ペナルティ法による目的関数生成における重み付け自動化

村田幸弘† 安藤竜弥† 阿部充志†

多制約条件を課した場合の，ペナルティ法を用いた汎用的な目的関数作成手法を提案する．制約条件が複数存在する場合，得られる解の精度はその重みづけの良し悪しに依存する．本研究では，各ペナルティ関数と最小化する評価関数の代表値の比を各重みとした．代表値は探索回数ごとに求められる関数値点列の近似曲線から求め，重みは規定回数ごとに自動的に更新する．これにより，各関数が偏りなく評価され，局所解に陥ることを回避できる．提案手法をテスト関数に適用した結果，制約条件を満足した高精度な解を得られることが分かった．

The Use of Automated Weighting to Generate an Objective Function by Penalty Method

Yukihiro Murata,† Ryuuya Ando† and Mitsushi Abe†

This paper proposes a method to generate an objective function with automated weighting, and how it can be applied to optimization algorithms, such as penalty method. In cases where a lot of constraints exist, the weighting is critical. In this paper, the ratio of the evaluated function value over each penalty function value is used to determine each weight. Each function value is smoothened through spline fitting. These weights should be updated automatically at predefined times. This proposed method could diminish the values of each function impartially, thereby avoiding obtaining local optimum solutions. In addition, this paper shows the results of benchmark tests which showed good performance at solving some test problems.

1. はじめに

近年，最適化計算の工学機器設計への応用は目覚ましく進展している．しかし，それにもかかわらず，制約条件を課した最適化には様々な問題が浮上している．これは，制約条件をどのような形で最適化問題に導入すれば最適解が得られるかが未解明であるためである．また，対象となる目的関数の形状を知ることが困難である場合が多く，さらに工学機器設計での制約条件もいっそう厳しくなっている．

このような背景の下，複数の制約条件を関数化し，それをペナルティ関数として目的関数に取り込み，無制約条件下での最適化問題に置き換える手法（ペナルティ法）で，工学機器を設計することに成功した(Parmee)[1]．しかしながら，ペナルティ関数に乗じる重み係数によって，その結果の良し悪しが左右されるため[2]，この手法を他の工学設計に応用することは困難であり，設計者が重み係数の設定，変更を繰り返して（トライアンドエラーによって）解を得るしかなかった．

一般的に，制約条件を満足する大域的最適解を得るためには，実行可能領域と実行不可能領域の双方を探索する必要がある[3]．このため，目的関数を適切に作成しないと，どちらかの領域に偏った探索が進行し最終的に局所解に陥ってしまう．

これまでもこのような問題を解決するための研究が行われてきた．近年では，遺伝的アルゴリズム（Genetic Algorithm: GA）による多制約条件付き最適化問題に関連する研究がさかんに行われている（COMOGA[4]，VEGA[5]，MOGA[6]，NPGA[7]）．GA は，初期値依存性の低い最適化アルゴリズムであるが，計算量が多い点が問題である．さらに，工学機器設計では，経験に基づいた良解が存在する場合が多く，軽微な変更の際には変更前の設計変数を初期値として利用できるため，スピードを重視した最適化アルゴリズムが望まれる．

そこで，汎用的な単点探索の単目的最適化アルゴリズムをそのまま利用可能な制約条件付き目的関数の生成手法を開発した．これにより，目的関数の記述部分

† 株式会社日立製作所電力電機開発研究所
Power & Industrial Systems R&D Laboratory, Hitachi, LTD.

を本手法に置き換えるのみで，制約条件付き最適化問題を扱うことができる．目的関数生成にはペナルティ法を用い，トライアンドエラーに依存していたペナルティ関数の重み付けを自動化している．

ペナルティ法では，各ペナルティ関数値と最小化する評価関数値の線形和を目的関数とするが，それらの関数には相関関係はないため，関数値に偏りがある場合には，ある特定の関数の最小化となりうる．これにより発生する問題として，探索ごとに探索領域が狭まるアルゴリズムを用いた場合，上記特定の関数を最小化する領域に探索が進行し，最終的に大域最適解近傍領域に到達できない可能性がある．本研究では，この対策として各ペナルティ関数に可変重みを乗じることで偏りを排除し，それぞれの制約条件を偏りなく満足させながら評価コスト関数を最小化した．ところで，重みを可変とするだけでは，重みの変動量が大きい場合は目的関数が固定されず，求める最適解が定まらないという問題がある．そこで，探索ごとに得られる各関数値からなる点列に振動の小さい近似曲線（平滑化自然スプライン関数）をあてはめ，曲線上の点を用いて重み付けを行うことで変動を抑える工夫を施した．

最後に本手法をよく知られたテスト関数に適用し，その有効性を検証する．

2. 目的関数作成手法概要

いま，最小化したい評価コスト関数を $f(\boldsymbol{x})$ とし，m 個の制約条件を $g_k(\boldsymbol{x}) \leq 0$（$k = 1, 2, \cdots, m$）とする．

Minimize: $f(\boldsymbol{x})$

subject to:

$g_k(\boldsymbol{x}) \leq 0, \quad k = 1, 2, \cdots, m.$

提案手法では，制約条件をコスト化した関数（ペナルティ関数とよぶ）$p_k(\boldsymbol{x})$（$k = 1, 2, \cdots, m$）を評価コスト関数 $f(\boldsymbol{x})$ に加算して目的関数 $o(\boldsymbol{x})$ を作成する．

$$o(\boldsymbol{x}) = f(\boldsymbol{x}) + \sum_{k=1}^{m} w_k p_k(\boldsymbol{x}). \tag{1}$$

ここで，ペナルティ関数 $p_k(\boldsymbol{x})$ に乗じた w_k（≥ 0）は重みである．また，$p_k(\boldsymbol{x})$ は制約対象の計算値の制約値からの逸脱量（≥ 0）と定義する．以下のテスト関数では，

$$p_k(\boldsymbol{x}) = \max\{0, g_k(\boldsymbol{x})\}, \tag{2}$$

とした．これらの各コスト関数値は最適化による探索が実施されるごとに求まる．

ところで，最適化アルゴリズムに焼きなまし法（Simulated Annealing: SA）を採用した場合，探索の進展にともなって探索領域が狭まる（温度が低下する）．この場合，式 (1) での $f(\boldsymbol{x})$ と $w_k p_k(\boldsymbol{x})$ を同程度に評価しなければ最適解を得られない可能性がある．**図 1** にその概念図を示す．探索開始時の初期設計変数を設定し，探索領域を点線円内部とする（図 1 左）．関数値に差があり，制約条件 1 のペナルティ関数が支配的であるとすると，制約条件 1 のみを満足するように探索が進行する．この結果，図 1 右に示すように制約条件 1 を満足する解は得られるものの，探索領域に大域的最適解が含まれなくなり，探索を継続しても求めるべき解には到達しない．固定重みの場合には，初期条件を基に重みを決定するが，探索過程で関数値は変動するため，上記の問題が発生する可能性がある．この状況を回避するためには，それぞれの関数を偏りなく低減させなければならない．つまり，探索過程においても評価コスト関数 $f(\boldsymbol{x})$ とペナルティ関数 $p_k(\boldsymbol{x})$ を同程度に最小化する重み付けが必要である．

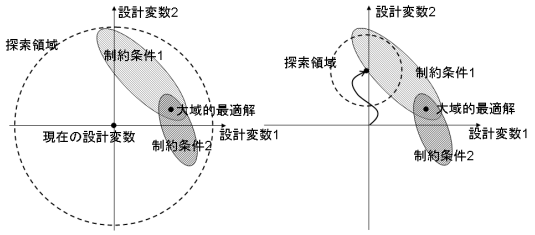

図 1 最適解にたどり着けない場合の概念図（左：探索序盤，右：制約条件 1 を満足する解を探索直後）

Fig. 1 Schematic drawing showing the example that the optimum is not to be obtained.

そこで，ペナルティ関数が評価コスト関数とおよそ同値となるような重み w_k をペナルティ関数に乗じることにより，関数間で偏りなく目的関数を最小化する．具体的には，重み w_k は各ペナルティ関数値で評価コスト関数値を規格化した値に相当する．

$$w_k \sim \left| \frac{f(\boldsymbol{x})}{p_k(\boldsymbol{x})} \right|. \tag{3}$$

ここで，重みを等式ではなく近似式で示したことに注意する．重み更新に際し，劇的に重みが変動した場合，つまり各関数値が大きく変動した場合，目的関数 $o(\boldsymbol{x})$ 自体が変動し，最小化対象が不明瞭となる．そこで，変動する各関数値そのものの比を重みとして算出するのではなく，変動する各関数それぞれにおいて探索回数をパラメータとした各関数値の近似曲線を描き，その曲線上の点を各関数値の代表値として選定し，その比を算出する．近似曲線には，ベジエ曲線などがよく用いられるが，変動を吸収した滑らかな近似曲線とはならない．そこで，各関数値に対して，それらの

値を反映しながらも滑らかさに重点を置いた近似曲線（平滑化自然スプライン関数）を採用した．

平滑化自然スプラインとは，以下に示す 2 つの量の線形結合を最小にすることによって座標上のデータ点列 $(x_1, y_1), (x_2, y_2), \cdots, (x_n, y_n)$（$a < x_1 < x_2 < \cdots < x_n < b$）を平滑化する手法である[8]．その量とは，

(1) データとそれに対応する平滑化曲線上の値との差を二乗し，それに重みを乗じた値の総和，
(2) データの対応する平滑化曲線上の値の l 階差分の二乗和（l は与えられた整数），

であり，これら 2 つの量を結合する割合は，データに対して忠実であるか滑らかであるかのどちらに重点をおくかで決まる．具体的には，正の重み ω_i および g を用いて σ を，

$$\sigma = \sum_{i=1}^{n} \omega_i \{f(x_i) - y_i\}^2 + g \int_a^b \{f^{(l)}(x)\}^2 dx, \quad (4)$$

と定義すれば，σ を可能な限り小さくする l 階連続微分可能な関数 $f(x)$ が求める近似曲線（平滑化自然スプライン関数）となる．正の重み g によって，滑らかさの度合いを調整できる．

探索回数と評価コスト関数値およびペナルティ関数値をそれぞれ横軸と縦軸とした n 個のデータ列 $(1, f_1(\boldsymbol{x})), \cdots, (n, f_n(\boldsymbol{x}))$ および $(1, p_{k,1}(\boldsymbol{x})), \cdots, (n, p_{k,n}(\boldsymbol{x}))$ に対して，n 個のデータ列を用いて描いた上記近似曲線上の n 個の点列をそれぞれ $(1, sf_1(\boldsymbol{x})), \cdots, (n, sf_n(\boldsymbol{x}))$ および $(1, sp_{k,1}(\boldsymbol{x})), \cdots, (n, sp_{k,n}(\boldsymbol{x}))$ とする．各近似曲線は各データ列に忠実ではあるものの，滑らかさに重点をおいているため，変動の少ない曲線となっている．なお，ペナルティ関数値がゼロとなるデータ（制約条件を満足するデータ）はデータ列から除外して近似曲線を算出する．ここで得られた近似曲線上の点から，式 (3) と同様に比を算出して重み w_k とする．

$$w_k = \left| \frac{sf_n(\boldsymbol{x})}{sp_{k,n}(\boldsymbol{x})} \right|. \quad (5)$$

重みは一定回数探索が進行するたびごとに更新する．ただし，すべての探索で k 番目の制約条件を満足する場合（つまり $p_{k,n}(\boldsymbol{x})$（$n = 1, 2, \cdots, n$）がゼロの場合），もしくは 1 回の探索のみ制約条件を満足しない場合（つまり $p_{k,n}(\boldsymbol{x})$（$n = 1, 2, \cdots, n$）のいずれか 1 つが非ゼロの場合）は，更新前の重みを引き継ぐこととする．

また，式 (5) では評価コスト関数内での各項が同程度となる重みを示したが，探索結果が制約条件を満足しない場合もある．トレードオフを許容する制約条件であればこの場合も解となる．しかし，その条件が必須である場合には，ペナルティ関数に支配的とならない程度の定数（> 1）を乗じる．これにより，相対的に必須制約条件に関するペナルティ関数値を大きくすることができ，他の制約条件に比して優先的に最小化される．しかし，定数を乗じたペナルティ関数のみが最小化されるような偏った最適化とはならない．これにより，探索序盤から上記比率程度に各項が保たれる．

3. 提案目的関数の最適化アルゴリズムへの組み込み

本論文では，単点探索の単目的最適化アルゴリズムとして，SA を採用した．SA は，溶解状態にある物質を冷却して結晶状態にするプロセスからヒントを得たアルゴリズムで，山登り法（Hill Climbing: HC）に確率的な遷移を導入した手法である．解の近傍を探索し，解が改善されれば，それに置き換えるが，解が改善されなくても，ある確率により置き換えるため，局所解に陥りにくいという利点がある．トレードオフの関係にある解の探索には，この改悪解の受け入れは有効である．また，探索スピードは HC とほぼ同等であり，GA などの多点探索手法より短時間に（最適）解に到達する．ここでは，設計変数の感度の違いに対応した SA である，Adaptive Simulated Annealing (ASA)[9] を最適化アルゴリズムとして用いた．

3.1 繰返し探索の概要

SA は設計変数の初期値依存性が高いため，設計変数初期値についても配慮する必要がある．今回の検証では設計変数初期値を設計変数領域の中間値とした．初期値依存性を排除するためには多点探索が有効であるが，今回はできる限り早く近傍解を得ることを狙い，探索（初期温度 $T_0 = 1.0$）によって得られた解を初期値として再度探索（初期温度 $T_0 = 1.0 \times 10^{-10}$）を実施することとした．温度は以下の検討を基に設定している．

ASA において，変量 P^i が範囲 $[A_i, B_i]$ にあるとき，

$$P_{k+1}^i = P_k^i + y^i (B_i - A_i), y^i \in [-1, 1], \quad (6)$$

により次の値を生成する[9]．変数 y_i は式 (7) に示す確率密度関数を用いてランダムに選ばれる．

$$g^i(y^i; T_i) = \frac{1}{2(|y^i| + T_i)\ln(1 + 1/T_i)}. \quad (7)$$

繰返し探索初回において温度は $T = 1.0 \times 10^{-20}$ 程度にまで低下しており，確率的探索手法であるがゆえの乱数の揺らぎによって最適解が得られていない可能性を考慮して，再探索での初期温度を設定した．確率

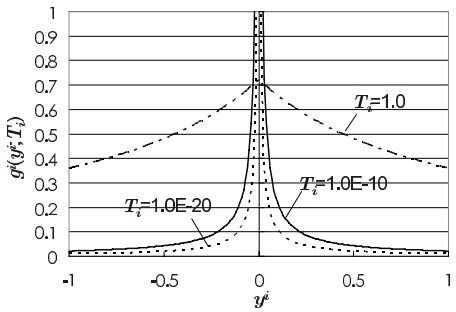

図 2 y^i 生成に用いる確率密度関数の例
Fig. 2 Examples of the probability density function.

密度関数の例を**図 2** に示す.

3.2 提案手法での重み更新頻度

提案する可変重みの更新頻度も得られる結果を左右すると考えられる．低い場合は，各関数値の変化が重みに反映されにくくなり，固定重みの場合の結果に近づく．また更新頻度が高い場合は，特に探索序盤において，目的関数が頻繁に変化するうえ，全体の計算時間に占める重み更新時間の割合が増大するため，その影響を受けにくい範囲でできる限り高い値が望ましい．

そこで，更新頻度を 1. 探索を 30 回行うごとに更新，2.50 回ごとに更新，3.100 回ごとに更新，4.200 回ごとに更新，5.300 回ごとに更新，6.400 回ごとに更新の 6 ケースについて，それぞれ何回探索を繰り返すことで理論解に到達するかを実験した．対象として Michalewicz らの不等式制約を課したテスト関数[10] $g01$ を選定した．

Minimize:

$$f(\boldsymbol{x}) = 5\sum_{i=1}^{4} x_i - 5\sum_{i=1}^{4} x_i^2 - \sum_{i=5}^{13} x_i \qquad (8)$$

subject to:

$$g_1(\boldsymbol{x}) = 2x_1+2x_2+x_{10}+x_{11}-10\leq 0 \qquad (9)$$
$$g_2(\boldsymbol{x}) = 2x_1+2x_3+x_{10}+x_{12}-10\leq 0 \qquad (10)$$
$$g_3(\boldsymbol{x}) = 2x_2+2x_3+x_{11}+x_{12}-10\leq 0 \qquad (11)$$
$$g_4(\boldsymbol{x}) = -8x_1 + x_{10} \leq 0 \qquad (12)$$
$$g_5(\boldsymbol{x}) = -8x_2 + x_{11} \leq 0 \qquad (13)$$
$$g_6(\boldsymbol{x}) = -8x_3 + x_{12} \leq 0 \qquad (14)$$
$$g_7(\boldsymbol{x}) = -2x_4 - x_5 + x_{10} \leq 0 \qquad (15)$$
$$g_8(\boldsymbol{x}) = -2x_6 - x_7 + x_{11} \leq 0 \qquad (16)$$
$$g_9(\boldsymbol{x}) = -2x_8 - x_9 + x_{12} \leq 0 \qquad (17)$$

設計変数：

$$0 \leq x_i \leq 1 (i = 1, \ldots, 9),$$
$$0\leq x_i\leq 100 (i=10,11,12),\ 0\leq x_{13}\leq 1. \qquad (18)$$

表 1 に算出した探索回数を示す．ここで示した探索回数は，再度探索（初期温度 $T_0 = 1 \times 10^{-10}$）した際の回数である．

表 1 重み更新頻度の違いによる最適解を得る繰返し探索回数
Table 1 Iteration number in order to get the optimal solution as parameters of weights update frequency.

重み更新頻度	探索回数
30 回ごと	6,744
50 回ごと	5,353
100 回ごと	4,142
200 回ごと	4,312
300 回ごと	5,269
400 回ごと	4,956

重み更新頻度が高い場合は，最小化している目的関数が頻繁に変化している，つまり最適解が変化していることを意味しており，最適解への到達が遅くなっていると考えられる．対象とする問題にも依存すると考えられるが，今回の検証では 200 回探索するごとに重みを更新することとした．

上記テスト関数における探索に必要な計算時間も計測した．探索に要する総計算時間は，使用する計算機環境（本検証では Pentium4 3 GHz を使用）や最適化アルゴリズムにも依存する．たとえば，4,312 回探索するのに要する時間は，固定重みでの 2 秒に対して提案手法は 145 秒であり，約 2 分 20 秒追加時間が必要となった．また，20,000 回探索するのに要する時間は，固定重みでの 7 秒に対して提案手法は 1,528 秒であり，追加時間は約 25 分であった．今回は過去の関数値をファイルの入出力で対応しており，コード化を工夫することで高速化が実現できると考える．

本手法は，過去の関数値を用いた重み付けであるため，重み更新頻度を高くすると過去の関数値数が増大し計算時間を要する．また，近似曲線を決定する際に使用する過去の関数値数を制限することで計算時間を短縮することも可能である．ただし，過去の関数値数の増大は更新する重みを徐々に固定する機能を果たすため，計算時間短縮を目的として過去の関数値数を削減することは，本手法の狙いを達成しない可能性があるため注意を要する．なお，本手法による追加時間は，制約条件数にほぼ比例し，設計変数の数には依存しない．

4. 提案手法の有効性の検証

4.1 提案手法による重みと関数値の推移

本節では，ペナルティ関数に乗じた重みおよび評価コスト関数値の推移を確認する．対象はテスト関数 $g01$ である．

ペナルティ関数は 9 個あり，重みをそれぞれ $w_1, w_2, \cdots, w_9$ としている．重みは 200 回探索するごとに更

表 2 ベンチマーク結果
Table 2 Benchmark test results.

関数# \ 手法	理論解	提案手法	平均値による重み	固定重み	PSO [12]
g01(最小化)	−15.000000	−15.000000	−15.000000	−15.000000	−15.0
g02(最大化)	0.803619	0.801036	0.798667	0.800786	0.7130
g04(最小化)	−30665.539	−30665.480371	−30665.341697	−30640.098934	−30665.5
g06(最小化)	−6961.81388	−6928.352727	−6311.215192	−6155.029455	−6961.7
g07(最小化)	24.306	24.651254	24.988766	24.809110	24.4420
g08(最大化)	0.095825	0.095825	0.095825	0.095825	0.09583
g09(最小化)	680.630	680.739309	681.007520	681.991025	680.657
g10(最小化)	7049.331	7101.708902	7256.727399	7347.159966	7131.01

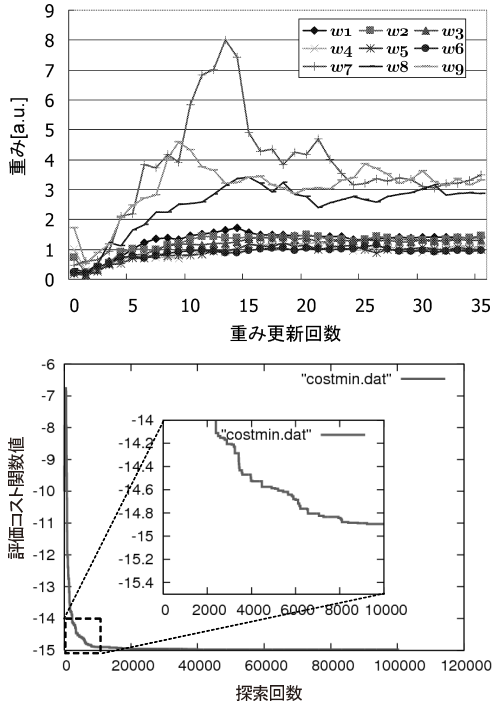

図 3 テスト関数 $g01$ での重み（上）と関数値（下）推移
Fig. 3 Transition of the weights (top) and the values of the test function $g01$ (bottom).

新する．つまり，**図 3** (上) には 7,000 回探索（35 回重み更新）まで各重みをプロットしている．また，得られた目的関数の最小化によって，制約条件を満足しながら評価コスト関数値を最小化していく推移を図 3 (下) に示す．プロットは各探索回数までの評価コスト関数最小値である．探索回数が 7,000 回近傍までは重みが変化し評価コスト関数値も急激に減少しているが，重みの振動が抑えられた後は緩やかに減少している．

4.2 ベンチマークテスト結果

提案手法による解が大域的最適解（理論解）にどの程度近づいているかを，テスト関数[10] を用いて検証した．比較対象として，1. 提案手法のように近似曲線から重みを算定せず，最近の関数値 200 点（$(1, f_{n-199}(\boldsymbol{x})), \cdots, (200, f_n(\boldsymbol{x}))$ および $(1, p_{k,n-199}(\boldsymbol{x})), \cdots, (200, p_{k,n}(\boldsymbol{x}))$）の平均値から算定した重みと，2. すべて固定重み 1 とした場合の 2 ケースの結果も併記する．最近の 200 点を採用する理由は，重み更新時の関数代表値を求めるためである．重みの更新頻度は提案手法と同じとする．テスト関数[10] を用いて得られる解を，理論解とあわせて **表 2** に示す．また，比較のため制約条件を考慮した Particle Swarm Optimization（PSO）[12] による結果を併記した．Particle 数を 20，反復回数 500 回とし，20 回実施した最良解を示す．ペナルティ関数を改良するのみで，近年開発された最適化アルゴリズムで得られる結果と同等程度の結果を得ており，提案する目的関数作成手法は有効に作用しているといえる．

提案手法および平均値を用いた手法と固定値（= 1）を用いた手法の 3 手法とも理論解もしくは近傍解を得ており，制約条件がそれほど厳しくない問題が含まれていたが，理論解から離れた解となっているものもある．以下では，テスト関数 $g02$，$g06$，$g10$ について考察する．

テスト関数 $g02$ は，周期的な非線形関数であり，数学的な理論解は知られていない（Keane, 1994）．本論文では文献 11) で示されている解を理論解としている．この関数は，**図 4**（ただし $n = 2$ とした）に示すような振動する関数値を持っており（多峰性関数であり），局所解に陥りやすい典型例である．

Maximize:

$$f(\boldsymbol{x}) = \left| \frac{\sum_{i=1}^{n} \cos^4(x_i) - 2\prod_{i=1}^{n} \cos^2(x_i)}{\sqrt{\sum_{i=1}^{n} i x_i^2}} \right| \tag{19}$$

subject to:

$$g_1(\boldsymbol{x}) = 0.75 - \prod_{i=1}^{n} x_i \leq 0 \tag{20}$$

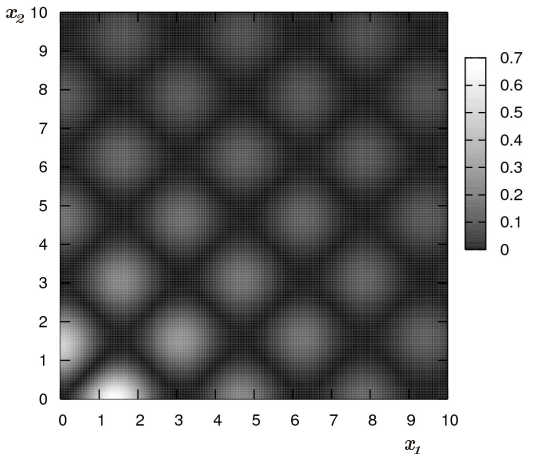

図 4 テスト関数 $g2$（$n = 2$）の等高線図
Fig. 4 Contour of the test function $g2$ ($n = 2$).

$$g_2(\boldsymbol{x}) = \sum_{i=1}^{n} x_i - 7.5n \leq 0 \tag{21}$$

設計変数：

$$n = 20, 0 \leq x_i \leq 10 (i = 1, \ldots, n). \tag{22}$$

本手法によって，$\boldsymbol{x} = (3.202508, 3.106059, 3.114214, 3.057234, 3.022873, 2.992847, 2.949357, 2.928433, 0.475081, 0.534223, 0.569699, 0.526425, 0.387949, 0.436212, 0.485470, 0.474745, 0.447031, 0.405011, 0.426908, 0.441689)$ において評価コスト関数値 0.801036 を得た．

今回得られた解は表 2 に示す理論解に到達していないが，その原因として以下の 2 点が考えられる．

第 1 点は，アルゴリズムに単探索の SA を用いており，そのパラメータ初期値依存性によって理論解に至っていない点である．SA での初期温度を上げることは対策の 1 つではあるが，上げすぎても探索が安定しない．多点探索はこのような問題には有効であるが，単点探索の早い解の探索の特徴を活かすため，3.1 節に示すように得られた解を初期値とした繰返し探索を一連の探索工程に取り入れた．

第 2 点はテスト関数が特徴的であり，期待する重み付けが困難である点である．提案する自動重み付け手法では，評価コスト関数 $f(x)$ とペナルティ関数 $g(x)$ の近似曲線上の各代表点の比を重みに代用している．このため，提案手法は，設計変数の変動範囲において各近似曲線が長い周期で変動する場合に限り，適切な重みを決定できる．しかし，各関数が短い周期で変動する場合には，滑らかな近似曲線はその振動に追随できず，近似曲線上の点が代表点とはならないことは自明である．よって，初期設計変数が最適解近傍（図 4 の原点近傍）であれば，単調増加もしくは減少で最適解を見つけることができるが，それ以外では短い周期で振動するため，適切な重みを設定できないと考えられる．

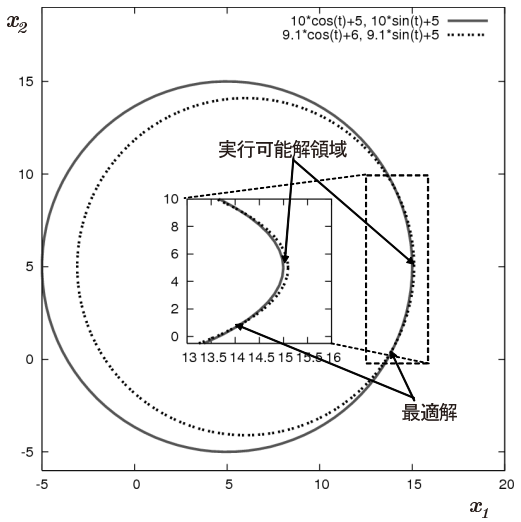

図 5 テスト関数 $g6$ の実行可能領域
Fig. 5 Feasible area of the constraints of $g6$.

テスト関数 $g06$ は多項式関数であるが，制約条件から，実行可能領域は**図 5** に示す狭い領域のみとなっている（Floudas and Pardalos，1987）．

Minimize:

$$f(\boldsymbol{x}) = (x_1 - 10)^3 + (x_2 - 20)^3 \tag{23}$$

subject to:

$$g_1(\boldsymbol{x}) = -(x_1 - 5)^2 - (x_2 - 5)^2 + 100 \leq 0 \tag{24}$$

$$g_2(\boldsymbol{x}) = (x_1 - 6)^2 + (x_2 - 5)^2 - 82.81 \leq 0 \tag{25}$$

設計変数：

$$13 \leq x_1 \leq 100, 0 \leq x_2 \leq 100. \tag{26}$$

つまり，実行不可能領域を探索する場合が多くなり，ペナルティ関数値が非ゼロとなる場合が多い．このような場合，平均値を用いた重みではそれ以前の関数値が反映されないため更新ごとに重みが変化する可能性が高くなると考えられる．そこで，**図 6** に提案手法による重みと平均値による重みを，重み更新回数 40 回以降においてプロットした．

平均値による重みでは振動が抑えられず，評価対象である目的関数が変化してその最適解が変動しており，この効果は得られる解の質に現れている．これに対し，提案手法では探索序盤の関数値も含めてその点列の近似曲線を算出して重みを決定しているため，重みの変動はより滑らかであり，目的関数の最適解の劇的な変化を抑制することができる．これにより，本来の目的である評価コスト関数の最小値に近い解を得ることが

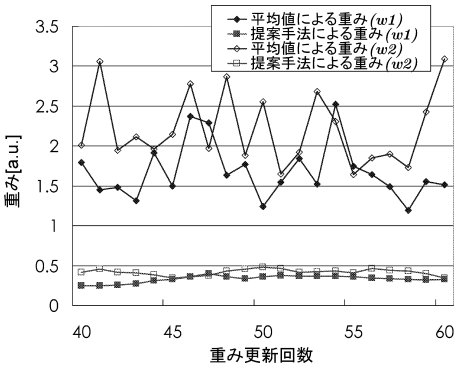

図 6 テスト関数 $g06$ における重みの変動量比較
Fig. 6 Fluctuations of the weights of test function $g06$ by proposal and average method.

できた．得られた解は，$\boldsymbol{x} = (14.109725, 0.872724)$ で，このときの評価コスト関数値は -6928.352727 である．

テスト関数 $g10$ は，最小化したい評価コスト関数は線形関数であり，制約条件は線形，非線形の混在した問題である（Hock and Schittkowski，1981）．

Minimize:

$$f(\boldsymbol{x}) = x_1 + x_2 + x_3 \tag{27}$$

subject to:

$$g_1(\boldsymbol{x}) = -1 + 0.0025(x_4 + x_6) \leq 0 \tag{28}$$

$$g_2(\boldsymbol{x}) = -1 + 0.0025(x_5 + x_7 - x_4) \leq 0 \tag{29}$$

$$g_3(\boldsymbol{x}) = -1 + 0.01(x_8 - x_5) \leq 0 \tag{30}$$

$$g_4(\boldsymbol{x}) = -x_1 x_6 + 833.33252 x_4 + 100 x_1 - 83333.333 \leq 0 \tag{31}$$

$$g_5(\boldsymbol{x}) = -x_2 x_7 + 1250 x_5 + x_2 x_4 - 1250 x_4 \leq 0 \tag{32}$$

$$g_6(\boldsymbol{x}) = -x_3 x_8 + 1250000 + x_3 x_5 - 2500 x_5 \leq 0 \tag{33}$$

設計変数：

$$100 \leq x_1 \leq 10000,\ 1000 \leq x_i \leq 10000 (i = 2, 3),$$
$$10 \leq x_i \leq 1000 (i = 4, \ldots, 8). \tag{34}$$

評価コスト関数には，3 つの設計変数しか用いられておらず，8 つの設計変数は制約条件によって依存性がある．このため，評価コスト関数に含まれる 3 つの設計変数を決定してその関数値を得たとしても，依存する他の 8 つの設計変数が制約条件を満足しない（infeasible）場合が考えられる．また，SA では，改悪する設計変数についても，温度に依存する受け入れ確率で最適解となりうると判断するため，解となりえない設計変数領域を探索して大域的最適解に至らない可能性がある．

本検証ではまず，ペナルティ法での重みを評価コスト関数値とペナルティ関数値との比が 1:1 となる重み付けで探索を実施した．得られた解は，$\boldsymbol{x} = (632.003, 5988.487, 1273.769, 124.888, 962.369, 264.997, 400.296, 383.069)$ において評価コスト関数値は 7894.259463 となった．この解は制約条件を満足していない．

そこで，前述したように制約条件を満足しないペナルティ関数にさらに定数を乗じた．これにより，$\boldsymbol{x} = (338.746868,\ 1678.268754,\ 5084.693280, 157.802650,\ 296.612303,\ 242.196880,\ 261.190189, 396.612294)$ において評価コスト関数値 7101.708902 と理論解近傍の実行可能解を得た．なお，他の 2 手法においても同じ定数を乗じている．

4.3 考　　察

本検証では，探索範囲の中心値を初期値とした単点探索を実施したが，得られる解の初期値依存性が高い場合，最適解が得られない可能性がある．ここでは，理論解から離れた解となったテスト関数 $g06$ について，初回の探索（初期温度 $T_0 = 1.0$）で得られた解近傍の設計変数をランダムに初期値として与え，多点探索を実施した．その結果，$\boldsymbol{x} = (14.0951393, 0.8432535)$ において，評価コスト関数値は -6961.484605 とさらに理論解に近づいた．1 章で述べたように，工学機器設計において特に軽微な変更を実施する場合には，良好な初期値がすでに存在するため単点探索による早い探索のメリットは大きい．ただし，広範な探索を実施する場合には複数の初期値を与え，初期値依存性にとらわれていないか確認する必要がある．

どんな問題に対しても大域的最適解を得るアルゴリズムは存在しない．しかし，特に工学分野においては，数学的に最適解でなくても，現状を改善した解が得られれば目的が達成される場合が多い．また，設計変数も同時に出力されているから，実行可能解であるかどうかは実際に制約対象を計算すれば容易に判断できる．

5. ま と め

本論文では，ペナルティ法を用いた目的関数の作成手法を提案し，テスト関数を用いてその性能を評価して提案手法の有効性を示した．提案する目的関数は簡素であるものの，局所解に陥らないガイドとして機能し，最適解もしくはその近傍解を探索できる．目的関数は，単探索の単目的最適化アルゴリズムにそのまま適用できるため，GA などの多探索アルゴリズムに比して短時間探索であり，汎用の単目的最適化ツールを利用できる．

今回は，単探索で単目的最適化アルゴリズムとして

SA を扱った．探索が進行するごとに探索範囲が狭まる（温度が低下する）探索において，特に制約条件を満足させる度合いに差がある場合には，必須以外の制約条件の満足に固執した探索は避けなければならない．特定の関数に偏った最小化を避けるために，評価コスト関数およびペナルティ関数はほぼ等しい値となるように重みを調整すべきである．本手法は，平滑化自然スプラインを用いた，滑らかに変動する重み付けを自動で行うことが可能で，目的関数の最適解の劇的な変化を抑制できる．これにより，探索序盤からすべての制約条件を考慮しながら，評価コスト関数を最小化する．

参 考 文 献

1) Parmee, I.: *The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realisation*, Springer-Verlag, Plymouth, UK (1998).
2) Richardson, J.T., Palmer, M.R., Liepins, G. and Hilliard, M.: Some guidelines for genetic algorithms with penalty functions, *Proc. 3rd Internaional Conference in Genetic Algorithms*, pp.191–197 (1989).
3) Aguirre, A.H., Rionda, S.B., Coello, C.A., Lizarraga, G. and Montes, E.M.: Handling Constraints using Multiobjective Optimization Concepts, *Int. J. Numer. Meth. Engng.*, Vol.59, No.15, pp.1989–2017 (2004).
4) Surry, P.D. and Radcliff, N.J.: The COMOGA Method: Constrained Optimisation by Multiobjective Generic Algorithms, *Control and Cybernetics*, Vol.26, No.3, pp.391–412 (1997).
5) Schaffer, J.D.: Multiple Objective Optimization with Vector Evaluated Genetic Algorithms, *Genetic Algorithms and their Applications, Proc. 1st International Conference on Genetic Algorithms*, Lawrence Erlbaum, pp.93–100 (1985).
6) Fonseca, C.M. and Fleming, P.J.: Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization, *Proc. 5th International Conference on Genetic Algorithms*, San Mateo, California, University of Illinois at Urbana-Champaign, Forrest, S. (Ed.), pp.416–423, Morgan Kauffman Publishers (1993).
7) Horn, J., Nafpliotis, N. and Goldberg, D.E.: A Niched Pareto Genetic Algorithm for Multiobjective Optimization, *Proc. 1st IEEE Conference on Ecolutionary Computation*, IEEE World Congress in Computational Intelligence, Vol.1, Piscateway, NJ, pp.82–87 (1994).
8) Schoenberg, I.J.: Spline functions and the problem of graduation, *Proc. National Academy of Sciences of the U.S.A.*, 52, pp.947–950 (1964).
9) Ingber, L.: Adaptive Simulated Annealing (ASA), [ftp.alumni.caltech.edu: /pub/ingber/ ASA-shar, ASA-shar.Z, ASA.tar.Z, ASA.tar.gz, ASA.zip], Lester Ingber Research, McLean, VA (1993).
10) Michalewicz, Z. and Schoenauer, M.: Evolutionary Algorithms for Constrained Parameter Optimization Problems, *Evolutionary Computation*, Vol.4, No.1, pp.1–32 (1996).
11) Runarsson, T.P. and Yao, X.: Stochastic Ranking for Constrained Evolutionary Optimization, *IEEE Trans. Evolutionary Computation*, Vol.4, No.3, pp.284–294 (2000).
12) Hu, X. and Eberhart, R.C.: Solving constrainted nonlinear optimization problems with particle swarm optimization, *Proc. 6th World Multiconference on Systemics, Cybernetics and Informatics* (2002).

(平成 19 年 1 月 18 日受付)
(平成 19 年 5 月 8 日再受付)
(平成 19 年 7 月 2 日採録)

村田 幸弘

東京大学大学院工学系研究科修了．現在，（株）日立製作所研究員．主に電磁場解析に関する研究に従事．博士（工学）．電気学会会員．

安藤 竜弥

慶應義塾大学大学院理工学研究科修了．現在，（株）日立製作所研究員．主に電磁場解析に関する研究に従事．電子情報通信学会会員．

阿部 充志

京都大学大学院工学研究科修了．現在，（株）日立製作所主任研究員．主に電磁場解析に関する研究に従事．APS，プラズマ核融合学会各会員．