ハイパフォーマンス
コンピューティング　88−8
（２００１．１０．２６）

高速球面調和関数変換法の安定性制御

須　田　礼　仁†

本論文では，我々が提案してきている高速球面調和関数変換法の前処理で必要となる安定性の制御について論ずる．分離点を次数範囲の下端に取ると，安定性制御において最も重要な問題は標本点の選択となる．最適な標本点集合を求めることは組合せ最適化問題となるので，我々は標本点集合の構成・改良・選択という 3 段階からなるヒューリスティックスキームを用いることを提案する．また，計算量について考察した上で，minimax と cmax という高速の探索アルゴリズムを提案する．さらに，これらのアルゴリズムで制御の対象とする目的関数について，計算量の観点も含めて論ずる．今回は実装と評価を行うことができなかったが，別の機会に報告したい．

Stability Control of Fast Spherical Harmonics Transform

Reiji Suda†

This paper discusses stability control of the fast spherical harmonics transform algorithm that we have proposed. Setting the lowest degree as the split point, the main problem of stability control becomes the selection of the sampling points, which are combinatorial optimization. We propose a three-fold heuristic scheme of generation, improvement, and selection of sets of sampling points. Considering the computational complexity from the practical point of view, we propose two fast search algorithms called minimax and cmax. The object functions and their approximations are also discussed. Implementation and experimental results are not included in this paper.

1. はじめに

球面調和関数変換は球面や球核などの領域における偏微分方程式の数値解法の道具として広く用いられている．異常気象や温暖化などで注目を集めている全球長期の気象シミュレーションでも現在用いられており，大規模変換の高速化は重要な課題である．

これに対し我々は，高速多重極子展開法に基づいた高速球面調和関数変換アルゴリズムを提案し，研究を進めてきた．これまでに，アルゴリズムのアイディアの提案[2]，実装評価[3]，誤差解析手法の提案[4] などについて論じてきた．今回はアルゴリズムの安定性の制御・最適化の手法とアルゴリズムについて，誤差制御と前処理の計算量を加味して再考する．我々のアルゴリズムの原理と誤差制御の方法については，紙面の都合で記述することができないので，既出の報告や論文を参照していただきたい．

1.1　安定性制御と標本点選択

今回考察する安定性制御とは，数値的な誤差の拡大を抑えることである．一般的には丸め誤差の爆発的な拡大を指して数値的不安定性と呼ぶが，我々のアルゴリズムでは内挿と直接計算に含まれる近似の誤差の拡大がより重要な問題である．

安定性制御の目的は 2 つあると考えられる．目的の一つは計算量の削減である．原則的には，安定性が高ければ，近似においてより大きな誤差が許されることになる．従って，安定性の最大化と計算量の最小化は基本的に両立する．もう一つの目的は計算精度の確保である．計算精度は丸め誤差によって制限を受けているので，安定性が低くなると達成できる計算精度が低くなってくる．アプリケーションが要求する計算精度を実現できなくなっては元も子もないので，安定性は高速計算法の生命に関わる問題である．両者を総合すると，**アプリケーションが要求する精度の範囲内で計算量を最小化すること**が目標ということになる．

さて，我々のアルゴリズムは，与えられた変換を内挿と分割統治で繰返し分解することにより高速化を実現する．その中で，数値的安定性に影響を及ぼし得るパラメタはいくつかあるが，本稿では内挿における**標本点集合の選択**に話題を絞る．

ルジャンドル陪関数変換を計算する点を**評価点**と呼んでいる．この評価点から適当な数の**標本点**を選び，あらかじめこれらの点におけるルジャンドル陪関数変換値を計算しておく．そして残りの評価点（**内挿点**と呼ぶ）における値は，FMM を用いた高速ルジャンドル陪関数内挿によって計算するのが我々のアルゴリズムの基本アイディアである．この内挿における標本点集合が数値的安定性に重大な影響を及ぼすことが明らかとなっている．我々のアルゴリズムはいわゆる D-H 法[1] とほぼ同じ手法に基づいているのだが，D-H 法が数値的に不安定であるのに対し，我々のアルゴリズムが安定であるのは，D-H 法が標本点に相当す

† 名古屋大学大学院 工学研究科 計算理工学専攻
Department of Computational Science and Engineering, Faculty of Engineering, Nagoya University

るものを等間隔点に制限しているのに対して，我々のアルゴリズムでは標本点集合を自由に選択できるところが本質的なのである．

標本点集合のほかに数値的安定性に大きく影響するパラメタに**分離点**がある．計算量的には，分離点は次数範囲の中央に取るのが最適であるが，数値的な安定性は分離点が次数範囲の下端点にあるときが最適であるらしい．分離点が下端点から離れると急激に安定性が低下して，達成精度が下がることが分かっている．将来はともかく現時点では安定性を優先させて，次数範囲の下端点に分離点を設定するのが賢明であると考えられる．一方，**分割点**と数値的な安定性との関係は何もわかっていない．分割点は計算量に関しても最適な設定が不明であり，現在では単純に次数の範囲を等分している．これらのパラメタについては，本稿ではこれ以上考察しない．

以上の議論に基づき，以下では安定性制御として標本点選択について論ずる．

なお，本稿では計算量のオーダー記号を以下のように用いる．計算量が関数 $g(n)$ で上から抑えられるときに $O(g(n))$，下から抑えられるときに $\Omega(g(n))$，両方抑えられるときに $\Theta(g(n))$ と記述する．また，最良計算量と最悪計算量に差があるときに，その中間の値を「目安」として用いるが，その際 $\theta(g(n))$ という表記を用いる．

2. 標本点選択スキーム

標本点選択が必要となる内挿計算は，分割統治で定義される二分木の各ノードでで発生するが，それらは互いに関連している．子ノードの内挿は評価点集合についてもスケーリングについても先祖ノードの内挿に全面的に依存するが，逆に親ノードの内挿に許される誤差の拡大率は子孫ノードの内挿の安定性に依存する．従って，完全な最適化のためには，二分木内のすべての内挿の標本点選択を一括して最適化する必要があるが，それは計算量的に無理がある．逐次的に処理を行う場合，親の内挿に対する子の内挿の依存は非常に本質的であるから，親の内挿から順に決定するトップダウンの処理が必然的な選択となる．

そこで，ある内挿の標本点を選択する際には，祖先ノードのすべての内挿がすでに決定しているものとする．子孫ノードの計算についても，現状では内挿に非依存で決めている分割点と分離点の情報が事前に得られるが，子孫ノードの内挿の標本点集合や誤差の限界は未知であるとする．

2.1 アルゴリズムの構成

標本点の選択は組合せ最適化である．しかし，かなり大規模な変換も実用的な時間で行いたいので，計算量は厳しく制約を受けており，かなり簡単で強力な近似最適化アルゴリズムが必要である．我々はアルゴリズムを次のような 3 段階で構成することを提案する．

(1) **標本点集合の構成**：与えられた評価点集合から n 個の標本点を選ばなければならない．そのようなアルゴリズムの中で最も基本的な手法であるが，空集合から出発して標本点を一つずつ追加することにより，標本点集合を構成する．標本点追加のプロセスは標本点の数だけ反復されるので，1 反復あたりの計算量の少ないアルゴリズムが必要である．

(2) **標本点集合の改良**：与えられた標本点集合からいくつかの標本点を取り除き，同数の別の点を標本点に加えることにより，目的関数の値を改善させる．同時に複数の点を入れ換えるのは計算量が大きくなってしまうので，1 組ずつ交換させるのが適切である．(i) **最適な 1 組の交換を求める**ことを考えると，現在の標本点集合のサイズ n と内挿点集合のサイズ m から nm 組の候補ができることになる．しかし，(ii) **最適な 1 点を取り除いてから最適な 1 点を加える**という方法にすれば，削除の候補は n 個，追加の候補は $m+1$ 個であるから，ずっと簡単である．しかし，2 次元の問題を 1 次元化しているという弱点がある．両者の中間的な方法として，(iii) **削除点に対して追加点が最適で，かつ追加点に対して削除点が最適となるような組を求める**という方法がある．2 次元を同時に見てはいるものの，1 次元探索を反復することで最適点が求まるのが特徴である．これらの改良アルゴリズムにおいて，目的関数は構成アルゴリズムのものと同じである必要はない．改良の回数を制限すれば，計算量のかかる目的関数を使用することができる．

(3) **標本点集合の選択**：いくつかのアルゴリズムで得られた標本点集合に対して目的関数を評価し，最適な集合を選択する．反復することはないから，最も計算量のかかる目的関数を用いることが許される．

2.2 前処理に許される計算量

以下，最終的な評価点の数を M，ルジャンドル陪関数の最大次数（切断周波数）を T とする．$M=\Theta(T)$ を仮定すると，前処理の計算量としてはアルファ値（後述）の計算量が最も多く，$\Theta(MT^2\log T)=\Theta(T^3\log T)$ である．しかもこの計算量は FMM により高速化されており，FMM を用いない場合の計算量は $\Theta(MT^3)$ となる．FMM を用いても T が小さい範囲ではこれに近い計算量の挙動を示すことになり，かなりの時間がかかる印象がする．非常に粗い近似値であればアルファ値を合計計算量 $\Theta(T^3)$ で求めることができるが，精度が悪く，あまりよくないようである．

実用的には，この $\Theta(T^3\log T)$ あるいは $\Theta(T^4)$ という計算量は限界である．計算量が T^4 に比例する場合，$T=100$ の前処理が 5 分で終了したとしても，$T=2000$ の前処理には 1 年半以上かかってしまう．従って，標本点選択のアルゴリズムの計算量は，できるだけアルファ値の計算量 $\Theta(T^3\log T)$ 以下に収めることが目標となる．

3. 高速探索アルゴリズム

この節では，標本点選択に用いる 2 つの探索アルゴリズムについて説明する．標本点選択はある種の最適化問題であるから，最適化の手法のいくつかが適用できるものと

思うが，以下に述べる 2 つのアルゴリズムと既存の手法との関係は，著者の不勉強のために不明である．読者から情報を頂ければ幸いである．

3.1 Minimax アルゴリズム

ここでは関数 $g_p(q)$ に対して

$$g(q) = \max_{p \in P}\{g_p(q)\}$$

を定義し，これを最小にする $q \in Q$ を求める問題を考える．単純なアルゴリズムとしては，$|P||Q|$ 個の $g_p(q)$ を計算してしまえばよい．しかし，ある p, q に対して $g_p(q)$ が p に関して最大かつ q に関して最小であることを証明できれば，他の値は計算しなくてもよいはずである．

ここで，何らかの方法で $g(q)$ の上限と下限 $\bar{g}(q)$ と $\underline{g}(q)$

$$\underline{g}(q) \le g(q) \le \bar{g}(q)$$

とが得られるものとする．これを用い，次のようなループで最大値最小化を実現することができる．これは分枝限定法に多少の工夫を加えたものになっている．

(1) **初期化**：最初にすべての q について $\bar{g}(q)$ と $\underline{g}(q)$ とを計算する．次に，$\bar{g}(q)$ を最小にする q_u および $\underline{g}(q)$ を最小にする q_1 と二番目に小さくする q_2 を求める．このとき $\bar{g}(q_u) \le \underline{g}(q_2)$ であれば q_u が最小値を達成する可能性がある．実際，$q_1 = q_u$ または $\bar{g}(q_u) \le \underline{g}(q_1)$ であれば最小値であるはずである．最小値が見つかればアルゴリズムを終了する．

(2) **初期最大値探索**：$g(q_u) = \max_{p \in P}\{g_p(q_u)\}$ を計算し，最大値を達成する p を p_* とおく．また，$\bar{g}(q_u) = \underline{g}(q_u) = g(q_u)$ と更新し，$q_* = q_u$ とおく．

(3) **下限値更新**：$\underline{g}(q) \ge \bar{g}(q_*)$ なる $q \ne q_*$ は最小値探索の候補にならないので，集合 Q から取り除く（縮小法）．それ以外の q に対して $g_{p_*}(q)$ を計算し，$g_{p_*}(q) > \underline{g}(q)$ なら $\underline{g}(q) = g_{p_*}(q)$ と更新する．改めて $\underline{g}(q)$ を最小にする q_1 と二番目に小さくする q_2 を求め，q_* が最小値を達成するかどうかをチェックする．最小値が見つかればアルゴリズムを終了する．

(4) **最大値探索**：$g(q_1)$ を計算し，最大値を達成する p を p_* とおく．また，$\bar{g}(q_1) = \underline{g}(q_1) = g(q_1)$ と更新し，$\bar{g}(q_1) < \bar{g}(q_*)$ なら $q_* = q_1$ と更新する．$g(q_1) \le \underline{g}(q_2)$ ならば q_1 が最小値を達成するとして終了する．そうでなければステップ (3) に戻る．

このアルゴリズムを以下では単に **minimax** と呼ぶことにし，計算量について考察する．上限値・下限値の計算量を $O(B)$ かつ $\Omega(\underline{B})$ とすると，初期化で $O(|Q|B)$ の計算量が必要である．最善の場合，これで最小値が見つかってしまうので，計算量の下限は $\Omega(|Q|\underline{B})$ となる．

ステップ (3) (4) からなるループは，$\min\{|P|, |Q|\}$ 回以内で停止する．なぜなら，すべての q に対して $g(q)$ が計算されれば当然最小値が求まるはずであるし，(4) において以前に p_* として選ばれたことがある p が再度 p_* に選ばれることはない（そうであれば $g(q_1) = \underline{g}(q_1)$ となり，q_1 は最小値を達成して終了する）からである．$g_p(q)$ の計算量を $O(F)$，$g(q)$ の計算量を $O(G)$ とすると，1 反復の計算量は $O(|Q|F + G)$ である．従って，minimax アルゴリズムの計算量の上限は $O(|Q|B + \min\{|P|, |Q|\}(|Q|F + G)$ となる．

また，$B = O(F)$ および $G = O(|P|F)$ と仮定することができるので，最悪でも $O(|P||Q|F)$ で計算量が抑えられ，すべての $g_p(q)$ を求める単純なアルゴリズムよりも悪くはならない．しかし，$g(q)$ を高速に求めるアルゴリズムがある場合，すべての $g(q)$ を計算して比較するアルゴリズムの計算量 $O(|Q|G)$ と minimax アルゴリズムの計算量どちらが少ないかは自明ではない．

Minimax アルゴリズムの反復回数を L とすると，$\theta(|Q|(B + LF) + LG)$ の計算量で最大値最小化が実現できる．このように，ループの反復回数が計算量に決定的な影響を与えるが，条件がよければループは数回で停止するようである．

3.2 Cmax アルゴリズム

次に，$|c_{ij}| = |a_i b_j/(y_i - x_j)|$ (但し $0 \le i < n$, $0 \le j < m$) の最大値を求める問題を考える．ここでは，y_i と x_j があらかじめソートされていれば，最大値は $\Theta(n + m)$ の計算時間で計算できることを示す．

まず，問題を $y_i < x_j$ の場合と $y_i > x_j$ の場合とに分割する．前者に対しては $|c_{ij}| = |a_i||b_j|/(x_j - y_i)$，後者に対しては $|c_{ij}| = |a_i||b_j|/(y_i - x_j)$ として，最大値を探す．まず，後者の問題を解くアルゴリズムを説明する．

添字 j を固定して考えると，$|a_i|/(y_i - x_j)$ の最大値を探せばよいことになる．今，y_i に対して二次元平面上の点 $p_i = (y_i, |a_i|)$ を，x_j に対して点 $q_j = (x_j, 0)$ を対応させる．すると，$y_i > x_j$ に対して $|a_i|/(y_i - x_j)$ は点 p_i と q_j を結ぶ直線の傾きである．従って，与えられた q_j に対してこのような最も傾きが急になるような p_i を求めればよいことになる．ここで q_j とそれよりも右にあるすべての点の集合に対する凸包を考える．このとき点 q_j は必ず凸包上にあり，最大値を与える p_i は反時計回りに見て q_j の一つ前の凸包上の点であることが分かる．

凸包の計算方法にはいくつかあるが，ここでは逐次添加法で構成する．すべての点を x 座標値でソートしておき，x 座標の大きいものから順に点を追加して凸包を構成してゆく．その過程において，上述の凸包がすべて中間状態として構成されるから，全体の凸包を構成するだけで，全ての j に対する $\max_{x_j < y_i}\{|a_i|/(y_i - x_j)\}$ が求まることになる．このとき，全体の凸包の構成の計算量は，グラハム走査と同じ原理により $\Theta(n + m)$ となる．

前半の $y_i < x_j$ の部分に対しては，x 座標の小さいものから順に凸包を構成ゆけば同様に $\Theta(n + m)$ の計算量で求めることができる．以上のアルゴリズムにより，$\max\{|c_{ij}|\}$ を $\Theta(n + m)$ の計算量で求めることができる．このアルゴリズムを **cmax** アルゴリズムと呼ぶことにする．

ここで，点があらかじめソートされていることは本質的である．この仮定が成り立たない場合，ソートのために

$O((n+m)\log(n+m))$ の計算時間が必要であることは明らかである．

4. 目的関数

前節までに，標本点選択のためのアルゴリズムの枠組みと，利用する 2 つの探索アルゴリズムについて説明した．この節では，これらのアルゴリズムを使用するために必要な目的関数について考察する．

4.1 C ノルムとその近似

論文 4) で論じているように，内挿によるルジャンドル陪関数変換は

$$\begin{pmatrix} A_S & A \end{pmatrix} \begin{pmatrix} I \\ \Theta \end{pmatrix} B \qquad (1)$$

のように書くことができる．ここで Θ が内挿を表し，$(A_S A)$ は内挿の結果から全評価点での関数値を内挿する計算，B は標本点での関数値を求める計算に対応する．内挿の誤差制御においては，

$$\gamma = \|AS_A^{-1}\|\|S_A\Theta S_B\|\|S_B^{-1}B\|$$

を用いて内挿の相対精度を

$$\|S_A(\tilde{\Theta}-\Theta)S_B\|/\|S_A\Theta S_B\| \le \epsilon/\gamma$$

となるように FMM の展開項数を設定する．ここでノルムは 2 ノルムとし，S_A は AS_A^{-1} の各列の 2 ノルムが 1 となるような対角行列，S_B は $S_B^{-1}B$ の各行の 2 ノルムが 1 となるような対角行列である．この γ の値が小さいほど安定性，計算量ともに有利である．我々はこの γ を **ABC ノルム**と呼んでいる．

ABC ノルムの中央の $\|S_A\Theta S_B\|$ が最も重要である．これを **C ノルム**と呼ぶことにする．同様に $\|AS_A^{-1}\|$ を A ノルム，$\|S_B^{-1}B\|$ を B ノルムと呼ぶ．しかし，A ノルムや B ノルムの値と，選ばれる標本点との関係が単純でなく制御が難しいため，現在は C ノルムだけが制御の対象である．子孫ノードの内挿の A ノルムや B ノルムを事前に制御することは，さらに難しいと思われる．

さて，すべての標本点が決定するまでの間は内挿行列は定義されないので，標本点集合の構成・改良アルゴリズムにおいては，この行列に関連する何らかの数値で代用しなければならない．このような代用数値としては，多項式や行列の視点から，いくつかの候補がある．

スケーリングがかけられた内挿行列 $C = S_A\Theta S_B$ の第 i,j 要素は，非分離のときは

$$c_{ij} = \frac{\alpha_i P_m^m(y_i)\omega(y_i)\beta_j}{(y_i-x_j)P_m^m(x_j)\omega_j(x_j)} \qquad (2)$$

となる．分離のときもほとんど同じ形である．選ばれた k 個の部分的な標本点集合に対して式 (2) で行列 C を定義する方法が，簡単かつ数値的安定性が高く，予測値としても有効である．上記の c_{ij} の式は，

$$a_i = \alpha_i P_m^m(y_i)\omega(y_i)$$
$$b_j = \beta_j/(P_m^m(x_j)\omega_j(x_j))$$

をあらかじめ計算しておけば，

$$c_{ij} = \frac{a_i b_j}{y_i - x_j}$$

と簡単に計算することができる．また，標本点集合が変化するときに，変化後の c_{ij} の値も簡単に計算することができる．また，$y_i - x_j$ などの上限・下限をあらかじめ計算しておけば，これらの行列要素の上限・下限が得られる．

さて，冪乗法を定数回で打ちきって C ノルムの近似値を求めることにすると，標本点が n 個，内挿点が m 個ある場合，$\Theta(mn)$ の計算量が必要である．この計算量では，標本点集合の選択には十分使える．しかし，標本点集合の改良に適用する場合には minimax アルゴリズムがうまく働いてくれなければならないし，標本点集合の構成に使うと $\Omega(T^4)$ の計算量が必要となってしまう．

そこで C ノルムの近似を目的関数とすることにより計算量を軽減することが考えられる．行列のノルムを近似する方法はいくつか考えられるが，行列の要素をすべて計算する必要がある方法はいずれも $\Omega(mn)$ の計算量が必要となってしまう．Minimax アルゴリズムか cmax アルゴリズムがうまく働く形に持ち込む必要がある．

行列の行または列の 2 ノルムの最大値は，行列の 2 ノルムの下限となる．また，$\sqrt{\min\{m,n\}}$ を掛ければフロベニウスノルムの上限すなわち 2 ノルムの上限となる．このことから，内挿行列の行・列の 2 ノルムの最大値を近似目的関数として用いることが可能である．計算量は明らかに $F = O(n+m)$ となり，minimax アルゴリズムを適用することができる．

また，行列要素の絶対値の最大値は，行列の 2 ノルムの下限となる．また，$\sqrt{nm}$ を掛ければ上限とすることができる．これから，行列要素の最大絶対値をノルムの近似として用いることが可能である．これは cmax アルゴリズムで計算することができる．

なお，過去の論文 3) などの構成アルゴリズムで使用していた目的関数は，新しい標本点 x_k に対する $\max_i\{|c_{ik}|\}$ あるいはスケーリングのついていない $\max_i\{|\theta_{ik}|\}$ であった．これらは C ノルムを上から抑えないので，十分な数値的安定性が得られない可能性がある．また類似のものとして，交差項を無視して $\max\{|a_i b_j|\}$ などを近似目的関数とすることも可能で，少ない計算量で次の標本点を選ぶことができる．この場合ある程度の数値的な安定性が得られるが，十分とは言い難いようである．

4.2 アルファ・ベータ積

ABC ノルムを最小化するのが目標であるが，当然限界がある．式 (1) で $B_I = \Theta B$ とすると，これは内挿点上の関数値を求める計算に対応するので，その各行は（順序を除いて）標本点の選択には依存しない．そこで，ABC ノルムに関する不等式

$$\gamma \ge \|AS_A^{-1}\|\|S_A B_I\|$$

の右辺の $\|S_A B_I\|$ を次のように評価することができる．まず，S_A の第 i 対角要素を α_i，B_I の i 行目の 2 ノルムを β_i とすると，$S_A B_I$ の i 行目の 2 ノルムは $\alpha_i\beta_i$ となる．

従って，

$$\|S_A B_I\| \geq \max_i \{\alpha_i \beta_i\}$$

である．また，$\|S_A B_I\|$ のフロベニウスノルムを考えると

$$\|S_A B_I\|^2 \leq \sum_i \alpha_i^2 \beta_i^2$$

であることも分かる．ここで α_i を**アルファ値**，β_i を**ベータ値**，$\alpha_i \beta_i$ を**アルファ・ベータ積**とそれぞれ呼んでいるが，上の 2 式から内挿点のアルファ・ベータ積が内挿の安定性に強い影響を与えることが分かる．

さらに，内挿全体に A スケールをかけて式 (1) を

$$\begin{pmatrix} A_S S^{-1} & A S_A^{-1} \end{pmatrix} \begin{pmatrix} I \\ S_A \Theta S^{-1} \end{pmatrix} SB$$

と書きなおしてみる．このとき $S_A \Theta S^{-1}$ は $S_A B_I$ に対する内挿行列と見なすことができるから，フェケテの原理[3]によって $|\alpha_i \theta_{ij} \bar{\alpha}_j^{-1}| \leq 1$ となる標本点選択が存在する．ここで S の第 j 対角要素を $\bar{\alpha}_j$ とおいた．このとき

$$S_A \Theta S^{-1} S S_B = S_A \Theta S_B$$

を考えると，S_B の第 j 対角要素を β_j とおけば

$$|\alpha_i \theta_{ij} \beta_j| = |\alpha_i \theta_{ij} \bar{\alpha}_j^{-1} \bar{\alpha}_j \beta_j| \leq |\bar{\alpha}_j \beta_j|$$

となる．これから，標本点のアルファ・ベータ積も C ノルムに影響することが分かる．フロベニウスノルムを考えれば

$$\|S_A \Theta S_B\|^2 \leq m \sum_j \bar{\alpha}_j^2 \beta_j^2$$

という評価も得られる (m は内挿点の数)．このように，スケールされた内挿行列の安定性は，標本点のアルファ・ベータ積とも密接な関係がある．

結局，すべての評価点のアルファ・ベータ積が内挿の安定性に密接な関係をもっていることが分かる．また，アルファ・ベータ積は直接計算における丸め誤差とその拡大に関係があることも明らかである．

このように，アルファ・ベータ積は，計算の安定性に関する，標本点選択に依存しない指標である．分割点と分離点が与えられているという仮定のもとでは，子ノードの内挿におけるアルファ・ベータ積は，親ノードの内挿が決まれば計算することができる．従って，これを現在の内挿の標本点選択アルゴリズムの目的関数に加えることがよいと考えられる．なお，上の議論からは，アルファ・ベータ積の**最大値**と**二乗和**の 2 つが目的関数の候補となる．

なお，アルファ・ベータ積は，直接計算の丸め誤差の影響の指標にはなるが，ドロッピングによる計算量の軽減効果とは一致しない．ドロッピングの効果をできるだけ維持するような目的関数も考えられるが，その有効性は以下の点から疑問である．まず，分離やシフトが行われると，関数値が大きく変るのでドロッピングの対象から外れてしまうことが多い．さらに，ルジャンドル陪関数のダンプは急激であるため，アルファ値が多少大きくなってもドロッピングによる計算量の変化は小さいと考えられ，次段が内挿ではないということがあらかじめ分かっている場合を除けば，むしろ安定性に計算量が敏感に反応する内挿を優先した方がよいと思われる．そもそも，関数値が大きくてドロッピングの対象になっていない行に対してはその意味がない．

4.3 子ノードのアルファ・ベータ積

具体的に次段のアルファ・ベータ積について考察する．現在考えている内挿式 (1) の残りの計算 B の計算方法は何通りかある．まず第 1 に，分割統治が停止して，B 全体が直接計算されるという場合がある．この場合，ベータ値は標本点のそれと同じ β_j で，アルファ値については $A_S + A\Theta$ の列の 2 ノルムとなる．このアルファ値を ${}_C\alpha_j$ と書くこととし，ここでのアルファ・ベータ積 ${}_C\alpha_j \beta_j$ を，**内挿後アルファ・ベータ積**と呼ぶことにする．

それ以外の場合には分割統治が行われる．すなわち，次数範囲が半分に分けられ，

$$B = \begin{pmatrix} B_L & B_H \end{pmatrix}$$

のように行列が分割される．しかし，分割自体は不安定性の要因にならないので，分離やシフトがない場合は内挿後アルファ・ベータ積を制御している限りは特段の配慮の必要がない．従って，内挿後アルファ・ベータ積で安定性制御ができる．

しかし，高次数側 B_H に対して内挿が行われる場合は分離やシフトが行われ，

$$B_H = T B'_H$$

のようになる．ここで T は分離あるいはシフトを表す行列で，B'_H は B_H とは異なる分離点で分離された変換行列である．この場合には，アルファ値 ${}_T\alpha_j$ は $(A_S + A\Theta)T$ の列のノルムとなり，ベータ値 ${}_T\beta_j$ は新しい分離点で計算される．従って，内挿後アルファ・ベータ積とは別に制御してやる必要があると考えられる．このアルファ・ベータ積を，**シフト後アルファ・ベータ積**と呼ぶことにする．

以上の議論から，内挿以降の計算の安定性制御のための目的関数としては内挿後アルファ・ベータ積とシフト後アルファ・ベータ積の 2 つが必要ということが分かる．

4.4 内挿後アルファ・ベータ積とその近似

内挿後アルファ・ベータ積についてさらに考察する．

まず，アルファ・ベータ積の二乗和の計算について考える．内挿後のアルファ値は $A_S + A\Theta$ の列の 2 ノルムである．第 1 項は単位ベクトルに，第 2 項は Θ の列に内挿演算を作用させれば計算できる．Θ の 1 列の計算に $\Theta(m)$ の計算量が必要で，さらに $\begin{pmatrix} A_S & A \end{pmatrix}$ を掛ける際に，FMM による高速内挿を用いて $\Theta(M)$ の計算量が必要である．これをすべての標本点について計算するので，合計 $\Theta(n(m+M))$ の計算量が必要である．ベータ値の方はすなおに計算して合計 $\Theta(n^2)$ である．この計算量では，標本点集合の選択にしか使えない．

アルファ・ベータ積の最大値を用いる場合には minimax アルゴリズムを利用することができる．適当な上限・下限を $B = \Theta(1)$ で実現し，$F = \Theta(m+M)$ を用いれば，構

成・改良アルゴリズムに利用することは可能と思われる．但し，実際に実用的な計算量で済むかどうかは，計算量が少なく有効な上限・下限が得られるかどうかにかかっている．

さらに簡単な近似的目的関数を求めるために $A_S + A\Theta$ を分解する．A の第 i 列を a_i，A_S の第 j 列を $\bar{a}_j$ と表す．このとき，内挿後アルファ値 ${}_C\alpha_j$ は $\|\bar{a}_j + \sum_i a_i\theta_{ij}\|$ となる．これから

$$ {}_C\alpha_j \le \|\bar{a}_j\| + \sum_i |\theta_{ij}|\|a_i\| = \bar{\alpha}_j + \sum_i \alpha_i|\theta_{ij}| $$

が得られる．これは $\Theta(m)$ 個の項からなるので，$F = \Theta(m)$ で計算でき，minimax アルゴリズムを適用することができる．これを**項別和近似**と呼ぶことにする．

さらに，近似を進め，上式右辺の下限に相当する $\max\{\bar{\alpha}_j, \alpha_i|\theta_{ij}|\}$ も ${}_C\alpha_j$ の近似値として考慮に値する．これは cmax アルゴリズムによりすべての j に対する値を $\Theta(n+m)$ で求めることが可能である．これを**最大項近似**と呼ぶことにする．これらの近似的目的関数は，二乗和と最大値の両方に適用可能である．

4.5 シフト後アルファ・ベータ積とその近似値

シフト後アルファ・ベータ積も，分離またはシフトが含まれることを除けば，前節の内挿後アルファ・ベータ積とそれほど変らない．

シフトが含まれる場合，

$$ \alpha_j^l = \|\bar{a}_j^0\tau_j^{0l} + \bar{a}_j^1\tau_j^{1l} + \sum_i (a_i^0\theta_{ij}^0\tau_j^{0l} + a_i^1\theta_{ij}^1\tau_j^{1l})\| $$

($l = 0, 1$) となる．しかしこのとき内挿後アルファ・ベータ積で使ったような

$$ \begin{aligned} \alpha_j^l &\le \|\bar{a}_j^0\||\tau_j^{0l}| + \|\bar{a}_j^1\||\tau_j^{1l}| \\ &\quad + \sum_i (\|a_i^0\||\theta_{ij}^0\tau_j^{0l}| + \|a_i^1\||\theta_{ij}^1\tau_j^{1l}|) \\ &\ge \max\{\|\bar{a}_j^0\||\tau_j^{0l}|, \|\bar{a}_j^1\||\tau_j^{1l}|, \\ &\qquad \|a_i^0\||\theta_{ij}^0\tau_j^{0l}|, \|a_i^1\||\theta_{ij}^1\tau_j^{1l}|\} \end{aligned} $$

といった項別近似は，目的関数としてはあまり適切ではないことが分かっている．これは，分離された $\bar{a}_j^0$ と $\bar{a}_j^1$ や，a_i^0 と a_i^1 などのベクトルのペアがほとんど同じ方向を向いている場合があり，互いにキャンセルしてしまうことがあるためである．このようにベクトルの方向がそろうのは分離ルジャンドル関数の性質であるらしい．そこで，これを考慮して分離ルジャンドル関数のペアを分けずに

$$ \begin{aligned} \alpha_j^l &\le \|\bar{a}_j^0\tau_j^{0l} + \bar{a}_j^1\tau_j^{1l}\| + \sum_i \|a_i^0\theta_{ij}^0\tau_j^{0l} + a_i^1\theta_{ij}^1\tau_j^{1l}\| \\ &\ge \max\{\|\bar{a}_j^0\tau_j^{0l} + \bar{a}_j^1\tau_j^{1l}\|, \|a_i^0\theta_{ij}^0\tau_j^{0l} + a_i^1\theta_{ij}^1\tau_j^{1l}\|\} \end{aligned} $$

などの項別近似を用いることにする．この際，あらかじめ $(\bar{a}_j^0, \bar{a}_j^1)$ および (a_i^0, a_i^1) といった内積を計算しておけば，それぞれの項は $\Theta(1)$ の計算量で計算することができる．これらの内積は単位ベクトルに対して内挿とその転置演算を連続して作用させることにより計算することができる．従って，これらの内積の計算に必要な計算量は $\Theta((n+m)M)$ となり，従来のアルファ値の計算に必要な計算量と同じオーダーである．但し，この方法を用いると cmax アルゴリズムは適用できないことになる．

以上，この節では C ノルム，内挿後アルファ・ベータ積，シフト後アルファ・ベータ積とそれらの近似について論じてきた．これらはいずれも重要なファクターであるので，実際に標本点選択アルゴリズムに適用する場合にはこれら 3 つの要素を総合して一つの目的関数を構成する必要がある．複数の目的関数の組み合わせ方はさまざまなものが考えられる．重みつき和には多少問題があるが，ある程度有効なようである．最大値は計算が不安定になりやすい．

5. さいごに

本稿では，我々が提案している高速球面調和関数変換法における安定性制御の手法について考察した．安定性制御の目的を明確化し，標本点集合を構成・改良・選択の 3 つのアルゴリズムで最適化することを提案した．また，C ノルム，内挿後アルファ・ベータ積，シフト後アルファ・ベータ積の 3 つが適切な目的関数であることを明らかにし，それぞれを計算量的な視点から考察した．さらに，minimax と cmax の 2 つの高速アルゴリズムを提案した．

本来ならば提案手法の実装と評価をここで報告すべきところであるが，著者の力不足でそれに至ることができなかった．しかし，理論的な部分についてはかなり議論が明確になったと思う．しかし実装の際には上限・下限関数を具体的に定義しなければならないし，それらを計算するために各種の前処理も必要になる．これらの検討と実装に目的関数自身のそれらの数倍の労力を必要とする．今後は提案手法の実現と評価に全力で取り組むつもりである．

謝辞

あらゆる意味で私達の研究を支えてくださっている杉原正顯教授に深く感謝いたします，また，有益なコメント下さっている未来開拓のプロジェクトの皆様（特に金田行雄教授，赤堀浩司氏），HPC 研究会の参加者の皆様にも感謝いたします．

この研究の一部は，学振未来開拓（地球規模流動現象解明のための計算理工学），文部科学省科研費，豊田理化学研究所の支援を受けています．

参 考 文 献

1) D. M. Healy Jr., D. Rockmore, P. J. Kostelec, and S. S. B. Moore, “FFTs for 2-Sphere — Improvements and Variations”, Tech. Rep. PCS-TR96-292, Dartmouth Univ., 1996.
2) 須田礼仁,「高速球面調和関数変換法」，情報処理学会研究報告 98-HPC-73，1998 年 10 月, pp. 37–42.
3) R. Suda and M. Takami, “A Fast Spherical Harmonics Transform Algorithm”, to appear in Math. Comp.
4) 須田礼仁，高見雅保,「高速球面調和関数変換法の誤差の解析と制御」，情報処理学会論文誌 ハイパフォーマンス，HPS-4 (掲載予定).