テクニカルノート

Orszagの高速Legendre多項式変換法の改良

森 明子[†,☆] 須田礼仁[†] 杉原正顯[†]

1986年にOrszagはLegendre多項式変換を含むSturm-Liouville固有関数変換の評価計算に対する高速算法を提案した．彼の算法は固有関数のWKB近似を利用して計算の一部をFFTで実行することにより，直接計算で$O(N^2)$かかるN次N点の評価計算を$O(N\log^2 N/\log\log N)$に改善できるものであるが，実際には計算に無駄が多く，精度も悪く実用的ではない．我々はLegendre多項式変換についてOrszagの算法の改良を提案する．我々の改良により，計算量は$O(N\log N)$に改善され，単精度程度の精度では$N\geq 256$で，倍精度程度の精度では$N\geq 512$で，それぞれ直接計算よりも高速になった．

An Improvement on Orszag's Fast Algorithm for Legendre Polynomial Transform

AKIKO MORI,[†,☆] REIJI SUDA[†] and MASAAKI SUGIHARA[†]

In 1986 Orszag proposed a fast algorithm for Sturm-Liouville eigenfunction transform including the Legendre polynomial transform. His algorithm, which exploits the speed of FFT through the WKB approximation, runs in time $O(N\log^2 N/\log\log N)$ for N-point evaluation of N-th order transform, while the direct computation requires $O(N^2)$ time. The algorithm is, however, not practical because of its low precision and algorithmic unsophisticatedness. In this note, we improve Orszag's algorithm for the Legendre polynomial transform. Our algorithm has computational complexity $O(N\log N)$, and is faster than the direct computation for $N\geq 256$ with relative error $\approx 10^{-8}$, and for $N\geq 512$ with relative error $\approx 10^{-14}$.

1. はじめに

近年，固有関数変換（Sturm-Liouville問題の固有関数で展開された関数の評価計算）が多用され，その高速化が重要な課題となっている．これに対して，Orszag[4]は固有関数のWKB近似に基づく高速変換法を提案した．しかし現在のところ実際にOrszagの算法を用いて応用計算をしたという報告は見当たらない．これは，WKB近似に基づく彼の算法は精度が悪く，アルゴリズムも複雑であるためと思われる．

しかし，Orszagの算法は，広範な固有関数変換に適用可能であるという他の高速変換法にない特長を持っており，さまざまなアプリケーションにおいて計算量の改善に貢献する可能性がある．そこで我々は，固有関数変換の1つであるLegendre多項式変換

$$A_j=\sum_{n=0}^{N-1}a_nP_n(\cos\theta_j)\quad(0\leq j<N)\qquad(1)$$

($\theta_j=(j+1/2)\pi/N$, $P_n(x)$はn次のLegendre多項式）を取り上げて，Orszagの算法の改良を試みた．その結果，近似公式の改良により高い精度を実現し，アルゴリズムの改良により計算量を削減することができた．本稿ではまずOrszagの算法とその問題点を説明した後，我々が提案する改良法と実装・評価の結果について報告する．

2. Orszagの算法とその問題点

2.1 Orszagの算法

1986年にOrszagは，固有関数変換，すなわちSturm-Liouville問題

$$\frac{d}{dx}\left(p(x)\frac{d}{dx}\psi_n(x)\right)+[\lambda_n w(x)-q(x)]\psi_n(x)=0$$

† 名古屋大学工学研究科
Graduate School of Engineering, Nagoya University
☆ 現在，株式会社日立製作所エンタープライズサーバ事業部
Presently with Enterprise Server Division, HITACHI, Ltd.

の固有関数 $\psi_n(x)$ で展開された関数の評価計算

$$A_j = \sum_{n=0}^{N-1} a_n \psi_n(x_j) \qquad (0 \le j < N) \qquad (2)$$

に対する高速算法を提案した．評価点 x_j が固有関数 $\psi_N(x)$ によって決まる特定の点のとき，固有関数 $\psi_n(x)$ に対するWKB近似

$$\psi_n(x_j) \approx (w(x_j)p(x_j))^{-1/4} \sin \xi_{nj} \qquad (3)$$
$$\xi_{nj} \approx nj\pi/N$$

はある定数 n_0 に対して

$$\min\{nj, n(N-j-1)\} > n_0 \qquad (4)$$

なる十分条件のもとで成立する．近似式(3)が必要な精度を満たす (n,j) の集合を S とすると，計算(2)は

$$A_j \approx (w(x_j)p(x_j))^{-1/4} \sum_{(n,j)\in S} a_n \sin\frac{nj\pi}{N} + \sum_{(n,j)\notin S} a_n \psi_n(x_j) \qquad (5)$$

のように近似できる．Orszagの算法は，式(5)の右辺第1項の一部にFFTを適用することによって高速化を実現する．

FFTを適用する具体的方法として，Orszagは適当な M_0, M_i, K_i $(1 \le i \le k)$ を用いて S に含まれる k 個の長方形

$$S^{(i)} = \{(n,j) \mid M_{i-1} < n \le M_i, \; K_i \le j < N - K_i\}$$

を定義して計算(2)を

$$A_j \approx \sum_{i=1}^{k} B_j^{(i)} + C_j^{(k)} \qquad (6)$$
$$B_j^{(i)} = (w(x_j)p(x_j))^{-1/4} \sum_{(n,j)\in S^{(i)}} a_n \sin\frac{nj\pi}{N}$$
$$C_j^{(k)} = \sum_{(n,j)\notin \cup_i S^{(i)}} a_n \psi_n(x_j)$$

のように近似し，各 $B_j^{(i)}$ $(1 \le i \le k)$ の計算に対してFFTを適用する方法を提案した．ここで長方形 $S^{(i)}$ のとり方を最適化すると漸近計算量が $O(N \log^2 N / \log\log N)$ となることが証明できる．

2.2 Orszagの算法の問題点

しかし実際にOrszagの算法を固有関数変換に適用しようとすると次のような問題が生じる．

(1) 精度

Orszagの論文はWKB近似の近似度に関してまったく触れていない．Legendre多項式について近似式(3)の近似精度を $N = 128$ から $N = 1024$ まで調べてみると，この範囲の N に対しては近似誤差がほとんど 10^{-2} 以上となり，実用精度には程遠い．この範囲の N では近似式(3)はまだ十分な精度で成立していないと見られる．

(2) アルゴリズム

Orszagの算法では，実際に近似が成立している範囲 S よりも狭い範囲にしかFFTを適用していない．また，式(6)のように多くの小領域に分割することにより計算量が余計にかかる．さらに，実際に計算するためにはFFTを適用する範囲 $S^{(i)}$ の具体的な設定が必要であるが，Orszagの論文にはこの設定方法についての明確な記述がない．

3. Orszagの方法の改良（問題点の克服）

前章で見たようにOrszagの算法はそのままでは実用にならない．本章では，Sturm-Liouville固有関数の1つであるLegendre多項式を取り上げ，精度とアルゴリズムに対する改良を提案する．

3.1 精度の改良（高精度近似式）

WKB近似式では精度が十分ではないので，Legendre多項式 $P_n(x)$ の高精度近似式でWKB近似式のようにFFTが適用できるものを調べた．その結果，次のStieljesによる漸近展開式（文献5) [p.193]）

$$P_n(\cos\theta) = P_n^{(p)}(\cos\theta) + O((n\sin\theta)^{-p-1/2}) \quad (0 < \theta < \pi) \quad (n \to \infty)$$
$$P_n^{(p)}(\cos\theta) = \frac{4}{\pi} \sum_{\nu=0}^{p-1} \frac{[(2\nu-1)!!]^2 (2n)!!}{(2\nu)!!(2n+2\nu+1)!!} \cdot \frac{\cos\{(n+\nu+1/2)\theta - (2\nu+1)\pi/4\}}{(2\sin\theta)^{\nu+1/2}} \qquad (7)$$

が良いことを見いだした．

3.2 アルゴリズムの改良

アルゴリズムについて，我々は以下のような改良を提案する．関数の評価点を $\theta_j = (j+1/2)\pi/N$ に選んでLegendre陪関数変換(1)を

$$A_j \approx \sum_{n=0}^{N-1} a_n P_n^{(p)}(\cos\theta_j) + \sum_{R_{nj}>\varepsilon} a_n R_{nj} \qquad (8)$$

と近似する．ここで右辺第1項にFFTを適用し，第2項は直接計算する．R_nj は近似誤差

$$R_{nj} = P_n(\cos\theta_j) - P_n^{(p)}(\cos\theta_j)$$

で，ε は精度を決定する閾値である．このアルゴリズムはFFTを全域に適用することにより近似の精度が

良い範囲に直接計算を持ち込まず無駄をなくしている．

3.3 提案算法の計算量

次に近似の次数 p を固定したときに，このアルゴリズムの計算量が $O(N\log N)$ となることを示す．式(8)の右辺第1項にはFFTを用いるので，計算量が $O(N\log N)$ であることは自明である．したがって，右辺第2項の計算量，すなわち $R_{nj}>\varepsilon$ になる (n,j) の範囲が $O(N\log N)$ であることを示せばよい．

近似次数 p を固定すると $R_{nj}=O((n\sin\theta)^{-p-1/2})$ となる[5]ことから，ある定数 n_p があって

$$n \geq n_p/\sin\theta$$

を満たす範囲で $R_{nj}\leq\varepsilon$ 以下になることが分かる．そこで，$n<n_p/\sin\theta$ となる範囲を**図1**のように

$$T_1=\{(n,j)\mid 0\leq j<j_p, 0\leq n<N\}$$
$$T_2=\{(n,j)\mid j_p\leq j<N-j_p, n<n_p/\sin\theta_j\}$$
$$T_3=\{(n,j)\mid N-j_p\leq j<N, 0\leq n<N\}$$

(j_p は $N\sin\theta_j<n_p$ となる最大の j) の3つの領域に分けて，それぞれの要素数を評価することにする．領域 T の要素数を $|T|$ で表すと，

$$|T_1|=|T_3|\approx n_pN/\pi$$
$$|T_2|\approx\frac{N}{\pi}\int_{\theta_{j_p}}^{\pi-\theta_{j_p}}\frac{n_p}{\sin\theta_{j_p}}d\theta$$

となる．T_1 と T_3 は $O(N)$ であり，T_2 の面積も積分を実行すれば $O(N\log N)$ となることは容易に分かる．以上により，全体の計算量は $O(N\log N)$ であることが証明された．

3.4 実装による提案算法の評価

次に提案算法の実装による評価について報告する．実装した計算機のCPUはalpha 21264 500 MHz，使用したコンパイラはDEC ccで最適化オプションは `-fast -non_shared -om` である．プログラミングにおいては，直接法も提案算法も，$\theta=\pi/2$ を中心とする対称性を用いることにより計算量をおよそ半分にする工夫を用いている．直接計算はflop（浮動小数点数演算）数が最小になるようにプログラミングした．提案算法のデータは，各サイズ N と閾値 ε に対し，近似次数 p を変化させて得られた計算量・計算時間の最小値である．

図2は直接法と提案算法のflop数を比較している．横軸にflop数をとっており，$N=128$, 256, 512の場合の直接法のflop数が縦の直線で示されている．縦軸は閾値 ε で，提案算法のflop数と ε との関係を点でプロットしてある．この図から，$N=256$ では単精度程度の精度が得られる $\varepsilon\leq 10^{-7}$ で，$N=512$ では倍精度程度の精度が得られる $\varepsilon=\leq 10^{-13}$ で，それぞれ提案算法が直接法よりも少ないflop数となることが分かる．**図3**は同様に計算時間を比較したもので，おおよそ同じような結果が得られている．

実験からStieltjesの近似公式が高い精度を与える

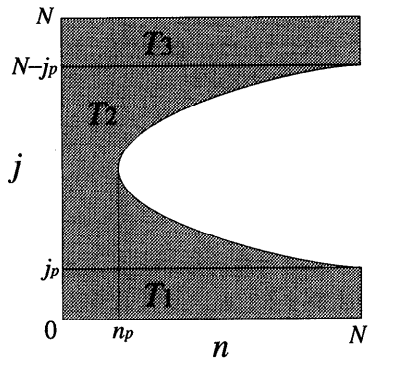


図1 計算量評価に用いる3つの領域

Fig. 1 Three regions used to estimate the computational complexity.

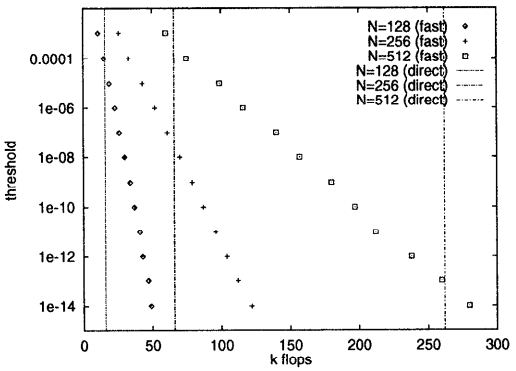


図2 直接計算と提案算法のflop数

Fig. 2 Flops counts for the direct computation and the proposed algorithm.

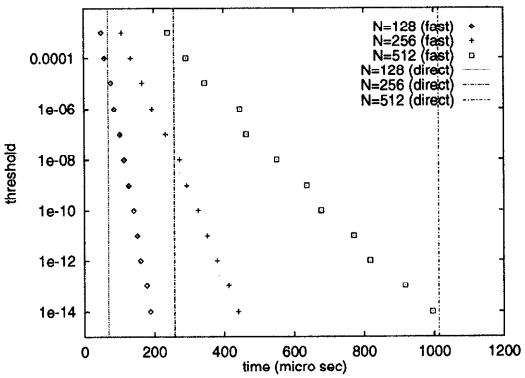


図3 直接計算と提案算法のCPU時間

Fig. 3 CPU times for the direct computation and the proposed algorithm.

ことが分かる．実験結果からは N を固定したときの提案アルゴリズムの計算量が $\log(1/\varepsilon)$ に比例しているように見受けられるが，この現象の理論的な裏付けについては今後検討をしてゆく予定である．

提案算法の相対誤差を，直接法と提案算法とで計算された関数値を並べたベクトルをそれぞれ b_D, b_F として，$\|b_D - b_F\|_2/\|b_D\|_2$ として評価した．展開係数 a_n には $[0,1)$ の一様乱数を用いた．その結果，相対誤差はつねにほぼ ε の 1/10 程度の数値となった．近似方法と誤差評価でノルムの違いがあるが，この結果は妥当なものといえる．しかし Stieltjes の近似式 (7) に現れる $(\sin\theta)^{-\nu-1/2}$ は $\theta \approx 0, \pi$ で発散するので，数値的な不安定性の原因となりうる．今回計算した範囲では不安定性は観測されなかったが，近似式をより安定なものに改良するなどの処置について検討の必要があるかもしれない．

4. まとめと考察

本稿では，Legendre 多項式変換の場合について，Sturm-Liouville 固有関数変換の高速算法である Orszag の方法を改良した．計算量は $O(N\log N)$ に改善され，単精度程度の精度では $N \geq 256$ で，倍精度程度の精度では $N \geq 512$ で，それぞれ直接法よりも高速となることが分かった．

本稿の方法は標本点が等分点の場合にしか適用できないが，Boyd の研究[3]と組み合わせることによって，他の任意の標本点での評価計算や，関数値から展開係数を求める展開計算も同じ計算量で可能となる．

Legendre 多項式変換に限ると，高速算法は Orszag のもの以外にもいくつかあり，計算量が $O(N\log N)$ のもの[1),2),6)]もある．しかし，他の算法に比べかなり簡単な計算で実現でき，高速化の原理が他の Sturm-Liouville 固有関数変換にも適用可能であるという点において，本算法は他の算法と一線を画する．他の変換における有効性は近似式 (7) に対応するものが得られるかどうかに依存する．近似度の良い解析的な式がない場合，最適近似を数値的に計算しそれを近似式に当てることも考えられる．今後は，本算法の考え方の基づく他の高速固有関数変換法の開発や近似式の最適化などについて研究を進めてゆく予定である．

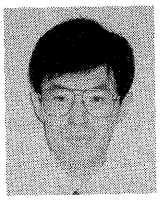
謝辞 本研究の一部は日本学術振興会未来開拓学術研究推進事業による．

参考文献

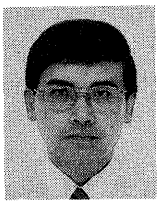
1) Alpert, B. and Rokhlin, V.: A Fast Algorithm for the Evaluation of Legendre Expansions, *J. Sci. Stat. Comput.*, Vol.12, pp.158–179 (1991).
2) Beylkin, G., Coifman, R.R. and Rokhlin, V.: Fast Wavelet Transforms and Numerical Algorithms I, *Comm. Pure Appl. Math.*, No.44, pp.141–183 (1991).
3) Boyd, J.P.: Multipole expansions and pseudospectral cardinal functions: A new generation of the fast Fourier transform, *J. Comp. Phys.*, Vol.103, pp.184–186 (1992).
4) Orszag, S.A.: Fast eigenfunction transform, *Science and Computers*, Rota, G.C. (Ed.), pp.23–30, Academic Press, New York (1986).
5) Szegö, G.: *Orthogonal Polynomials*, AMS, New York (1959).
6) 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，No.98-HPC-73, pp.37–42 (1998).

(平成11年5月27日受付)
(平成11年7月 1 日採録)

森 明子
1974年生．1997年岐阜大学工学部土木工学科卒業．1999年名古屋大学大学院工学研究科計算理工学専攻修士課程修了．現在日立製作所エンタープライズサーバ事業部勤務．



須田 礼仁（正会員）
1968年生．1991年東京大学理学部情報科学科卒業．1993年同大学院理学系研究科情報科学専攻修士課程修了．1996年同博士（理学）．現在名古屋大学大学院工学研究科計算理工学専攻講師．日本応用数理学会会員．



杉原 正顯（正会員）
1954年生．1977年東京大学工学部計数工学科卒業．1982年同大学院工学系研究科博士課程計数工学専攻修了．工学博士．現在，名古屋大学大学院工学研究科計算理工学専攻教授．日本応用数理学会，日本数学会各会員．