Recent Research in Science and Technology 2011, 3(7): 77-84
ISSN: 2076-5061
www.scholarjournals.org

www.recent-science.com

RRST-Mathematics

Relative Accuracy of Numerical Quadrature Formulas - Trapezoid Rule, Simpson's 1/3 Rule, Simpson's 3/8 Rule and Boole's Rule

R. B. Srivastava*, Jai Singh Yadav

Department of Mathematics, M. L. K. P. G. College, Balrampur, U. P., India

Article Info

Article History

Received : 18-05-2011
Revised : 02-07-2011
Accepted : 12-07-2011

***Corresponding Author**

Tel : +919415036245

Email:
rambux@gmail.com

Abstract

We have calculated the definite integral by dividing the interval of integration [-1, 1] into 96 equal parts in trapezoidal rule, 192 equal parts in Simpson's 1/3 rule, 288 equal parts in Simpson's 3/8 rule and 384 equal parts in Boole's rule by developing computer programs in C++ language. Error in the values of integral calculated by quadrature formulas is minimum when upper limit is in the neighborhood of zero and lower limit is -1. Accuracy of the quadrature formulas has been found in the order- Simpson's three-eighth rule > Simpson's one-third rule > Boole's rule > Trapezoidal rule.

Key Words: Trapezoidal rule, Simpson's rule, Boole's rule, Quadrature formula, Definite Integral

Introduction

The field of numerical analysis predates the invention of modern computers by many centuries. Linear interpolation was already in use more than 2000 years ago. Many great mathematicians of the past were preoccupied by numerical analysis, as it obvious from the names of important algorithms like Newton's method, Lagrange interpolation polynomial, Gaussian elimination, or Euler's method [1].

To facilitate computations by hand, large books were produced with formulas and tables of data such as interpolation points and function coefficients [2-4]. Using these tables, often calculated out to 16 decimal places or more for some functions, one could look up values to plug into the formulas given and achieve very good numerical estimates of some functions. The canonical work in the field is the NIST publication edited by Abramowitz and Stegun, [5] a 1000-plus page book of a very large number of commonly used formulas and functions and their values at many points. The function values are no longer very useful when a computer is available, but the large listing of formulas can still be very handy.

The mechanical calculator was also developed as a tool for hand computation. These calculators evolved into electronic computers in the 1940s, and it was then found that these computers were also useful for administrative purposes. But the invention of the computer also influenced the field of numerical analysis, since now longer and more complicated calculations could be done.

Numerical quadrature is another name for numerical integration, which refers to the approximation of an integral $\int f(x)dx$ of some function f(x) by a discrete summation $\Sigma w_i f(x_i)$ over points x_i with some weights w_i. There are many methods of numerical quadrature corresponding to different choices of points x_i and weights w_i, from Euler integration to sophisticated methods such as Gaussian quadrature,[6] with varying degrees of accuracy for various types of functions f(x).

Popular methods use one of the Newton–Cotes formulas (like the midpoint rule or Simpson's rule) or Gaussian quadrature. These methods rely on a "divide and conquer" strategy, whereby an integral on a relatively large set is broken down into integrals on smaller sets. In higher dimensions, where these methods become prohibitively expensive in terms of computational effort, one may use Monte Carlo or quasi-Monte Carlo methods or, in modestly large dimensions, the method of sparse grids.

More accurate integration formulas with smaller truncation error can be obtained by interpolating several data points with higher-order interpolating polynomials. For example, the fourth-order interpolating polynomial $P_4(t)$ between five data points leads to the Boole's rule of numerical integration. The Boole's rule has the global truncation error of order $O(h^6)$. However, the higher-order interpolating polynomials often do not provide good approximations for integrals because they tend to oscillate wildly between the samples (polynomial wiggle). As a result, they are seldom used past Boole's rule. Another popular numerical algorithm is used instead to reduce the truncation error of numerical integration. This is Romberg integration based on the Richardson extrapolation algorithm

In numerical analysis, the Newton–Cotes formulas are a group of formulas for numerical integration (also called quadrature) based on evaluating the integrand at n+ 1 equally-spaced point. Newton–Cotes formulas can be useful if the value of the integrand at equally-spaced points is given. If it is possible to change the points at which the integrand is

evaluated, then other methods such as Gaussian quadrature [6-7] and Clenshaw–Curtis quadrature [8] are probably more suitable.

It is assumed that the value of a function f is known at equally spaced points x_i, for i = 0, …, n. There are two types of Newton–Cotes formulas, the "closed" type which uses the function value at all points, and the "open" type which does not use the function values at the endpoints. The closed Newton-Cotes formula [9-10] of degree n is stated as

$$\int_a^b f(x)\,dx \approx \sum_{i=0}^{n} w_i\, f(x_i)$$

where $x_i = h\, i + x_0$, with h (called the step size) equal to $(x_n - x_0)/n$. The w_i are called weights.

As can be seen in the following derivation the weights are derived from the Lagrange basis polynomials. This means they depend only on the x_i and not on the function f. Let L(x) be the interpolation polynomial in the Lagrange form for the given data points $(x_0, f(x_0))$, …, $(x_n, f(x_n))$, then

$$\int_a^b f(x)\,dx \approx \int_a^b L(x)\,dx = \int_a^b \sum_{i=0}^{n} f(x_i)\, l_i(x)\,dx = \sum_{i=0}^{n} f(x_i) \underbrace{\int_a^b l_i(x)\,dx}_{w_i}.$$

The open Newton–Cotes formula of degree n is stated as

$$\int_a^b f(x)\,dx \approx \sum_{i=1}^{n-1} w_i\, f(x_i)$$

A Newton–Cotes formula of any degree n can be constructed. However, for large n a Newton–Cotes rule can sometimes suffer from catastrophic Runge's phenomenon where the error grows exponentially for large n. Methods such as Gaussian quadrature and Clenshaw–Curtis quadrature with unequally spaced points (clustered at the endpoints of the integration interval) are stable and much more accurate, and are normally preferred to Newton–Cotes. If these methods can not be used, because the integrand is only given at the fixed equi-distributed grid, then Runge's phenomenon can be avoided by using a composite rule, as explained in next section.

Material and Method

In order to find out the relative numerical accuracy of quadrature formulas, we have calculated the definite integral

$$\int_{-1}^{1} (x^2 + 2)\,dx$$

by dividing the interval of integration [-1, 1] into 96 equal parts in trapezoidal rule, 192 equal parts in Simpson's 1/3 rule, 288 equal parts in Simpson's 3/8 rule and 384 equal parts in Boole's rule. These rules[11-13] are given below-

Trapezoidal Rule:

$$\int_{x1}^{xn} f(x)dx = \sum_{k=1}^{n-1} \int_{xk}^{xk+1} f(x)dx$$

$$= \sum_{k=1}^{n-1} h[\,f(x_{k+1}) + f(x_k)]/2$$

where $h=x_{k+1} - x_k$

Simpson's 1/3 Rule:

$$\int_{x1}^{xn} f(x)dx = \sum_{k=1}^{(n-1)/2} \int_{x2k-1}^{x2k+1} f(x)dx$$

$$= \sum_{k=1}^{(n-1)/2} h[\,f(x_{2k-1}) + 4f(x_{2k}) + f(x_{2k+1})]/3$$

where $h=x_{2k+1} - x_{2k}$, n>3 and n are odd numbers

Simpson's 3/8 Rule:

$$\int_{x1}^{xn} f(x)dx = \sum_{k=1}^{(n-1)/3} \int_{x3k-2}^{x3k+1} f(x)dx$$

$$= \sum_{k=1}^{(n-1)/3} h[3f(x_{3k-2}) + 9f(x_{3k-1}) + 9f(x_{3k}) + 3f(x_{3k+1})]/8$$

where $h=x_{3k+1} - x_{3k}$, (n-1)/3 are positive integers.

Boole's Rule:

$$\int_{x1}^{xn} f(x)dx = \sum_{k=1}^{(n-1)/4} \int_{x4k-3}^{x4k+1} f(x)dx$$

$$= \sum_{k=1}^{(n-1)/4} h[14f(x_{4k-3}) + 64f(x_{4k-2}) + 24f(x_{4k-1}) + 64f(x_{4k}) + 14f(x_{4k+1})]/45$$

where $h = x_{4k-3} - x_{4k-2} = x_{4k-2} - x_{4k-1} = \ldots = x_{4k+2} - x_{4k+1}$, (n-1)/4 are positive integers.

The values of the definite integral have been obtained with the help of following C++ programs developed by us-

C++ Program for Trapezoidal Rule

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
//Trapezoidal rule
void main(void)
{
        FILE *fpt;
        int i, n=97;
        float a, b, h, x[200];
        double s=0.0,ev;
        double f(float x);
        double g(float x);
        clrscr();
        // filename
        fpt=fopen("jst1.txt", "w");
        printf("a= ");
        scanf("%f", &a);
        printf("b= ");
        scanf("%f", &b);
        h=(b-a)/(n-1);
```

```
                fprintf(fpt,"a= %6.2f\n",a);
                fprintf(fpt,"b= %5.2f\n",b);
                fprintf(fpt,"h= %10.8f\n",h);
                //Function
                fprintf(fpt,"f(x)=x^2+2\n");
                fprintf(fpt,"S.No. Lower Limit   Upper Limit
                Value by Trepezoidal      Exact Value\n");
                for(i=1; i<=n; i++)
                x[i]=a+(i-1)*h;
                for(i=1; i<n; i++)
                {
                  s=s+h*(f(x[i+1])+f(x[i]))/2;
                  ev=g(x[i+1])-g(x[1]);
                  fprintf(fpt,"%4d   %7.2f  %8.3f %22.15f
                %22.15f\n",i,a,x[i+1],s,ev);
                }

                fclose(fpt);
}
double f(float x)
{
                double r;
                r=x*x+2;
                return(r);
}
double g(float x)
{
                 double r;
                 r=x*x*x/3 + 2*x;
                 return(r);
}
```

C++ Program for Simpson's 1/3Rule

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
//Simpson's rule
void main(void)
{
  FILE *fpt;
  int i, n=193;
  float a, b, h, x[200];
  double s=0.0,ev;
  double f(float x);
  double g(float x);
  clrscr();
  //to change filename
  fpt=fopen("jss1.txt", "w");
  printf("a= ");
  scanf("%f", &a);
  printf("b= ");
  scanf("%f", &b);
  h=(b-a)/(n-1);
  fprintf(fpt,"a= %6.2f\n",a);
  fprintf(fpt,"b= %5.2f\n",b);
  fprintf(fpt,"h= %10.8f\n",h);
  //Function
  fprintf(fpt,"f(x)=x^2+2\n");
  fprintf(fpt,"S.No. Lower Limit    Upper Limit      Value by
  Simpson1/3        Exact Value\n");
  for(i=1; i<=n; i++)
 x[i]=a+(i-1)*h;
  for(i=1; i<=(n-1)/2; i++)
  {
    s=s+h*(f(x[2*i-1])+4*f(x[2*i])+f(x[2*i+1]))/3;
    ev=g(x[2*i+1])-g(x[1]);
    fprintf(fpt,"%4d  %7.2f  %8.3f %22.15f
   %22.15f\n",i,a,x[2*i+1],s,ev);
  }

  fclose(fpt);
 }
double f(float x)
{
  double r;
  r=x*x+2;
  return(r);
}
double g(float x)
{
   double r;
   r=x*x*x/3 + 2*x;
   return(r);
}
```

C++ Program for Simpson's 3/8 Rule

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
//Simpson's 3/8 rule
void main(void)
{
  FILE *fpt;
  int i, n=289;
  float a, b, h, x[300];
  double s=0.0,ev;
  double f(float x);
  double g(float x);
  clrscr();
  //to change filename
  fpt=fopen("jsst1.txt", "w");
  printf("a= ");
  scanf("%f", &a);
  printf("b= ");
  scanf("%f", &b);
  h=(b-a)/(n-1);
  fprintf(fpt,"a= %6.2f\n",a);
  fprintf(fpt,"b= %5.2f\n",b);
  fprintf(fpt,"h= %10.8f\n",h);
  //Function
  fprintf(fpt,"f(x)=x^2+2\n");
  fprintf(fpt,"S.No. Lower Limit    Upper Limit      Value by
 Simpson3/8       Exact Value\n");
  for(i=1; i<=n; i++)
 x[i]=a+(i-1)*h;
  for(i=1; i<=(n-1)/3; i++)
  {
 s=s+h*(3*f(x[3*i-2])+9*f(x[3*i-1])+9*f(x[3*i])+3*f(x[3*i+1]))
        /8;
    ev=g(x[3*i+1])-g(x[1]);
```

```
    fprintf(fpt,"%4d  %7.2f  %8.3f %22.15f
%22.15f\n",i,a,x[3*i+1],s,ev);
  }

  fclose(fpt);
}
double f(float x)
{
  double r;
  r=x*x+2;
  return(r);
}
double g(float x)
{
   double r;
   r=x*x*x/3 + 2*x;
   return(r);
}
```

C++ Program for Boole's Rule

```
#include<conio.h>
#include<stdio.h>
#include<math.h>
//boole's rule
void main(void)
{
  FILE *fpt;
  int i, n=385;
  float a, b, h, x[400];
  double s=0.0,ev;
  double f(float x);
  double g(float x);
  clrscr();
  //to change filename
  fpt=fopen("jsb1.txt", "w");
  printf("a= ");
  scanf("%f", &a);
  printf("b= ");
  scanf("%f", &b);
  h=(b-a)/(n-1);
  fprintf(fpt,"a= %6.2f\n",a);
  fprintf(fpt,"b= %5.2f\n",b);
  fprintf(fpt,"h= %10.8f\n",h);
  //Function
  fprintf(fpt,"f(x)=x^2+2\n");
  fprintf(fpt,"S.No. Lower Limit   Upper Limit      Value by
  Booles         Exact Value\n");
  for(i=1; i<=n; i++)
x[i]=a+(i-1)*h;
  for(i=1; i<=(n-1)/4; i++)
  {
    s=s+h*(14*f(x[4*i-3])+64*f(x[4*i-2])+24*f(x[4*i-1])+
    64*f(x[4*i])+14* f(x[4*i+1])) /45;
    ev=g(x[4*i+1])-g(x[1]);
    fprintf(fpt,"%4d  %7.2f  %8.3f %22.15f
%22.15f\n",i,a,x[4*i+1],s,ev);
  }
  fclose(fpt);
}
double f(float x)
{
  double r;
  r=x*x+2;
  return(r);
}
double g(float x)
{
   double r;
   r=x*x*x/3 + 2*x;
   return(r);
}
```

Result and Discussion

Values of the integral have been obtained at different upper limits in the interval [-1, 1] by keeping lower limit -1 with the help of computer programs developed by us. Methods of evaluation of the integral are trapezoidal rule, Simpson's one-third rule, Simpson's three-eighth rule and Boole's rule. Exact values of integral at different upper limits have also been obtained and then compared with the calculated values.

Values of integral at different upper limits in the interval [-1, 1] by keeping lower limit -1 have been obtained by trapezoidal rule and are shown in Table-1. This Table also contains the exact values, error in calculated values and percentage error in calculated values. Graph between errors at various upper limits are shown in Graph-1 which indicates that the error increases as upper limit increases. Close look to Table-1 indicates that the maximum error is 0.0001448563 and average percentage error is 0.0029852961.

With the help of Simpson's one-third rule the value of integral has been calculated and shown in Table-2. Error in the value of integral is shown in Graph-2. It is clear from Graph-2 that the error oscillates if upper limit is less than -1. At the upper limit -1, the value of error increases suddenly. Maximum error in the values of integral obtained by Simpson's one-third rule is 0.0000001794 and average percentage error is 0.0000034667.

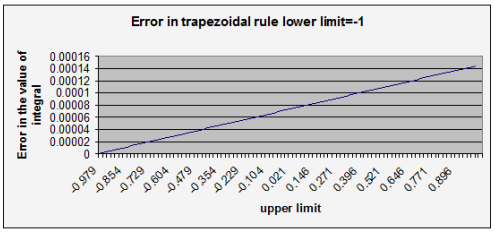
Table-3 contains the values of integral obtained by Simpson's three-eighth rule, exact value of integral, error in the calculated value and percentage error. Errors between calculated and observed values of integral at different upper limits are shown in Graph-3. It is clear from Graph-3 that the error oscillates as the upper limit approached to -1. Maximum error in the values of integral obtained by Simpson's three-eighth rule is 0.0000000723 and average percentage error is 0.0000033255.

Table-4 contains the values of integral obtained by Boole's rule, exact value of integral, error in the calculated value and percentage error. Errors between calculated and observed values of integral at different upper limits are shown in Graph-4. It is clear from Graph-4 that the error oscillates and it amazingly increases as upper limit becomes -1. Maximum error in the values of integral obtained by Simpson's three-eighth rule is 0.0000001806 and average percentage error is 0.0000034753.

It is clear that the errors in the value of integral obtained by trapezoidal rule are very high as compared to the other rules. Graph between maximum error and average percentage error for Simpson's one-third rule, Simpson's three-eighth rule and Boole's rule are shown in Graph-5 and Graph-6 respectively.

Table-1: Values of integral obtained by trapezoidal rule at different upper limits

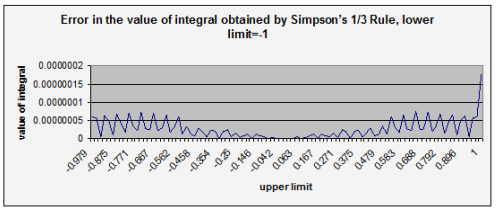
S. No.	Lower Limit	Upper Limit	Value Obtained by Trapezoidal Rule	Exact Value	Error	Percentage Error
1	-1	-0.979	0.0620704956	0.0620689275	0.0000015681	0.0025263550
11	-1	-0.771	0.6390109786	0.6389944333	0.0000165453	0.0025892777
21	-1	-0.562	1.1490388405	1.1490071615	0.0000316791	0.0027570818
31	-1	-0.354	1.6102385759	1.6101918373	0.0000467386	0.0029026739
41	-1	-0.146	2.0406946773	2.0406328768	0.0000618005	0.0030284989
51	-1	0.063	2.4584916367	2.4584147733	0.0000768634	0.0031265428
61	-1	0.271	2.8817139462	2.8816220201	0.0000919261	0.0031900829
71	-1	0.479	3.3284460983	3.3283391106	0.0001069878	0.0032144494
81	-1	0.688	3.8167725857	3.8166505380	0.0001220477	0.0031977698
91	-1	0.896	4.3647779020	4.3646407122	0.0001371898	0.0031432095
96	-1	1.000	4.6668115230	4.6666666667	0.0001448563	0.0031040636



Graph-1: Error in the value of integral obtained by trapezoidal rule at different upper limits, lower limit=-1

Table-2: Values of integral obtained by Simpson's one-third rule at different upper limits, lower limit=-1

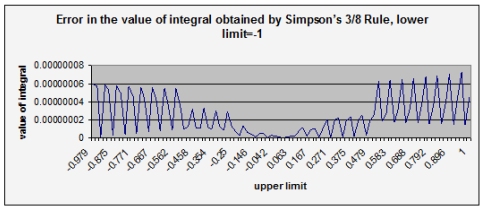
S.No.	Lower Limit	Upper Limit	Value Obtained by Simpson's 1/3 Rule	Exact Value	Error	Percentage Error
1	-1	-0.979	0.0620689877	0.0620689275	0.0000000602	0.0000970279
11	-1	-0.771	0.6389944004	0.6389944333	0.0000000329	0.0000051457
21	-1	-0.562	1.1490071925	1.1490071615	0.0000000311	0.0000027040
31	-1	-0.354	1.6101918573	1.6101918373	0.0000000200	0.0000012426
41	-1	-0.146	2.0406328883	2.0406328768	0.0000000115	0.0000005642
51	-1	0.063	2.4584147772	2.4584147733	0.0000000040	0.0000001607
61	-1	0.271	2.8816220165	2.8816220201	0.0000000036	0.0000001252
71	-1	0.479	3.3283390980	3.3283391106	0.0000000125	0.0000003759
81	-1	0.688	3.8166505137	3.8166505380	0.0000000243	0.0000006356
91	-1	0.896	4.3646407603	4.3646407122	0.0000000481	0.0000011013
96	-1	1.000	4.6666668461	4.6666666667	0.0000001794	0.0000038440



Graph-2: Error in the value of integral obtained by Simpson's one-third rule at different upper limits, lower limit=-1

Table-3: Values of integral obtained by Simpson's three-eighth rule at different upper limits

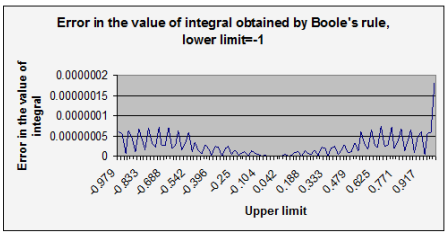
S. No.	Lower Limit	Upper Limit	Value Obtained by Simpson's 3/8 Rule	Exact Value	Error	Percentage Error
1	-1	-0.979	0.0620689872	0.0620689275	0.0000000597	0.0000961161
11	-1	-0.771	0.6389943869	0.6389944333	0.0000000464	0.0000072597
21	-1	-0.562	1.1490071702	1.1490071615	0.0000000087	0.0000007569
31	-1	-0.354	1.6101918278	1.6101918373	0.0000000095	0.0000005875
41	-1	-0.146	2.0406328508	2.0406328467	0.0000000041	0.0000002025
51	-1	0.063	2.4584147307	2.4584147285	0.0000000022	0.0000000908
61	-1	0.271	2.8816219586	2.8816219583	0.0000000003	0.0000000102
71	-1	0.479	3.3283390256	3.3283390441	0.0000000185	0.0000005546
81	-1	0.688	3.8166504230	3.8166503906	0.0000000324	0.0000008492
91	-1	0.896	4.3646406417	4.3646407122	0.0000000705	0.0000016158
94	-1	0.958	4.5433787087	4.5433787811	0.0000000723	0.0000015924
96	-1	1.000	4.6666667119	4.6666666667	0.0000000452	0.0000009695



Graph-3: Error in the value of integral obtained by Simpson's three-eighth rule at different upper limits, lower limit=-1

Table-4: Values of integral obtained by Boole's rule at different upper limits

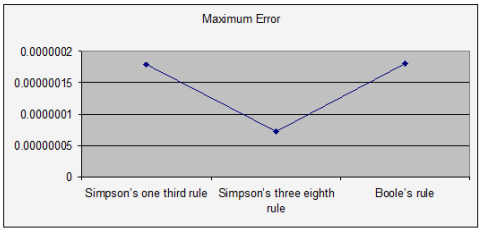
S. No.	Lower Limit	Upper Limit	Value Obtained by Boole's Rule	Exact Value	Error	Percentage Error
1	-1	-0.979	0.0620689884	0.0620689275	0.0000000609	0.0000980875
11	-1	-0.771	0.6389944010	0.6389944333	0.0000000323	0.0000050500
21	-1	-0.562	1.1490071932	1.1490071615	0.0000000318	0.0000027661
31	-1	-0.354	1.6101918579	1.6101918373	0.0000000207	0.0000012835
41	-1	-0.146	2.0406328890	2.0406328768	0.0000000122	0.0000005968
51	-1	0.063	2.4584147779	2.4584147733	0.0000000046	0.0000001877
61	-1	0.271	2.8816220170	2.8816220201	0.0000000030	0.0000001053
71	-1	0.479	3.3283390987	3.3283391106	0.0000000118	0.0000003554
81	-1	0.688	3.8166505146	3.8166505380	0.0000000234	0.0000006120
91	-1	0.896	4.3646407607	4.3646407122	0.0000000485	0.0000011104
96	-1	1	4.6666668473	4.6666666667	0.0000001806	0.0000038710



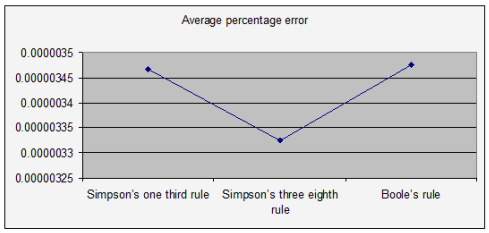
Graph-4: Error in the value of integral obtained by Boole's rule at different upper limits, lower limit=-1

Table-5: Maximum error and average percentage error in the various methods

Method	Minimum Error	Maximum Error	Average percentage error
Trapezoidal rule	0.0000015681	0.0001448563	0.0029852961
Simpson's one third rule	0.0000000001	0.0000001794	0.0000034667
Simpson's three eighth rule	0.0000000003	0.0000000723	0.0000033255
Boole's rule	0.0000000009	0.0000001806	0.0000034753



Graph-5: Maximum error Simpson's one-third rule, Simpson's three-eighth rule and Boole's rule



Graph-6: Average percentage error Simpson's one-third rule, Simpson's three-eighth rule and Boole's rule

Conclusion

Value of integral calculated by trapezoidal rule is accurate up to three places after decimal point while it is accurate up to 6 places after decimal point in Simpson's one-third, Simpson's three-eighth and Boole's rule. Values of minimum errors, maximum errors and average percentage errors are shown in Table-5 which indicates that the values calculated by Simpson's three-eighth rule is very reliable as compared to trapezoidal rule, Simpson's one-third rule and Boole's rule. Accuracy of the quadrature formulas has been found in the following order-

Simpson's three-eighth rule > Simpson's one-third rule > Boole's rule > trapezoidal rule

It is evident that Simpson's three-eight rule is better approximation for the value of integral as compared to other rules.

References

[1] Burden, Richard L.; J. Douglas Faires, (2000), Numerical Analysis (7th Ed. ed.), Brooks/Cole.

[2] Gilat, Amos, (2004), MATLAB: An Introduction with Applications (2nd edition ed.), John Wiley & Sons.

[3] Hildebrand, F. B. ,(1974), Introduction to Numerical Analysis (2nd edition ed.), McGraw-Hill.

[4] Leader, Jeffery J., (2004), Numerical Analysis and Scientific Computation, Addison Wesley.

[5] M. Abramowitz and I. A. Stegun eds., (1972), Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables. New York: Dover.

[6] Gil, Amparo; Segura, Javier; Temme, Nico M., (2007), "§5.3: Gauss quadrature", Numerical Methods for Special Functions, SIAM

[7] Press, William H.; Flannery, Brian P.; Teukolsky, Saul A.; Vetterling, William T., (1988), "§4.5: Gaussian Quadratures and Orthogonal Polynomials", Numerical Recipes in C (2nd ed.)

[8] Josef Stoer and Roland Bulirsch, (1980). Introduction to Numerical Analysis. New York: Springer-Verlag.

[9] Atkinson, Kendall A., (1989). An Introduction to Numerical Analysis (2nd edition ed.), John Wiley & Sons.

[10] Burden, Richard L. and Faires, J. Douglas, (2000). Numerical Analysis (7th edition ed.), Brooks/Cole.

[11] George E. Forsythe, Michael A. Malcolm, and Cleve B. Moler, (1977), Computer Methods for Mathematical Computations. Englewood Cliffs, NJ: Prentice-Hall.

[12] William H. Press, Brian P. Flannery, Saul A. Teukolsky, William T. Vetterling, (1988), Numerical Recipes in C. Cambridge, UK: Cambridge University Press.

[13] Jon M. Smith,(1974), Recent Developments in Numerical Integration, J. Dynam. Sys., Measurement and Control 96, Ser. G-1, No. 1, 61-70.