\alpha$ および $\varepsilon$ 制約法は，制約条件を緩和することにより，等式制約を含むような制約条件の厳しい問題に対しても適用することができる．

(4) 目的関数と各制約の多目的問題として解く方法
　目的関数と一般に複数の制約関数を多目的最適化問題として解く方法である [21, 22]．制約が複雑な問題に有効であると期待されるが，多目的最適化問題は一目的問題と比較すると非常に困難な問題であり，一般に多くの計算量を必要とするという問題点がある．

　本研究では，進化的アルゴリズムなどの集団に基づく最適化において，集団から適切なペナルティ係数を設定するための情報として，等価ペナルティ係数値 (Equivalent Penalty Coefficient Value, EPC) を提案するとともに，EPC を用いてペナルティ係数を動的に調整する方法を提案する．本手法を工業設計問題に適用することによりその有効性を示す．
　本論文の構成は次の通りである．**2.** で制約付き最適化問題とペナルティ関数法について簡単に説明する．**3.** で等価ペナルティ係数値 (EPC) およびその調整法を提案する．**4.** で提案手法について説明し，ベンチマーク関数に関する実験結果を示す．**5.** はまとめである．

## 2　制約付き最適化問題

### 2.1　定義

　本論文では，次のような不等式制約，等式制約，上下限制約を持つ最適化問題 (P) を考える．目的関数および制約条件がともに線形の場合が線形計画問題，その他の場合が非線形計画問題である．

$$
\begin{aligned}
(P)\ \text{minimize}\quad & f(\boldsymbol{x}) \\
\text{subject to}\quad & g_j(\boldsymbol{x}) \le 0,\ j=1,\ldots,q \\
& h_j(\boldsymbol{x}) = 0,\ j=q+1,\ldots,m \\
& l_i \le x_i \le u_i,\ i=1,\ldots,n
\end{aligned}
\tag{1}
$$

ここで，$\boldsymbol{x} = (x_1, \cdots, x_n)$ は $n$ 次元決定変数ベクトル，$f(\boldsymbol{x})$ は 目的関数，$g_j(\boldsymbol{x}) \leq 0$ は $q$ 個の不等式制約，$h_j(\boldsymbol{x}) = 0$ は $m-q$ 個の等式制約であり，$f, g, h$ は線形あるいは非線形の実数値関数である．$l_i, u_i$ はそれぞれ，$n$ 個の決定変数 $x_i$ の下限値，上限値である．さらに，以下では全ての制約を満足する領域を実行可能領域 (feasible region)，上下限制約を満足する領域を探索領域 (search space) と呼ぶことにする．

### 2.2 ペナルティ関数法

制約付き最適化では，目的関数の最適化と制約の最適化という 2 つの最適化を同時に行う必要がある．ペナルティ関数法では，目的関数 $f(\boldsymbol{x})$ に制約逸脱度 $\phi(\boldsymbol{x})$ $(\geq 0)$ をペナルティとして加えることにより，制約付き最適化問題を以下のような制約なし最適化問題に変換する．

$$F(\boldsymbol{x}) \quad = \quad f(\boldsymbol{x}) + \rho\phi(\boldsymbol{x}) \tag{2}$$

ここで，$F$ は拡張目的関数，$\rho$ はペナルティ係数 $(\rho > 0)$ である．ペナルティ係数を $\rho \to \infty$ とすると，$F(\boldsymbol{x})$ を最小化するには制約逸脱度は $\phi(\boldsymbol{x}) \to 0$ となり，実行可能解を得ることができる．

制約逸脱度 $\phi(\boldsymbol{x})$ の定義としては，以下の例がある．

$$\phi(\boldsymbol{x}) \quad = \quad \max\{\max_j\{0, g_j(\boldsymbol{x})\}, \max_j |h_j(\boldsymbol{x})|\} \tag{3}$$

$$\phi(\boldsymbol{x}) \quad = \quad \sum_j ||max\{0, g_j(\boldsymbol{x})\}||^p + \sum_j ||h_j(\boldsymbol{x})||^p \tag{4}$$

ここで $p$ は正数である．

## 3 集団的降下法における等価ペナルティ係数値

### 3.1 集団的降下法

DE や PSO などのように解集団による最適化の際に降下法を利用した最適化法である集団的降下法 (population-based descent method) について説明する． 集団的降下法は一般に以下のように記述できる．

1. 初期化: 解をランダムに生成し集団 $P = \{\boldsymbol{x}_i, i = 1, 2, \cdots, N\}$($N$ は解の数) を構成する．
2. 評価: 全ての解を評価する．
3. 終了判定: 終了条件を満足すれば終了する．
4. 各解に対して，
   (a) 生成: 各解 $\boldsymbol{x}_i$ と集団 $P$ の情報に基づき新しい解 $\boldsymbol{x}'_i$ を生成する．
   (b) 評価: 新しい解を評価する．
   (c) 更新: 新しい解が古い解より良ければ，古い解を新しい解で置換する．
5. 3. へ戻る

### 3.2 等価ペナルティ係数値

集団的降下法では，古い解 $\boldsymbol{x}_i$ と新しい解 $\boldsymbol{x}'_i$ を比較し，新しい解が古い解より良ければ古い解を新しい解で置換する．ペナルティ関数法を使用する場合，拡張目的関数値による評価結果はペナルティ係数の値によって一般には異なる．

$\boldsymbol{x}'_i$ の関数値と制約逸脱度の両者が $\boldsymbol{x}_i$ より優れている場合は，ペナルティ係数の値に無関係に $\boldsymbol{x}'$ は拡張目的関数値が良い解となる，すなわち，任意の $\rho$ について $F(\boldsymbol{x}') < F(\boldsymbol{x})$ が成立する．$\boldsymbol{x}$ と $\boldsymbol{x}'$ の関係が逆の場合も同様である．また，関数値と制約逸脱度が同じ場合は，任意の $\rho$ について $F(\boldsymbol{x}) = F(\boldsymbol{x}')$ が成立する．したがって，以下の条件が成り立つ場合にはペナルティ係数を決定する必要が無い．

$$f(\boldsymbol{x}') \le f(\boldsymbol{x}) \text{ and } \phi(\boldsymbol{x}') \le \phi(\boldsymbol{x}) \quad \text{or} \quad f(\boldsymbol{x}) \le f(\boldsymbol{x}') \text{ and } \phi(\boldsymbol{x}) \le \phi(\boldsymbol{x}') \tag{5}$$

これ以外の場合，関数値が優れているが制約逸脱度が劣っている解と関数値が劣っているが制約逸脱度が優れている解が存在する．前者が $\boldsymbol{x}_i$ 後者が $\boldsymbol{x}'_i$ と仮定すると，$f(\boldsymbol{x}_i) < f(\boldsymbol{x}'_i)$, $\phi(\boldsymbol{x}_i) > \phi(\boldsymbol{x}'_i)$ である．この 2 つの解の拡張関数値が一致するペナルティ係数値を等価ペナルティ係数値 (equivalent penalty coefficient value, EPC) と呼ぶことにする．EPC を $\rho_i$ で表現すると以下の関係が成立する．

$$\begin{aligned} f(\boldsymbol{x}_i) + \rho_i\phi(\boldsymbol{x}_i) &= f(\boldsymbol{x}'_i) + \rho_i\phi(\boldsymbol{x}'_i) \qquad (6)\\ \rho_i &= \frac{f(\boldsymbol{x}'_i) - f(\boldsymbol{x}_i)}{\phi(\boldsymbol{x}_i) - \phi(\boldsymbol{x}'_i)} \qquad (7) \end{aligned}$$

このとき，ペナルティ係数 $\rho$ が EPC より大きい，すなわち $\rho > \rho_i$，ならば，$F(\boldsymbol{x}_i) > F(\boldsymbol{x}'_i)$ となるため，制約逸脱度が優れている $\boldsymbol{x}'_i$ が良い解となる．逆にペナルティ係数が EPC より小さい，すなわち $\rho < \rho_i$，ならば，$F(\boldsymbol{x}_i) < F(\boldsymbol{x}'_i)$ となるため，目的関数値が優れている $\boldsymbol{x}_i$ が良い解となる．

式 (5) を満たさない解 $\boldsymbol{x}_i$，$\boldsymbol{x}'_i$ に対する EPC $\rho_i$ を昇順にソートしたリストを $H = \{\rho_k | \rho_k < \rho_{k+1}, k = 1, 2, \cdots\}$ とする．適応的にペナルティ係数を制御するために，アルゴリズムパラメータとして制約優先率 (constraint priority rate)$R_{cp}$ を導入する．このとき，$H$ の $R_{cp}|H|$ 番目の要素の値をペナルティ係数 $\rho$ に設定するために，以下のように線形補間を利用する．

$$\rho = \begin{cases} R_{cp}|H|\rho_1, & \text{if } \lfloor R_{cp}|H| \rfloor < 1 \\ R_{cp}\rho_{|H|}, & \text{if } R_{cp} > 1 \\ \rho_{\lfloor R_{cp}|H| \rfloor} + (R_{cp}|H| - \lfloor R_{cp}|H| \rfloor)(\rho_{\lceil R_{cp}|H| \rceil} - \rho_{\lfloor R_{cp}|H| \rfloor}), & otherwise \end{cases} \tag{8}$$

ただし，$|H|$ は $H$ の要素数，$\lfloor a \rfloor$ は $a$ 以下の最大の整数，$\lceil a \rceil$ は $a$ 以上の最小の整数である．

$R_{cp} = 0$ のときは $\rho = 0$ となり，目的関数値のみの最適化を行うことになる．$R_{cp} > 1$ のときは $\rho > \rho_{|H|}$ となり，2 つの解の比較で制約逸脱度が優先されることになるため，制約逸脱度のみの最適化を行うことになる．すなわち，ペナルティ係数を $\infty$ に増加させるのと同じ効果がある．

このため，$R_{cp}$ を小さい値から 1 を超えた値，例えば 1.1 まで増加させることにより，実行可能解を得ることができる．本研究では，$R_{cp}$ を固定した場合について考察する．

このように等価ペナルティ係数値は，集団的降下法のように，古い解と新しい解を一対比較する集団的最適化手法に用いることができる．

## 4 提案手法

本研究では集団的降下法として，差分進化 (Differential Evolution, DE) を採用し，DE に EPC に基づく適応的ペナルティ法を導入した DEEPC(DE using Equivalent Penalty Coefficient value for constrained optimization) を提案する。

### 4.1 差分進化

差分進化 (DE) は Storn and Price[23, 24] によって提案された進化的アルゴリズムである．DE は確率的な直接探索法であり，解集団を用いた多点探索を行う．DE には幾つかの形式が提案されており，DE/best/1/bin や DE/rand/1/exp などがよく知られている．これらは，DE/$base$/$num$/$cross$ という記法で表現される．“$base$” は基本ベクトルとなる親の選択方法を指定する．例えば，DE/rand/$num$/$cross$ は基本ベクトルのための親を集団からランダムに選択し，DE/best/$num$/$cross$ は集団の最良個体を選択する．“$num$” は基本ベクトルを変異させるための差分ベクトルの個数を指定する．“$cross$” は子を生成するために使用する交叉方法を指定する．例えば，DE/$base$/$num$/bin は一定の確率で遺伝子を交換する交叉 (binomial crossover) を用い，DE/$base$/$num$/exp は，指数関数的に減少する確率で遺伝子を交換する交叉 (exponential crossover) を用いる．本研究では，差分ベクトル数を 1 ($num = 1$) とした DE/rand/1/exp を用いる．

### 4.2 提案手法のアルゴリズム

提案手法 DEEPC のアルゴリズムは以下の通りである．

**Step0** 初期化．$N$ 個の初期個体 $\boldsymbol{x}_i$ を探索空間内に生成し，初期集団 $P = \{\boldsymbol{x}_i \mid i = 1, 2, \cdots, N\}$ を構成する．全ての個体を評価する．

**Step1** 終了判定．終了条件を満足すれば，アルゴリズムは終了する．終了条件としては，最大関数評価回数を用いる．

**Step2** DE 操作（子個体の生成）．全ての個体 $\boldsymbol{x}_i$ $(i = 1, 2, \cdots, N)$ を親ベクトルとして選択し，以下の操作を順に行う．3 個体 $\boldsymbol{x}_{r1}, \boldsymbol{x}_{r2}, \boldsymbol{x}_{r3}$ を，$\boldsymbol{x}_i$ および互いに重複しないようにランダムに選択する．変異ベクトル $\boldsymbol{m}_i = \boldsymbol{x}_{r1} + F(\boldsymbol{x}_{r2} - \boldsymbol{x}_{r3})$ を生成する．ここで，$F$ はスケーリングパラメータである．変異ベクトル $\boldsymbol{m}_i$ と親個体 $\boldsymbol{x}_i$ を交叉し，子個体 $\boldsymbol{x}_i^{\mathrm{child}}$ を生成し，評価する．

**Step3** ペナルティ係数の決定．EPC ($\rho_i$) を次のようにして求める．式 (5) を満足する個体の EPC を $\infty$ とする．その他の個体の EPC を式 (7) で求める．リスト $\{\rho_i\}$ を昇順にソートする．$\infty$ の要素を削除し $H$ を求める．式 (8) に基づきペナルティ係数を決定する．

**Step4** 生存者選択 (世代交代）．各親個体とその子個体を順に比較するために，各親個体と子個体の拡張目的関数値を求める．拡張目的関数値を比較し，子個体 $\boldsymbol{x}_i^{\mathrm{child}}$ が親個体 $\boldsymbol{x}_i$ よりも良ければ子個体が生存者となり，親を子で置換する．

**Step5** Step1 に戻る．

### 4.3 実験条件

本研究では，DE に関するパラメータは，個体数 $N$=20, スケーリングパラメータ $F$=0.8，交叉率 $CR$=0.95 とし，提案手法のパラメータ制約優先率を $R_{cp}$=0.9 とした．目的関数の最大評価回数 $FE_{\max}$ を 2,500 回，5,000 回，10,000 回の 3 通りの場合について実験を行った．各問題について 30 回独立に試行を行い，各試行における最良個体の目的関数値を求めた．ただし，最良個体とは，制約逸脱度が最小の個体であり，制約逸脱度が同じ場合は目的関数値が最小の個体である．

### 4.4 Himmerblau の問題

Himmerblau の問題は以下のように定式化される．

$$
\begin{aligned}
\text{Minimize } f(\boldsymbol{x}) &= 5.3578547x_3^2 + 0.8356891x_1x_5 + 37.293239x_1 - 40792.141 \\
\text{Subject to } g_1(\boldsymbol{x}) &= 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5, \\
g_2(\boldsymbol{x}) &= 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2, \\
g_3(\boldsymbol{x}) &= 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4, \\
0 \le g_1(\boldsymbol{x}) &\le 92, 90 \le g_2(\boldsymbol{x}) \le 110, 20 \le g_3(\boldsymbol{x}) \le 25, \\
78 \le x_1 &\le 102, 33 \le x_2 \le 45, 27 \le x_3, x_4, x_5 \le 45.
\end{aligned}
$$

表 1 に実験結果を示す．提案手法以外の結果は，文献 [25] に基づくものであり，MGA は多目的 GA と解の優越関係によるアルゴリズム，Gen は遺伝的アルゴリズムを利用したアルゴリズム，GRG は Generalized Reduced Gradient 法，Death は death penalty 法，その他はペナルティに基づくアルゴリズムであり，ペナルティ係数を固定する static penalty, 探索ステップ数によりペナルティ係数を変化させる dynamic penalty, simulated annealing のように温度によりペナルティ係数を変化させる annealing penalty, 複数の探索点の状態によりペナルティ係数を決定する adaptive penalty，解の集団と 2 種類のペナルティル係数のための集団を用いて共進化させる Coevolutionary penalty 法である．

各アルゴリズムによる試行中の最良値，平均値，最悪値および標準偏差を示した．良い結果を示したアルゴリズムは DEEPC, MGA, Co-evolutionary である．DEEPC は関数評価回数が 2,500 回の時点でも全ての項目で他のアルゴリズムより優れている．したがって，DEEPC は効率よく探索を行える安定したアルゴリズムであるといえる．

表 1: Result of Himmerblau's problem

| Algorithm | FEs | Best | Average | Worst | S.D. |
|---|---|---|---|---|---|
| DEEPC | 2500 | **-31025.1166** | **-31021.3957** | **-31012.1760** | **3.2432** |
| | 5000 | **-31025.5554** | **-31025.5189** | **-31025.3960** | **0.0379** |
| | 10000 | **-31025.5602** | **-31025.5602** | **-31025.5602** | **0.0000** |
| MGA | 5,000 | -31005.7966 | -30862.8735 | -30721.0418 | 73.240 |
| Gen | | -30183.576 | N/A | N/A | N/A |
| GRG | | -30373.949 | N/A | N/A | N/A |
| Co-evolutionary | 900,000 | -31020.859 | -30984.2407 | -30792.4077 | 73.6335 |
| Static | 5,000 | -30790.2716 | -30446.4618 | -29834.3847 | 226.3428 |
| Dynamic | 5,000 | -30903.877 | -30539.9156 | -30106.2498 | 200.035 |
| Annealing | 5,000 | -30829.201 | -30442.126 | -29773.085 | 244.619 |
| Adaptive | 5,000 | -30903.877 | -30448.007 | -29926.1544 | 249.485 |
| Death | 5,000 | -30790.271 | -30429.371 | -29834.385 | 234.555 |

### 4.5 角材の溶接の設計

角材の剪断応力 ($\tau$)，曲げ応力 ($sigma$)，台の座屈荷重 ($P_c$)，角材の端のたわみ ($\delta$) などの制約の下でコストが最小となる角材の溶接を設計する．図 1 のように，4 つの決定変数 $h(x_1)$，$l(x_2)$，$t(x_3)$，$b(x_4)$ により設計する．

この問題は以下のように定式化される．

$$
\begin{aligned}
&\text{Minimize } f(\boldsymbol{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(14+x_2)\\
&\text{Subject to } g_1(\boldsymbol{x}) = \tau(\boldsymbol{x}) - \tau_{max} \le 0,\ g_2(\boldsymbol{x}) = \sigma(\boldsymbol{x}) - \sigma_{max} \le 0,\ g_3(\boldsymbol{x}) = x_1 - x_4 \le 0,\\
&\quad g_4(\boldsymbol{x}) = 0.10471x_1^2 + 0.04811x_3x_4(14+x_2) - 5 \le 0,\ g_5(\boldsymbol{x}) = 0.125 - x_1 \le 0,\\
&\quad g_6(\boldsymbol{x}) = \delta(\boldsymbol{x}) - \delta_{max} \le 0,\ g_7(\boldsymbol{x}) = P - P_c(\boldsymbol{x}) \le 0,\ 0.1 \le x_1, x_4 \le 2, 0.1 \le x_2, x_3 \le 10,\\
&\quad \text{where}\\
&\quad \tau = \sqrt{\tau'^2 + 2\tau'\tau''\frac{x_2}{2R} + \tau''^2},\ \tau' = \frac{P}{\sqrt{2}x_1x_2}, \tau'' = \frac{MR}{J}, M = P\left(L + \frac{x_2}{2}\right),\\
&\quad R = \sqrt{\frac{x_2^2 + (x_1+x_3)^2}{4}},\ J = 2\sqrt{2}x_1x_2\left(\frac{x_2^2}{12} + \frac{(x_1+x_3)^2}{4}\right),\\
&\quad \sigma(\boldsymbol{x}) = \frac{6PL}{x_4x_3^2},\ \delta(\boldsymbol{x}) = \frac{4PL^3}{Ex_3^3x_4},\ Pc(\boldsymbol{x}) = \frac{4.013E\sqrt{x_3^2x_4^6/36}}{L^2}\left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right),\\
&\quad P = 6000lb, L = 14in, \delta_{max} = 0.25in,\ E = 30\times10^6psi, G = 12\times10^6psi,\\
&\quad \tau_{max} = 13600psi, \sigma_{max} = 30000psi.
\end{aligned}
$$

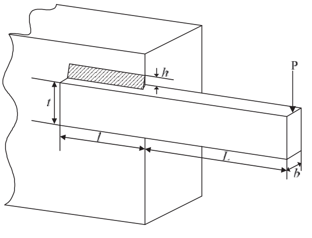


図 1: Welded beam design

表 2 に実験結果を示す．良い結果を示したアルゴリズムは DEEPC, MGA, Co-evolutionary である．DEEPC は関数評価回数が 2,500 回の時点で最悪値および標準偏差以外の項目で他の全てのアルゴリズムより優れており，非常に精度の高い最良解を見つけている．なお，評価回数 5,000 回の時点で全ての項目で他の全てのアルゴリズムより優れている．したがって，DEEPC は効率よく探索を行えるアルゴリズムであるといえる．

表 2: Result of welded beam problem

| Algorithm | FEs | Best | Average | Worst | S.D. |
|---|---|---|---|---|---|
| DEEPC | 2500 | **1.7279** | **1.7431** | 1.7906 | 0.0142 |
| | 5000 | **1.7249** | **1.7253** | **1.7273** | **0.0005** |
| | 10000 | **1.7249** | **1.7249** | **1.7249** | **0.0000** |
| MGA | 5,000 | 1.8245 | 1.9190 | 1.9950 | 0.05377 |
| Co-evolutionary | 900,000 | 1.7483 | 1.7720 | 1.7858 | 0.01122 |
| Static | 5,000 | 2.0469 | 2.9728 | 4.5741 | 0.6196 |
| Dynamic | 5,000 | 2.1062 | 3.1556 | 5.0359 | 0.7006 |
| Annealing | 5,000 | 2.0713 | 2.9533 | 4.1261 | 0.4902 |
| Adaptive | 5,000 | 1.9589 | 2.9898 | 4.84036 | 0.6515 |
| Death | 5,000 | 2.0821 | 3.1158 | 4.5138 | 0.6625 |

## 4.6 圧力容器の設計

半球状のキャップが両端に付いている円筒状の容器において，材料，形成，溶接に必要なコストを最小化する問題である．図 2 に示すように，$T_s$(シェルの厚み)，$T_h$(キャップの厚み)，$R$(内径)，$L$(円筒状の長さ)の 4 変数を設計する．このうち，$T_s$ と $T_h$ は利用可能な筒状鋼板の厚みから，0.0625 インチの整数倍である．

この問題は以下のように定式化される．

$$
\begin{aligned}
&\text{Minimize } f(\boldsymbol{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
&\text{Subject to } g_1(\boldsymbol{x}) = -x_1 + 0.0193x_3 \leq 0,\ g_2(\boldsymbol{x}) = -x_2 + 0.00954x_3 \leq 0, \\
&\qquad g_3(\boldsymbol{x}) = -\pi x_3^2x_4 - 4\pi/3x_3^3 + 1296000 \leq 0,\ g_4(\boldsymbol{x}) = x_4 - 240 \leq 0, \\
&\qquad x_1, x_2 = 0.0625i, i \in \{1, 2, \cdots, 99\}, 10 \leq x_3, x_4 \leq 200.
\end{aligned}
$$

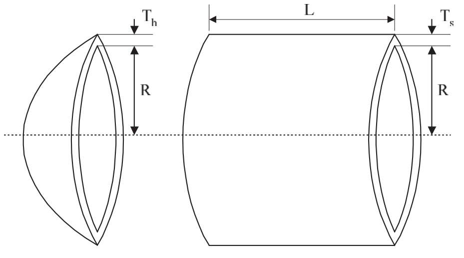


図 2: Pressure Vessel design

表 3 に実験結果を示す．Deb は Genetic Adaptive Search，Kannan は拡張 Lagrangian Multiplier 法，Sandgen は Branch and Bound 法によるものである．

良い結果を示したアルゴリズムは DEEPC, MGA, Co-evolutionary である．DEEPC は関数評価回数が 2,500 回の時点で標準偏差以外の項目で他のアルゴリズムより優れており，非常に精度の高い最良解を見つけている．なお，5,000 回の時点で全ての項目で他のアルゴリズムより優れている．したがって，DEEPC は効率よく探索を行えるアルゴリズムであるといえる

表 3: Result of Pressure Vessel problem

| Algorithm | FEs | Best | Average | Worst | S.D. |
|---|---|---|---|---|---|
| DEEPC | 2500 | **6065.4514** | **6106.0972** | **6238.1967** | 44.0454 |
| | 5000 | **6059.7152** | **6059.8638** | **6061.5215** | **0.3324** |
| | 10000 | **6059.7143** | **6059.7143** | **6059.7143** | **0.0000** |
| MGA | 50,000 | 6069.3267 | 6263.7925 | 6403.4500 | 97.9445 |
| Deb | | 6410.3811 | N/A | N/A | N/A |
| Kannan | | 7198.0428 | N/A | N/A | N/A |
| Sandgen | | 8129.1036 | N/A | N/A | N/A |
| Co-evolutinary | 900,000 | 6288.7445 | 6293.8432 | 6308.1497 | 7.4133 |
| Static | 2,500,000 | 6110.8117 | 6656.2616 | 7242.2035 | 320.8196 |
| Dynamic | 2,500,000 | 6213.6923 | 6691.5606 | 7445.6923 | 322.7647 |
| Annealing | 2,500,000 | 6127.4143 | 6660.8631 | 7380.4810 | 330.7516 |
| Adaptive | 2,500,000 | 6110.8117 | 6689.6049 | 7411.2532 | 330.4483 |
| Death | 2,500,000 | 6127.4143 | 6616.9333 | 7572.6591 | 358.8497 |

## 5 あとがき

制約付き最適化問題を制約なしの問題に変換するペナルティ関数法に対して，等価ペナルティ係数値 EPC を提案し，集団的降下法において EPC に基づきペナルティ係数を動的に制御する方法を提案した．本論文では，提案手法を差分進化に適用した DEEPC を構成した．DEEPC により幾つかの代表的な制約付き最適化問題を解くことにより，DEEPC の有効性を示した．

今後は，アルゴリズムパラメータである $R_{cp}$ の制御について考察するとともに，本手法を様々なアルゴリズムに適用し，その性能を調べることを予定している．

**謝辞** 本研究は，JSPS 科研費 26350443, 17K00311 の助成を受けて行われた．

## 参考文献

[1] Michalewicz, Z.: Genetic Algorithms, Numerical Optimization and Constraints, in *Proc. of the 6th International Conference on Genetic Algorithms*, pp. 151–158, Pittsburgh (1995).

[2] Michalewicz, Z.: A Survey of Constraint Handling Techniques in Evolutionary Computation Methods, in *Proc. of the 4th Annual Conference on Evolutionary Programming*, pp. 135–155, The MIT Press, Cambridge, Massachusetts (1995).

[3] Michalewicz, Z. and Schoenauer, M.: Evolutionary Algorithms for Constrained Parameter Optimization Problems, *Evolutionary Computation*, Vol. 4, No. 1, pp. 1–32 (1996).

[4] Kennedy, J. and Eberhart, R. C.: Particle Swarm Optimization, in *Proc. of IEEE International Conference on Neural Networks*, pp. 1942–1948, Perth, Australia (1995).

[5] Shi, Y. and Eberhart, R.: A Modified Particle Swarm Optimizer, in *Proc. of IEEE International Conference on Evolutionary Computation*, pp. 69–73, Anchorage (1998).

[6] Kennedy, J. and Eberhart, R. C.: *Swarm Intelligence*, Morgan Kaufmann, San Francisco (2001).

[7] Parsopoulos, K. E. and Vrahatis, M. N.: Particle Swarm Optimization Method for Constrained Optimization Problems, in Sincak, P., Vascak, J. and et al., eds., *Intelligent Technologies — Theory and Application: New Trends in Intelligent Technologies*, Vol. 76 of *Frontiers in Artificial Intelligence and Applications*, pp. 214–220, IOS Press (2002).

[8] Deb, K.: An Efficient Constraint Handling Method for Genetic Algorithms, *Computer Methods in Applied Mechanics and Engineering*, Vol. 186, No. 2/4, pp. 311–338 (2000).

[9] Runarsson, T. P. and Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization, *IEEE Trans. on Evolutionary Computation*, Vol. 4, No. 3, pp. 284–294 (2000).

[10] 高濱徹行, 阪井節子：制約付き非線形最適化手法 $\alpha$ 制約法によるファジー制御ルールの最適化, 電子情報通信学会論文誌, Vol. J82-A, No. 5, pp. 658–668 (1999).

[11] Takahama, T. and Sakai, S.: Tuning Fuzzy Control Rules by the $\alpha$ Constrained Method which Solves Constrained Nonlinear Optimization Problems, *Electronics and Communications in Japan, Part 3: Fundamental Electronic Science*, Vol. 83, No. 9, pp. 1–12 (2000).

[12] 高濱徹行, 阪井節子：$\alpha$ 制約 Simplex 法によるファジー制御ルールの学習, 電子情報通信学会論文誌, Vol. J83-D-I, No. 7, pp. 770–779 (2000).

[13] Takahama, T. and Sakai, S.: Learning Fuzzy Control Rules by $\alpha$-Constrained Simplex Method, *Systems and Computers in Japan*, Vol. 34, No. 6, pp. 80–90 (2003).

[14] 高濱徹行, 阪井節子：$\alpha$ 制約遺伝的アルゴリズム $\alpha$GA による制約付き最適化, 電子情報通信学会論文誌, Vol. J86-D-I, No. 4, pp. 198–207 (2003).

[15] Takahama, T. and Sakai, S.: Constrained Optimization by $\alpha$ Constrained Genetic Algorithm ($\alpha$GA), *Systems and Computers in Japan*, Vol. 35, No. 5, pp. 11–22 (2004).

[16] Takahama, T. and Sakai, S.: Constrained Optimization by Combining the $\alpha$ Constrained Method with Particle Swarm Optimization, in *Proc. of Joint 2nd International Conference on Soft Computing and Intelligent Systems and 5th International Symposium on Advanced Intelligent Systems* (2004).

[17] Takahama, T. and Sakai, S.: Constrained Optimization by the $\alpha$ Constrained Particle Swarm Optimizer, *Journal of Advanced Computational Intelligence and Intelligent Informatics*, Vol. 9, No. 3, pp. 282–289 (2005).

[18] Takahama, T. and Sakai, S.: Constrained Optimization by Applying the $\alpha$ Constrained Method to the Nonlinear Simplex Method with Mutations, *IEEE Trans. on Evolutionary Computation*, Vol. 9, No. 5, pp. 437–451 (2005).

[19] Takahama, T. and Sakai, S.: Constrained Optimization by $\varepsilon$ Constrained Particle Swarm Optimizer with $\varepsilon$-level Control, in *Proc. of the 4th IEEE International Workshop on Soft Computing as Transdisciplinary Science and Technology (WSTST'05)*, pp. 1019–1029 (2005).

[20] Takahama, T., Sakai, S. and Iwane, N.: Constrained Optimization by the $\varepsilon$ Constrained Hybrid Algorithm of Particle Swarm Optimization and Genetic Algorithm, in *Proc. of the 18th Australian Joint Conference on Artificial Intelligence*, pp. 389–400 (2005), Lecture Notes in Computer Science 3809.

[21] Camponogara, E. and Talukdar, S. N.: A Genetic Algorithm for Constrained and Multiobjective Optimization, in Alander, J. T. ed., *3rd Nordic Workshop on Genetic Algorithms and Their Applications (3NWGA)*, pp. 49–62, Vaasa, Finland (1997), University of Vaasa.

[22] Surry, P. D. and Radcliffe, N. J.: The COMOGA Method: Constrained Optimisation by Multiobjective Genetic Algorithms, *Control and Cybernetics*, Vol. 26, No. 3, pp. 391–412 (1997).

[23] Storn, R. and Price, K.: Minimizing the Real Functions of the ICEC'96 Contest by Differential Evolution, in *Proc. of the International Conference on Evolutionary Computation*, pp. 842–844 (1996).

[24] Storn, R. and Price, K.: Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces, *Journal of Global Optimization*, Vol. 11, pp. 341–359 (1997).

[25] Coello, C. A. C.: Theoretical and Numerical Constraint-Handling Techniques used with Evolutionary Algorithms: A Survey of the State of the Art, *Computer Methods in Applied Mechanics and Engineering*, Vol. 191, No. 11-12, pp. 1245–1287 (2002).