Sequential Approximate Optimization using RBF network (basic examination on the sampling function)

RBFネットワークによる逐次近似最適化
（サンプル関数の基礎的検討）

北山哲士[*1]　荒川雅生　山崎光悦

Sequential Approximate Optimization Using RBF Network
(Basic Examination on the Sampling Function)

Satoshi KITAYAMA, Masao ARAKAWA, Koetsu YAMAZAKI
College of Science and Engineering, Kanazawa University
Kakuma-machi, Kanazawa, 920-1192, Japan

One of the important issues on the Sequential Approximate Optimization (SAO) is the sampling strategy. The sampling strategy for SAO using the Radial Basis Function (RBF) network is proposed in this paper. The proposed sampling strategy consists of three parts, which are called the density function, the boundary function, and random sampling. In order to add the new sampling points effectively, the density function and the boundary fuction are constructed by the RBF network. The objective of the density function is to find the sparse region in the design variable space and is to add the new sampling points in this region. In the constrained optimization problems, at least, one or more constraints will be active. As the result, it is desirable to add the new sampling points on the constratins. The objective of the boundary function is to add the new sampling points on the boundary. In addition, the random sampling is also introduced to spread the search region. The algorithm of proposed sampling strategy is described in detail. Through the numerical examples, the validity is examined.

Key Words **: Engineering Optimization, Optimal Design, Sequential Approximate Optimization, RBF Network, Global Optimization, System Engineering**

1. 緒言

近年の最適化手法の進展により，実務レベルの設計問題に対して最適設計を行うことが身近なものになりつつある．最適化手法の進展の背景にはコンピュータの飛躍的な発達があり，特に遺伝的アルゴリズム(GA)やParticle Swarm Optimization(PSO)などに代表されるメタヒューリスティクスと称される方法を直接的に適用して最適設計を行うことも可能にしている．しかし，限られた時間という制約の下で，数理計画法に代表される関数の感度（勾配）を用いる方法やメタヒューリスティクスを直接的に用いて最適設計を行う場合，基本的には1回のシミュレーションに要する計算コストの低い設計問題に対して有効であると考える方がよい．また実験を基調とするような設計問題を対象とする場合などは，各種最適化手法を直接的に適用して最適設計を行うことは困難である．つまり，実務レベルの設計問題に対して最適化を行う場合の重要な課題は，ファンクションコール（シミュレーション回数や実験回数）を減らし，厳密ではないものの近似的な最適解を求めるという点にある．

そこで，ファンクションコールを減らし，厳密でなくとも近似的な最適解を求める方法の一つとして，応答曲面を用いた最適設計が挙げられる．この方法では，実質的なファンクションコールは，応答値を算出するために必要なサンプル点の数だけとなる．特に実験計画法に基づく応答曲面を用いた最適設計では，直交表を用いて効率的に実験を行うことが可能であり，その有効性が広く知られている[(1),(2)]．しかし，実験計画法に基づく応答曲面の場合，本質的な問題は設計変数の水準値の設定方法，すなわちサンプル点の配置方法であり，また関数空間を基本的には二次近似するため，その有効性は非線形性の弱い範囲までと考えるのが妥当である．応答曲面の次数を増やすことにより，多峰性関数を近似することも可能であるが，その際にはいくつかの注意が必要である[(3)]．

一方，非線形性の弱い関数から多峰性関数まで幅広く良好に近似する方法として，KrigingやRBFネットワークが挙げられる[(4-9)]．特にRBFネットワークの場合は，計算の手軽さや応答値に誤差が含まれるような問題にも柔軟に対応でき，実験を基調とする最適設計問題にも対応できる有力な方法の一つであると考えられる．また実験計画法に基づく応答曲面法と大きく異なるのは，KrigingやRBFネットワークでは，図1に示すような，サンプル点を逐次追加することにより段階的に

* 原稿受付　平成？？年？月　日
*1正員，金沢大学理工研究域（〒920-1192　金沢市角間町）．
*2正員，香川大学工学部（〒761-0396　高松市林町2217-20）．
E-mail: kitagon@t.kanazawa-u.ac.jp

応答曲面の精度を向上させ，精度の高い最適解を求める，いわゆる逐次近似最適化が可能な点である[10]．

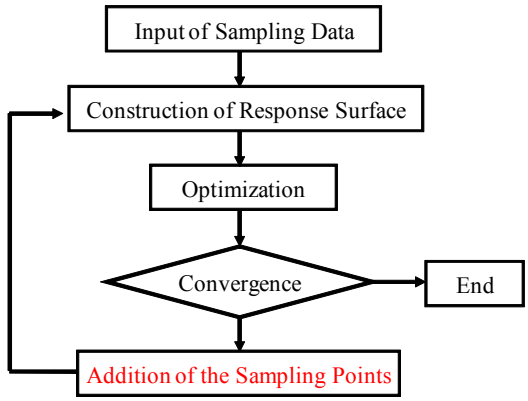


Fig.1 The flow of sequential approximate optimization

KrigingやRBFネットワークを用いて逐次近似最適化を行う場合，サンプル点をどのように配置すればよいかという点に関しては，現在の重要な研究課題の一つであり，精力的に研究が行われている[8],[11-15]．これらの研究において，重要とされていることをまとめると，次のようになる．

（1）設計変数空間におけてサンプル点の疎な領域を見つけ，その領域内にサンプル点を追加する．これは，局所的最適解に捕捉されず，大域的最適解を見つけるために行われる．

（2）応答曲面の最適解を追加サンプル点としたり，最適解の近傍にサンプル点を追加する．これは応答曲面の局所的精度の向上を狙い行われる．仮に大域的最適解が見つからなくとも，精度の高い次善の局所的最適解であることを期待して行われる．

筆者らは，RBFネットワークを用いた逐次近似最適化において，中山らの方法[8]を改良し，上記2点を同時に達成する方法を提示した[16]．しかし，中山らの方法も筆者らの方法も，基本的にはランダム性に大きく依存した方法であることは否めない．また，制約条件は基本的にペナルティ関数として扱っていたため，実行可能領域と非実行可能領域の境界を明確にすることはできなかった．有制約最適化問題では，多くの場合，一つ以上の制約条件がアクティブになることを考えれば，目的関数と制約条件の応答曲面を個別に作成し，実行可能領域と非実行可能領域の境界を明確にすることは有益であると考えられる．

これらを踏まえ，RBFネットワークを用いた逐次近似最適化における研究課題をまとめると，次のようになると思われる．

（P1）設計変数空間におけるサンプル点の疎な領域を確定的に見つける方法．

（P2）実行可能領域と非実行可能領域の境界を明確にする方法．制約条件毎に応答曲面を作成し，サンプル点の実行可能性の有無を判断し，その結果から境界を明確にし，境界上付近にサンプル点を配置することにより，制約条件の近似精度，さらには最適解の精度が向上すると期待できる．

（P3）ランダム性の考慮．ある程度のランダム性を考慮することで，サンプル点配置のばらつきを持たせる．また外挿領域へサンプル点を配置することも可能となり，探索空間の拡大へつながる．

なお，著者らの一人は，サンプル点配置の指針を与える推奨関数と称する関数を提案しているが[17]，この関数はサンプル点の粗密を判断する関数と実行可能性を判断する関数を掛け合わせていることから，必ずしも上記(P1),(P2)を個別に満足する方法ではない．

そこで本論文では，RBFネットワークを用いて上記(P1),(P2)を満足すると考えられるサンプル関数を構築する．具体的な方法として，(P1)に対しては，密度関数と称する関数を構築し，密度関数を最小にする解を追加サンプル点とする．また(P2)に対しては，境界関数と称する関数を構築し，境界関数を最小とする解を追加サンプル点とする．数値計算例を通じて，本論文で提示する方法の有効性の基礎的検討を行う．

2. 問題設定とRBFネットワーク

2.1 問題設定　本論文では，次の問題を扱う．

$$f(\boldsymbol{x}) \to \min \tag{1}$$

$$g_k(\boldsymbol{x}) \le 0 \quad k = 1,2,\cdots,ncon \tag{2}$$

$$x_I^L \le x_I \le x_I^U \quad I = 1,2,\cdots,n \tag{3}$$

$f(\boldsymbol{x})$ は最小化する目的関数であり，$g_k(\boldsymbol{x})$ は挙動制約条件である．また $ncon$ は挙動制約条件の数を表す．x_I は I 番目の設計変数であり，x_I^L と x_I^U はそれぞれ側面制約条件の下限値と上限値を表す．n は設計変数の数を表す．本論文では，式(1),(2)をRBFネットワークを用いてそれぞれ個別に近似する．

2.2 RBFネットワークの学習　RBFネットワークは3層型のフィードフォワード型のニューラルネットワークである．このネットワークでは基底関数として一般にガウス関数が用いられ，その重ね合わせにより応答曲面 $\hat{y}(\boldsymbol{x})$ を作成する．応答曲面は

$$\hat{y}(\boldsymbol{x}) = \sum_{j=1}^{m} w_j h_j(\boldsymbol{x}) \tag{4}$$

で表され，m は中間層素子数，w_i は重みを表す．また $h_j(\boldsymbol{x})$ は基底関数であり，次式で与えられる．

$$h_j(\boldsymbol{x}) = \exp\left(-\frac{(\boldsymbol{x}-\boldsymbol{x}_j)^T(\boldsymbol{x}-\boldsymbol{x}_j)}{r_j^2}\right) \tag{5}$$

上式において $\boldsymbol{x}_j$ と r_j はそれぞれ j 番目の基底関数の中心と半径である．学習用データ $\boldsymbol{x}_i$ と対になる教師データ y_i （$i=1,2,\cdots,p$）とすると，RBFネットワークにおける学習は次式を最小化する問題となる．

$$E=\sum_{i=1}^{p}(y_i-\hat{y}(\boldsymbol{x}_i))^2+\sum_{j=1}^{m}\lambda_j w_j^2\rightarrow \min \qquad (6)$$

式(6)において，第1項はネットワークの出力値と教師データの誤差の二乗和であり，第2項の λ_j は一部の素子だけが過剰に反応するのを避けるための重みに対するパラメータであり，これによりデータに含まれるノイズの影響を抑制するとともに，学習過程における正規性を保証している．RBFネットワークの学習とは，式(6)を満足する重みベクトル $\boldsymbol{w}$ を見つけることであり，次の式により求まる．

$$\boldsymbol{w}=(\boldsymbol{H}^T\boldsymbol{H}+\boldsymbol{\Lambda})^{-1}\boldsymbol{H}^T\boldsymbol{y} \qquad (7)$$

ここで上式の $\boldsymbol{H}$，$\boldsymbol{\Lambda}$，$\boldsymbol{y}$ はそれぞれ

$$\boldsymbol{H}=\begin{bmatrix} h_1(\boldsymbol{x}_1) & h_2(\boldsymbol{x}_1) & \cdots & h_m(\boldsymbol{x}_1)\\ h_1(\boldsymbol{x}_2) & h_2(\boldsymbol{x}_2) & \cdots & h_m(\boldsymbol{x}_2)\\ \vdots & \vdots & \ddots & \vdots\\ h_1(\boldsymbol{x}_p) & h_2(\boldsymbol{x}_p) & \cdots & h_m(\boldsymbol{x}_p)\end{bmatrix} \qquad (8)$$

$$\boldsymbol{\Lambda}=\begin{bmatrix} \lambda_1 & 0 & \cdots & 0\\ 0 & \lambda_2 & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & 0 & \lambda_m\end{bmatrix} \qquad (9)$$

$$\boldsymbol{y}=(y_1,y_2,\cdots,y_p)^T \qquad (10)$$

で与えられる．さらに式（7）において，

$$\boldsymbol{A}=\boldsymbol{H}^T\boldsymbol{H}+\boldsymbol{\Lambda} \qquad (11)$$

とおけば，RBFネットワークの学習は，$\boldsymbol{A}^{-1}$ を求めることに帰着される．さらにその性能を特徴付けるものとして，追加学習と忘却が存在する．特に追加学習はマトリックスの操作のみで簡単に行うことができ，逐次近似最適化が容易となる．以降では，サンプル点の数 p と中間層素子数 m は同じとする．また基底関数の中心はサンプル点上におくものとする．そのため，サンプル点が追加されれば，基底関数の数が増加し，応答値の計算も行うことになる．

2.3　半径式について　筆者らは式(5)中の r_j を簡易的に決定するために，中山らの式[8]を参考に新たな半径式を提案した[16]．

$$r_j=\frac{d_{j,\max}}{\sqrt{n}\sqrt[n]{m-1}} \qquad (12)$$

ここで，n は設計変数の数，m はサンプル点数，$d_{j,\max}$ は j 番目のサンプル点からの最大距離を表している．式（12）は各基底関数に適用され，サンプル点の疎密が生じるような場合に有効である．なお，RBFネットワークやサポートベクターマシン等において，基底関数にガウス関数がよく用いられているが，ガウス関数の半径は非常に重要なものである．この半径の決定に際し，サンプル点のばらつきを考慮し，簡易的に半径を決定する研究は筆者の調べる範囲では見当たらない．また，次節で述べる適応的スケーリングと式(12)を併用することが極めて重要である．

2.4　適応的スケーリング　式（12）の半径式では，次のような条件を満足する必要があった[16]．

（R1）サンプル点の数を考慮すること．
（R2）サンプル点間の距離を考慮すること．
（R3）設計変数の数を考慮すること．
（R4）サンプル点数が増加するにつれ，設計変数の数に依らず，ある同じ値(すなわち1)に収束すること．

上記の条件を導く際，最も重要なことは，はじめに各設計変数を適切にスケーリングし，設計変数空間のゆがみを修正することである．I 番目の設計変数 x_I をスケーリングするために，以下の式を用いる．

$$X_I=\frac{x_I-x_I^L}{x_I^U-x_I^L}\times s \quad I=1,2,\cdots,n \qquad (13)$$

ここで $s(>0)$ はスケーリング係数を表す．式(13)によってスケーリングされた設計変数空間において式(12)を計算する．ただし，スケーリング係数を適切に設定しなければ，（R4）を満足しなくなる可能性がある．そこで，以下に示すような適応的スケーリングを導入する．具体的なアルゴリズムを以下に示す．

(STEP1)スケーリング係数 $s(>0)$ を設定．
(STEP2)式(13)による設計変数のスケーリング．
(STEP3)スケーリングされた設計変数空間において，式(12)による半径値を算出．
(STEP4)半径値の最小値 $r_{\min}$

$$r_{\min}=\min_{1\le j\le m}\{r_j\} \qquad (14)$$

を見つける．$r_{\min}\le 1$であれば，

$$s=\alpha\times s \quad (\alpha>1) \qquad (15)$$

として，STEP2へ戻る．そうでなけば，終了．

本論文では，初期スケーリング係数を $s=1$，式(15)中の α を $\alpha=1.2$ としている．

3.　サンプル関数

本章では，RBFネットワークを用いた密度関数および境界関数の構築方法について記述する．なお，以下の議論では，適応的スケーリングにより設計変数はすべてスケーリングされているものとする．

3.1　密度関数　設計変数空間内のサンプル点の疎な領域を確定的に見つけることが密度関数の目的で

ある．そこで，サンプル点の疎な領域に極小値を生成し，その最小値を取る点を，設計空間内におけるサンプル点の疎な点とする．このような性質を持つ関数をRBFネットワークで作成することを考える．

そこで，サンプル点の存在する位置に常に+1をネットワークの出力としてを与え，サンプル点の有無を学習させ，設計変数空間におけるサンプル点の疎密を判断させることとした．サンプル点が m 個ある場合の具体的な流れを以下に示す．

(D-STEP1) 式(10)のすべての成分を+1とした $m\times 1$ ベクトル

$$\boldsymbol{y}^D=(1,1,\cdots,1)^T \tag{16}$$

を用意する．

(D-STEP2) 密度関数の重み $\boldsymbol{w}^D$ を以下の式から求める．

$$\boldsymbol{w}^D=(\boldsymbol{H}^T\boldsymbol{H}+\boldsymbol{\Lambda})\boldsymbol{H}^T\boldsymbol{y}^D \tag{17}$$

(D-STEP3) 密度関数 $D(\boldsymbol{x})$ を最小化する．

$$D(\boldsymbol{x})=\sum_{j=1}^{m}w_j^D h_j(\boldsymbol{x})\rightarrow \min \tag{18}$$

そして，求めた最適解を追加サンプル点とする．

なお，密度関数 $D(\boldsymbol{x})$ の作成に関し，基底関数の半径には式(12)を用いる．

密度関数がサンプル点の疎な領域で極小値を生成する理由は，ネットワークの出力値が一定値に抑えているため，重み $\boldsymbol{w}^D$ に影響を与えるのはサンプル点間の距離となり(注)，その結果，サンプル点間の距離が大きければ大きいほど，密度関数は，サンプル点間の距離が大きな領域で，より小さな極小値を生成するようになる．密度関数の一例を図2に示す．図2中の●はサンプル点を表している．

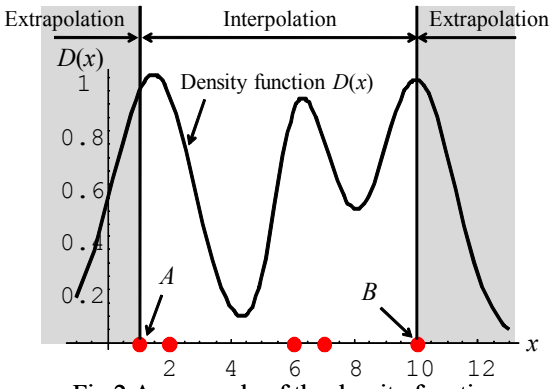


Fig.2 An example of the density function

なお，基底関数の半径値が適切でなければ，図2のようなサンプル点の疎な領域に極小値は生成されないことに注意されたい．図2のような関数を作成する際は，適用的スケーリングにより設計変数空間をスケーリングしておき，その後式(12)により基底関数の半径を決定する．

注： 式(12)は，与えられたデータから半径値を簡易的に決めるものであるため，密度関数に影響を与える要素は式(5)の分子（すなわちサンプル点間の距離）となる．

3.2 境界関数

与えられたサンプル点のみから実行可能領域と非実行可能領域の境界を判別することは，極めて困難である．しかし有制約最適設計問題では，一般に一つ以上の制約条件がアクティブになることを考えれば，実行可能領域と非実行可能領域の境界を明確にし，さらに境界上付近にサンプル点を追加することは重要であると考えられる．そこで，以下のようにして境界関数 $B_k(\boldsymbol{x})$ ($k=1,2,\cdots,ncon$) を作成する．境界関数は，すべての制約条件に対して作成する．

(B-STEP1) k 番目の制約条件 $g_k(\boldsymbol{x})$ を対象とする．m 個のサンプル点の中で，k 番目の制約条件の最大値 $g_{k,\max}$ を見つける．

$$g_{k,\max}=\left|g_k(\boldsymbol{x}_i)\right| \quad i=1,2,\cdots,m \tag{19}$$

(B-STEP2) 次式により，k 番目の制約条件に対する境界関数 $B_k(\boldsymbol{x})$ のネットワークの出力 $\boldsymbol{y}_k^B$ を決める．

$$\boldsymbol{y}_k^B=(y_{1,k}^B,y_{2,k}^B,\cdots,y_{m,k}^B)^T \quad i=1,2,\cdots,m \tag{20}$$

ただし，ネットワークの出力 $\boldsymbol{y}_k^B$ の各成分は，

$$y_{i,k}^B=-1+\left|\frac{g_k(\boldsymbol{x}_i)}{g_{k,\max}(\boldsymbol{x}_i)}\right|\times 2 \quad i=1,2,\cdots,m \tag{21}$$

で与える．

(B-STEP3) k 番目の制約条件に対する境界関数 $B_k(\boldsymbol{x})$ の重み $\boldsymbol{w}_k^B$ を計算する．

$$\boldsymbol{w}_k^B=(\boldsymbol{H}^T\boldsymbol{H}+\boldsymbol{\Lambda})^{-1}\boldsymbol{H}^T\boldsymbol{y}_k^B \tag{22}$$

ただし，重み $\boldsymbol{w}_k^B$ の各成分は次の通りである．

$$\boldsymbol{w}_k^B=(w_{1,k}^B,w_{2,k}^B,\cdots,w_{m,k}^B)^T \tag{23}$$

(B-STEP4) k 番目の制約条件に対する境界関数 $B_k(\boldsymbol{x})$ の最小値を見つける．

$$B_k(\boldsymbol{x})=\sum_{j=1}^{m}w_{j,k}^B h_j(\boldsymbol{x})\rightarrow \min \tag{24}$$

そして，求めた最適解を k 番目の制約条件に対する追加サンプル点とする．

上記(B-STEP1)～(B-STEP4)を制約条件の数だけ，繰り返す．

ここで，境界関数に用いる半径式について注意が必要である．例えば図3に示すように，境界 $g_k(\boldsymbol{x})=0$ 付近に二つのサンプル点（●で表示）が存在し，その距離が近い場合を考える．なお，サンプル点は境界付近以外にも存在するものとする．

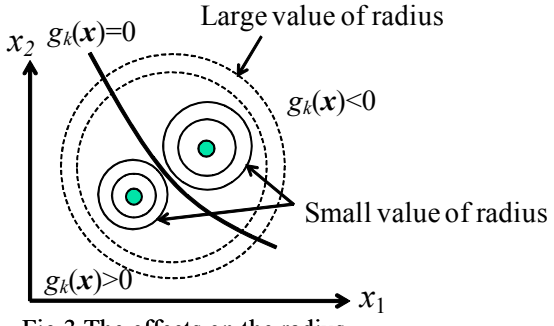


Fig.3 The effects on the radius

このとき，単純に式(12)を用いた場合，分子 $d_{j,\max}$ の影響により，境界関数の等高線は例えば図3中の破線のようになり，境界 $g_k(\boldsymbol{x})=0$ 付近に極小値を生成しないことがわかった．そのため，式(12)を用いて得られた値よりも小さな値にすることにより，凹凸の激しい関数を作成することが可能となる．例えば図3中の実線で示すような境界関数の等高線が得られ，境界付近に極小値を生成させることが期待できる．しかし，適切な半径値の設定は極めて困難である．そこで本論文では，境界関数に用いる基底関数の半径式に，以下の式を用いた．

$$r_j = \frac{1}{2}\frac{d_{j,\max}}{\sqrt{n}\sqrt[n]{m-1}} \tag{25}$$

3．3　密度関数および境界関数の側面制約条件

応答曲面が基本的にはサンプル点間を内挿する関数であることを考慮すれば，サンプル点の存在しない外挿領域を含めて考えることは避けるほうが望ましい（図2参照）．そこで，密度関数および境界関数の側面制約条件は次のように与えた．

$$\hat{x}_I^L = \min_{1\le i\le m} x_{i,I} \quad I=1,2,\cdots,n \tag{26}$$

$$\hat{x}_I^U = \max_{1\le i\le m} x_{i,I} \quad I=1,2,\cdots,n \tag{27}$$

ここで $\hat{x}_I^L$ と $\hat{x}_I^U$ は，それぞれ側面制約条件の下限値と上限値を表しており，$x_{i,I}$ は i 番目のサンプル点 $\boldsymbol{x}_i$ の I 番目の設計変数を表している．図2の例では，点Aと点Bが密度関数と境界関数の側面制約となる．

3．4　密度関数および境界関数による追加サンプル点の数

本論文では，密度関数および境界関数を用いた場合の追加サンプル点の数を，以下のように設定した．

$$\mathrm{int}(n/2) \tag{28}$$

ここで int は四捨五入の関数を表す．

3．5　追加サンプル点の総数

図1に示した逐次近似最適化の流れの中で，一度に追加する追加サンプル点の総数(図1中の「Addition of the Sampling Points」に相当する部分)は次のようになる．

$$1+2\times\mathrm{int}(n/2)+ncon\times\mathrm{int}(n/2) \tag{29}$$

上式において，各項の意味は次の通りである．

第1項：　応答曲面の最適解を追加．

第2項：　密度関数 $D(\boldsymbol{x})$ およびランダム性を考慮した追加サンプル点の数．

第3項：　境界関数 $B_k(\boldsymbol{x})$（$k=1,2,\cdots,ncon$）による追加

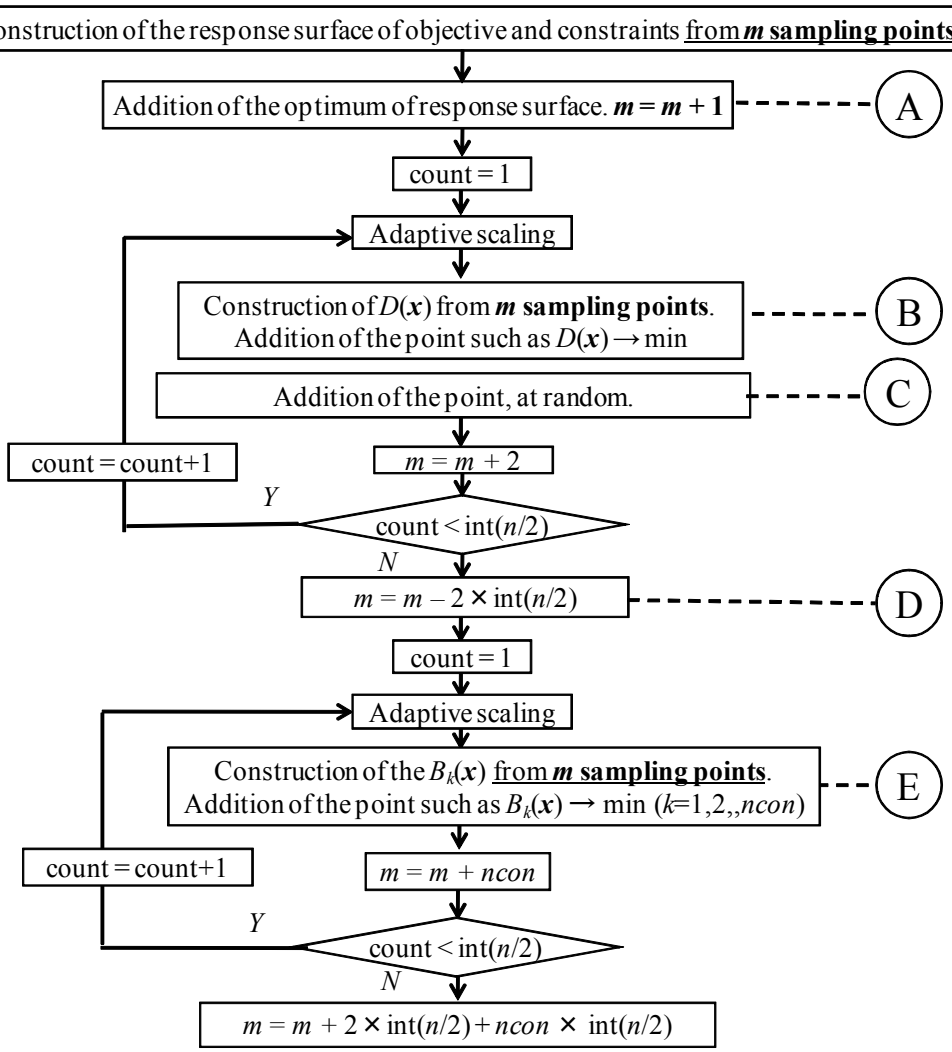


Fig.4 Proposed sampling algorithm

サンプル点の数.

3.6 大域的最適化 密度関数および境界関数は，多峰性関数となる．そのため，密度関数および境界関数の最小値の探索には，大域的最適化法が必要となる．両関数の最小値探索問題は，無制約最適化問題となるため，本論文ではPSOを用いた．PSOの詳細については，例えば文献(22)を参照されたい.

4. サンプル点追加のアルゴリズム

本論文で提示するRBFネットワークを利用したサンプル点追加のアルゴリズムを図4に示す．以下，同図中のA～Dについて説明する.

4.1 応答曲面の最適解を追加（A） 目的関数と制約条件の応答曲面を個別に作成し，これから得られた最適解を直接，追加サンプル点とする．応答曲面の最適解が真の制約条件を満足していない可能性も考えられるが，m 個のサンプル点から得られた貴重な情報であるため利用する．また，この時点で，サンプル点総数を

$$m = m+1 \tag{30}$$

と更新する.

4.2 密度関数の構築（B） 応答曲面の最適解を加えたサンプル点を含めて，密度関数を構築し，密度関数の最適解を追加サンプル点とする.

4.3 ランダム性の考慮（C） ランダム性を考慮することで，サンプル点間のばらつきを持たせることを期待して行う．また，外挿領域にもサンプル点を配置することも期待でき，探索空間を拡大することも期待してランダム性を考慮する.

4.4 サンプル点数を戻す（D） 密度関数およびランダムに配置したサンプル点は，制約条件を考慮せず配置しているため，境界関数の精度に大きな影響を与える．そのため，密度関数およびランダムに配置したサンプル点を除き，一旦，式(30)で表わされる応答曲面の最適解を追加したサンプル点数，すなわち図4中Aのサンプル点数に戻す.

4.5 境界関数の構築（E） 応答曲面の最適解を加えた式(30)のサンプル点数から，制約条件ごとに境界関数を構築し，境界関数の最適解を追加サンプル点とする.

5. 数値計算例

本論文で提示するサンプル点配置戦略を考慮した逐次近似最適化システムの有効性を数値計算例を通じて検討する．実際の設計では，目的関数や制約条件を設計変数の陽な形で表すことが困難である場合が多いが，本論文で提示する方法の基礎的検討を行うため，数値計算例では，関数形が設計変数の陽な形で与えられているものを扱う．また，設計者自身が満足した段階でシステムを終了させることも可能であるが，以下の数値計算例では，最大サンプル点数 $m_{\max}$ を指定することにより終了するよう設定した．また，応答曲面の最適解を求めるために用いたPSOでは，探索点数を30，最大探索回数を500とした.

5.1 有制約最適化問題 次の問題を考える.

$$f(\boldsymbol{x}) = -(x_1-1)^2-(x_2-0.5)^2 \to \min \tag{31}$$

$$g_1(\boldsymbol{x}) = \frac{[(x_1-3)^2+(x_2+2)^2]\exp(-x_2^7)}{12} - 1 \le 0 \tag{32}$$

$$g_2(\boldsymbol{x}) = (10x_1+x_2)/7 - 1 \le 0 \tag{33}$$

$$g_3(\boldsymbol{x}) = \frac{(x_1-0.5)^2+(x_2-0.5)^2}{0.2} - 1 \le 0 \tag{34}$$

$$0 \le \boldsymbol{x} \le 1 \tag{35}$$

この問題の実行可能領域と局所的最適解 $\boldsymbol{x}_L$ および大域的最適解 $\boldsymbol{x}_G$ を図5に示す．図5から明らかなように，実行可能領域が分離しており，また制約条件 $g_1(\boldsymbol{x})$ と $g_3(\boldsymbol{x})$ がアクティブな状態で $\boldsymbol{x}_L$ および $\boldsymbol{x}_G$ が得られる．なお，

$$\boldsymbol{x}_G = (0.2016, 0.8332)^T, \quad f(\boldsymbol{x}_G) = -0.7484 \tag{36}$$

$$\boldsymbol{x}_L = (0.2623, 0.1223)^T, \quad f(\boldsymbol{x}_L) = -0.6867 \tag{37}$$

である.

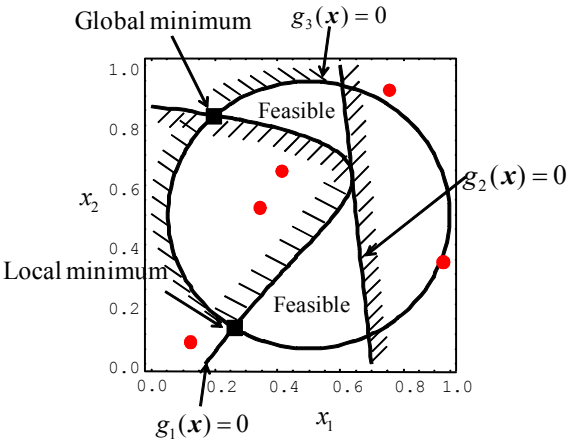
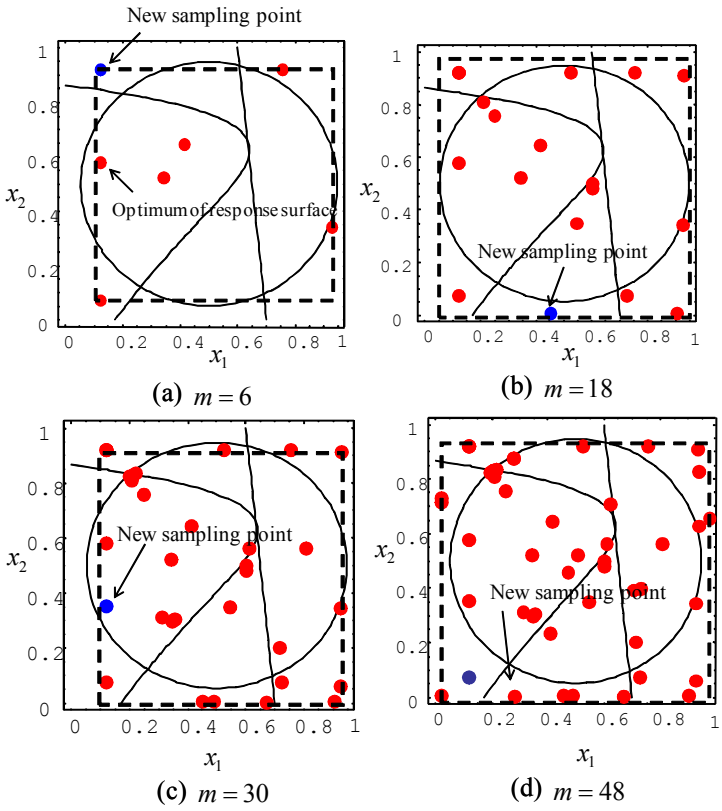


Fig.5 Feasible region, global and local minimum

この問題に対し，ラテン超方格実験(Latin Hypercube Design:LHD)に基づき初期サンプル点を5点配置し，最大サンプル点数を $m_{\max}=50$ とした．図5中の●は初期サンプル点を示す．目的関数および各制約条件を個別にRBFネットワークで応答曲面近似し，最適解を求める．密度関数および境界関数によって追加されたサンプル点を図6と図7の●で示す．図6中の点線は，密度関数の探索領域を表している.

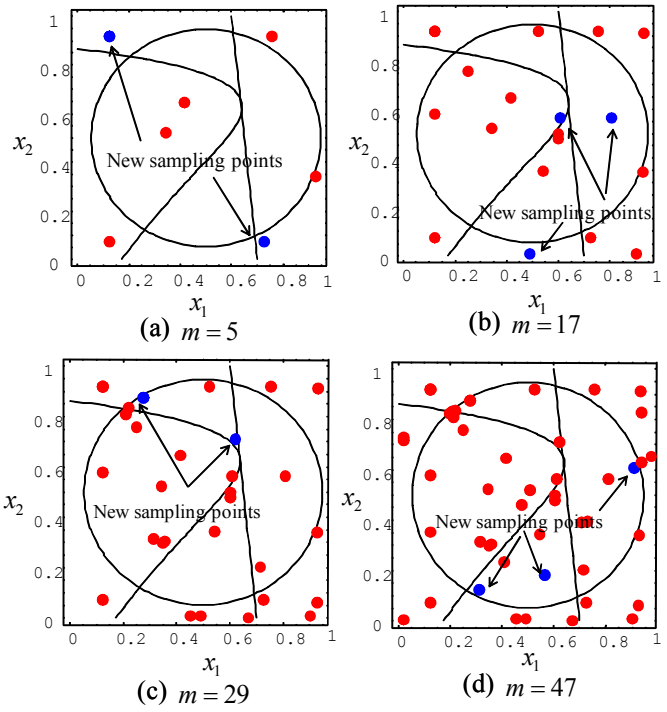
密度関数による追加サンプル点では，サンプル点数



(a) $m=6$ (b) $m=18$

(c) $m=30$ (d) $m=48$

Fig.6 New sampling point by using the density function



(a) $m=5$ (b) $m=17$

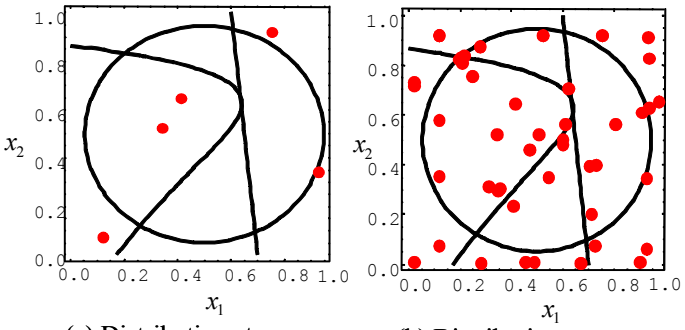
(c) $m=29$ (d) $m=47$

Fig.7 New sampling points by using the boundary function

が少ない場合は，密度関数の探索領域の側面制約条件がアクティブとなる点に密度関数の最小値を取りやすいが，サンプル点数が増加するにつれ，探索領域内部にも密度関数を最小とする点があることがわかる．

一方，境界関数を利用した場合，必ずしも各制約条件上（$g_1(\boldsymbol{x})=g_2(\boldsymbol{x})=g_3(\boldsymbol{x})=0$）に確実に境界関数を最小とする点がこないが，境界付近にサンプル点を追加できていることがわかる．

図8に，初期サンプル点と最終的なサンプル点の配置状況を示す．



(a) Distribution at $m=5$ (b) Distribution at $m=50$

Fig.8 Distribution of sampling points

図8より，定性的ではあるが，大域的最適解付近にサンプル点が集まっており，また設計変数空間全体的にサンプル点が配置されており，さらに境界上付近にもいくつかサンプル点があることがわかる．応答曲面の最適解 $\tilde{\boldsymbol{x}}_G$ における目的関数値 $f(\tilde{\boldsymbol{x}}_G)$ と各制約条件値 $g_k(\tilde{\boldsymbol{x}}_G)$（$k=1,2,3$）の履歴を表1に示す．なお，最終的に

Table 1 Objective and constraints at the approximate global minimum

Number of sampling points	$f(\tilde{\boldsymbol{x}}_G)$	$g_1(\tilde{\boldsymbol{x}}_G)$	$g_2(\tilde{\boldsymbol{x}}_G)$	$g_3(\tilde{\boldsymbol{x}}_G)$
5	-0.7818	0.2189	-0.7468	-0.2441
11	-0.6321	0.0966	-0.5380	-0.3510
17	-0.7245	0.0414	-0.5886	-0.0920
23	-0.7257	-0.0159	-0.5686	-0.0309
29	-0.7510	0.0158	-0.6006	-0.0100
35	-0.7468	0.0000	-0.5919	0.0000
41	-0.7468	0.0000	-0.5919	0.0000
47	-0.7468	0.0000	-0.5919	0.0000

Table 2 Results of benchmark problems

	Test1	Test2	Test3	Test4	Test5
m_{max}	15	50	50	50	50
Global minimum of objective	-12.871	-1.4565	0	11.4371	-0.7483
Minimum of objective	-12.8708	-1.4557	4.3601E-04	11.4426	-0.7486
Maximum of objective	-12.3941	-1.3107	9.3849E-03	11.9480	-0.7431
Average of objective	-12.7723	-1.4061	3.5725E-03	11.6164	-0.7467
Standard deviatiation of objective	1.4444E-01	4.8181E-02	3.1839E-03	1.7970E-01	1.7961E-03

得られた応答曲面の最適解 $\tilde{\boldsymbol{x}}_G$ と目的関数値 $f(\tilde{\boldsymbol{x}}_G)$ は次の通りである．

$$\tilde{\boldsymbol{x}}_G = (0.2024, 0.8327)^T, \quad f(\tilde{\boldsymbol{x}}_G) = -0.7468 \qquad (38)$$

表1より，サンプル点数 $m = 29$ のときには，大域的最適解を見つけていることがわかる．なお，制約条件 $g_1(\boldsymbol{x})$ を多少違反しているが，制約条件が正規化されていることを考慮すれば，制約条件を満足していると考えてよい．

5．2　ベンチマーク問題への適用結果　基礎的な検討として，設計変数の少ない問題に対して，本論文で提示する逐次近似最適化システムの有効性を検討する．付録に記載するいくつかのベンチマーク問題に対し，逐次近似最適化を実行した結果を表2に示す．初期サンプル点配置は，LHDにより決め，すべての問題において，初期サンプル点数を5とした．表2中の $m_{\max}$ は最大サンプル点数を表しており，乱数の種を変えて10回試行した．

表2より，比較的少ないサンプル点数で精度の高い大域的最適解を見つけることができており，本論文で提示する逐次近似最適化システムの有効性の一端が確認できる．

5．3　コイルバネの重量最小化問題　文献（18）のコイルバネの重量最小化問題を考える．設計変数はワイヤの直径 $d(=x_1)$，コイルの平均直径 $D(=x_2)$，コイルの巻数 $N(=x_3)$ であり，すべて連続変数である．最適設計問題は次のように定式化される．

$$f(\boldsymbol{x}) = (2 + x_3)x_1^2 x_2 \to \min \qquad (39)$$

$$g_1(\boldsymbol{x}) = 1 - x_2^3 x_3 / (71785 x_1^4) \le 0 \qquad (40)$$

$$g_2(\boldsymbol{x}) = \frac{4x_2^2 - x_1 x_2}{12566(x_2 x_1^3 - x_1^4)} + \frac{1}{5108 x_1^2} - 1 \le 0 \qquad (41)$$

$$g_3(\boldsymbol{x}) = 1 - 140.45 x_1 / (x_2^2 x_3) \le 0 \qquad (42)$$

$$g_4(\boldsymbol{x}) = (x_1 + x_2)/1.5 - 1 \le 0 \qquad (43)$$

$$0.05 \le x_1 \le 2.00 \qquad (44)$$

$$0.25 \le x_2 \le 1.30 \qquad (45)$$

$$2.00 \le x_3 \le 15.0 \qquad (46)$$

高々3変数の問題であるが，実行可能領域内の最適解を見つけにくい問題であり，さらに多峰性が激しいため，非常に多くの計算を要するとされている．そのため多点探索型最適化手法のベンチマーク問題としてもよく用いられている[19–21]．

はじめに，表3に示すように直交表 $L9$ を用いて各設計変数の初期サンプル点を決め，その後，最大サンプル点数を $m_{\max} = 150$ とし，逐次近似最適化を11回行った結果を表4 に示す．なお，本節で扱う最適設計問題は，筆者らも一連の研究で扱っているため[16]，本論文で提示するサンプル関数の有効性を比較のため，その結果も同表中に併せて示す．

Table 3 Orthogonal array of L9

	x_1	x_2	x_3
No.1	0.05	0.25	2
No.2	0.05	0.775	8.5
No.3	0.05	1.3	15
No.4	1.025	0.25	8.5
No.5	1.025	0.075	15
No.6	1.025	1.3	2
No.7	2	0.25	15
No.8	2	0.075	2
No.9	2	1.3	8.5

表4 より，本論文で提示した方法のほうが，以前の研究で得られた方法よりも，目的関数値および11回試行したときの目的関数値の平均も大幅に改善されており，ファンクションコールも激減していることがわかる．目的関数値が大幅に改善された理由は，制約条件 $g_1(\boldsymbol{x})$，$g_2(\boldsymbol{x})$ がよりアクティブに近づいているためで

Table 4 Comparison of results of the optimum design of tension/compression spring

Design variables	Best solutions found					
	Arora [18]	Coello [19]	Ray [20]	Hu [21]	Previous report	This research
x_1	0.053396	0.05148	0.050417	0.051466	0.052062	0.050000
x_2	0.39918	0.351661	0.321532	0.351384	0.337205	0.314777
x_3	9.1854	11.632201	13.979915	11.60866	13.831074	14.650042
$g_1(\boldsymbol{x})$	0.000019	-0.00208	-0.001926	-0.003336	-0.005994	-0.018820
$g_2(\boldsymbol{x})$	-0.000018	-0.00011	-0.012944	-0.00011	-0.062925	-0.006566
$g_3(\boldsymbol{x})$	-4.123832	-4.026318	-3.89943	-4.026318	-3.649392	-3.837790
$g_4(\boldsymbol{x})$	-0.698283	-0.731239	-0.752034	-0.731324	-0.740489	-0.756815
$f(\boldsymbol{x})$	0.01273	0.012705	0.01306	0.012667	0.014469	0.013103
Function Call	N/A	900000	1291	N/A	82	66
Average of $f(\boldsymbol{x})$	N/A	0.012769	0.013436	0.012719	0.016013	0.013273
Worst of $f(\boldsymbol{x})$	N/A	0.012822	0.01358	N/A	0.017655	0.013643

ある．これは，ランダム性に大きく依存した以前の方法よりも，本論文で提示する密度関数および境界関数の有効性を示す一つの結果であると考えられる．

6. 考察

6.1 密度関数について　密度関数は，設計変数空間におけるサンプル点の疎な領域を見つけることが目的であった．この点については，数値計算例で示したように，この目的をある程度，達成できているものと考えられる．サンプル点が少ない段階では，図6に示したように密度関数の側面制約条件上に最適解をとることが多い．サンプル点の少ない段階で，内挿領域に最小値を生成させるのであれば，半径値をより小さくとる必要がある．ただし半径値を小さくとると，サンプル点の少ない段階で密度関数は凹凸の激しいものとなり，どのような大域的最適化法を用いても，密度関数の局所的最適解発見に留まる可能性が大きいことに注意が必要である．そのため，本論文では意図的に式(12)を用いることで，サンプル点の少ない段階では，比較的密度関数の凹凸をなくし，サンプル点が増加するにつれて，密度関数の凹凸を激しくしている．

6.2 境界関数について　サンプル点の少ない段階では，境界関数の最小値は必ずしも境界($g_k(\boldsymbol{x})=0$)に生成されなかったが，サンプル点が増加するにつれ，境界付近に最小値が生成されることがわかった．有制約最適設計問題では，一般に一つ以上の制約条件がアクティブになることが多く，制約条件が増加した場合は，アクティブな制約条件を対象とし，境界関数を作成すればよいと思われる．しかし，複数の制約条件により実行可能領域が分離しているような問題では，局所的最適解の発見に留まることが予想されるため，本論文ではすべての制約条件を対象とし，境界関数を作成した．実務レベルの設計問題では，多くのファンクションコールは許容できないことが多く，アクティブな制約条件のみを対象とし，境界関数を構築することが実用的であると考えられる．

6.3 半径に関する考察　本論文では，RBFネットワークの半径の決定に関し，密度関数には式(12)，境界関数には式(25)を用いている．設計変数をスケーリングせずにこれらの式を単純に用いると，設計変数の範囲やオーダーが異なる場合，有効に作用するとは限らない．そのため，2.4節で示した適応的スケーリングを用いて設計変数をスケーリングした後に式(12),(25)を用いることが極めて重要である．サポートベクターマシンやKriging等のガウス関数を用いた方法では，半径を簡素に適切に決めることが求められており，本論文で提示した適応的スケーリングおよび半径式は，実用上，有効であると考えている．

7. 結言

本論文では，逐次近似最適化における重要な研究課題である**追加サンプル点の配置方法について**，RBFネットワークを用いた追加サンプル点の配置方法を提示した．本論文で提示した追加サンプル点の配置方法は，サンプル点の疎な領域を見つけるための密度関数，実行可能領域と非実行可能領域の境界上にサンプル点を配置する境界関数をそれぞれRBFネットワークを用いて構築し，それらの最小値をとる点を新たな追加サンプル点とし，さらに探索領域を拡大するため，ランダム性を考慮した方法である．本論文で提示した方法の特徴は，RBFネットワークによる応答曲面の特徴を生かし，積極的に密度関数および境界関数を多峰性関数にする点にある．また，RBFネットワークにより応答曲面を作成するサブルーチンを作成していれば，そのサブルーチンを何度も使うことが可能であり，簡単に実行できる．簡単な数値計算例を通じ，基礎的検討を行い，本論文で提示する方法の有効性の一端を確認した．

本研究を遂行するにあたり，適切なご助言を頂いた山川宏先生（早稲田大学），杉本博之先生（北海学園大学），中山弘隆先生（甲南大学）には，特に感謝したい．

参考文献

(1) Myers, R.H., Montgomery, D.C., *Reponse Surface Methodology -Process and Product Optimization Using Designed Experiments*, Wiley Inter-Science, (1995).

(2) Kashiwamura, T., Mori, Y., Shiratori, M., Yu, O., Maruyama, O., Optimum Design of Frame Column Subjected to Axial Crushing by Statistical Optimization Method, *Trans. of the Japan Society of Mechanical Engineers, Series A*, Vol.62, No.603, (1996), pp.2422-2427.

(3) Yamakawa, H., ed., *Handbook of Optimum Design*, (2003), pp.162-170, Asakura Shoten, (in Japanese)

(4) Donald, R.J., Schonlau, M., Welch, W.J., Efficient Global Optimization of Expensive Black-Box Functions, *J. of Global Optimization*, Vol. 13, (1998), pp.455-492.

(5) Simpson, W., Mauery, T.M., Korte, J.J., Mistree, F., Kriging Models for Global Approximation in Simulation-Based Multidisciplinary Design Optimization, *AIAA Journal*, Vol.39, No.12, (2001), pp.2233-2241.

(6) Martin, J.D., Simpson, W., Use of Kriging Models to Ap-

proximate Deterministic Computer Models, *AIAA Journal*, Vol.43, No.4, (2005), pp.853-863.
(7) Muller A. A., Messac, A., Extended Radial Basis Functions: More Flexible and Effective Metamodeling, *AIAA Journal,* Vol.43, No.6, (2005), pp.1306-1315.
(8) Nakayama, H., Arakawa, M., Sasaki, R., Simulation-Based Optimization Using Computational Intelligence, *Optimization and Engieering*, Vol. 3, (2002), pp.201-214.
(9) Wang, G.G., Shan, S., Review of Metamodeling Techniques in Support of Egnieering Design Optimization, *Trans. of ASME, J. of Mechanical Design*, Vol. 129, (2007), pp.370-380.
(10) Nakayama, H., et. al., *Multiobjective Optimization and Design Engineering*, Gendaitosho, (2007). (in Japanese)
(11) Huang, D., Allen, T.T., Notz, W.I., Zeng, N., Global Optimization of Stochastic Blax-Box Systems via Sequential Kriging Meta-Models, *J. of Global Optimization*, Vol. 34, (2006), pp.441-466.
(12) Shao, T., Krishnamurty, S., A Clustering-Based Surrogate Model Upadating Approach to Simulation-Based Engineering Design, *Tras. of ASME, J. of Mechanical Design*, Vol. 130, (2008), pp.041101-1-041101-13.
(13) Sharif, B., Wang, G.G., EiMekkawy, T.Y., Mode Pursuing Sampling Method for Discrete Variable Optimization on Expensive Black-Box Functions, *Trans. of ASME, J. of Mechanical Design*, Vol. 130, (2008), pp.021402-1-021402-11.
(14) Todoroki, A., Kawakami, Y., Optimal Design of Wind Turbine Blade of CF/GF Hybrid Composites, *Trans. of JSCES*, Paper No. 20080012.
(15) Shan, S., Wang, G.G., An Efficient Pareto Set Identification Approach for Multiobjective Optimization on Black-Box Functions, *Trans. of ASME, J. of Mechanical Design*, Vol. 127, (2005), pp.866-874.
(16) Kitayama,S., Arakawa, M., Yamazaki, K.,Global Optimization by Generalized Random Tunneling Algorithm (5th report: Approximate Optimization Using RBF Network), *Trans.of the Japan Society of Mechanical Engineers, Series C*, Vol.73, No.729, (2007), pp.1299-1306.
(17) Arakawa, M., Nakayama, H., Ishikawa, H., Approximate Optimization Using RBF Network and Genetic Range Genetic Algorithm (Proposal of Base Function and Basic Consideration), *Trans. of the Japan Society of Mechanical Engineers, Series C*, Vol.70, No.697, (2004), pp.2674-2681.
(18) Arora, J.S., *Introduction to Optimum Design*, (1989), McGraw-Hill, New York.
(19) Coello Coello, C.A., Use of a Self-Adaptive Penalty Approach for Engineering Optimization Problems, *Computers in Industry*, Vol.41, (2000), pp.113-127.
(20) Ray, T., Saini, P., Engineering Design Optimization Using Swarm with an Intelligent Information Sharing among Individuals, *Engineering Optimization*, Vol.33, (2001), pp.735-748.
(21) Hu, X. H., Eberhart, R.C., Shi, Y.H., Engineering Optimization with Particle Swarm, *IEEE Swarm Intelligence Symposium*, (2003), pp.53-57.
(22) Parsopoulos, K.E.,Vrahatis M.N., Recent approaches to global optimization problems through Particle Swarm Optimization, *Natural Computing*,**1**,(2002),pp.235-306.

付 録

5. 2節で用いたベンチマーク問題に示す.

Test1:

$$f(\boldsymbol{x}) = \sum_{i=1}^{5} i\cos[(i+1)x_1 + i] \rightarrow \min$$
$$0 \le x_1 \le 7.5$$

Test2:

$$f(\boldsymbol{x}) = 2 + 0.01(x_2 - x_1^2)^2 + (1-x_1)^2 + 2(2-x_2)^2 + 7\sin(0.5x_1)\sin(0.7x_1x_2) \rightarrow \min$$
$$0 \le \boldsymbol{x} \le 5$$

Test3:

$$f(\boldsymbol{x}) = \sum_{i=1}^{2} |x_i \sin x_i + 0.1x_i| \rightarrow \min$$
$$-10 \le \boldsymbol{x} \le 10$$

Test4:

$$f(\boldsymbol{x}) = x_1^2 + x_2^2 \rightarrow \min$$
$$g_1(\boldsymbol{x}) = -(x_1+4)^2/3 - (x_2 - 0.1)^2 + 20 \le 0$$
$$-6 \le x_1 \le 4, \quad -4 \le x_2 \le 6$$

Test5:

$$f(\boldsymbol{x}) = -(x_1-1)^2 - (x_2-0.5)^2 \rightarrow \min$$
$$g_1(\boldsymbol{x}) = [(x_1-3)^2 + (x_2+2)^2]\exp(-x_2^7) - 12 \le 0$$
$$g_2(\boldsymbol{x}) = 10x_1 + x_2 - 7 \le 0$$
$$g_3(\boldsymbol{x}) = (x_1 - 0.5)^2 + (x_2 - 0.5)^2 - 0.2 \le 0$$
$$0 \le \boldsymbol{x} \le 1$$