2007－HPC－113 (4)
2007／12／7

理論物理学計算への多倍長ライブラリの適用

濱口信行

nobuyuki.hamaguchi.sa@hitachi.com

(株) 日立製作所 ソフトウエア事業部

概要

理論物理学の計算に$IR-vertex$ダイアグラムがある.ここで,$IR-vertex$ダイアグラムは,ファイマンパラメータの解析積分を使用せずに,直接導き出される積分を言う.多倍長ライブラリを使用する事により,非常に小さい光子質量$10^{-160}GeV$で,解析解と10桁,$10^{-240}GeV$で8桁まで一致する積分結果を得る事が出来た.

Appling multi-precision library to Computation in Theoretical Physics

Nobuyuki Hamaguchi

Software Division,Hitachi,Ltd.

Abstract

We have succeeded in obtaining very precise integration results for IR-vertex diagrams, where the integration is carried out directly without performing any analytic integration of Feynman parameters. Applying the Multi-precision library to numerical integration, we have obtained integration results which agree with the analytic solution to 10 digits even for such a very small photon mass as 10^{-160} GeV and 8 digits for 10^{-240} GeV in the IR-vertex diagram.

1. はじめに

標準モデルのわく組みで,多重ループ積分の自動計算をめざし,その手始めに,scalar one-loop integral of three-point Vertex function [1] に着手した.この計算では,以前は,光子質量は10^{-15} GeV の範囲を扱っていたが,本来0で,積分が収束するために,導入されたものであるので,可能な限り小さい方が良い.10^{-15} GeV という値は演算精度からくる制限であり,目的を達成するため

には, 多倍長計算が必要となる. 積分方法は,Gauss-kronrod 法 [2], 誤差評価による区間 2 分割の Gauss-Legendre 法二重指数関数型積分法 [5], 算法は ε -算法を使用している.[1] [3]
演算時間の測定では, Intel Xeon 3.06GHz (2cpu L2 (512KB/cpu)), Os: Red Hat Enterprose Linux ES release 3 (taroon) Kernel: 2.4.21-4 Elsmp on an i686 Intel Fortran V8.1 -O2 , で 1 cpu で実行している.(注)

2. IR-vertex diagram

IR-vertex diagram を数式で表すと
$\lim_{\epsilon\to 0}\int_0^1\int_0^{1-x}\frac{1}{d(x,y)+ei}dy\,dx$ の計算となる.
ここで $d(x,y)=-xys+(x+y)^2m^2+(1-x-y)\lambda^2 \quad e=-xys\epsilon$ または $e=\epsilon$ を示す.

$\lim_{\epsilon\to 0}$ は, 特異点がある場合, 0 に収束する ϵ の点列 ϵ_n に関する積分計算を行い, 積分列 I_n を求め, $\epsilon\to 0$ の場合の I を外挿する事を示しており, 点列 ϵ_n は正の等比数列, $\frac{\epsilon_{n+1}}{\epsilon_n}<1$ をとっている.

実際の問題では, $s=q^2 \quad squared \quad energy \quad m \quad electron \quad mass \quad \lambda \quad photon \quad mass$ を表している.[4]
λ は $photon \quad mass$ より, 本来 0 なので, できるだけ小さな値で計算できる事が重要となる.

この積分の解析解は, $s<0, s>>m^2>>\lambda^2$ の場合, $I=\frac{1}{-s}[\ln(\frac{-s}{m^2})\ln(\frac{m^2}{\lambda^2})+\frac{1}{2}\ln^2(\frac{-s}{m^2})-\frac{\pi^2}{6}]$ で, $s>0$ の場合は s で解析接続をして,
実数部分は, $\frac{1}{-s}[\ln(\frac{m^2}{-s})\ln(\frac{\lambda^2}{m^2})+\frac{1}{2}\ln^2(\frac{m^2}{-s})-\frac{2\pi^2}{3}]$ となり,
虚数部分は, $\frac{1}{-s}\ln(\frac{\lambda^2}{s})$ となる.

この式の相対誤差は $|\frac{m^2}{s}|$ となる.

今回は, $s=\pm 500^2 \quad m=0.0005$ とし, λ を変化させているので, [4]
解析解のもつ相対誤差は 10^{-12} となる.

3. 二重指数関数型積分法による計算.

最終目標は, $s>0$ の場合を求めるのであるが, 演算精度, 計算結果の精度, 演算時間を見るために, $s<0$ の場合を検討した.
$s<0$, の場合, 被積分関数には, 特異点が無く, $-xys, (x+y)^2m^2, (1-x-y)\lambda^2$ の項は全て 0 または正となる.このため結果の精度に影響を与えるのは, 加算

時の桁落ちとなる.

二重指数関数型積分では,演算精度と分点数(きざみ幅)により,結果の精度と演算時間が定まるので,$s<0$ の場合の計算に最も適している.

λ を変化させ,解析解と相対誤差 10^{-6}以内となる結果を得るのに必要な分点数と演算時間をもとめた.(表 1)

表 1: 二重指数関数型積分による計算結果

演算精度	分点数	$\lambda=10^{-n}$ の n	実行時間 (秒)	同一演算量での実行時間比
4 倍精度	496	21	0.61	1
8 倍精度	1184	58	15.42	4.43
16 倍精度	2760	138	232.48	12.30
32 倍精度	6240	291	4915.67	50.92
32 倍精度	14600	1000 (注 1)	26534.61	50.20
32 倍精度	29200	2000 (注 1)	106241.14	50.25
4 倍精度	49700	2500 (注 2)	5318.81	0.87

(注 1) 変数変換区間を $[-1,1]$ から $[0,1]$ に変更している.
また,$\lambda=10^{-160}$の様に,n が大きいと,指数部が 11 ビットだと,λ^2が表現出来なくなるので,指数部の拡張が必要となる.
(注 2) 指数部が 15 ビットでも,λ^2が表現出来なくなるので,その回避策と,高速化 (高速,高精度総和計算を使用) を施している.

4. GAUSS 型積分法による計算.

$s>0$ では被積分関数には特異点があり,その特異点はかならずしも端点ではないので,$GAUSS$ 型積分法が適している.算法は ε − 算法を使用する.ただし, ε − 算法で精度の良い結果を得るには初期値の選択に以下の注意点がある.

(1)ϵの初期値を小さくとると個々の積分の精度が悪くなる.
(2)ϵの初期値を大きくとると ε 算法の数列の収束が悪くなる.

今回の問題での初期値は以下の様に定めている.
$s>0, s>4m^2, s>>\lambda$ とする.
$e=\epsilon$とすると,積分した結果に $\ln(\lambda^4+\epsilon^2)$ の項が現れる.
これがϵの漸近展開をなすには,$\lambda^4>\epsilon^2$,すなわち,$\epsilon<\lambda^2$.
この事から, $\varepsilon_0=\lambda^2$としている.

$s > 0$ の場合は $s < 0$ を解析接続したものであるから,必要な演算精度は $s < 0$ の結果をもとにした.$s > 0$ だと,近接する数の減算の桁落ちが加わるので,$\lambda = 10^{-20}$までのλに対し,4 倍精度で実行し,精度に対する影響を見た.
(倍精度だと 10^{-9}でも十分な精度がでない事は,$s < 0$ の結果より分かる.)

(1) 比較的大きなλでの計算

$\lambda = 10^{-9}$から 10^{-20}までの計算では $e = \epsilon$で計算した.結果を表 2 に示す.

表 2: 4 倍精度演算での実数部誤差評価

λ	解析値	実測値
10^{-9}	-0.44013021324646545 41D-02	-0.44013018600037737 5D-02
10^{-12}	-0.59282487631777760 50D-02	-0.59282487441356334 0D-02
10^{-15}	-0.74551954189733904 47D-02	-0.74551949090186498 8D-02
10^{-20}	-0.10000106512007698 1D-01	-0.10005049189025186 2D-01

$\lambda = 10^{-20}$の場合が精度の良くない結果となっている.これを 8 倍精度で実行した結果は $-0.1000010647321952 17D-01$ で良い精度となっている.演算時間は,4 倍精度の 6 倍の演算量で,65913 秒 (4 倍精度 3254 秒), で演算量を削減する方式を検討する必要が生じた.

$s > 0$ は $s < 0$ を解析接続したものであるため,e=-xy ε として計算すると,$\ln(\lambda^4 + \varepsilon^2)$ の項が,$\ln \lambda^4$ となり,$s > \varepsilon$ まで, 大きく取れ, 精度を落とす事なく (むしろ良くなり), かつ演算量を削減できる.

実測結果は以下の様になっている.

表 3: $e = -xy\epsilon$ とした時の実数部誤差評価

λ	解析値	実測値
10^{-9}	-0.44013021324646545 41D-02	-0.44013021345763848 5D-02
10^{-12}	-0.59282487631777760 50D-02	-0.59282487261014749 1D-02
10^{-15}	-0.74551954189733904 47D-02	-0.74551964139596940 0D-02
10^{-20}	-0.10000106512007698 1D-01	-0.99993381801278475 4D-02

表 3 に示すとおり,表 2 で示した $e = \epsilon$の結果より,よくなっている.

$GAUSS$ 型の誤差評価による区間分割法は ε − 算法で実行した場合,精度により,大きく演算量が異なってくる.

上記 2 ケースでの演算時間を比較すると表 4 の様になっている.

表 4: 4 倍精度演算での実行時間 (秒)

λ	$e = \epsilon$	$e = -xy\epsilon$	性能向上比
10^{-9}	1006.713	505.57	1.99
10^{-12}	1819.767	542.21	3.36
10^{-15}	2684.821	505.80	5.31
10^{-20}	3254.224	522.48	6.22

表 4 に示すとおり, $e = -xy\ \varepsilon$ とすると演算時間が大幅に改善されている. この事から, 更に小さい λ での計算では, $e = -xy\ \varepsilon$ で行っている.

(2) 小さい λ での計算

$\lambda = 10^{-30}$ から 10^{-240} までの計算では $e = -xy\epsilon$ で計算した.結果を表 5 および表 6 に示す.演算精度をあげる事により, 10^{-160} までは相対誤差 10^{-10} 以下, 10^{-240} までは相対誤差 10^{-8} 以下と十分な精度となっている.

表 5: 実数部演算結果

λ	演算精度	解析値	実測値
10^{-30}	8 倍精度	-0.1508992869804822915D-01	-0.150899286980769753D-01
10^{-60}	16 倍精度	-0.3035939525622601542D-01	-0.30359395252562854951D-01
10^{-80}	16 倍精度	-0.4053903962834453960D-01	-0.405390396284235075D-01
10^{-160}	32 倍精度	-0.8125761711681863632 3D-01	-0.81257617075281066 6D-01
10^{-200}	32 倍精度	-0.10161690586105568468	-0.101616905857448388
10^{-240}	32 倍精度	-0.12197619460529273304	-0.121976194600966288

表 6: 虚数部演算結果

λ	演算精度	解析値	実測値
10^{-30}	8 倍精度	0.1892298396155253893D-02	0.18922983961589882 2D-02
10^{-60}	16 倍精度	0.3628406655134965448D-02	0.362840665514227622D-02
10^{-80}	16 倍精度	0.4785812161121439818D-02	0.47858121612547853 2D-02
10^{-160}	32 倍精度	0.9415434185067337298 4D-02	0.941543418501223556D-02
10^{-200}	32 倍精度	0.1173024519704028603 8D-01	0.11730245306496004 3D-01
10^{-240}	32 倍精度	0.1404505620901323477 8D-01	0.140450563290801095D-01

5. まとめ.

多倍長ライブラリは, 数値積分の定量的な精度評価および精度向上に非常に有効な事がわかった. 今後は更に複雑な問題での精度向上, 高速化, 多重積分での積分法と算法に関して検討を行う.

6. 参照文献

(1) Computation of loop integrals using extrapolation Elise de Doncker,Yoshimitsu Shimizu,junpei Fujimoto,Fukuko yuasa Computer Physics Communications 159(2004) 145-156
(2) QUADPACK A Subroutine Package for Automatic Integration R.Piessens E.de Doncker-Kapenga C.W.Uberhuber D.K.Kahaner
(3) 伊理正夫 数値計算 朝倉書店 1981
(4) J. Fujimoto, M. Igarashi, N. Nakazawa, Y. Shimizu and K. Tobimatsu, Progress of Theoretical Physics, Supplement No. 100 (1990) 1-379
(5) 杉原正顯, 室田一雄 数値計算法の数理 岩波書店 2003

7. 謝辞

精度検証にあたり御指導いただきました 高エネルギー加速器研究機構の 藤本先生, 石川先生, 湯浅先生, 金子先生, 及び総合研究大学院大学の清水先生に感謝致します.

(注)