

# Romberg Integration

*nalib*

```
> restart;
> libname:="c:/nalib",libname;
libname := "/nalib", "/Library/Frameworks/Maple.framework/Versions/15/lib",
"/Library/Frameworks/Maple.framework/Versions/15/toolbox/NAG/lib"
> with(numanal);
[SOR, SOR_dir, adaptq, adaptq_dir, bezier, bezier_dir, bisection, bisection_dir, chop, chop_dir,
clamped_spline, clamped_spline_dir, divided_diff, divided_diff_dir, extrap, extrap_dir,
falseposition, falseposition_dir, fixedpoint, fixedpoint_dir, gaussseidel, gaussseidel_dir, hermite,
hermite_dd, hermite_dd_dir, hermite_dir, horner, horner_dir, jacobi, jacobi_dir, muller,
muller_dir, natural_spline, natural_spline_dir, newton, newton_dir, romberg, romberg_dir,
secant, secant_dir, steffensen, steffensen_dir]
```

*Page 136, Number 1h*

We will use Romberg integration to approximate the value of  $\int_0^{\frac{\pi}{4}} \cos(x)^2 dx$ . First, let's find the exact value of the integral.

```
> f:=(cos(x))^2;
f:=cos(x)^2
> actual_int:=int(f,x = 0 .. Pi/4);
actual_int := 1/4 + 1/8 pi
```

Let's create a Romberg table with 5 rows. First, we check the directions for **romberg**.

```
> romberg_dir();
romberg returns an approximation to a definite integral.
```

The arguments for romberg are:  
(1)the function being integrated  
(2)the lower limit of integration  
(3)the upper limit of integration  
(4)number of rows (0 if stopping on a tolerance)  
(5)tolerance (0 if choosing a number of rows)  
(6)variable for returning the integral approximation

If assigning the result to a variable, have the variable and the 6th argument the same.

If S is the variable for returning the integral's value and has already been given a value, the procedure should be preceded by the statement:  
S:='S'

Since the **romberg** table can only provide up to 10 decimal digits, the individual elements of the table are printed separately with up to 15 decimal digits in case more accuracy is needed and **Digits** has been

increased.

```
> S:=romberg(f,0,evalf(Pi/4),5,0,S);
```

```
R[1,1] = 0.589048622500000
R[2,1] = 0.629713943900000
R[2,2] = 0.643269051000000
R[3,1] = 0.639478032000000
R[3,2] = 0.642732728000000
R[3,3] = 0.642696973100000
R[4,1] = 0.641895374700000
R[4,2] = 0.642701155600000
R[4,3] = 0.642699050800000
R[4,4] = 0.642699083800000
R[5,1] = 0.642498251900000
R[5,2] = 0.642699211000000
R[5,3] = 0.642699081400000
R[5,4] = 0.642699081900000
R[5,5] = 0.642699081900000
```

Romberg Integration Table:

```
0.58904862
0.62971394    0.64326905
0.63947803    0.64273273    0.64269697
0.64189537    0.64270116    0.64269905    0.64269908
0.64249825    0.64269921    0.64269908    0.64269908
0.64269908
```

```
S := 0.6426990819
```

Let's check the amount of error in this approximation.

```
> approx_error:=evalf(abs(actual_int-S));
approx_error := 1. 10-10
```

This is pretty good. Suppose we want an error of less than  $10^{-13}$ . We first increase [Digits](#).

```
> Digits:=20;
```

```
Digits := 20
```

```
> S:='S';
```

```
S := S
```

```
> S:=romberg(f,0,evalf(Pi/4),0,10(-13),S);
```

```
R[1,1] = 0.589048622548086
R[2,1] = 0.629713943940854
R[2,2] = 0.643269051071777
R[3,1] = 0.639478031941918
R[3,2] = 0.642732727942273
R[3,3] = 0.642696973066972
R[4,1] = 0.641895374670267
R[4,2] = 0.642701155579716
R[4,3] = 0.642699050755546
R[4,4] = 0.642699083734729
```

```

R[5,1] = 0.642498251819992
R[5,2] = 0.642699210869901
R[5,3] = 0.642699081222580
R[5,4] = 0.642699081706183
R[5,5] = 0.642699081698228
R[6,1] = 0.642648880278729
R[6,2] = 0.642699089764974
R[6,3] = 0.642699081691313
R[6,4] = 0.642699081698753
R[6,5] = 0.642699081698724
R[6,6] = 0.642699081698724
R[7,1] = 0.642686531721749
R[7,2] = 0.642699082202756
R[7,3] = 0.642699081698608
R[7,4] = 0.642699081698724
R[7,5] = 0.642699081698724
R[7,6] = 0.642699081698724
R[7,7] = 0.642699081698724
R[8,1] = 0.642695944228106
R[8,2] = 0.642699081730224
R[8,3] = 0.642699081698722
R[8,4] = 0.642699081698724
R[8,5] = 0.642699081698724
R[8,6] = 0.642699081698724
R[8,7] = 0.642699081698724
R[8,8] = 0.642699081698724

```

Romberg Integration Table:

```

0.58904862
0.62971394    0.64326905
0.63947803    0.64273273    0.64269697
0.64189537    0.64270116    0.64269905    0.64269908
0.64249825    0.64269921    0.64269908    0.64269908
0.64269908
0.64264888    0.64269909    0.64269908    0.64269908
0.64269908
0.64268653    0.64269908    0.64269908    0.64269908
0.64269908
0.64269594    0.64269908    0.64269908    0.64269908
0.64269908

0.64269908
0.64269908    0.64269908
0.64269908    0.64269908    0.64269908

```

$S := 0.64269908169872415480$

Let's check the error here.

```
> approx_error:=evalf(abs(actual_int-S));  
approx_error := 1. 10-20
```

### Page 136, Number 5

We wish to approximate  $\int_1^5 f \, dx$  by the Romberg method knowing only that  $f(1) = 2.4142$ ,

$f(2) = 2.6734$ ,  $f(3) = 2.8974$ ,  $f(4) = 3.0976$ , and  $f(5) = 3.2804$ . With only 5 points, we are limited to using the trapezoidal rule with 1, 2, and 4 subintervals, yielding 3 rows. So as to be able to use our algorithms, we write the following **piecewise function**  $g$ . We could use any formula that has the correct values for each domain value.

```
> g:=piecewise(x<1.5,2.4142,x<2.5,2.6734,x<3.5,2.8974,x<4.5,3.0976,  
3.2804);
```

$$g := \begin{cases} 2.4142 & x < 1.5 \\ 2.6734 & x < 2.5 \\ 2.8974 & x < 3.5 \\ 3.0976 & x < 4.5 \\ 3.2804 & \text{otherwise} \end{cases}$$

Now we apply romberg's method using 3 rows.

```
> S:='S';  
S := S
```

```
> S:=romberg(g,1,5,3,0,S);
```

```
R[1,1] = 11.389200000000000  
R[2,1] = 11.489400000000000  
R[2,2] = 11.522800000000000  
R[3,1] = 11.515700000000000  
R[3,2] = 11.524466666666667  
R[3,3] = 11.524577777777778
```

Romberg Integration Table:

```
11.38920000  
11.48940000    11.52280000  
11.51570000    11.52446667    11.52457778
```

```
S := 11.524577777777778
```

### Numerical Analysis

#### Page 136, Number 1h

We load the [NumericalAnalysis](#) package.

```
> with(Student[NumericalAnalysis]):
```

We now use the [Quadrature](#) command with **method** = romberg, and **output**=value, information, or plot.

```
> s:=Quadrature(f,x=0..Pi/4,method=romberg,output=value);
```

s := 0.64269908169872415481

When using **output = information** or **output=plot**, we do not assign the **Quadrature** command to a variable.

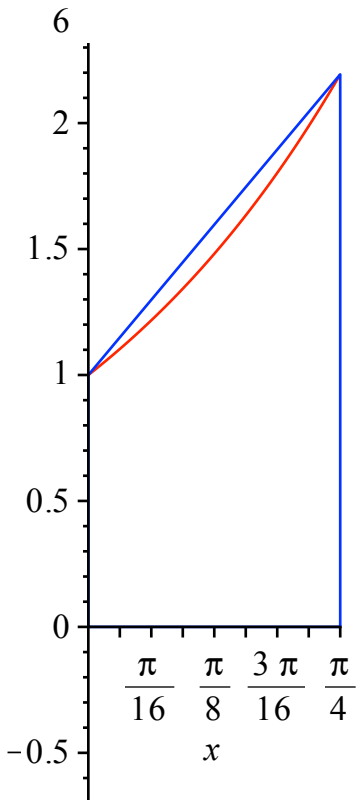
```
> Quadrature(f,x=0..Pi/4,method=romberg,output=information);  
INTEGRAL: Int(cos(x)^2,x=0..1/4*Pi) = 0.642699082  
APPROXIMATION METHOD: Romberg Integration Method with 9 Applications of  
Trapezoidal Rule
```

```
----- INFORMATION TABLE -----  
-----  
Approximate Value          Absolute Error          Relative Error  
0.642699082                0                       0 %  
----- ROMBERG INTEGRATION TABLE -----  
-----  
0.5890486  
0.6297139    0.6432691  
0.6394780    0.6427327    0.6426970  
0.6418954    0.6427012    0.6426991    0.6426991  
0.6424983    0.6426992    0.6426991    0.6426991    0.6426991  
0.6426489    0.6426991    0.6426991    0.6426991    0.6426991  
0.6426991  
0.6426865    0.6426991    0.6426991    0.6426991    0.6426991  
0.6426991    0.6426991  
0.6426959    0.6426991    0.6426991    0.6426991    0.6426991  
0.6426991    0.6426991    0.6426991  
0.6426983    0.6426991    0.6426991    0.6426991    0.6426991  
0.6426991    0.6426991    0.6426991    0.6426991
```

```
-----  
Number of Function Evaluations:    257
```

```
> Quadrature(exp(x),x=0..Pi/4,method=romberg,output=plot);
```

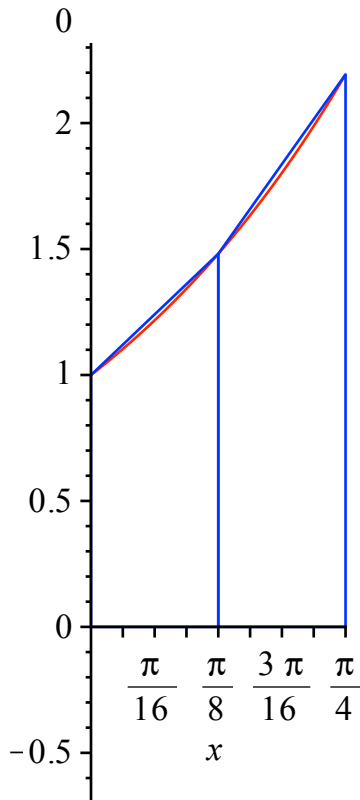
$R_{1,1}$   
 Approximation:  
 1.193280050738015456\



$f(x)$

Partitions: 1

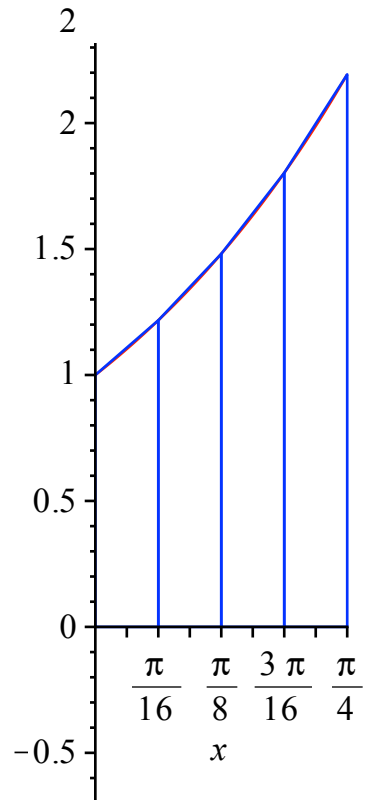
$R_{2,1}$   
 Area:  
 1.208575679488131816\



$f(x)$

Partitions: 2

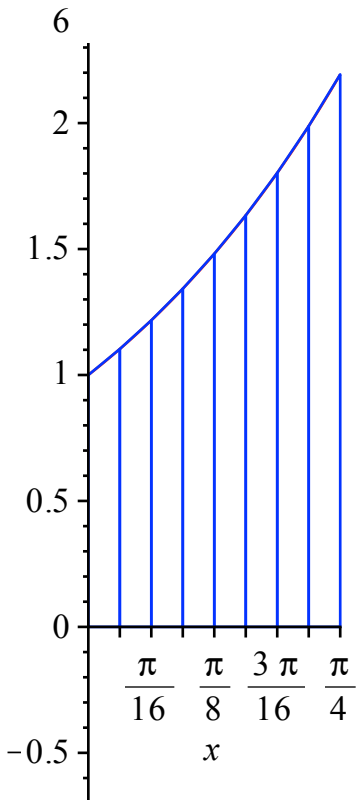
$R_{3,1}$   
 Area:  
 1.197111314250758827\



$f(x)$

Partitions: 4

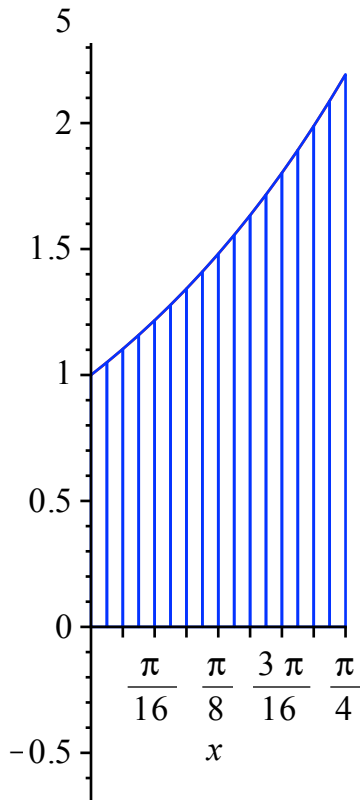
$R_{4,1}$   
Area:  
1.194238327968424998\



$f(x)$

Partitions: 8

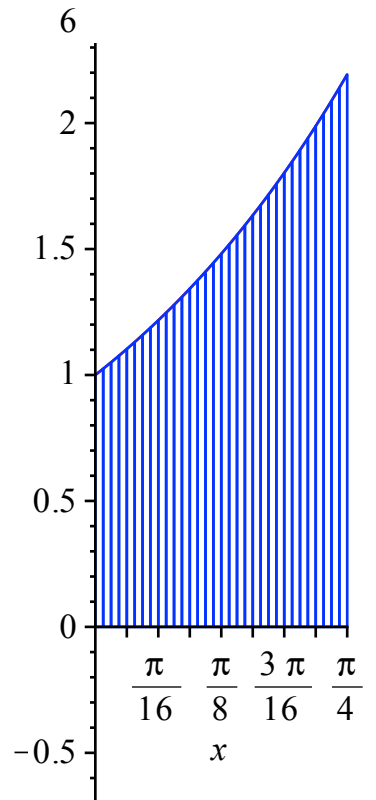
$R_{5,1}$   
Area:  
1.193519648904942769\



$f(x)$

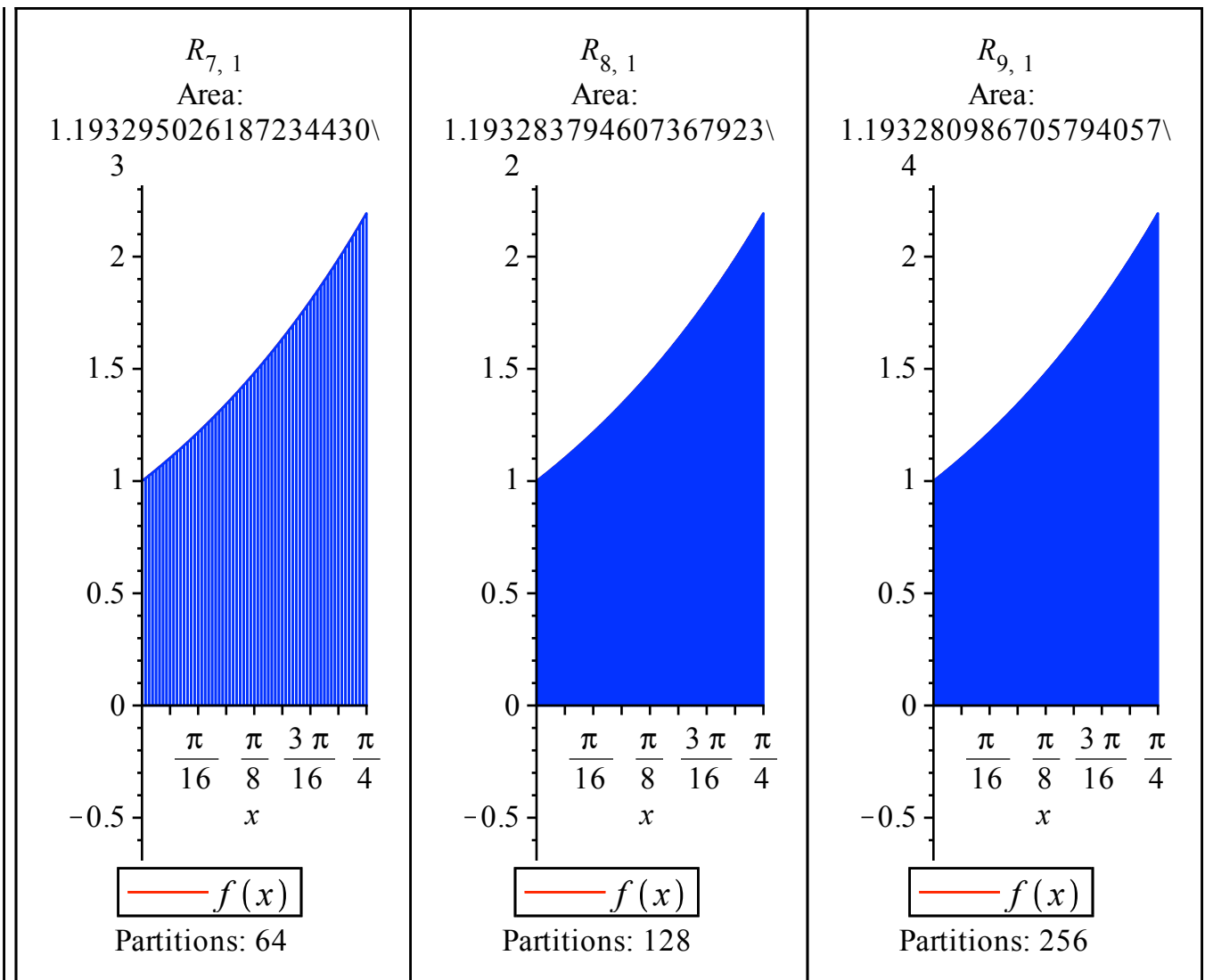
Partitions: 16

$R_{6,1}$   
Area:  
1.193339952083843131\



$f(x)$

Partitions: 32



The default appears to be nine applications of the trapezoidal rule or, equivalently, 9 Rows in the romberg table. You can choose the number of rows, say to 5, by using `method=romberg[5]`.

```
> Quadrature(f, x=0..Pi/4, method=romberg[5], output=information);
INTEGRAL: Int(cos(x)^2, x=0..1/4*Pi) = 0.642699082
APPROXIMATION METHOD: Romberg Integration Method with 5 Applications of Trapezoidal Rule
```

```
----- INFORMATION TABLE -----
-----
Approximate Value      Absolute Error      Relative Error
0.642699082           4.9576192e-13     7.714e-11 %
----- ROMBERG INTEGRATION TABLE -----
-----
0.5890486
0.6297139      0.6432691
0.6394780      0.6427327      0.6426970
0.6418954      0.6427012      0.6426991      0.6426991
0.6424983      0.6426992      0.6426991      0.6426991      0.6426991
-----
```

```
Number of Function Evaluations:      17
```

```

> Quadrature(g,x=1..5,method=romberg[3],output=information);
INTEGRAL: Int(piecewise(x < 1.5,2.4142,x < 2.5,2.6734,x < 3.5,2.8974,x
< 4.5,3.0976,3.2804),x=1..5) = 11.5157
APPROXIMATION METHOD: Romberg Integration Method with 3 Applications of
Trapezoidal Rule
----- INFORMATION TABLE -----
-----
Approximate Value          Absolute Error          Relative Error
      11.5245778          0.00887777778          0.07709 %
----- ROMBERG INTEGRATION TABLE -----
-----
11.3892000
11.4894000    11.5228000
11.5157000    11.5244667    11.5245778
-----
Number of Function Evaluations:      5

```