多倍長計算手法

平成26年度第1四半期

目次

1.はじめに

数値積分では,積分計算そのものにデータ形式に依存するものがあり、また数学的には発散する積分値列をある範囲内でしか適用できないものがあります。この資料ではこれらをまとめた結果を記載しています。
また量子モンテカルロ法によるスペクトル計算で物理的に有効とするためにはパラメータの設定である条件がつきます。この条件下ではleee754-2008データ形式での限界が存在します。それを求めるため今回新たに548倍精度演算を作成して計算可能なパラメータ領域を求めました。

2.数値積分の注意事項

簡単な例として以下の2つを考えます。

$$(1)\int_0^1 \frac{1}{x(1-x)+\lambda^2}\,dx \fallingdotseq 4\log\left(\frac{1}{\lambda}\right) \quad (0<\lambda<<1)$$

$$(2)\int_0^1 \frac{1}{xM^2+(1-x)\lambda^2}dx \fallingdotseq \frac{1}{M^2}\left(\log\left(\frac{1}{\lambda^2}\right)+\log M^2\right)$$

$$(0<<M, 0<\lambda<<1)$$

(1)で$\int_{1-\varepsilon}^1 \frac{1}{x(1-x)+\lambda^2}dx \fallingdotseq \frac{\varepsilon}{\varepsilon+\lambda^2}, \varepsilon=\frac{\lambda^2}{\log\left(\frac{1}{\lambda}\right)-1}$でこの

積分値は$\log\left(\frac{1}{\lambda}\right)$で積分区間[0,1]での積分値の$\frac{1}{4}$を

占める事になります。例えば倍精度の場合,$1-\varepsilon$では

$\varepsilon=2^{-53}$より小さく取れませんから,$\lambda=10^{-30}$だと,

$\varepsilon=10^{-62}$と取れれば,34.53...となる積分値が1となってしまいます。

dd形式の4倍精度ですと,$1-\varepsilon$で$a=1, b=-\varepsilon$で10^{-308}まで

とれますのでλが10^{-30}と小さくても充分な精度で計算ができます。

ieee754−2008形式の4倍精度では$\varepsilon=2^{-113} \fallingdotseq 10^{-34}$と倍精度

よりは小さなλまで精度良く計算できますが,$\lambda=10^{-30}$だと精度

を保った計算が困難になります。

(2)の$I=\int_0^1 \frac{1}{xM^2+(1-x)\lambda^2}dx \fallingdotseq \frac{1}{M^2}(\log(\frac{1}{\lambda^2})+\log M^2)$

$(0<<M, 0<\lambda<<1)$を考えます。

$I=A\log(\frac{1}{\lambda^2})+B \qquad A=\frac{1}{M^2}, B=\frac{1}{M^2}\log M^2$

のA, Bを$\lambda_n=10^{-n}$のときの積分値y_n列から求めるとします。

ここで$\lambda^2=0$なら$I=\lim_{\varepsilon->0}\frac{1}{M}\ln\frac{1}{\varepsilon}$

で$\lambda^2=0$なら$I->\infty$と考えられますが、ここで数値積分特有の問題が発生します。

端点特異点がある場合以下の場合があります。

積分値が有界の場合(例$\int_0^1 \frac{1}{\sqrt{x}}dx=2$)

積分値が有界でない場合(例$\int_0^1 \frac{1}{x}dx=\infty$)

数値積分は内点x_k, 重みw_kでの和$\sum_{k=1}^{n} f(x_k)w_k$で

表わされますので, 積分値が有界でない場合は正しく計算

出来ずに$\int_0^1 \frac{1}{x}dx=\ln(\frac{1}{\varepsilon})$　(εは0でない計算機で表現できる最小数)

となります。このためλが比較的大きい場合は$A\times\ln(\frac{1}{\lambda^2})+B$の

形になりますが、計算機では収束列を作りますので,

λが小さくなると、有界な単調数列は収束しますので

$A\times\ln(\frac{1}{\lambda^2})+B$の形からは外れてきます。

このため, $A\times\ln(\frac{1}{\lambda^2})+B$の形をだすには, 出来るだけ0,1に近い数値を

とる必要があります。またA, Bを求める際のnの大きさに制限が

加わる事になります。$\lambda=0$の場合の計算機上での積分値をy_∞

とします。$M=100$としますと$y_{30}=64\log 10\times 10^{-4}, \varepsilon=10^{-50}$

だと$y_\infty=50\log 10\times 10^{-4}, \varepsilon=10^{-100}$だと$y_\infty=100\log 10\times 10^{-4}$.

以上からこの様なケースでは, 指数部15ビットのデータ形式を持つ

変数を2つつなぎ合わせた演算が適していると言えます。性能も考慮

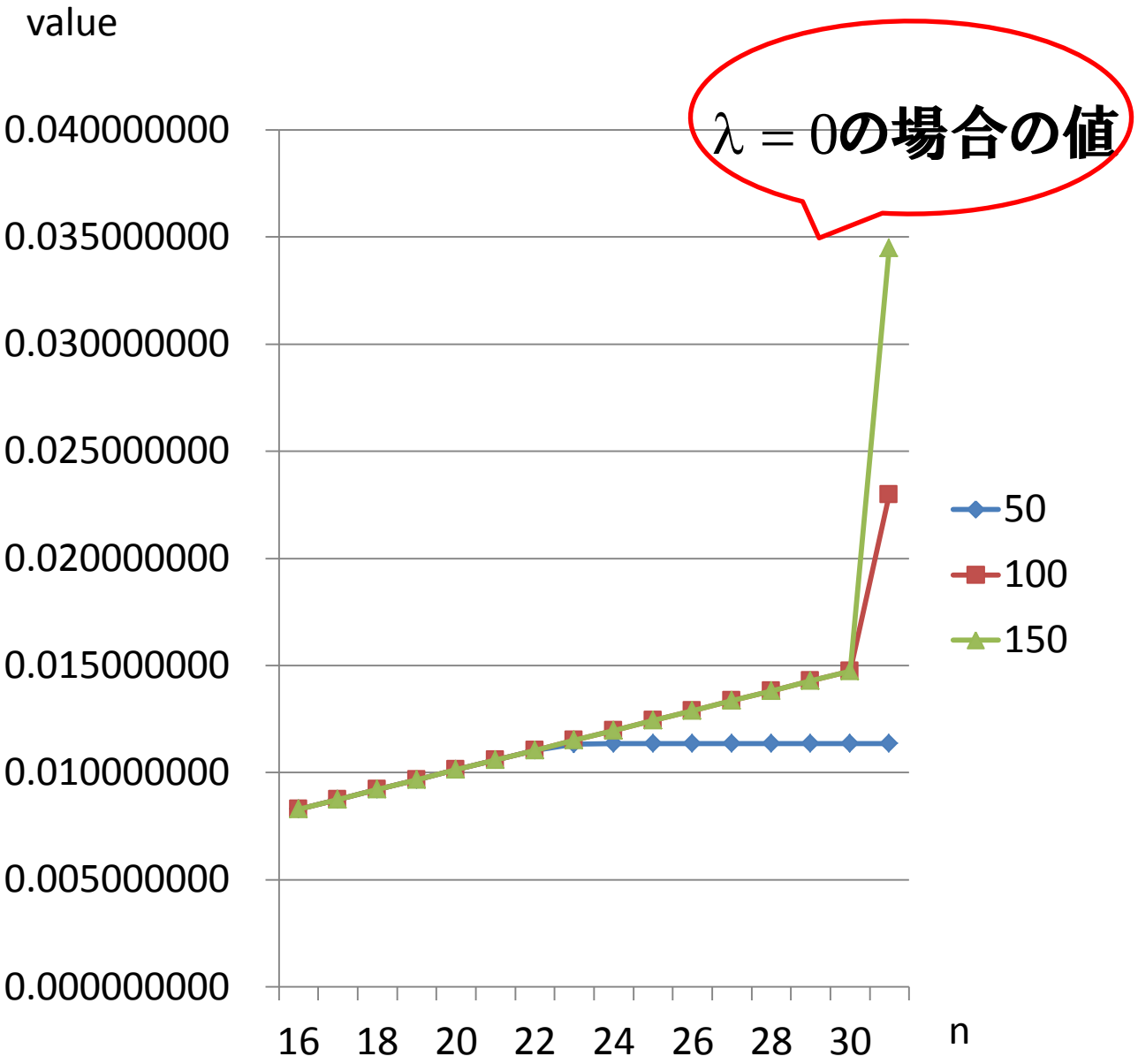
すればcの拡張倍精度(80ビット)変数を2つつなげた演算が最も適して

います。(2)の場合の実測例を次にしめします。

$$\int_0^1 \frac{1}{M^2 x + (1-x)\lambda^2} dx$$ の計算例

$M = 100, \lambda = 10^{-n} (16 \le n \le 30)$ と $\lambda = 0$

$eps = 10^{-m} (m = 50,100,150)$



$$\lambda = 10^{-n}$$

$\varepsilon = 10^{-100}, 10^{-150}$ **の場合の一次差分の値**

$\lambda = 10^{-n} (16 \le n \le 29)$

n	100	150
16	0.000460517018599	0.000460517018599
17	0.000460517018599	0.000460517018599
18	0.000460517018599	0.000460517018599
19	0.000460517018599	0.000460517018599
20	0.000460517018599	0.000460517018599
21	0.000460517018599	0.000460517018599
22	0.000460517018599	0.000460517018599
23	0.000460517018599	0.000460517018599
24	0.000460517018599	0.000460517018599
25	0.000460517018599	0.000460517018599
26	0.000460517018599	0.000460517018599
27	0.000460517018599	0.000460517018599
28	0.000460517018599	0.000460517018599
29	0.000460517018599	0.000460517018599

(2)のケースの積分値I

$I = A \times \log(\frac{1}{\lambda^2}) + B$**となっていると**

一次差分は $2 \times \log(10) / M^2 = 0.0004605170186$

となり結果は良く一致しています。

3. 拡張倍精度+拡張倍精度演算の結果

倍精度変数を2つつなげた形式では計算出来ない問題をcの拡張倍精度(80ビット)変数を2つつなげた形式で計算を行いました。

ここでは,使用する定数 $\frac{\pi}{2}$, $\log(2)$は整数演算方式の8倍精度定数から,整数演算方式の仮数部64ビットを抜き出した4倍精度変数を作り,8倍精度定数から先に作成した4倍精度変数を引きこれから仮数部64ビットの4倍精度変数を作成し,2つの仮数部64ビットの4倍精度変数を作成して(FORTRAN),これをそれぞれcの拡張倍精度に割り当てて作成しています。

3.1　1次元積分

一次元積分

$$I=\int_0^1 \frac{1}{-x^2+x+\lambda^2}dx \fallingdotseq 4\times\ln\left(\frac{1}{\lambda}\right)$$

$\lambda=10^{-m}$, **拡張倍精度＋拡張倍精 度**

$n=262144$

実行結果一覧

m	実測値	解析近似解
2450	2.2565333911E+04	2.2565333911E+04
2451	2.2574544252E+04	2.2574544252E+04
2452	2.2583754592E+04	2.2583754592E+04
2453	2.2592964932E+04	2.2592964932E+04
2454	2.2602175273E+04	2.2602175273E+04
2455	2.2611385613E+04	2.2611385613E+04
2456	2.2620595895E+04	2.2620595954E+04
2457	2.2629800419E+04	2.2629806294E+04
2458	2.2638505230E+04	2.2639016634E+04
2459	2.2641485194E+04	2.2648226975E+04
2460	2.2641557723E+04	2.2657437315E+04
2461	2.2641558463E+04	2.2666647655E+04
2462	2.2641558470E+04	2.2675857996E+04
2463	2.2641558470E+04	2.2685068336E+04
2464	2.2641558470E+04	2.2694278677E+04
2465	2.2641558470E+04	2.2703489017E+04

青,紫色表示の結果は数値表現範囲の制限に影響を受けた事を示しています。

3.2　2次元積分

inf　ra　　vtx

$$I=\int_0^1\int_0^{1-x}\frac{1}{D}dydx$$

$$D=-xys+(x+y)^2m^2+(1-x-y)\lambda^2$$

$s<0$の場合解析近似解

$$I=\frac{1}{-s}\left[\ln\left(\frac{m^2}{-s}\right)\ln\left(\frac{\lambda^2}{m^2}\right)+\frac{1}{2}\ln^2\left(\frac{m^2}{-s}\right)-\frac{\pi^2}{6}\right]$$

非常に精度の良い計算結果となっています。!!

実測値

```
 size =16384
elapse time(sec)=66.772168
ramda =1.00000000000005550e-1000
result=  5.08822892088460067e-01
```

```
 size =65536
elapse time(sec)=1959.780368
ramda =1.00000000000013321e-2400
result=  1.22139755086583201e+00
```

```
 size =65536
elapse time(sec)=2859.353386
ramda =1.00000000000013599e-2450
result=  1.24684666050009526e+00
```

解析近似解

```
  s=  -250000.00000000000000000000000000
  m**2=  2.50000000000000000000000000000000E-0007
  ramda**2=  1.00000000000000000000000000000000E-2000

  ztotal=  0.50882241995459883061265175094468 2
```

```
  s=  -250000.00000000000000000000000000
  m**2=  2.50000000000000000000000000000000E-0007
  ramda**2=  1.00000000000000000000000000000000E-4800

  ztotal=  1.22139752600289552323213971357172
```

```
  s=  -250000.00000000000000000000000000
  m**2=  2.50000000000000000000000000000000E-0007
  ramda**2=  1.00000000000000000000000000000000E-4900

  ztotal=  1.24684663693319183368283571223697
```

3.3　3次元積分

inf ra　box

$$I=\int_0^1\int_0^{1-x}\int_0^{1-x-y}\frac{1}{D^2}dzdydx$$

$$D=-xys-tz(1-x-y-z)+(x+y)\lambda^2$$
$$+(1-x-y-z)(1-x-y)m_e{}^2$$
$$+z(1-x-y)m_f{}^2$$

解析解$$I=\frac{1}{-s(-t+m_f{}^2)}\ln\left(\frac{-s}{\lambda^2}\right)\ln\frac{(-t+m_f{}^2)^2}{m_f{}^2m_e{}^2}$$

$$s<0,t<0,m_e{}^2<<|s|,-t$$

$$s=-500^2,t=-150^2,m_f=150,m_e=0.0005$$**に固定**

この積分計算ではプログラミングで次ページに示します注意点があります。

二重指数関数型積分でのinf ra box のDD形式の分岐点

積分変数変換区間は[0,1]とします。

$x = 1 - \varepsilon$においてx = a + bからx = 1 − c

に表現形式を変更するとします。$(a, b, c > 0)$

$h = 0.5^6$として,εを定めると,

$$2y = \ln(\frac{1}{\varepsilon}), t = \ln(\frac{2}{\pi} y + \sqrt{(\frac{2}{\pi} y)^2 + 4})$$

$n = \frac{t}{h}$よりnが求まります。分点数$N_h = 2n.$

最初に$\varepsilon = \lambda^2$として分点数を定めます。

次に分岐点の$\varepsilon = 2^{-m}$のn(これをn_m)を求めて

$k = n + n_m$として以下の様に分点x30(i)を求めます。

```
if (i.ge.k)  then
x30(i) = 1.0q0 /(1.0q0 + exp(−2.0q0 * y))
else
x30(i) = exp(y) /(exp(y) + exp(−y))
end    if
```

また分点数Nを任意に決める場合

$h = 0.5^6 \times N_h / N, k = k \times N / N_h$と変更します。

例としては, $N = 2048, \lambda = 10^{-50}, \varepsilon = 10^{-100}$から

$\varepsilon = 10^{-120}$に変更して精度の良い結果が得られた

ものを示しました。

解析解

ramda= 0.1000000D-29 kai= 0.35617368129182453 8D-06
ramda= 0.1000000D-34 kai= 0.41063620430777801 2D-06
ramda= 0.1000000D-39 kai= 0.46509872732373148 6D-06
ramda= 0.1000000D-44 kai= 0.51956125033968496 0D-06
ramda= 0.1000000D-49 kai= 0.57402377335563843 4D-06

実測値

size =2048
elapse time(sec)=2734.381061
ramda = 1.00000000000000008e-30
result= 3.561736812918009 54e-07

size =2048
elapse time(sec)=2735.784601
ramda = 1.00000000000000001e-35
result= 4.106362043073803 23e-07

size =2048
elapse time(sec)=2733.080820
ramda = 9.99999999999999 29e-41
result= 4.650987273338546 73e-07

size =2048
elapse time(sec)=2734.544721
ramda = 9.99999999999999 84e-46
result= 5.195612472270477 70e-07

size =2048
elapse time(sec)=2738.138126
ramda = 1.00000000000000001e-50
result= 5.620315551030949 41e-07

変数変換区間変更前

size =2048
elapse time(sec)=2735.340927
ramda = 1.00000000000000001e-50
result= 5.740237743580000 10e-07

変数変換区間変更後

更に小さいλに関しては以下の様になっています。

解析解

```
   ramda=  0.1000000-199  kai= 0.22078994638342426 5D-05
   ramda=  0.1000000-249  kai= 0.27525246939937773 9D-05
   ramda=  0.1000000-299  kai= 0.32971499241533121 3D-05
   ramda=  0.1000000-349  kai= 0.38417751543128468 7D-05
   ramda=  0.1000000-399  kai= 0.43864003844723816 1D-05
   ramda=  0.1000000-449  kai= 0.49310256146319163 5D-05
   ramda=  0.1000000-499  kai= 0.54756508447914510 9D-05
```

実測値

```
 size =8192
elapse time(sec)=169611.184812
ramda = 9.99999999999999982e-201
result=  2.20789939326117987e-06
```

```
 size =8192
elapse time(sec)=176674.035061
ramda = 1.00000000000001388e-250
result=  2.75253476866793786e-06
```

```
size =8192
elapse time(sec)=176035.891126
ramda = 1.00000000000001665e-300
result=  3.29713382570765961e-06
```

```
 size =8192
elapse time(sec)=175589.716968
ramda = 1.00000000000001943e-350
result=  3.84150079631083170e-06
```

```
 size =8192
elapse time(sec)=186976.452800
ramda = 1.00000000000002220e-400
result=  4.38556073884776220e-06
```

```
 size =16384
elapse time(sec)=1366835.826775
ramda = 1.00000000000002498e-450
result=  4.93101872610781070e-06
```

```
 size =16384
elapse time(sec)=1371994.157738
ramda = 1.00000000000002775e-500
result=  5.47562342320869000e-06
```

$\lambda = 10^{-500}$ の詳細

解析解 ＝5.47565084 479145D - 06

size =16384
elapse time(sec)=1371994.157738
ramda = 1.00000000000002775e-500
result= 5.47562342320869000e-06

T2Kでの結果

size =32768
elapse time(sec)=63260.473200
ramda = 1.00000000000002776e-500
result= 5.47565084649954684e-06

これらの結果から小さいλに対しては,並列化(MPI化)によってNを増やして精度の良い結果が得られる事がわかります。

4.量子モンテカルロ法による物性スペクトル計算

これまでは, $U=10, N=100, L=448$を固定して
βに対して正しく計算するのに必要な演算精度
を求めてきましたが、物理的に意味をなすには
$L/\beta=20$が必要なため, 今回はこの点からβに
対する必要演算精度を求める様にしています。
これまで$N=20$で正しい結果が得られれば,
$N=100$で正しい結果が得られ大きな差がない
事から性能面から$U=10, N=20$としています。
$N=20, N=100$の関係の検証は,68倍精度演算,
128倍精度演算実行で行いました。
また実行に際しては、メモリの制約もあり, 絶対値
最小値でのチェックのため, $L/\beta=20+\alpha(0<\alpha<<1)$
となる様に変更した場合があります。

βに対して必要な演算精度は以下の様になります。

$\beta / L = a$(一定), $U = 10$とする。$(0 < a << 1)$

$\beta = 10, L = 200, a = 0.05$

$$\sqrt{U\beta L} = \beta\sqrt{\frac{U}{a}} = 10\sqrt{2} \times \beta$$

$$\text{必要ビット数} = \ln_2 e^{\beta \times 10\sqrt{2}} = \frac{10\sqrt{2} \times \beta}{\ln 2} = 20.40278893 \times \beta$$

β	必要ビット数	必要精度	L
100	2040	68	2000
180	3673	128	3600
250	5101	188	5000
300	6121	248	6000
400	8162	308	8000
500	10202	368	10000
600	12242	428	12000
700	14282	488	14000
800	16323	548	16000

ここで548倍精度演算では,ieee754-2008データ形式の数値範囲の制限により,様子が他の演算精度の場合と異なります。

4.1　548倍精度演算のヒルベルト行列計算

誤差は解$x = (1,1,\ldots,1)^T$と比較していますので、小さなNに対しては差が0と正しく計算しているかの検証とは言えません。そこでNを変えながら正しい事を検証しました。

548倍精度ヒルベルト行列　x5570

N	最大誤差	実行時間（秒）	条件数（ビット数）	有効ビット−条件数
100	0.000D−0000	182.566	501	0.000D−0000
200	0.000D−0000	1304.570	1010	0.000D−0000
300	1.647D−4819	4368.843	1518	4.140D−4818
400	1.656D−4667	10544.205	2022	2.168D−4666
500	4.445D−4513	21630.467	2530	1.817D−4513

誤差が出るサイズ

条件数1138ビットとなるN

N	条件数（ビット数）
223	1127
224	1132
225	1137
226	1142
227	1147
228	1152

理論上はN = 226で誤差0以外の値が出る事になるが実測値はN = 227

4.2　8+60*k (1≤k≤9) 倍精度演算での結果

測定結果一覧

$8+60\times k \quad (1\le k\le 8)$

倍精度演算での実行結果

$(U=10, N=20)$

理論値は解析近似解の値を採用しています。

項番	β	L	演算精度
1	100	2000	68倍精度
2	180	3600	128倍精度
3	250	5000	188倍精度
4	300	6000	248倍精度
5	400	8000	308倍精度
6	500	10016	368倍精度
7	600	12032	428倍精度
8	10000/14	14336	488倍精度

最小値一覧

備考	理論値	実測値	実測値(N=100)
1	0.567D-307	0.330D-307	0.330D-307
2	0.120D-552	0.514D-553	0.512D-553
3	0.130D-767	0.476D-768	
4	0.370D-921	0.126D-921	
5	0.299D-1228	0.927D-1229	
6	0.143D-1536	0.424D-1537	
7	0.686D-1845	0.199D-1845	
8	0.296D-2197	0.849D-2198	

N=100でも結果は影響なし。

488倍精度演算までは問題なしと言えます。

548倍精度演算での調査結果

```
n=           8
l=       16000
beta, dt=    800.00000000000000000000000000000000
 5.00E-02
u, t0=    10.00000000000000000000000000000000
           1
fac=  0.70710678118654752440084436210484 9
For denominator: mstep=           1
For green func: mstep2=           1

ispin=           1
QDR  =  1.13949637284986456231601074493146 7E+4911
I+QDR=  1.13949637284986456231601074493146 7E+4911

ispin=           2
QDR  =  4.84493967547754402433739442571308 3E+4911
I+QDR=  4.84493967547754402433739442571308 3E+4911

   0.0000000000D+00   -0.2500000000D+00
```

```
n=           8
l=       15744
beta, dt=    800.00000000000000000000000000000000
 5.08E-02
u, t0=    10.00000000000000000000000000000000
           1
fac=  0.71283243564025129743160436923030 5
For denominator: mstep=           1
For green func: mstep2=           1

ispin=           1
QDR  =  3.37063287003268167588917330241850 4E+4929
I+QDR=  3.37063287003268167588917330241850 4E+4929

ispin=           2
QDR  =  1.05636585510384055575844764887041 4E+4931
I+QDR=  1.05636585510384055575844764887041 4E+4931

   0.0000000000D+00   -0.1000000000D+01
```

$\beta = 800, \mathrm{N} = 20, \mathrm{U} = 10, \mathrm{L} = 15774$

最小値　**理論値** $= 0.688 \times 10^{-2437}$

実測値 $= 0.197 \times 10^{-2437}$

$\mathrm{L} / \beta = 15744 / 800 = 19.68 < 20$

4.3 ieee754-2008データ形式での限界

$L/\beta = 20$を$\beta/L = 0.05$として記述しました。

$\beta/L = a$(一定), $U = 10$とします。$(0 < a << 1)$

$\beta = 10, L = 200, a = 0.05$

$$\sqrt{U\beta L} = \beta\sqrt{\frac{U}{a}} = 10\sqrt{2}\times\beta$$

必要ビット数 $\ln_2 e^{\beta\times 10\sqrt{2}} = \frac{10\sqrt{2}\times\beta}{\ln 2} = 20.40278893\times\beta$

ここで, $\beta = 800, L = 16000$とすると必要ビット数は16323,

最大数 $= 10^{4913.481171}$から548倍精度演算で計算可能

となります。ただし$V = QDR$の最終段階での絶対値の最大数

$e^{\sqrt{U\beta L}}, I + V = Q(D + Q^T R^{-1})R$を求める際の有効ビット数

$16383 - \frac{\sqrt{U\beta L}}{\ln 2}$は$16383 - \alpha \geq \frac{\sqrt{U\beta L}}{\ln 2}$の必要がある.

これは$e^{\sqrt{U\beta L}}$付近の逆数の減算結果の持つ有効ビットには

制限があるためです。すなわち,10^{-4930}付近のa, bでは

$a - b$の結果は指数部が負となる可能性があり, $a - b = 0$と

する必要が出てくる事によります。$n = 8$

$\beta = 800, L = 15808$で結果不正, $\beta = 800, L = 15744$で正しい結果

が得られている事から$159 \leq \alpha \leq 192$となります。

$a = 0.05$を守る範囲では$\beta = 790, L = 15808$で548倍精度で正しい結果、

$\beta = 800, L = 15744, a = 0.050813$の範囲で548倍精度で正しい結果が

得られています。

絶対温度12.5K まで計算可能!!