ハイパフォーマンス
コンピューティング 84－2
（2000．12．8）

高速球面調和関数変換法の誤差の解析と制御

須田礼仁† 高見雅保†

球面調和関数変換は通常、切断波数 M に対して計算量が $O(M^3)$ となる直接法で計算されている。これに対し、我々は計算量が $O(M^2 \log M)$ となる高速変換アルゴリズムを提案し、実装・評価をしてきた。今回は、我々のアルゴリズムの近似誤差の解析と制御について報告する。まず、行列とノルムを用いた誤差評価の方法を導入し、複数の近似の相互作用の解析方法を提案する。続いて誤差の制御の方法と、それに伴い必要となる、標本点選択アルゴリズムの変更、行ドロッピングの導入について説明する。最後に、提案手法によりが安定に誤差を制御できることについて、裏付け的な考察を行う。

Error Analysis and Control of Fast Spherical Harmonics Transform

REIJI SUDA † and MASAYASU TAKAMI†

Spherical harmonics transform is usually computed directly in time $O(M^3)$ for cut-off frequency M. We have proposed a fast transform algorithm which is accelerated by FMM (Fast Multipole Method) through polynomial interpolation and runs in time $O(M^2 \log M)$. In this paper, we discuss on analysis and control of the approximation errors of our algorithm. First a method of error estimation with matrix norm is introduced, then the algorithms of error control, sampling point selection, and row dropping are explained. We also discuss on the stability of our algorithm, and show that the numerical stability is guaranteed by the split function scaling.

1. はじめに

球面調和関数変換はフーリエ変換とルジャンドル陪関数変換に分解される。フーリエ変換の部分は高速フーリエ変換法 (FFT) により、切断周波数 M に対して $O(M^2 \log M)$ の計算量で計算することができる。しかしルジャンドル陪関数変換に対しては、単純で効率のよい高速変換法が存在しないため、通常は $O(M^3)$ の計算量がかかる直接法が用いられている。

これに対し、ルジャンドル陪関数変換を高速に行うためのアルゴリズムの研究がいくつか行われてきた。中でもD-H 法と呼ばれるフーリエ変換を利用した高速アルゴリズム[5]と、Mohlenkamp による高速ウェーブレット変換を利用したアルゴリズム[6]は、ライブラリとして実現されつつある（その他の研究については紙面の都合で述べる余裕がないので、参考文献を参照されたい）。但し、D-H 法はそのままでは数値的に不安定であり、Mohlenkamp のアルゴリズムは速度的に課題が残っている。我々は高速多重極子展開法 (FMM) を用いた、計算量が $O(M^2 \log M)$ のルジャンドル陪関数の安定で高速な近似変換アルゴリズムを提案・評価してきた[1)~4)]。

我々のアルゴリズムは近似アルゴリズムであるため、誤差の解析と制御はアルゴリズムの不可欠な要素である。しかし、これまでは実装の結果の精度しか評価をしてきておらず、解析と制御は行われてこなかった。今回、ようやく誤差の解析と制御についてある程度の知見が得られたので、それについて報告をしたい。

† 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, Nagoya University

2. アルゴリズムの行列表現

ルジャンドル陪関数変換

$$g^m(\mu_i) = \sum_{n=m}^{M} g_n^m P_n^m(\mu_i) \qquad (1)$$

は、行列 $(A)_{ij} = P_j^m(\mu_i)$ とベクトル $(x)_j = g_j^m$ との積とみなすことができる。我々の高速変換アルゴリズムは、この行列 A をさまざまな方法で分解・分離し、その一部の計算を FMM で高速化している。この節では、我々のアルゴリズムの構成要素と、その行列表現について述べる。

2.1 内　　挿

我々のアルゴリズムで最も重要な部分は、FMM による多項式の高速内挿である。ルジャンドル陪関数 $P_n^m(x)$ はある $m-n$ 次多項式 $q_n^m(x)$ を用いて

$$P_n^m(x) = P_m^m(x) q_n^m(x)$$

のように表現できる。このためルジャンドル陪関数変換 (1) から $g^m(x)/P_m^m(x)$ を計算すると $M-m$ 次の多項式になる。従って、適当な $M-m+1$ 点上での関数 $g^m(x)$ の値が分かっていれば、任意の点での関数値は**内挿**により計算することができる。Lagrange の内挿公式によれば、この内挿は

$$g^m(y_i) = P_m^m(y_i) \sum_j \frac{\omega_j(y_i)}{\omega_j(x_j)} \frac{g^m(x_j)}{P_m^m(x_j)}$$

と表現できる。ここで

$$\omega(x) = \prod (x - x_j)$$
$$\omega_j(x) = \omega(x)/(x - x_j)$$

である。行列表示をするために

$$(A^{\mathcal{Z}})_{ij} = P_j^m(z_i) \quad z_i \in \mathcal{Z}$$
$$P^{\mathcal{Z}} = \mathrm{diag}(P_m^m(z_i)) \quad z_i \in \mathcal{Z}$$
$$(\Omega^{\mathcal{X}\mathcal{Y}})_{ij} = \omega_j(y_i)/\omega_j(x_j) \quad x_j \in \mathcal{X},\ y_i \in \mathcal{Y}$$

とすると、内挿によるルジャンドル陪関数変換は

$$A^{\mathcal{X}+\mathcal{Y}} = \begin{pmatrix} I \\ P^{\mathcal{Y}} \Omega^{\mathcal{X}\mathcal{Y}} (P^{\mathcal{X}})^{-1} \end{pmatrix} A^{\mathcal{X}}$$

と表現できる。ここで I は単位行列である。

2.2 分　　割

さらに $A^{\mathcal{X}}$ の計算も高速化するために、分割統治法を利用する。次数 n の範囲 $\mathcal{N}$ を適当に**分割**して $\mathcal{N} = \mathcal{N}_0 + \mathcal{N}_1$ とする。これは

$$(A_{\mathcal{N}}^{\mathcal{X}})_{ij} = P_j^m(x_i) \quad j \in \mathcal{N}, x_i \in \mathcal{X}$$

とすると

$$A_{\mathcal{N}}^{\mathcal{X}} = \begin{pmatrix} A_{\mathcal{N}_0}^{\mathcal{X}} & A_{\mathcal{N}_1}^{\mathcal{X}} \end{pmatrix}$$

ということになる。さらに、ベクトルの方も分割して

$$A_{\mathcal{N}}^{\mathcal{X}} x_{\mathcal{N}} = A_{\mathcal{N}_0}^{\mathcal{X}} x_{\mathcal{N}_0} + A_{\mathcal{N}_1}^{\mathcal{X}} x_{\mathcal{N}_1}$$

のように表記した方が後で都合がよい。分割された 2 つの行列のうち、次数の範囲が下半分となっている $A_{\mathcal{N}_0}^{\mathcal{X}}$ には再び内挿と分割統治法が適用できる。

2.3 分離と分離内挿

一方の $A_{\mathcal{N}_1}^{\mathcal{X}}$ にはそのままでは内挿が適用できない。これは $P_m^m(x)$ で割っても低い次数の多項式に帰着できないためである。これを解決するには、**split Legendre function** を導入する必要がある。ルジャンドル陪関数は

$$P_n^m(x) = P_{n,\nu}^{m,0}(x) + P_{n,\nu}^{m,1}(x)$$
$$P_{n,\nu}^{m,l}(x) = P_{\nu+l}^m(x) q_{n,\nu}^{m,l}(x)$$

のように**分離**することができる。ここで、$q_{n,\nu}^{m,l}(x)$ は次数が $|n - \nu + l - 1| - 1$ 次の多項式である。これに従いルジャンドル陪関数変換も

$$g^m(x) = \sum_{l=0,1} P_{\nu+l}^m(x) \sum_n g_n^m q_{n,\nu}^{m,l}(x)$$

のように分離される。これは

$$(S_{\mathcal{N},\nu}^{\mathcal{X},l})_{ij} = P_{j,\nu}^{m,l}(x_i)$$

を用いると

$$A_{\mathcal{N}}^{\mathcal{X}} = \begin{pmatrix} I & I \end{pmatrix} \begin{pmatrix} S_{\mathcal{N},\nu}^{\mathcal{X},0} \\ S_{\mathcal{N},\nu}^{\mathcal{X},1} \end{pmatrix}$$

と表現できる。数値的な安定性と計算量の両方を考慮に入れると、ν は $\mathcal{N}$ の最小値に設定するのがよい。

分離を行うと、効率的な内挿がいつでも行える。これを行列であらわすと

$$\begin{pmatrix} S_{\mathcal{N},\nu}^{\mathcal{X}+\mathcal{Y},0} \\ S_{\mathcal{N},\nu}^{\mathcal{X}+\mathcal{Y},1} \end{pmatrix} = \begin{pmatrix} I & 0 \\ \Theta^0 & 0 \\ 0 & I \\ 0 & \Theta^1 \end{pmatrix} \begin{pmatrix} S_{\mathcal{N},\nu}^{\mathcal{X},0} \\ S_{\mathcal{N},\nu}^{\mathcal{X},1} \end{pmatrix}$$

となる。但し、

$$\Theta^l = P_{\nu+l}^{\mathcal{Y}} \Omega^{\mathcal{X}\mathcal{Y}} (P_{\nu+l}^{\mathcal{X}})^{-1}$$

である。

2.4 分離点のシフト

分離しているルジャンドル陪関数変換に対して分割統治法を適用したときには、分割されたうち少なくとも一方に対しては分離点を ν とは異なる値にしなければならなくなる。新しい分離点を μ としたとき、

$$T_{\nu,\mu}^{l,\lambda} = \mathrm{diag}(P_{\nu+l,\mu}^{m,\lambda}(x_i)/P_{\mu+\lambda}^m(x_i)) \tag{2}$$

を用いて

$$\begin{pmatrix} S_{\mathcal{N},\nu}^{\mathcal{X},0} \\ S_{\mathcal{N},\nu}^{\mathcal{X},1} \end{pmatrix} = \begin{pmatrix} T_{\nu,\mu}^{0,0} & T_{\nu,\mu}^{1,0} \\ T_{\nu,\mu}^{1,0} & T_{\nu,\mu}^{1,1} \end{pmatrix} \begin{pmatrix} S_{\mathcal{N},\mu}^{\mathcal{X},0} \\ S_{\mathcal{N},\mu}^{\mathcal{X},1} \end{pmatrix}$$

により**分離点のシフト**をすることができる。

3. 誤差解析の方針

球面調和関数変換の計算は行列の演算であるから、標準的な行列演算の誤差解析の手法を用いることができる。以下では行列のノルム解析を用いて誤差の解析を行うが、行列とベクトルのノルムはすべて一貫性を満たしているものを仮定する。また、行列 A に対して、近似が含まれた計算を $\tilde{A}$ で表現するものとする。丸め誤差は近似の誤差よりも十分小さく、無視できるものとする。

3.1 誤差の解析

前節で説明したように、我々のアルゴリズムには (1) 内挿, (2) 分割, (3) 分離, (4) シフトがあり、これに (5) 直接計算を加えた 5 つが主な計算である。

分割: 前節で見たように、行列の分割に従ってベクトルも分割してやると、分割は行列・ベクトルの和として表現できる。演算 $A + B$ に対して、近似の誤差は

$$\|(\tilde{A} + \tilde{B}) - (A + B)\| \le \|\tilde{A} - A\| + \|\tilde{B} - B\|$$

となるから、分割されたそれぞれの演算について誤差を解析・制御してやればよい。分割自身は誤差を考える必要は（丸め誤差を無視すれば）ない。

分離: 分離についても、行列表現が $\begin{pmatrix} I & I \end{pmatrix}$ であるから、それ自身の誤差を考える必要はない。

シフト: 分離点のシフトについても近似は入れられないので、丸め誤差まで正しい計算をしていると仮定することができる。但し、シフトを表す行列は分離のような簡単なものではないので、これが他の原因で生じた誤差をどのように伝播するかについて検討を要する。

直接計算: 直接計算でも近似が入る。これは、ルジャンドル陪関数 $P_n^m(x)$ の次のような性質によるものである。ルジャンドル陪関数 $P_n^m(x)$ は x が 1 に近づくと、およそ $(1-x)^{m/2}$ の速さでダンプする。これは $n \leq 2m$ の範囲で顕著に現れる。このため、大規模な問題では倍精度浮動小数では関数値を表現しきれずにアンダーフローしてしまうし、組立除法で変換計算を行うとオーバーフローを起こしてしまう。この問題を解決するためには、結果にほとんど影響を及ぼさないほど関数値が小さくなる (m, n, x) の範囲について、ルジャンドル陪関数変換から計算を省略してしまわなければならない。この計算の省略を、以下では**ドロッピング**と呼ぶ。直接計算では、このドロッピングで発生する誤差の大きさと、発生した誤差の出力への伝播について、解析と制御を行う必要がある。

内挿: 内挿では FMM を用いて近似計算をしているので誤差が発生する。さらに、内挿は常に計算の途中にあるので、他の計算で発生した**誤差の伝播**についても考慮しなければならない。

3.2 単純な誤差解析はなぜうまくいかないか

問題がすべて相対誤差だけで話が済めば簡単である。変換結果の各点の相対誤差を抑えるためには、高速アルゴリズムを行列の近似とみなしたときに、行列の各要素が一定の相対誤差で近似されていなければならない。しかし、我々のアルゴリズムでは直接計算にドロッピングを取り入れているために、これは決して満足されない。我々の近似アルゴリズムのような行列計算の誤差の制御は、絶対誤差によるか、(実質的には同じであるが) ノルムの相対誤差によらなければ不可能である。

一方で、絶対誤差ですべてが済めばこれも話は簡単であるが、これもうまくゆかない。高速アルゴリズムの中で出てくる行列のノルムがすべて適当なレベルに抑えられれば、それに従って絶対誤差も抑えられる。しかし、分離点シフトを表現する行列 (2) のノルムは、分離点でのルジャンドル陪関数 $P_{\mu,\lambda}^m(x)$ が (たまたま x が零点に近いために) 小さいときに、非常に大きくなってしまうのである。

このように、相対誤差や絶対誤差による単純な解析はうまくゆかない。しかし、以下に示すように、**スケーリング**を用いることにより、我々のアルゴリズムの誤差の解析と制御を行うことができることが分かってきた。

4. 誤差の制御

4.1 誤差を考える範囲

近似の誤差を考える必要がある演算は、内挿と直接計算である。これらの計算はいずれも行列で表現できる。分割統治法の分割に従ってベクトルも分割すると行列計算が和に分解でき、分割された部分和の誤差を個別に扱えばよい。従って、以下では分割統治法で分割された次数 $\mathcal{N}$ の範囲のうち、考えている演算に関する部分だけを考察する。

直接計算: 分割統治法の分割部分のうち、ある直接計算 D に関係ない部分を省略してしまうと、誤差を考えるべき計算は D の出力を利用する部分だけでよいことになる。この部分を適当な行列 A で表現すると、誤差を考えるべき計算全体は AD と表される。この A は直接計算 D の出力を内挿によりすべての評価点の上での値を求める計算に相当する。

内挿: 同様に、ある内挿計算 C に関して誤差を考えるべき計算の全体は、適当な行列 A, B を用いて ACB と表すことができる。このうち B は内挿の入力となる標本点上での関数値を計算する部分、A は内挿の結果をさらに内挿してすべての評価点での値を計算する部分に相当する。

4.2 近似の相互作用の扱い

直接計算: 直接計算 D の近似を行う際に考えるべき計算は、ある行列 A を用いて AD と表される。ここで A は直接計算 D の結果をすべての評価点上に内挿する計算であるが、これは厳密な内挿ではなく、近似を含む内挿 $\tilde{A}$ としてやることが適当である。こうすることで、内挿の近似と直接法の近似との相互作用をふくめて

$$\|\tilde{A}\tilde{D} - AD\| \leq \|\tilde{A}(\tilde{D} - D)\| + \|(\tilde{A} - A)D\|$$

のように誤差を評価・制御することができるからである。右辺第 1 項は、直接計算の誤差を制御するときには、近似を含む内挿に基づいて行うべきことを示しており、右辺第 2 項は、内挿の誤差を制御するときには、右につく直接計算は厳密な計算を仮定して制御すべきことを示している。

内挿: 内挿 C の近似を扱う場合は、A には近似を含む計算を、B には厳密な計算を用いるべきである。これにより、

$$\begin{aligned}\|\tilde{A}\tilde{C}\tilde{B} - ACB\| \leq \|\tilde{A}\tilde{C}(\tilde{B} - B)\| \\ + \|\tilde{A}(\tilde{C} - C)B\| + \|(\tilde{A} - A)CB\|\end{aligned}$$

のように誤差が評価できる。

このように、制御の対象となっている行列の左につく行列 A は近似を含むものを考える。そこで、本論文では $\tilde{A}$ の代わりに単に A を用いる。

4.3 誤差制御のアルゴリズム

直接計算: 直接計算の誤差制御においては、$\|A(\tilde{D} - D)\|$ をある定数 ϵ で抑えることが目標となる。単純に

$$\|A(\tilde{D} - D)\| \leq \|A\|\|\tilde{D} - D\|$$

としたくなるが、分離点シフトのために $\|A\|$ は相当大きな数になってしまう可能性がある。

この問題を解決する 1 つの方法としては、$A(\tilde{D} - D)$ を

$$A(\tilde{D} - D) = \sum_i a_i(\tilde{d}_i^T - d_i^T)$$

のように分解することが考えられる。ここで a_i は A の i 列目、d_i^T は D の i 行目である。これに基づくと、

$$\|A(\tilde{D} - D)\| \leq \sum_i \|a_i\|\|\tilde{d}_i^T - d_i^T\| \qquad (3)$$

となり、A の列のノルムに応じて D の行の近似を制御してやればよいことになる。具体的に誤差の 2 ノルムを ϵ 以下に抑えるためには、

$$\|a_i\|_2\|\tilde{d}_i^T - d_i^T\|_2 \leq \epsilon/N \qquad (4)$$

としてやればよい。ここで N は D の行の数である。

ほぼ同じことを行列ノルムを使ってもできる。

$\alpha_i = \|a_i\|$

$S_A = \mathrm{diag}(\alpha_i)$

を定義してやると、

$$\|A(\tilde{D} - D)\| \leq \|AS_A^{-1}\|\|S_A(\tilde{D} - D)\| \qquad (5)$$

となる。以下では α_i を**アルファ値**、S_A を**スケーリング行列**と呼ぶことにする。アルファ値 α_i は、直接計算の結果の第 i 要素が、最終的なルジャンドル陪関数変換の出力に与える影響の大きさを示している。

この誤差制御式は、S_A の対角要素が α_i 以外のものでも成立する。一般に、行列積のノルムをノルムの積で抑える不等式はかなり緩いが、スケーリングを適切にしてやれば（つまり行列の条件をよくすれば）ある程度タイトにすることができる。ここでは、不等式 (5) ができるだけタイトになるようなスケーリング行列を選ぶのが望ましい。アルファ値を用いたスケーリング行列は AS_A^{-1} の各列のノルムを一様にするが、これは行列の条件を改良する方法のもっとも基本的なものであり、不等式をある程度タイトにすることが期待できる。

具体的に 2 ノルムを用いて誤差を抑えるには、

$$\|S_A(\tilde{D} - D)\|_2 \leq \|S_A(\tilde{D} - D)\|_F \leq \epsilon/\|AS_A^{-1}\|_2$$

とすればよい。ここでスケーリング行列にアルファ値を用いれば $\|AS_A^{-1}\|_2 \leq \sqrt{N}$ であるから、式 (4) は十分条件になる。つまり、行列ノルムによるバウンド (5) は行ごとに分解したバウンド (3) より決して悪くない。

内挿: 内挿について同様に考えると、

$$\|A(\tilde{C} - C)B\| \leq \sum_{i,j} \|a_i\|\|\tilde{c}_{ij} - c_{ij}\|\|b_j^T\|$$

あるいは

$$\|A(\tilde{C} - C)B\| \leq \|AS_A^{-1}\|\|S_A(\tilde{C} - C)S_B\|\|S_B^{-1}B\|$$

のように制御することができる。**アルファ値、ベータ値、スケーリング行列**という呼び方も同様に用いることにする。ベータ値 $\beta_j = \|b_j^T\|$ は、内挿の第 j 標本点での関数値の大きさを表している。また、順変換では行列を転置して使うので、この内挿（内挿点での情報を標本点に圧縮していることになる）の結果の第 j 要素が、最終的なルジャンドル陪関数順変換の結果に与える影響の大きさを示していることにもなる。

以上のように、アルファ値、ベータ値を用いてスケーリングする方法を、**アルファ・ベータ・スケーリング**と呼ぶことにする。

4.4 標本点選択アルゴリズムの修正

内挿の誤差制御で FMM などにより近似をするとき、計算量は近似の相対精度に依存すると考えられる。要求される精度は絶対精度であるので、計算量をできるだけ抑えるためには相対精度の分母となる $\alpha_i\beta_j|c_{ij}|$ が小さいことが望ましい。

これまでに提案してきた標本点選択のアルゴリズムは、$|c_{ij}|$ をできるだけ小さくするように設計されてきた。これまでの経験からは、これでもそれほど悪くないことがわかっている。しかし、α_i と β_j の効果を考慮に入れたほうが、より $\alpha_i\beta_j|c_{ij}|$ を小さくでき、計算量を減らすことができる可能性がある。

まず、非分離のときには

$$c_{ij} = \frac{P_m^m(y_i)\omega_j(y_i)}{P_m^m(x_j)\omega_j(x_j)}$$

となる。ここで従来のアルゴリズム[2]のように $P_m^m(x)\omega_j(x)$ を

$$w(x) = P_m^m(x)\prod_{k<j}(x - x_k)$$

で近似してやることにする。さらに α_i と β_j が入っているので、j 番目の標本点として

$$\gamma_j = \max_{i \neq j}\{\alpha_i\beta_j w(y_i)/w(x_j)\}$$

で定義される γ_j を最小にする x_j を選ぶことになる。

分離のときには $l = 0, 1$ に対する内挿を同時に考え、

$$\gamma_j = \max_{i \neq j}\max_{l=0,1}\{\alpha_i\beta_j w^l(y_i)/w^l(x_j)\}$$

を最小にする x_j を j 番目の標本点として選ぶ。いずれも具体的なアルゴリズムは従来の方法[2),4)]を拡張することで構成できるので省略する。

このアルゴリズムでは、$\alpha_i P_\nu^m(y_i)$ が大きいものと $\beta_j/P_\nu^m(x_j)$ が小さいものが標本点に選ばれやすい。前者は出力への影響が大きいもので、これを標本点に選ぶことで近似の対象からはずす効果があるのであろう。後者は精度のよい入力を選ぶ、あるいは順変換で誤差が拡大しにくいものを選ぶということだろうか。いずれにせよ、$\alpha_i \approx \beta_i^{-1}$ となっていれば、両方の効果が一致してすなおに標本点が選ばれる。また、$\alpha_i\beta_i$ が小さいものは計算への影響が小さいので、どのように計算してもあまり問題にならない。しかし、$\alpha_i\beta_i$ が大きいものは計算結果への影響が大きいことを意味する。これを内挿点に選べば逆変換で、標本点に選べば順変換で、誤差の拡大を招く。従って、$\max_i\{\alpha_i\beta_i\}$ が大きくなるほど、内挿における精度を確保するのが難しくなる。つまり、この値は内挿計算の安定性の指標となっている。この値をこの内挿における**最大アルファ・ベータ積**と呼ぶことにする。

4.5 行ドロッピング

上述の誤差制御を実装してみると、そのままではうまくゆかないことが分かる。それは、ルジャンドル陪関数のダンプのために、B のいくつかの行の全要素がアンダーフローを起こしてしまうことによる。このとき $\beta_j = \|b_j^T\|$ は 0 となり、S_B^{-1} が発散してしまう。この問題を避けるためには、以下に述べる**行ドロッピング**の処理が内挿計算の前に不可欠である。

内挿計算 ACB において、CB の部分を取り出すと、これは内挿 C の標本点と内挿点を合わせた全評価点においてルジャンドル陪関数変換を行う計算になっている。従って、これをある直接計算 D で表現することができ、全体は AD として表現することができる。

ここで先の直接計算の近似の誤差評価式 (3) を用いる。ここでもし $\|a_i\|\|d_i^T\|$ が十分に小さければ、計算 D における第 i 行の計算をすべて省略（行ドロッピング）しても、それによる誤差は十分に小さく抑えられる。そこで、この $\|a_i\|\|d_i^T\|$ が十分小さい行を行ごとドロッピングして、近似 $\tilde{D}$ を構成する。その上で、$\tilde{D}$ を内挿を用いて $\tilde{C}B$ で近似をする。

このようにアルゴリズムを変更すると、B は $\tilde{D}$ の一部の行を取り出した行列になる。従って、$\|a_i\|\|b_i^T\|$ は行ドロッピングの閾値よりも大きい。従って $\|a_i\|$ がオーバーフローするほど大きくならない限り、$\|b_i^T\|$ がアンダーフローすることはない。このように行ドロッピングを導入することにより、アンダーフローを起こすほど絶対値が変化するルジャンドル陪関数にも対応できるようになる。

5. アルゴリズムの安定性についての考察

すでに述べたように、内挿において最大アルファ・ベータ積 $\max_i\{\alpha_i\beta_i\}$ が大きくなると、要求される相対精度が高くなり、誤差を抑えるためにより多くの計算量が必要となる。直接計算においても同様で、$\max_i\{\|a_i\|\|d_i\|\}$ が大きくなると誤差を抑えるために計算量がかかるようになる。そして、これらの値が極端に大きくなると、要求精度が有効数字レベルを超えて、計算が不安定化してしまう。

ところが、これまでの数値実験では、最大アルファ・ベータ積はそれほど大きくなることはなく、安定に誤差の制御ができているようである。アルファ値は非常に大きくなるし、ベータ値もそれなりの値になるのに、積が小さく収まるのは、偶然ではなさそうである。その原因を現在のところでは次のように理解している。

5.1 非分離内挿の安定性

まず、分離が行われていない部分について考える。内挿 ACB に対して、新しい標本点アルゴリズムにより、$\|S_ACS_B\|$ を小さくするように標本点が選ばれる。このとき、S_ACS_B の各行・各列の大きさ（ノルム）が極端に異なることはノルムを小さくすることに対して有利でないであろう。すると、S_ACS_B の各行・各列の大きさはだいたいそろっているということが期待できる。

次の内挿は $\hat{A}=AC$, $\hat{C}\hat{B}=B$ という設定で行われる。このとき $\hat{S}_A$ は $\hat{A}=AC$ の各列のノルムに基づいて決められるが、AS_A^{-1} の列ノルムが一様で、S_ACS_B の行・列のノルムも大体一様だとすると、$AC=(AS_A^{-1})(S_ACS_B)S_B^{-1}$ の列のノルムは、大体 S_B^{-1} に応じた大きさになりそうである。つまり、$\hat{S}_A\approx S_B^{-1}$ である。S_B は 2 つの内挿に対して共通であるから、内挿 $\hat{C}$ に対して、アルファ・ベータ積はおよそ一定の値になることが期待できる。

5.2 分離開始時点での安定性

次に、非分離から分離に移るところについて考える。ここでは分離点のシフトはない。次の内挿 $\hat{C}$ を考えるところまではほぼ同じである。分離がなされるが、行列表現が $(\ I\quad I\ )$ であるから、先の考察によれば、

$$\hat{A}=\begin{pmatrix} AC \\ AC \end{pmatrix}$$

$$\hat{S}_A\approx\begin{pmatrix} S_B^{-1} & 0 \\ 0 & S_B^{-1} \end{pmatrix}$$

となる。しかし、新しい B スケール行列 $\hat{S}_B$ を計算する際、$\hat{B}$ は分離されているので、もとの S_B とは異なるものが得られる。この $\hat{B}$ は

$$\hat{B}=\begin{pmatrix} B^0 \\ B^1 \end{pmatrix}$$

のようになっているが、分離の性質上 $B^0+B^1=B$ である。従って、$b_i^0+b_i^1=b_i$ すなわち $\|b_i^0\|+\|b_i^1\|\geq\|b_i\|$ となる（簡単のため、行ベクトルを示す ^{T} を省略した）。

これは、アルファ・ベータ積が大きくなる可能性を示している。しかし、実際には $\|b_i^0\|+\|b_i^1\|$ は $\|b_i\|$ に比べて極端に大きくなることはなく、切断周波数 M に対して最大でも $\sqrt{M}$ 倍程度で抑えられているようである（但し、n 方向のスケーリングに依存する。この結果は、ルジャンドル陪関数の積分が定数になるように正規化されている場合である）。これを**分離の安定性**と呼ぶことにする。この安定性の原因はよく分かっていないが、古くから知られている組み立て除法による変換のアルゴリズムの安定性と密接な関係があることは確かである。

5.3 分離内挿の安定性

最後に、分離が行われている場合について考える。これまで同様、内挿の結果である ACS_B の列はおよそ一様になると期待する。次の内挿の前にシフトが入らない場合には、これまでと話は変わらない。

シフトが入る場合には、分離点シフトを表す行列を T として、次の内挿 $\hat{C}$ に対して $\hat{A}=ACT$, $\hat{B}=T^{-1}B$ となる。しかし、これだけから $\hat{S}_A\approx\hat{S}_B^{-1}$ とは言い難い。T は（各行・各列の非零要素数が 2 であるとしても）対角行列ではないからである。

しかし、実際にはこの $\hat{S}_A\approx\hat{S}_B^{-1}$ がおおよそ成り立っているらしい。$\hat{S}_A$ は $ACT\hat{S}_A^{-1}$ の各列のノルムを一様にする。ここで C は ACS_B の列ノルムを一様にすると期待すると、$\hat{S}_A^{-1}$ はほとんど $S_B^{-1}T$ の列の大きさで決まると期待される。すなわち、$S_B^{-1}T\hat{S}_A^{-1}$ の各列の大きさはほぼ一様になると期待される。この $S_B^{-1}T\hat{S}_A^{-1}$ の要素は $T^{l,\lambda}/(\beta^l\hat{\alpha}^\lambda)$ である。ここで

$$P_{n,\nu}^{m,l}=T_{\nu,\mu}^{l,0}P_{n,\mu}^{m,0}+T_{\nu,\mu}^{l,1}P_{n,\mu}^{m,1}$$

であることに注意する。これで両辺の絶対値を取ると

$$1\leq|T_{\nu,\mu}^{l,0}|\frac{|P_{n,\mu}^{m,0}|}{|P_{n,\nu}^{m,l}|}+|T_{\nu,\mu}^{l,1}|\frac{|P_{n,\mu}^{m,1}|}{|P_{n,\nu}^{m,l}|}$$

となる。この不等式も分離の安定性から、右辺はそれほど大きな値にはならない。

さて、β^l はこの分母に出てくる $P_{n,\nu}^{m,l}$ を要素とするベクトルのノルムである。一方、分子に出てくる $P_{n,\mu}^{m,\lambda}$ を要

素とするベクトルのノルムは $\hat{\beta}^{\lambda}$ となる。これらの値を乱暴に置き換えてしまうと

$$1 \approx |T^{l,0}|\frac{\hat{\beta}^0}{\beta^l} + |T^{l,1}|\frac{\hat{\beta}^1}{\beta^l}$$

となる。これから $S_B^{-1}T\hat{S}_B$ の要素が分離の安定性と同じ位の安定性を示すことが期待される。これと $S_B^{-1}T\hat{S}_A^{-1}$ とを比較すると、$\hat{S}_A^{-1} \approx \hat{S}_B$ となることが期待できる。

これで、すべての場合についてアルファ・ベータ積があまり大きくならないことの説明ができた。これらの説明は「証明」ではないが、実際に観測される安定性をある程度説明していると思われる。結論をまとめると、アルファ・ベータ積が拡大されてもせいぜい分離の不安定性と同程度であるということである。

5.4 分離関数スケーリング

さて、前節の考察では、$S_B^{-1}T\hat{S}_B$ の要素の大きさは、分離の不安定性と同じ程度であると期待された。そこで具体的なスケーリングとして $\zeta_n^l(x) = \|b_i^l\|_\infty$ を考える。つまり

$$\zeta_\nu^l(x) = \max_{\nu \le n \le M}\{|P_{n,\nu}^{m,l}(x)|\}$$

である。この ζ_ν^l によるスケーリングを**分離関数スケーリング**と呼ぶことにする。さらに

$$\tau_{\nu,\mu}^{l,\lambda}(x) = \zeta_\nu^l(x)^{-1}T_{\nu,\mu}^{l,\lambda}(x)\zeta_\mu^\lambda(x)$$

を考えると、これは分離点シフト係数を ζ でスケーリングしたものに他ならない。

数値実験によると $\tau_{\nu,\mu}^{l,\lambda}(x)$ の絶対値は最大でもおよそ $\sqrt{M}$ の程度にしかならない。この大きさは、分離の不安定性とほとんど一致している（これも n 方向のスケーリングに依存する）。

このようにスケーリングしてやると分離点シフトの係数がかなり小さく抑えられるということは、スケーリングしないときの分離点シフト行列のノルムの大きさは見かけのもので、実質的には不安定性の源にならないことを意味している。しかも $\zeta^l(x_i)$ と β_i^l の関係から、分離している直接計算に分離関数スケーリングを適用しても安定であることが分かる。非分離の場合には若干損をするが、分離の不安定性である $\sqrt{M}$ 倍程度である。内挿も分離関数スケーリングで安定になるように標本点を選ぶことができる。つまり、アルファ・ベータ・スケーリングに代えて分離関数スケーリングを統一的に用いると、我々のアルゴリズムの全体を安定に計算できることが保証できるのである。

しかし、分離関数スケーリングとアルファ・ベータ・スケーリングのどちらが有利かは自明ではない。アルファ・ベータ・スケーリングは非分離の内挿に対して有利であるが、分離された後で利益の払い戻しが必要となる。分離関数スケーリングは分離内挿に有利であるが、非分離内挿で犠牲を払わなければならない。分離関数スケーリングがグローバルに安定性を与えるスケーリングであるのに対して、アルファ・ベータ・スケーリングは局所的な最適性を目指したスケーリングである。この両者の間に極端な矛盾を生じないことが、我々のアルゴリズムの安定性の原因の１つであるらしい。しかもこの点は、組立除法による直交関数変換のアルゴリズムの安定性と密接な関係がある。我々のアルゴリズムは、組立除法により安定に変換が計算できる種類の直交関数変換に対して一般的に適用可能ではないかと期待される。

6. まとめと課題

本論文では、我々が提案してきている高速ルジャンドル陪関数変換法の誤差について考察をした。スケーリングを行えば、行列ノルムを用いて誤差が解析・制御ができることが分かった。また、我々のアルゴリズムの安定性の原因も少し明らかとなった。

本稿では数値実験の結果については全く報告することができなかった。現在のところ、提案する誤差制御法を実装し、誤差を制御することには成功している。しかし、提案手法の有効性を検証するためには、内挿・直接計算・行ドロッピングの個別の誤差が期待した通りの値になっているかどうか、不等式のゆるさはどの程度なのか、スケーリングの方法と誤差の拡大にはどのような関係があるのか、行列・ベクトルのノルムはどのように計算するべきなのか（精度と計算量の両面から）などについて、実装・評価をして行かなければならない。

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、有益なコメントを下さっている未来開拓のプロジェクトの皆様、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）、豊田理化学研究所、および文部省科研費の支援を受けています。

参 考 文 献

1) 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.
2) 高見雅保、須田礼仁、杉原正顯、「FMM による Legendre 陪関数変換の高速化」、情報処理学会研究報告 99-HPC-78、1999 年 10 月、pp. 1–6.
3) 高見雅保、須田礼仁、「Legendre 陪関数変換の高速計算に伴う FMM の改良」、豊田研究報告、第 53 巻、2000 年 5 月、pp. 77–84.
4) 須田礼仁、高見雅保、「高速球面調和関数変換法の精度と速度」、情報処理学会研究報告 2000-HPC-83、2000 年 10 月、pp. 19–24.
5) http://www.cs.dartmouth.edu/~geelong/sphere/
6) http://amath.colorado.edu/appm/faculty/mjm/libftsh.html