リーマンのツェータ関数（実数）の数値計算

2016.9.16　　鈴木　実

1　はじめに

ガンマ関数は gcc に用意されているが，リーマンのツェータ関数は用意されていないようだ．ボーズ・アインシュタイン積分にはリーマンのツェータ関数が現れる．これを簡単に計算したい．実数値だけでよいので，リーマンのツェータ関数を簡単に計算できるプログラムが必要である．リーマンのツェータ関数は収束が遅いという典型的な例であるそうだ．それで収束が速く，かつ特異点 $s=1$ $(s=1+2\pi in/\log 2)$ (n はある整数) を除く全ての複素平面で収束する次の展開式を用いて計算する C プログラムを作成した．この展開式は Wikipedia にあるように，1930 年ごろに Konrad Knopp によって予想され，1930 年に Helmut Hasse によって証明されたということである．

$$\zeta(s)=\frac{1}{1-s^{1-s}}\sum_{n=0}^{\infty}\frac{1}{2^{n+1}}\sum_{k=0}^{n}(-1)^k\binom{n}{k}(k+1)^{-s} \tag{1}$$

この展開式は複素数についても成立つのでプログラムを複素数に対応させれば複素数の数値計算が可能である．

プログラムソース

以下のプログラムは上の展開式で$n = 50$まで計算し，約 14 桁の精度が得られる．計算速度は上げるには NMAX を小さくし，計算精度を上げるには NMAX を大きくすれば良い．

```
/* Riemann_zeta_f.c  -o  zeta */
/* 2016.9.16 by M. Suzuki */

#include <stdio.h>
#include <stdlib.h>
#include <math.h>

#define    NMAX    50

double binom(int n, int k)
{
    int i;
    double x, z;
    if(k==0) return z=n;
    z=1;
    i=k;
    while(k>0)
    {
        z*=n;
        n--;
        k--;
        if(n==0) n=1;
    }
    while(i>0)
    {
        z/=i;
        i--;
    }
    return z;
}

int main(int argc, char *argv[])
{
    double s, t, x, y, z;
    int i, j, n, k;

    s=atof(argv[1]);
    if(s==1) exit(0);
    z=0;
    for(n=0; n<NMAX+1; n++)
    {
        t=1/pow(2.0, n+1);
        y=0;
        for(k=0;k<n+1;k++)
        {
            x=pow(k+1, -s)*binom(n, k);
            if((k % 2)==1) x*=-1;
            y+=x;
        }
        t*=y;
        z+=t;
    }
    z/=(1-pow(2.0, 1-s));
    printf("%20.16lf\n", z);
}
```