Global Journal of Pure and Applied Mathematics.
ISSN 0973-1768 Volume 19, Number 2 (2023), pp. 651-676

http://www.ripublication.com/gjpam.htm

Barnes G-Function and the Derivative of the Integral of a Theta Function

Darrell Cox[1]

[1]*Department of Mathematics, Grayson County College, United States*
E-mail: darkencox273@gmail.com

Abstract

The Riemann zeta function is generalized using the Barnes G-function. There is also a functional equation for this generalized zeta function. There appear to be zeros for all complex numbers with a real part of 1/2. Derivatives are used to derive these results.
Keywords: Riemann zeta function, Barnes G-function.

1. INTRODUCTION

Equation (3) in section 1.3 of Edward's [1] book is

$$\Pi(s) = \lim_{N\to\infty} \frac{1 \cdot 2 \cdots N}{(s+1)(s+2)\cdots(s+N)}(N+1)^s \tag{1}$$

This equation is valid for all s in the halfplane Re $s > -1$. (Edwards uses the notation $\Pi(s-1)$ instead of $\Gamma(s)$.)

2. BARNES G-FUNCTION AND A THETA FUNCTION

An equation given in Kargin's [2] article relating $\zeta(s)$ to the series $\theta(x) = \sum_{n=1}^{\infty} e^{-n^2\pi x}$ is

$$\frac{\Gamma(\frac{1}{2}s)\zeta(s)}{\pi^{s/2}} = \sum_{n=1}^{\infty} \int_0^{\infty} x^{s/2-1} e^{-n^2\pi x} dx = \int_0^{\infty} x^{s/2-1}\theta(x) dx \tag{2}$$

In the following, the partition function is $\{\lambda_k\}$=$\{n$ with multiplicity $n\}$, n=1, 2, 3,.... Equation A.16 in Voros [3] article is

$$\Theta^B(t) = \sum_{n=1}^{\infty} ne^{-tn} = -\frac{d}{dt}(\frac{1}{e^t-1}) \tag{3}$$

Equation A.17 (the reflection formula for $\Gamma(z)$) is

$$Z^B(s) = \sum_{n=1}^{\infty} n \cdot n^{-s}\zeta(s-1) \tag{4}$$

Multiplying $\frac{\Pi(\frac{s}{2})\zeta(s)}{\pi^{s/2}}$ and $\Theta^B(s)$ gives

$$\zeta_1(s) = \Pi(\frac{s}{2})\Theta^B(s)\zeta(s)\pi^{-s/2} \tag{5}$$

$\zeta_1(s)$ divided by $\zeta(s)$ gives a logarithmic spiral. A plot of $\zeta_1(s)$ and $\zeta(s)$ for $s = (0.5, 51.3720769775)$ (the average of the tenth and eleventh non-trivial zeta function zeros) and $n \leq 1000$ is

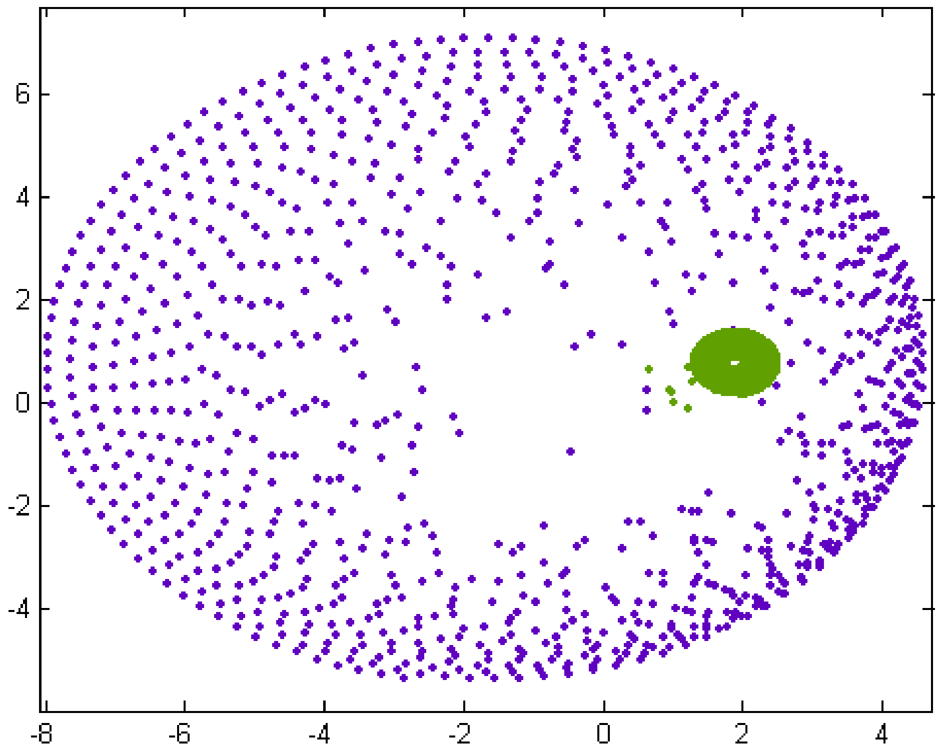

Figure 1:

The result of the division is

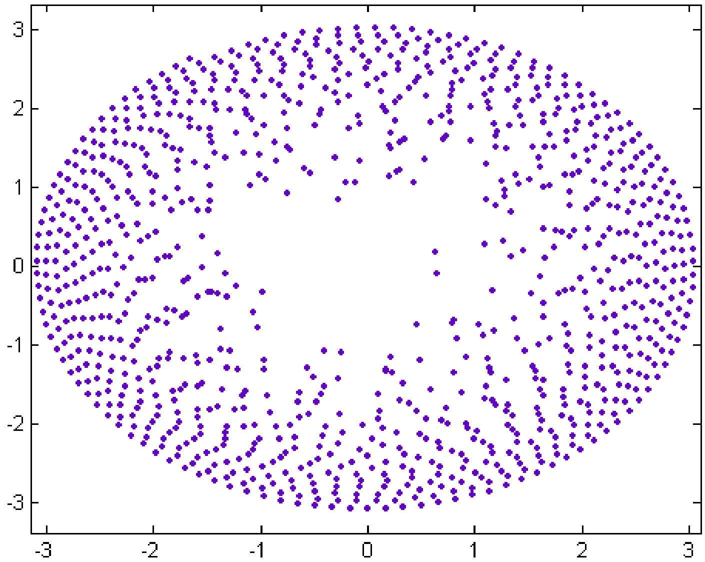

Figure 2:

A plot of the logarithms of the n values of the inflection points (where the curve decreases toward the x-axis and then increases) is

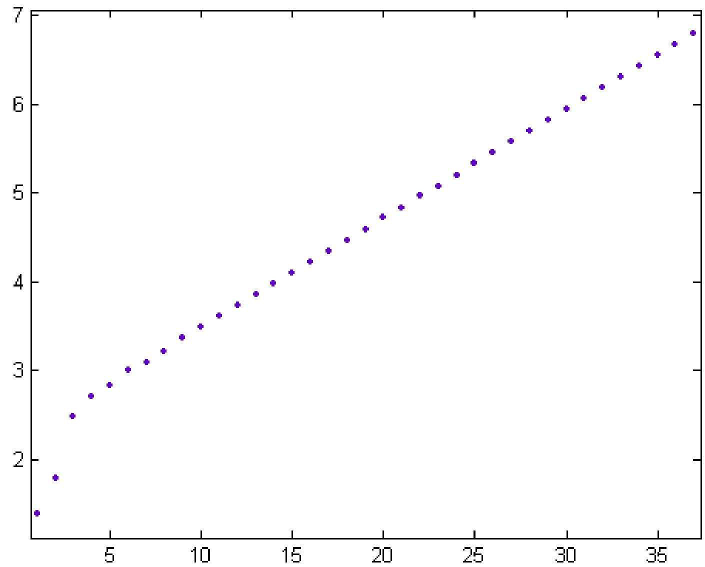

Figure 3:

For a linear least-squares fit of the curve (disregarding the first three n values), $p_1 = 0.1232$ with a 95% confidence interval of (0.1229, 0.1236), $p_2 = 2.245$ with a 95% confidence interval of (2.236, 2.253), SSE=0.003553, R-squared=0.9999, and RMSE=0.01051.

A plot of $\zeta_1(s)$ and $\zeta(s)$ for $s = (0.5, 49.77383247767)$ (the tenth zeta function zero) and $n \leq 1000$ is

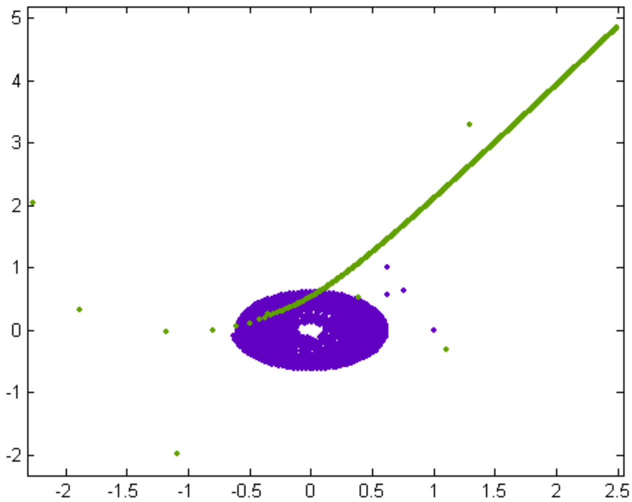

Figure 4:

A plot of the result of the division is

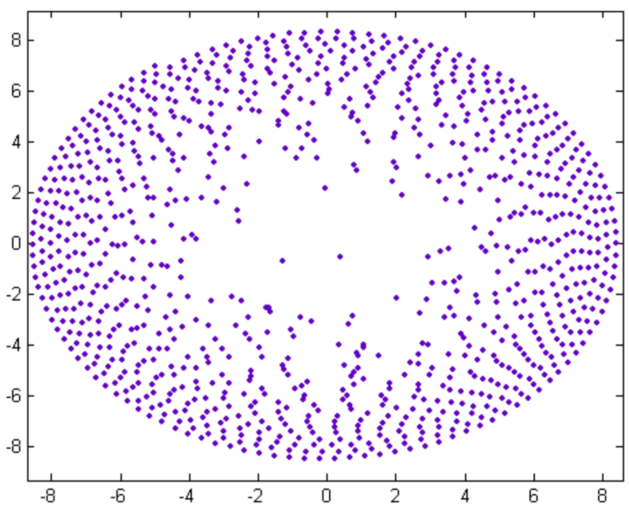

Figure 5:

A plot of the logarithms of the n values of the inflection points is

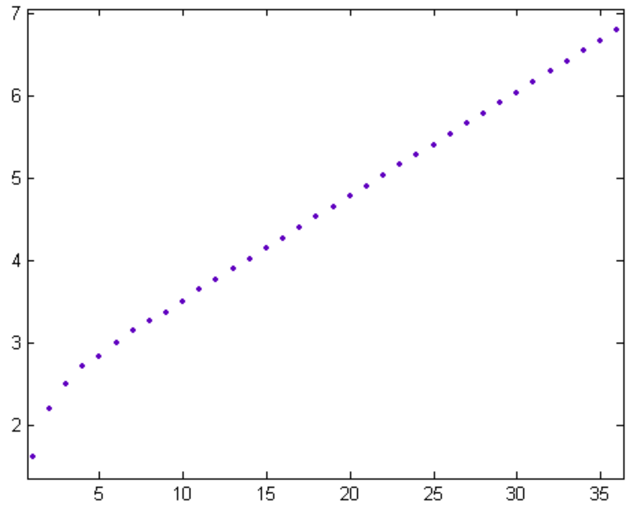

Figure 6:

For a linear least-squares fit of the curve (disregarding the first five n values), $p_1 = 0.1265$ with a 95% confidence interval of (0.1263, 0.1267), $p_2 = 2.241$ with a 95% confidence interval of (2.237, 2.245), SSE=0.0005179, R-squared=1, and RMSE=0.004226.
A plot of $\zeta_1(s)$ and $\zeta(s)$ for $s = (0.5, 52.97032147771446)$ (the eleventh zeta function zero) and $n \leq 1000$ is

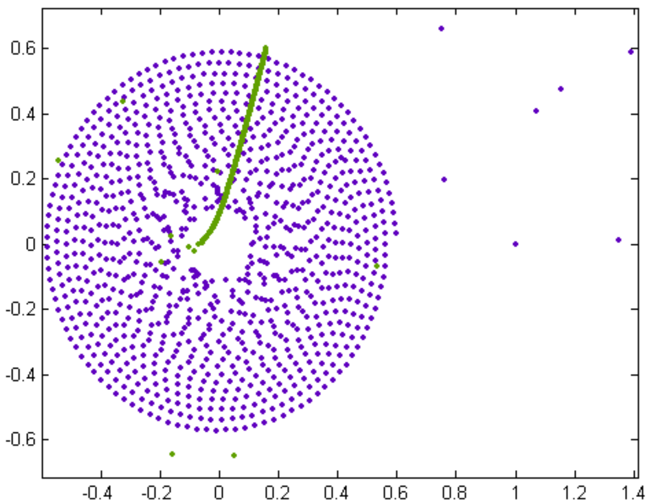

Figure 7:

A plot of the result of the division is

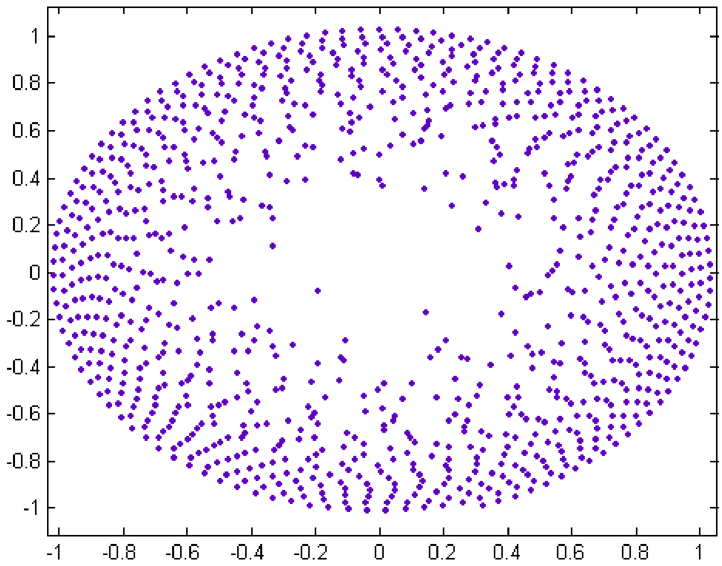

Figure 8:

A plot of the logarithms of the n values of the inflection points is

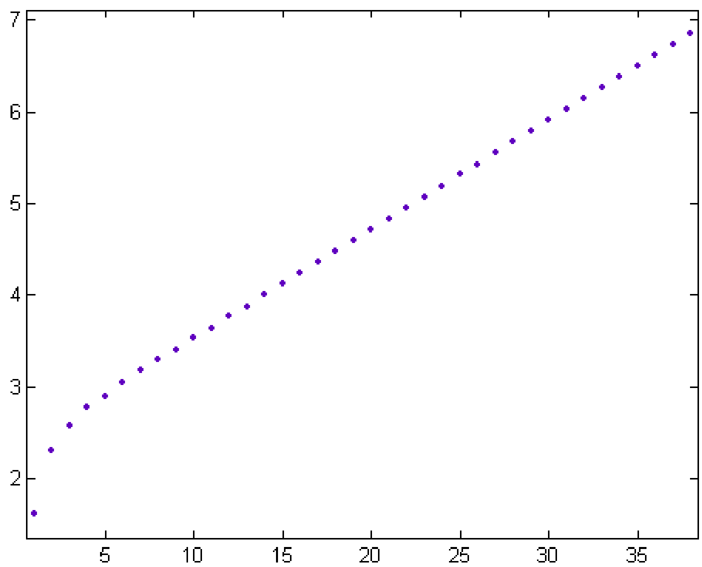

Figure 9:

For a linear least-squares fit of the curve (disregarding the first three n values), $p_1 =$ 0.1189 with a 95% confidence interval of (0.1188, 0.1191), $p_2 = 2.2336$ with a 95% confidence interval of (2.333, 2.234), SSE=0.000564, R-squared=1, and RMSE=0.004265.

The average slope for the tenth and eleventh zeta function zeros (0.1227) is almost equal to the slope for the average s values (0.1232). Note that $\zeta(s)$ is not centered for the average s values but the resulting logarithmic spirals from the divisions are all centered.

Riemann's functional equation (equation (5) in section 1.6 of Edwards' book) is

$$\Pi(\frac{s}{2}-1)\pi^{-s/2}\zeta(s) = \Pi(\frac{1-s}{2}-1)\pi^{-(1-s)/2}\zeta(1-s) \tag{6}$$

A plot of the left and right sides of the equation for the first non-trivial zeta function zero ($s = (0.5, 14.134725191735)$) and $n \leq 100000$ is

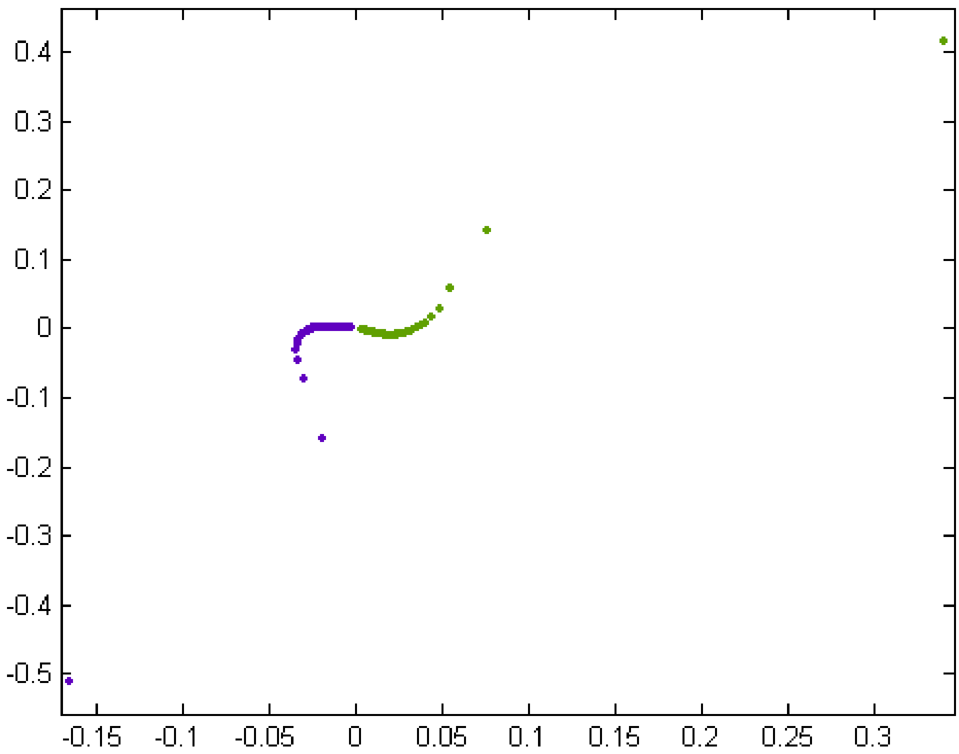

Figure 10:

Such curves are typical for zeta function zeros.

$\zeta_1(s)$ can be substituted into the functional equation. A plot of the left and right sides of the equation for the first zeta function zero and $n \leq 1000$ is

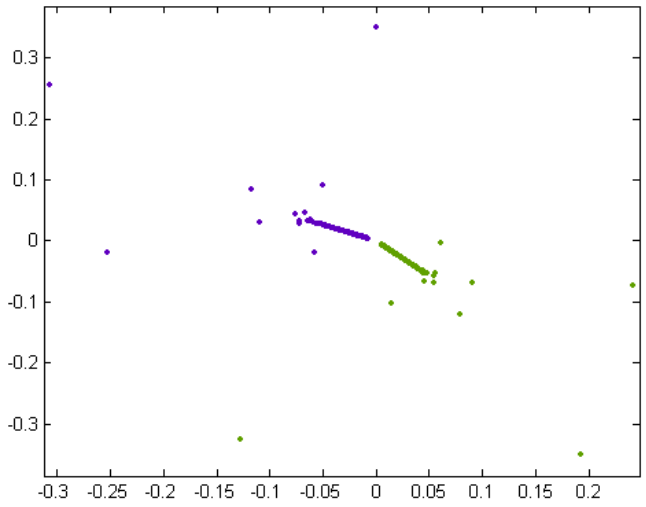

Figure 11:

3. ANOTHER RATIO INVOLVING THE BARNES G-FUNCTION

A plot of $\Theta^B(s)\pi^{-s/2}$ for the second zeta function zero and $n \leq 1000$ is

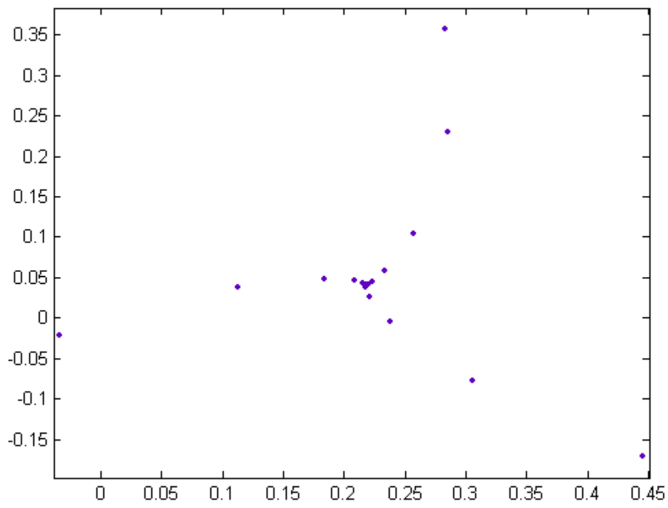

Figure 12:

The curve converges to (0.2178, 0.0414).

A plot of $\Theta^B(s)\pi^{-s/2}$ for the fourth zeta function zero and $n \leq 1000$ is

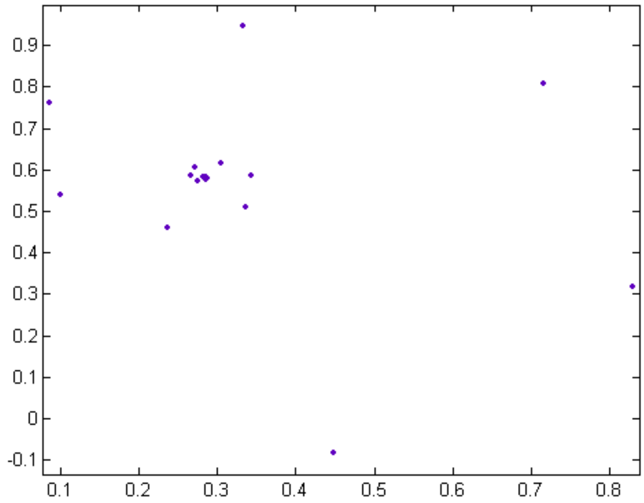

Figure 13:

The curve converges to (0.2835, 0.5824).

A plot of $\Theta^B(s)\pi^{-s/2}$ for the seventh zeta function zero and $n \leq 1000$ is

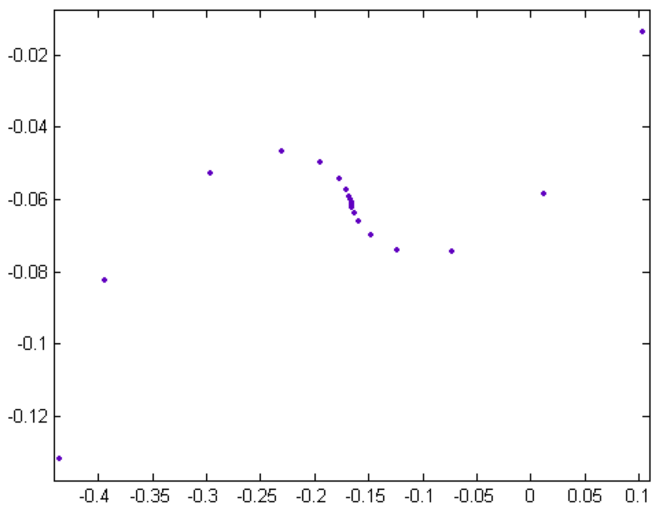

Figure 14:

The curve converges to (−0.1659, −0.0610).

A plot of $\Theta^B(s)\pi^{-s/2}$ for $s = (0.5, 13.5)$ and $n \leq 1000$ is

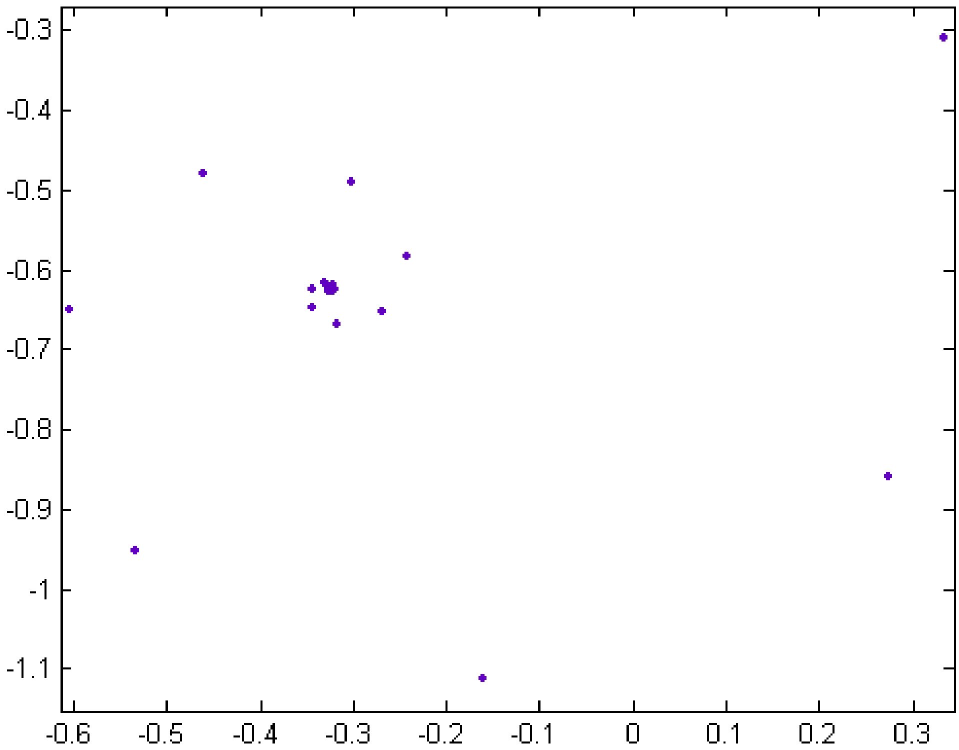

Figure 15:

The curve converges to (−0.3251, −0.6256).

For complex numbers with a real part of 1/2, the radius of convergence appears to be 1.

4. A DERIVATIVE INVOLVING THE REFLECTION FORMULA

In a variant of the above, the expression

$$\Pi(\frac{s}{2})\Theta^B(s)\zeta(s) \tag{7}$$

is multiplied by the following expression.

$$\frac{2\pi\zeta(s)}{\Pi(1-s)Z^B(s)} \tag{8}$$

This gives $\zeta_2(s)$. A plot of $\zeta_2(s)$ for the first zeta function zero and $n \leq 1000$ is

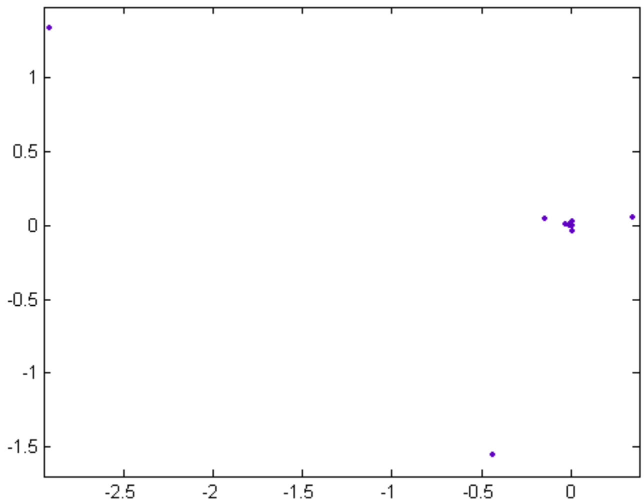

Figure 16:

A plot of the curve for $100 \leq n \leq 1000$ is

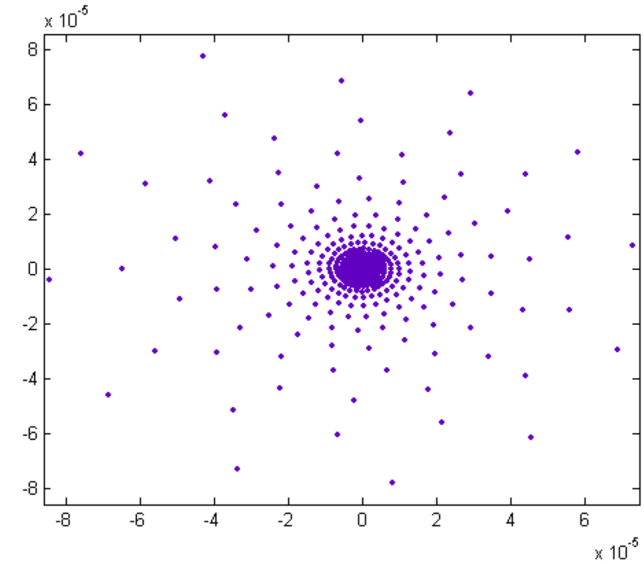

Figure 17:

A plot of the logarithms of the n values of the inflection points is

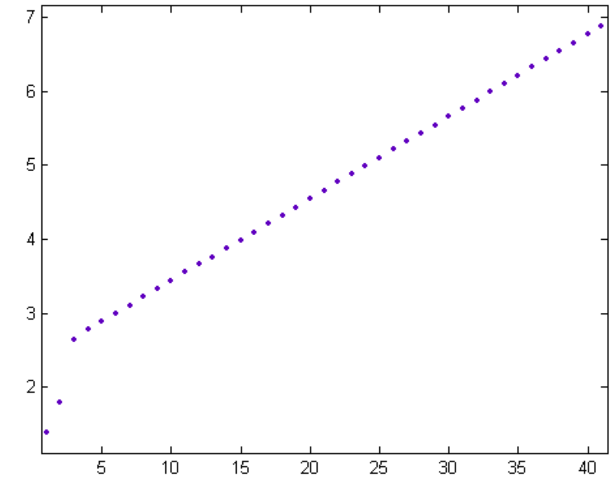

Figure 18:

For a linear least-squares fit of the curve (disregarding the first two n values), $p_1 = 0.1111$ with a 95% confidence interval of (0.111, 0.113), $p_2 = 2.323$ with a 95% confidence interval of (2.319, 2.327), SSE=0.001177, R-squared=1, and RMSE=0.00564. The slope is about a fourth of that for the first zeta function zero.
A plot of $\Pi(\frac{s}{2})\Theta^B(s)\zeta(s)$ and $\frac{2\pi\zeta(s)}{\Pi(1-s)Z^B(s)}$ is

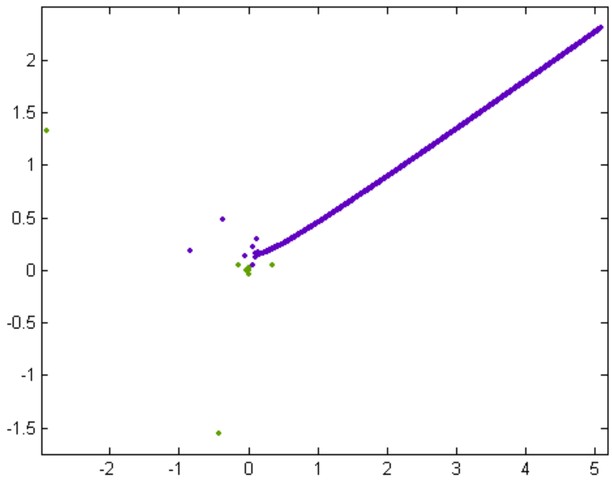

Figure 19:

$\Pi(\frac{s}{2})\Theta^B(s)\zeta(s)$ can be interpreted as being the derivative of $\frac{2\pi\zeta(s)}{\Pi(1-s)Z^B(s)}$.
$\zeta_2(s)$ can also be substituted into the functional equation. A plot of the logarithms of the n values of the inflection points for the left side of the equation is

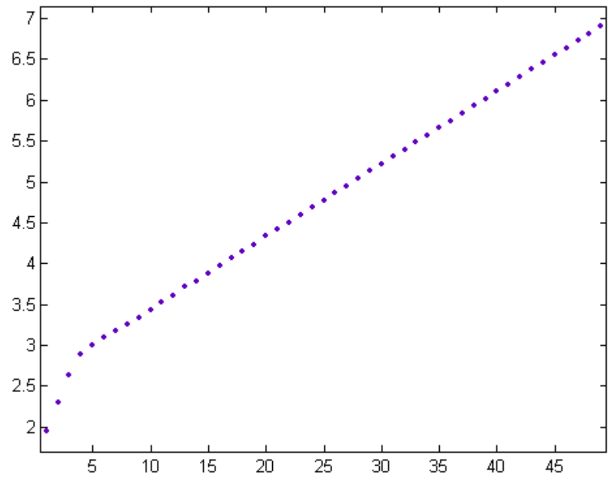

Figure 20:

For a linear least-squares fit of the curve (disregarding the first three n values), $p_1 = 0.089$ with a 95% confidence interval of (0.0889, 0.08911), $p_2 = 2.546$ with a 95% confidence interval of (2.542, 2.549), SSE=0.0009971, R-squared=1, and RMSE=0.004761. A plot of the logarithms of the n values of the inflection points for the right side of the equation is

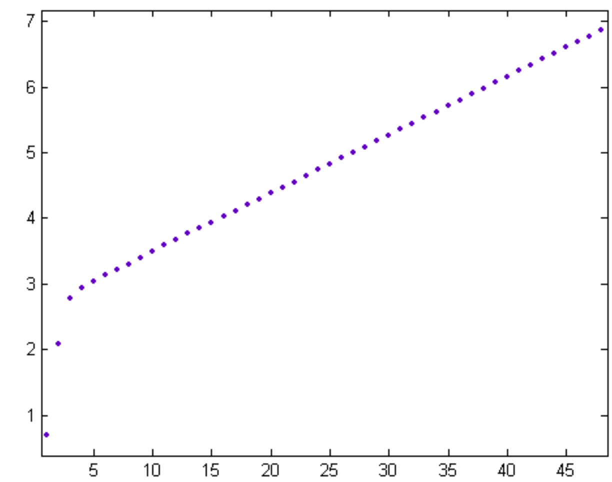

Figure 21:

For a linear least-squares fit of the curve (disregarding the first three n values), $p_1 = 0.0889$ with a 95% confidence interval of (0.08881, 0.08899), SSE=0.0006367, R-squared=1, and RMSE=0.003848. The parameters are almost the same.
A plot of the superimposed logarithmic spirals for $100 \leq n \leq 1000$ is

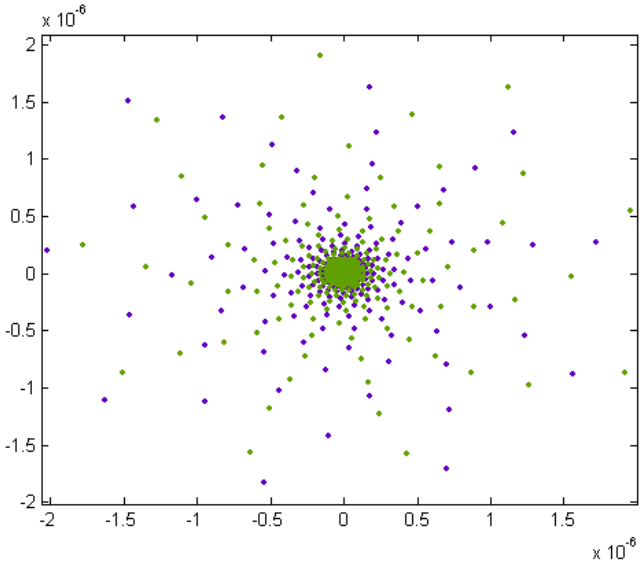

Figure 22:

A plot of $\zeta_2(s)$ for $s = (0.5, 51.3720769775)$ and $100 \leq n \leq 1000$ is

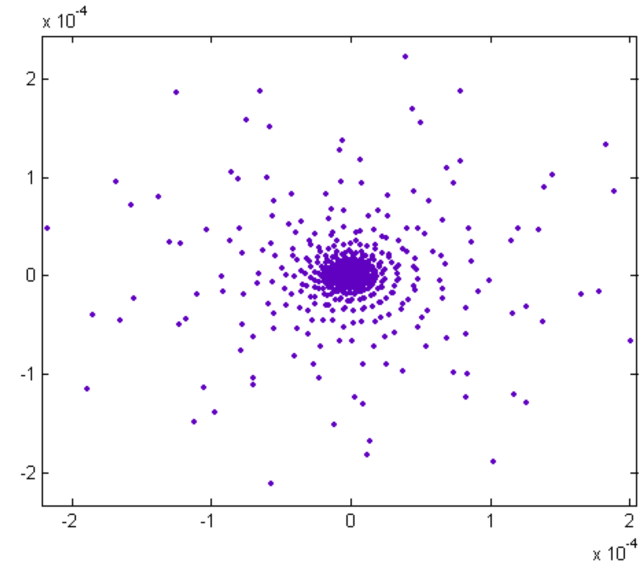

Figure 23:

The logarithmic spiral is still centered on (0, 0).

5. METHODS

C code for computing the two generalizations of the Riemann zeta function is as follows. The prime look-up table contains the primes less than 1500000.

```
//
// compute Mobius function
//
#include <math.h>
#include <stdio.h>
#include ”table5.h” // prime look-up table
extern char *malloc();
int mobius(unsigned int a, unsigned int *table, unsigned int tsize) {
unsigned int i,count,p;
if (a==1)
    return(1);
count=0;
for (i=0; i<tsize; i++) {
    p=table[i];
    if (p>a)
        break;
    if (a==(a/p)*p) {
        a=a/p;
        if (a==(a/p)*p)
            return(0);
        count=count+1;
        if (a==1)
            break;
        }
    }
if ((count&1)==0)
    return(1);
else
    return(-1);
}
//
// compute Euler’s phi function
//
int mobius(unsigned int a, unsigned int *t, unsigned int tsize);
unsigned int nueuler(unsigned int n, unsigned int *table,
```

```
unsigned int tsize) {
unsigned int d;
int sum;
if (n==1)
     return(1);
sum=0;
for (d=1; d<=n; d++) {
     if (n==(n/d)*d)
          sum=sum+(n/d)*mobius(d, table, tsize);
     }
return((unsigned int)sum);
}
//
// Generalization of Riemann zeta function
//
unsigned int nueuler(unsigned int a, unsigned int *table, unsigned int tsize);
unsigned int max=1000;
double s=.50; // 0.50 usually
//double t=51.3720769775;
//double t=144.556414146;
double t=14.13472514173470;
//double t=21.02203963877156;
//double t=25.01085758014569;
//double t=30.42487612585951;
//double t=32.93506158773919;
//double t=37.58617815882568;
//double t=40.91871901214750;
//double t=43.32707328091500;
//double t=48.00515088116716;
//double t=49.77383247767230;
//double t=52.97032147771446;
//double t=56.44624769706339;
//double t=59.34704400260235;
//double t=60.83177852460981;
//double t=65.11254404808160;
//double t=67.07981052949417;
//double t=69.54640171117399;
//double t=72.06715767448191;
//double t=75.70469069908393;
```

```
//double t=77.14484006887480;
double pi=3.14159265359;
unsigned int n=1; // select n
unsigned int xmin=0;
unsigned int out=1; // 2 for top, 3 for left, 4 for bottom, 5 for right
// usually set to 4 when the spiral is not output
unsigned int flag=1; // if set, multiply by π^{-s/2}
unsigned int skip=0; // set to 1 to get first part, 2 to get second part
unsigned int tsize=114155; // size of prime look-up table
void main() {
unsigned int temp,x;
double temp1,temps,tempt,prods,a,b,c,d,e,f,g,sums,sumt,sumr,sumi;
double *rsave,*isave,oldsumr,oldsumi,R,I,z;
FILE *Outfp;
Outfp = fopen("double4.dat","w");
if (max>1500000) {
    printf("max too large \n");
    return;
    }
rsave=(double*) malloc(16000004);
if (rsave==NULL)
    return;
isave=(double*) malloc(16000004);
if (isave==NULL)
    return;
//
// Π(s/2)ζ(s)Θ^B(s)π^{-s/2}
//
if (s>=0.0)
    temp1=pow(pi,s/2.0);
else {
    temp1=pow(pi,-s/2.0);
    temp1=1.0/temp1;
    }
temps=temp1*(cos(t*log(pi/2.0)));
tempt=temp1*(sin(t*log(pi/2.0)));
a=temps*temps+tempt*tempt;
e=temps/a;
f=-tempt/a;
```

```
prods=1.0;
sums=0.0;
sumt=0.0;
sumr=0.0;
sumi=0.0;
for (x=1; x<=max; x++) {
    if (n==0)
        temp=nueuler(x,table,tsize);
    else
        temp=x;
    prods=prods*(double)temp/((double)temp+s);
    if (s>=0.0)
        temp1=pow((double)(x+1),s/2.0);
    else {
        temp1=pow((double)(x+1),-s/2.0);
        temp1=1.0/temp1;
        }
    temps=temp1*(cos(t*log(x+1)));
    tempt=temp1*(sin(t*log(x+1)));
    a=prods*temps-tempt;
    b=prods*tempt+temps;
    if (s>=0.0)
        temp1=pow((double)x,s);
    else {
        temp1=pow((double)x,-s);
        temp1=1.0/temp1;
        }
    temps=temp1*(cos(t*log(x)));
    tempt=temp1*(sin(t*log(x)));
    temp1=temps*temps+tempt*tempt;
    c=temps/temp1;
    d=tempt/temp1;
    sums=sums+c;
    sumt=sumt-d;
    temps=a*sums-b*sumt;
    tempt=a*sumt+b*sums;
//
// Barnes G-function
//
```

```
    temp1=exp((double)x*s);
    a=temp1*(cos((double)x*t));
    b=temp1*(sin((double)x*t));
    g=a*a+b*b;
    sumr=sumr+a*(double)x/g;
    sumi=sumi-b*(double)x/g;
    a=temps*sumr-tempt*sumi;
    b=temps*sumi+tempt*sumr;
    if (flag!=0) {
        temps=a*e-b*f;
        tempt=a*f+b*e;
        a=temps;
        b=tempt;
        }
    rsave[x-1]=a;
    isave[x-1]=b;
    }
sumr=0.0;
sumi=0.0;
oldsumi=0.0;
oldsumr=0.0;
for (x=1; x<=max; x++) {     if (n==0)
        temp=nueuler(x,table,tsize);
    else
        temp=x;
    if (s>=0.0)          temp1=pow((double)temp,s);
    else {
        temp1=pow((double)temp,-s);
        temp1=1.0/temp1;
        }
    R=temp1*cos(t*log((double)temp));
    I=temp1*sin(t*log((double)temp));
    temp1=R*R+I*I;
    sumr=sumr+R/temp1;
    sumi=sumi-I/temp1;
    c=sumr;
    d=sumi;
    a=rsave[x-1];
    b=isave[x-1];
```

```
    if (skip==0) {
        z=c*c+d*d;
        temps=(c*a+d*b)/z;
        tempt=(c*b-d*a)/z;
        tempt=-tempt;
        }     if (skip==1) {
        temps=a;
        tempt=b;
        }
    if (skip==2) {
        temps=c;
        tempt=d;
        }
    if (x>xmin) {
        if (out==1)
            fprintf(Outfp," %.10llf, %.10llf \n",temps,tempt);
        if ((out==2)&&((oldsumr>0.0)&&(temps<0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        if ((out==3)&&((oldsumi>0.0)&&(tempt<0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        if ((out==4)&&((oldsumr<0.0)&&(temps>0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        if ((out==5)&&((oldsumi<0.0)&&(tempt>0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        oldsumr=temps;
        oldsumi=tempt;
        }
    if (x==(x/1000)*1000)
        printf(" x=%d \n",x);
indent }
fclose(Outfp);
return;
}

#include <math.h>
#include <stdio.h>
#include "table5.h" // prime look-up table
extern char *malloc();
//
```

```
// Generalizaiton of Riemann Zeta Function
//
unsigned int nueuler(unsigned int a, unsigned int *table, unsigned int tsize);
unsigned int max=1000;
double s=.50; // 0.50 usually
//double t=13.5;
double t=14.13472514173470;
//double t=21.02203963877156;
//double t=25.01085758014569;
//double t=30.42487612585951;
//double t=32.93506158773919;
//double t=37.58617815882568;
//double t=40.91871901214750;
//double t=43.32707328091500;
//double t=48.00515088116716;
//double t=49.77383247767230;
//double t=52.97032147771446;
//double t=56.44624769706339;
//double t=59.34704400260235;
//double t=60.83177852460981;
//double t=65.11254404808160;
//double t=67.07981052949417;
//double t=69.54640171117399;
//double t=72.06715767448191;
//double t=75.70469069908393;
//double t=77.14484006887480;
double pi=3.14159265359;
unsigned int n=1; // select n
unsigned int xmin=0;
unsigned int out=4; // 2 for top, 3 for left, 4 for bottom, 5 for right
// usually set to 4 when the spiral is not output
unsigned int tsize=114155; // size of prime look-up table
void main() {
unsigned int temp,x;
double temp1,temps,tempt,prods,a,b,c,d,sums,sumt,sumr,sumi,g,olds,oldt;
double *rsave,*isave,z,R,I;
FILE *Outfp;
Outfp = fopen("doubley.dat","w");
rsave=(double*) malloc(16000004);
```

```
if (rsave==NULL)
    return;
isave=(double*) malloc(16000004);
if (isave==NULL)
    return;
if (max>1500000) {    printf("max too large \n");
    return;
    }
//
// 2πζ(s) / (Π(1−s)Z^B(s))
// Π(s/2)Θ^B(s)ζ(s)
//
prods=1.0;
sums=0.0;
sumt=0.0;
sumr=0.0;
sumi=0.0;
for (x=1; x<=max; x++) {
    if (n==0)
        temp=nueuler(x,table,tsize);
    else
        temp=x;
    prods=prods*(double)temp/((double)temp+s);
    z=1.0-s;
    if (z>=0.0)
        temp1=pow((double)(x+1),z);
    else {
        temp1=pow((double)(x+1),-z);
        temp1=1.0/temp1;
        }
    temps=temp1*(cos(t*log(x+1)));
    tempt=temp1*(sin(t*log(x+1)));
    a=prods*temps-tempt;
    b=prods*tempt+temps;
    if (s>=0.0)
        temp1=pow((double)x,s);
    else {
        temp1=pow((double)x,-s);
        temp1=1.0/temp1;
```

```
        }
    temps=temp1*(cos(t*log(x)));
    tempt=temp1*(sin(t*log(x)));
    temp1=temps*temps+tempt*tempt;
    c=temps/temp1;
    d=tempt/temp1;
    sums=sums+c;
    sumt=sumt-d;
    z=a*a+b*b;
    temps=(a*sums+b*sumt)/z;
    tempt=(a*sumt-b*sums)/z;
    tempt=-tempt;
    c=temps*2.0*pi;
    d=tempt*2.0*pi;
    rsave[x-1]=c;
    isave[x-1]=d;
    }
sumr=0.0;
sumi=0.0;
sums=0.0;
sumt=0.0;
for (x=1; x<=max; x++) {
    if (n==0)
        temp=nueuler(x,table,tsize);
    else
        temp=x;
    if (s>=0.0)
        temp1=pow((double)temp,s);
    else {
        temp1=pow((double)temp,-s);
        temp1=1.0/temp1;
        }
    R=temp1*cos(t*log((double)temp));
    I=temp1*sin(t*log((double)temp));
    temp1=R*R+I*I;
    sumr=sumr+R/temp1;
    sumi=sumi-I/temp1;
    a=sumr*(double)temp;
    b=sumi*(double)temp;
```

```
    z=s-1.0;
    if (z>=0.0)
        temp1=pow((double)temp,z);
    else {
        temp1=pow((double)temp,-z);
        temp1=1.0/temp1;
        }
    R=temp1*cos(t*log((double)temp));
    I=temp1*sin(t*log((double)temp));
    temp1=R*R+I*I;
    sums=sums+R/temp1;
    sumt=sumt-I/temp1;
    c=a*sums-b*sumt;
    d=a*sumt+b*sums;
    a=rsave[x-1];
    b=isave[x-1];
    z=c*c+d*d;
    temps=(c*a+d*b)/z;
    tempt=(c*b-d*a)/z;
    tempt=-tempt;
    c=tempt;
    d=temps;
    rsave[x-1]=c;
    isave[x-1]=d;
    }
olds=0.0;
oldt=0.0;
prods=1.0;
sums=0.0;
sumt=0.0;
sumr=0.0;
sumi=0.0;
for (x=1; x<=max; x++) {
    if (n==0)
        temp=nueuler(x,table,tsize);
    else
        temp=x;
    prods=prods*(double)temp/((double)temp+s);
    if (s>=0.0)
```

```
        temp1=pow((double)(x+1),s/2.0);
    else {
        temp1=pow((double)(x+1),-s/2.0);
        temp1=1.0/temp1;
        }
    temps=temp1*(cos(t*log(x+1)));
    tempt=temp1*(sin(t*log(x+1)));
    a=prods*temps-tempt;
    b=prods*tempt+temps;
    if (s>=0.0)
    temp1=pow((double)x,s);
    else {
        temp1=pow((double)x,-s);
        temp1=1.0/temp1;
        }
    temps=temp1*(cos(t*log(x)));
    tempt=temp1*(sin(t*log(x)));
    temp1=temps*temps+tempt*tempt;
    c=temps/temp1;
    d=tempt/temp1;
    sums=sums+c;
    sumt=sumt-d;
    temps=a*sums-b*sumt;
    tempt=a*sumt+b*sums;
//
// Barnes G-function
//
    temp1=exp((double)x*s);
    a=temp1*(cos((double)x*t));
    b=temp1*(sin((double)x*t));
    g=a*a+b*b;
    sumr=sumr+a*(double)x/g;
    sumi=sumi-b*(double)x/g;
    a=temps*sumr-tempt*sumi;
    b=temps*sumi+tempt*sumr;
    c=rsave[x-1];
    d=isave[x-1];
    temps=a*c-b*d;
    tempt=a*d+b*c;
```

```
    if (x>xmin) {
        if (out==1)
            fprintf(Outfp," %.10llf %.10llf \n",temps,tempt);
        if ((out==2)&&((olds>0.0)&&(temps<0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        if ((out==3)&&((oldt>0.0)&&(tempt<0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        if ((out==4)&&((olds<0.0)&&(temps>0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        if ((out==5)&&((oldt<0.0)&&(tempt>0.0)))
            fprintf(Outfp," %.10llf \n",log((double)x));
        olds=temps;
        oldt=tempt;
        }
    }
fclose(Outfp);
return;
}
```

REFERENCES

[1] H. M. Edwards, *Riemann's Zeta Function*, Dover, (1974)

[2] Vladislav Kargin, Statistical properties of zeta functions' zeros, Probability Surveys, Vol. 11 (2014) 121-160

[3] A. Voros, *Communications in Mathematical Physics*, 439-465(1987), Springer-Verlag