Numerical Integration II

Mark A. Austin

University of Maryland

austin@umd.edu
ENCE 201, Fall Semester 2023

November 13, 2023

Overview

Motivating Idea

Summary of Results: For the Trapezoid Rule:

$$I = \int_a^b f(x)dx = T_n - \frac{|f^2(\xi)|}{12}h^2(b-a). \tag{1}$$

where $[a \leq \xi \leq b]$. The method is $O(h^2)$ accurate.

For Simpson's Rule:

$$I = \int_a^b f(x)dx = S_n - \frac{h^4}{180}(b-a)|f^4(\xi)| \tag{2}$$

where $[a \leq \xi \leq b]$ and $h = (b-a)/n$ is the step length. The method is $O(h^4)$ accurate.

Motivating Idea

Motivating Idea

Reduce computational effort by finding numerical integration techniques that have high order (i.e., $E(h) = O(h^n)$, $n \geq 4$) error estimates.

Two Algorithms:

- **Gauss Quadrature.** Relax constraint that integration points need to be equally spaced.
- **Romberg Integration** (Richardson's Extrapolation). Apply Richardson's extrapolation repeatedly on the trapezium rule. Then, systematically combine error expansions to cancel out error terms of order h^2, h^4, h^6, $\cdots$.

Gaussian Quadrature

(Johann Karl Friedrich Gauss, 1777-1855)

Gaussian Quadrature

Strategy: Release condition for equally spaced sub-intervals. Transform integration integral [a,b] $\rightarrow$ [-1,1], i.e.,

$$\int_a^b f(x)dx \longrightarrow \int_{-1}^{1} \phi(u)du \tag{3}$$

where

$$\phi(u) = \frac{(b-a)}{2} f\left(\frac{(b-a)u}{2} + \frac{(a+b)}{2}\right) \tag{4}$$

Next, determine weights (w_i) and positions (u_i) such that

$$\int_{-1}^{1} \phi(u)du = w_o\phi(u_o) + w_1\phi(u_1) + \cdots \tag{5}$$

Gaussian Quadrature

One-Point Quadrature: Integrate a linear function exactly.

$$\int_{-1}^{1} \phi(u)du = w_o\phi(u_o). \tag{6}$$

Two-Point Quadrature: Integrate a cubic function exactly.

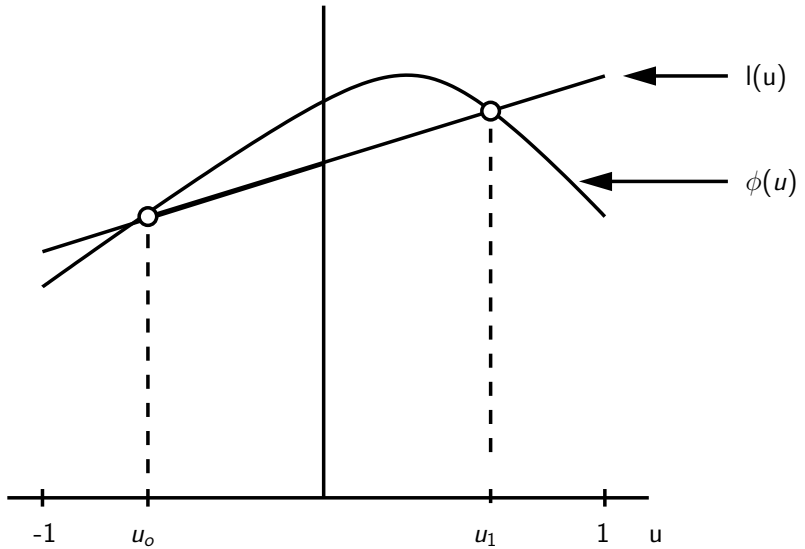
$$\int_{-1}^{1} \phi(u)du = w_o\phi(u_o) + w_1\phi(u_1) \tag{7}$$

Three-Point Quadrature: Integrate a fifth order function exactly.

$$\int_{-1}^{1} \phi(u)du = w_o\phi(u_o) + w_1\phi(u_1) + w_2\phi(u_2) \tag{8}$$

Gaussian Quadrature

Sketch of Derivation: Two-point quadrature:



We seek points $(u_o,\ u_1)$ and weights $(w_o,\ w_1)$ such that:

$$\int_{-1}^{1} \phi(u)du = \int_{-1}^{1} a+bu+cu^2+du^3du = w_o\phi(u_o)+w_1\phi(u_1). \quad (9)$$

Gaussian Quadrature

Let, $I(u) = \alpha_o + \alpha_1 u$, i.e., a straight line interpolating the points.

Can write the cubic equation as:

$$\begin{aligned}\phi(u) &= a + bu + cu^2 + du^3 \\ &= (\alpha_o + \alpha_1 u) + (u - u_o)(u - u_1)(\beta_o + \beta_1 u).\end{aligned}$$

Our goal is to choose w_o, w_1, u_o and u_1 so that

$$\int_{-1}^{1} \phi(u) - I(u)du = \int_{-1}^{1} (u - u_o)(u - u_1)(\beta_o + \beta_1 u)du = 0. \quad (10)$$

Set $\beta_o = 1$, $\beta_1 = 0 \rightarrow 2/3 + 2u_o u_1 = 0$.
Set $\beta_o = 0$, $\beta_1 = 1 \rightarrow u_o + u_1 = 0$.

Gaussian Quadrature

Solving: $u_1 = -u_o = \frac{1}{\sqrt{3}}$.

Need weights w_o and w_1

$$\int_{-1}^{1} \phi(u)du = \int_{-1}^{1} I(u)du = \int_{-1}^{1} (\alpha_o + \alpha_1 u)du = 2\alpha_o. \qquad (11)$$

From $I(u_o)$ and $I(u_1)$:

$$\int_{-1}^{1} \phi(u)du = w_o(\alpha_o + \alpha_1 u_o) + w_1(\alpha_o + \alpha_1 u_1). \qquad (12)$$

This equation must work for all values of α_o and α_1. This gives $w_o + w_1 = 1$ and $w_o - w_1 = 0$. Thus, $w_o = w_1 = 1$.

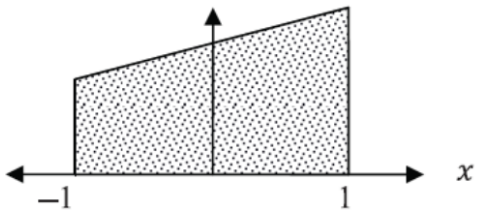
Gaussian Quadrature

Gauss Quadrature Weights (w_i) and Integration Points (u_i):

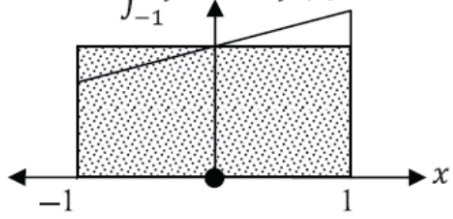
n	w_i	u_i
2	1.0 1.0	$-1/\sqrt{3}$ $1/\sqrt{3}$
3	5/9 8/9 5/9	-0.774596669 0.000000000 0.774596669
4	0.3478548 0.6521452 0.6521452 0.3478548	-0.8611363 -0.3399810 0.3399810 0.8611363

Gaussian Quadrature: Weight and Integration Points

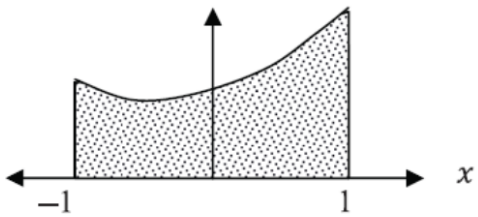
$f = a_0 + a_1 x$



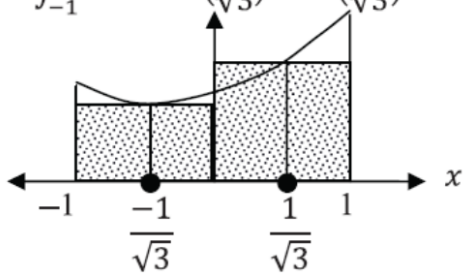
$\int_{-1}^{1} f dx = 2f(0)$



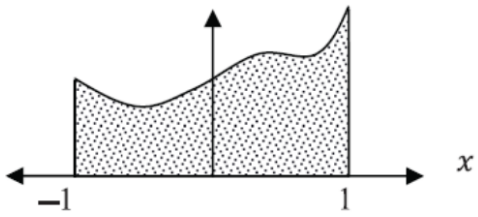
$f = a_0 + a_1 x + a_2 x^2 + a_3 x^3$



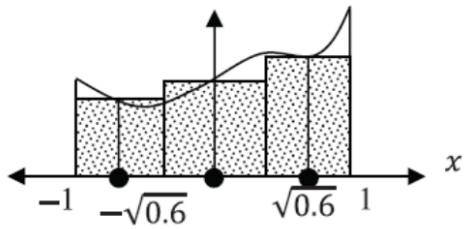
$\int_{-1}^{1} f dx = f\left(\frac{1}{\sqrt{3}}\right) + f\left(\frac{-1}{\sqrt{3}}\right)$



$f = a_0 + a_1 x + a_2 x^2 + a_3 x^3 + a_4 x^4 + a_5 x^5$



$\int_{-1}^{1} f dx = \frac{5}{9} f(\sqrt{0.6}) + \frac{8}{9} f(0) + \frac{5}{9} f(-\sqrt{0.6})$



Gaussian Quadrature

Example 1. $I = \int_0^{\pi} sin(x)dx = 2.0$

Coordinate Transformation: Map [0,π] $\rightarrow$ [-1,1].

Let,

$$x = \frac{\pi}{2}[1 + u] \rightarrow dx = \frac{\pi}{2}du. \tag{13}$$

Hence,

$$I = \int_0^{\pi} sin(x)dx = \frac{\pi}{2}\int_{-1}^{1} sin(\frac{\pi}{2}(1 + u))du. \tag{14}$$

Gaussian Quadrature

Two-Point Quadrature: $w_o = w_1 = 1.0$; $u_0 = \frac{-1}{\sqrt{3}}$; $u_1 = \frac{1}{\sqrt{3}}$.

$$I \approx \frac{\pi}{2}\left[\sin(\frac{\pi}{2}(1-\frac{1}{\sqrt{3}})) + \sin(\frac{\pi}{2}(1+\frac{1}{\sqrt{3}}))\right] = 1.93581957. \quad (15)$$

Three-Point Quadrature: $w_o = \frac{5}{9}$, $w_1 = \frac{8}{9}$, $w_2 = \frac{5}{9}$, $u_0 =$ -0.774596669, $u_1 = 0$, $u_2 = 0.774596669$.

$$I \approx \frac{\pi}{2}\left[\frac{5}{9}\sin(\frac{\pi}{2}(1+u_o)) + \frac{8}{9}\sin(\frac{\pi}{2}(1+u_1) + \frac{5}{9}\sin(\frac{\pi}{2}(1+u_2))\right]$$
$$= 2.00138891.$$

Gauss Integration

Abbreviated Python Code:

```
1
2   # Define mathematical functions ...
3
4   def f1(x):
5       return math.sin(x);
6
7   def main():
8       print("--- ");
9       print("--- Integrate math.sin(x) from [0,pi] with Two-Point Quadrature ... ");
10
11      w0 = 1.0
12      u0 = - 1.0/math.sqrt(3.0)
13      x0 = (math.pi/2.0)*(1 + u0)
14      w1 = 1.0
15      u1 =   1.0/math.sqrt(3.0)
16      x1 = (math.pi/2.0)*(1 + u1)
17
18      I = (math.pi/2.0)*( w0*f1(x0) + w1*f1(x1) )
19
20      print("--- ");
21      print("---   w0 = {:14.6e}, u0 = {:14.6e}, x0 = {:14.6e}".format( w0, u0, x0 ) )
22      print("---   w1 = {:14.6e}, u1 = {:14.6e}, x1 = {:14.6e}".format( w1, u1, x1 ) )
23      print("---   Integral sin(x) dx = {:14.8e} ...".format( I ) )
24
25      print("--- ");
26      print("--- Integrate math.sin(x) from [0,pi] with Three-Point Quadrature ... ");
27
28      w0 = 5.0/9.0
```

Gauss Integration

Abbreviated Python Code:

```
28      w0 = 5.0/9.0
29      u0 = - 0.774596669
30      x0 = (math.pi/2.0)*(1 + u0)
31
32      w1 = 8.0/9.0
33      u1 = 0.0
34      x1 = (math.pi/2.0)*(1 + u1)
35
36      w2 = 5.0/9.0
37      u2 = 0.774596669
38      x2 = (math.pi/2.0)*(1 + u2)
39
40      I = (math.pi/2.0)*( w0*f1(x0) + w1*f1(x1) + w2*f1(x2) )
41
42      print("--- ");
43      print("---   w0 = {:14.6e}, u0 = {:14.6e}, x0 = {:14.6e}".format( w0, u0, x0 ) )
44      print("---   w1 = {:14.6e}, u1 = {:14.6e}, x1 = {:14.6e}".format( w1, u1, x1 ) )
45      print("---   w2 = {:14.6e}, u2 = {:14.6e}, x2 = {:14.6e}".format( w2, u2, x2 ) )
46      print("---   Integral sin(x) dx = {:14.8e} ...".format( I ) )
```

Numerical Output:

```
--- Integrate math.sin(x) from [0,pi] with Two-Point Quadrature ...
---
---   w0 =   1.000000e+00, u0 =  -5.773503e-01, x0 =   6.638966e-01
---   w1 =   1.000000e+00, u1 =   5.773503e-01, x1 =   2.477696e+00
---   Integral sin(x) dx = 1.93581957e+00 ...

      Absolute error = 0.0641804
      Relative Error = 0.0320902

--- Integrate math.sin(x) from [0,pi] with Three-Point Quadrature ...
---
---   w0 =   5.555556e-01, u0 =  -7.745967e-01, x0 =   3.540627e-01
---   w1 =   8.888889e-01, u1 =   0.000000e+00, x1 =   1.570796e+00
---   w2 =   5.555556e-01, u2 =   7.745967e-01, x2 =   2.787530e+00
---   Integral sin(x) dx = 2.00138891e+00 ...

      Absolute error = 0.0013889
      Relative Error = 0.0006944
```

Gauss Integration

Example 2. Evaluate $I = \int_0^4 xe^{2x}dx$.

Analytic Solution.

$$I = \int_0^4 xe^{2x}dx = \left[\frac{x}{2}e^{2x} - \frac{1}{4}e^{2x}\right]_0^4 = 5,216.92. \tag{16}$$

Coordinate Transformation: Map [0,4] $\rightarrow$ [-1,1]. Let,

$$x = \frac{4}{2}[1+u] \rightarrow dx = \frac{4}{2}du. \tag{17}$$

Hence,

$$I = \int_0^4 xe^{2x}dx = 2\int_{-1}^1 2(1+u)e^{4(1+u)}du \tag{18}$$

Gauss Integration

Numerical Results:

```
--- Integrate x*exp(2x) from [0,4] with Two-Point Quadrature ...

---    w0 =    1.000000e+00, u0 =  -5.773503e-01, x0 =    8.452995e-01
---    w1 =    1.000000e+00, u1 =   5.773503e-01, x1 =    3.154701e+00
---    Integral x*exp(2x) dx = 3.47754394e+03 ...

       Absolute error = 1,739.38
       Relative Error = 0.33341

--- Integrate x*exp(2x) from [0,4] with Three-Point Quadrature ...

---    w0 =    5.555556e-01, u0 =  -7.745967e-01, x0 =    4.508067e-01
---    w1 =    8.888889e-01, u1 =   0.000000e+00, x1 =    2.000000e+00
---    w2 =    5.555556e-01, u2 =   7.745967e-01, x2 =    3.549193e+00
---    Integral x*exp(2x) dx = 4.96710668e+03 ...

       Absolute error = 249.82
       Relative Error = 0.04789
```

Gauss Integration

Numerical Results:

```
--- Integrate x*exp(2x) from [0,4] with Four-Point Quadrature ...

---    w0 =   3.478548e-01, u0 =  -8.611363e-01, x0 =   2.777274e-01
---    w1 =   6.521452e-01, u1 =  -3.399810e-01, x1 =   1.320038e+00
---    w2 =   6.521452e-01, u2 =   3.399810e-01, x2 =   2.679962e+00
---    w3 =   3.478548e-01, u3 =   8.611363e-01, x3 =   3.722273e+00
---    Integral x*exp(2x) dx = 5.19754375e+03 ...

       Absolute error = 19.38
       Relative Error = 0.00371
```

Romberg Integration

(Werner Romberg, 1909-2003)

Romberg Integration

Let $T_n(h)$ denote trapezium rule applied to an interval [a,b] with n subintervals of length h (i.e., b-a = nh).

Idea: We want to obtain a sequence of approximations by applying the trapezium rule with successively halved interval lengths (i.e, h, h/2, h/4, etc),

$$T_n(h), T_{2n}(h/2), T_{4n}(h/4), T_{8n}(h/8), \cdots \tag{19}$$

Strategy: From the previous tutorial, the error associated with trapezium rule has the form:

$$E_T = Ch^2 f''(\epsilon), \quad \text{where} \quad a \leq \epsilon \leq b. \tag{20}$$

Hence, $h \to 0$ implies $E_T \to$ error series converges.

Romberg Integration

Sketch of Derivation: We assume:

$$I(a,b) = T_n + h^2 E_2 + h^4 E_4 + h^6 E_6 + h^8 E_8 + \cdots \tag{21}$$

$$I(a,b) = T_{2n} + \left(\frac{h}{2}\right)^2 E_2 + \left(\frac{h}{4}\right)^2 E_4 + \left(\frac{h}{6}\right)^2 E_6 + \left(\frac{h}{8}\right)^2 E_8 + \cdots \tag{22}$$

First Level of Romberg Integration: $O(h^4)$ accuracy.

Combine equations: 4 (22) - (21):

$$(4-1)I(a,b) = 4T_{2n} - T_n + \left(\frac{h}{4} - 1\right)^2 h^4 E_4 + \cdots \tag{23}$$

Romberg Integration

Hence,

$$I(a,b) = R_{22} + \frac{1}{4}h^4 E_4 + \cdots = \left[\frac{4\,T_{2n} - T_n}{4-1}\right] + \frac{1}{4}h^4 E_4 + \cdots \quad (24)$$

Repeating for T_{2n} and T_{4n}:

$$I(a,b) = R_{32} + h^4 E_4' + \cdots = \left[\frac{4\,T_{4n} - T_{2n}}{4-1}\right] + h^4 E_4' + \cdots \quad (25)$$

and T_{4n} and T_{8n}:

$$I(a,b) = R_{42} + h^4 E_4' + \cdots = \left[\frac{4\,T_{8n} - T_{4n}}{4-1}\right] + h^4 E_4' + \cdots \quad (26)$$

Romberg Integration

Second Level of Romberg Integration: $O(h^6)$ accuracy.

$$R_{33} = \left[\frac{4^2 R_{32} - R_{22}}{4^2 - 1} \right]. \tag{27}$$

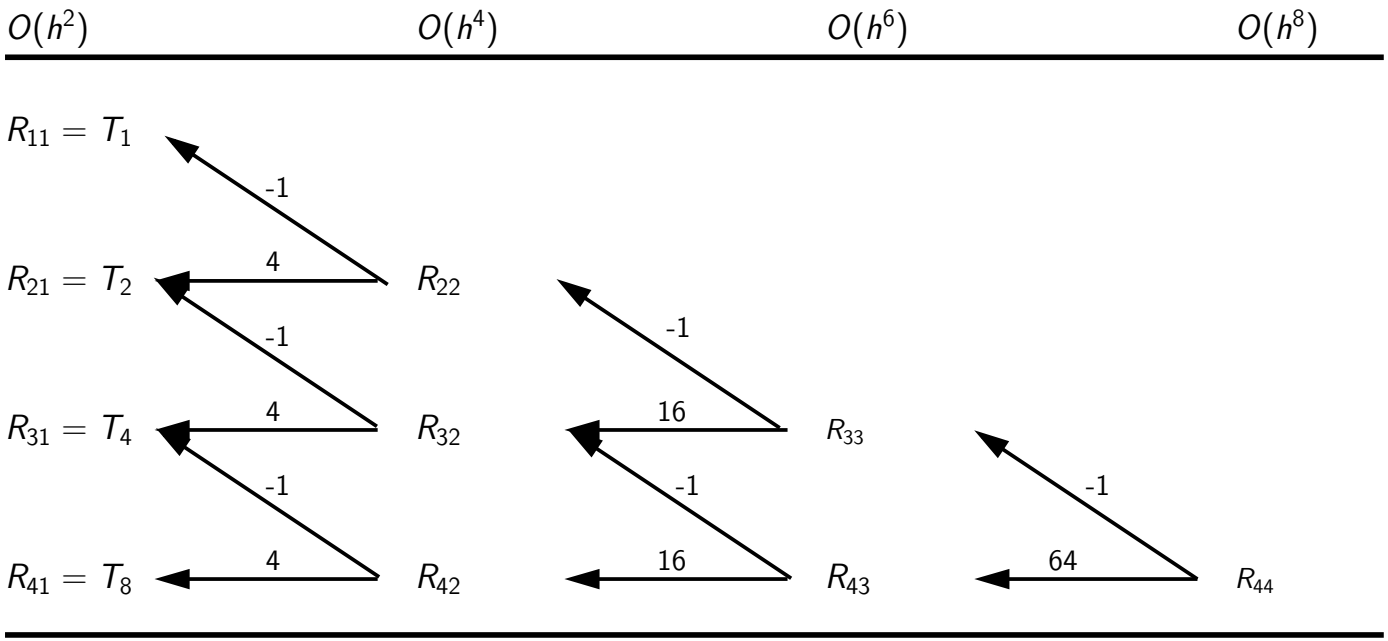
$$R_{43} = \left[\frac{4^2 R_{42} - R_{32}}{4^2 - 1} \right]. \tag{28}$$

Third Level of Romberg Integration: $O(h^8)$ accuracy.

$$R_{44} = \left[\frac{4^3 R_{43} - R_{33}}{4^3 - 1} \right]. \tag{29}$$

Romberg Integration

Computational Procedure:



Extrapolation Update:

$$R_{jk} = \left[\frac{4^{k-1} R_(j, k-1) - R_(j-1, k-1)}{4^{k-1} - 1} \right]. \qquad (30)$$

Romberg Integration

Example 1. $I = \int_0^{\pi} sin(x)dx = 2.0$

Consider the data set (5 points):

x	0.0	$\pi/4$	$\pi/2$	$3\pi/4$	π
sin(x)	0.0	$1/\sqrt{2}$	1.0	$1/\sqrt{2}$	0.0

Applying Trapezoid Rule for 1, 2, 4 and 8 intervals:

$$R_{11} = T_1 = \frac{\pi}{2}[f(0) + f(\pi)] = 0.000000 \tag{31}$$

$$R_{21} = T_2 = \frac{\pi}{4}[f(0) + 2f(\pi/2) + f(\pi)] = 1.57079633 \tag{32}$$

Continuing: $R_{31} = T_4 = 1.89611890$; $R_{41} = T_8 = 1.97423160$.

Romberg Integration

First Column of Extrapolation: $O(h^4)$

$$R_{22} = \left[\frac{4R_{21} - R_{11}}{(4-1)}\right] = 2.0944. \tag{33}$$

$$R_{32} = \left[\frac{4R_{31} - R_{21}}{(4-1)}\right] = 2.0046. \tag{34}$$

$$R_{42} = \left[\frac{4R_{41} - R_{31}}{(4-1)}\right] = 2.0003. \tag{35}$$

Second Column of Extrapolation: $O(h^6)$

$$R_{33} = \left[\frac{16R_{32} - R_{22}}{(4^2-1)}\right] = 1.9986. \tag{36}$$

Romberg Integration

Second Column of Extrapolation: Continued ...

$$R_{43} = \left[\frac{16R_{42} - R_{32}}{(4^2 - 1)} \right] = 2.0003. \tag{37}$$

Third Column of Extrapolation: $O(h^8)$

$$R_{44} = \left[\frac{64R_{43} - R_{33}}{(4^3 - 1)} \right] = 2.0000. \tag{38}$$

Homework Exercise: Show R_{22} is equivalent to one step of Simpson's Rule.

Romberg Integration

Python Source Code:

```
1   # ============================================================================
2   # TestIntegrationRomberg01.py: Use Romberg Algorithm to integrate functions.
3   #
4   # Written By: Mark Austin                                          July 2023
5   # ============================================================================
6
7   import math;
8   import Integration;
9
10  # Define mathematical functions ...
11
12  def f1(x):
13      return math.sin(x);
14
15  # main method ...
16
17  def main():
18      print("--- ");
19      print("--- Case Study 1: Integrate math.sin(x) from [0,pi],  nointervals = 4 ... ");
20      print("--- ========================================================== ... ");
21
22      # Initialize problem setup ...
23
24      a = 0.0;
25      b = math.pi
26      nointervals = 4
27
28      print("--- Inputs:")
```

Romberg Integration

Python Source Code:

```
29        print("---   a = {:9.4f} ...".format(a) )
30        print("---   b = {:9.4f} ...".format(b) )
31        print("---   no intervals = {:d} ...".format(nointervals) )
32
33        # Compute numerical solution to integral ..
34
35        print("--- Execution:")
36        xi = Integration.romberg( f1, a, b, nointervals )
37
38        # Summary of computations ...
39
40        print("--- Output:")
41        print("---   integral = {:14.8e} ...".format( xi ) )
42
43    # call the main method ...
44
45    main()
```

Romberg Integration

Abbreviated Output:

```
--- Inputs:
---   a = 0.0000, b = 3.1416 ...
---   no intervals = 4 ...
--- Execution:
---   Initialize Romberg Integration Table ...
---   Compute trapezoid rule for first column ...
---   Iterate over levels of refinement ...

Matrix: Romberg Integration Table (instantiated)
  1.92367069e-16   0.00000000e+00   0.00000000e+00   0.00000000e+00
  1.57079633e+00   2.09439510e+00   0.00000000e+00   0.00000000e+00
  1.89611890e+00   2.00455975e+00   1.99857073e+00   0.00000000e+00
  1.97423160e+00   2.00026917e+00   1.99998313e+00   2.00000555e+00

--- Output:
---   integral = 2.00000555e+00 ...

      Absolute error = 0.00000555
      Relative Error = 0.00000277
```

Romberg Integration

Example 2. Evaluate $I = \int_0^4 xe^{2x}dx$.

Analytic Solution.

$$I = \int_0^4 xe^{2x}dx = \left[\frac{x}{2}e^{2x} - \frac{1}{4}e^{2x}\right]_0^4 = 5,216.92. \tag{39}$$

One Step of Trapezoid Rule (n = 1).

$$T_1 = \left[\frac{4-0}{2}\right][f(0) + f(4)] = 23,847.66. \tag{40}$$

Two Steps of Trapezoid Rule (n = 2).

$$T_2 = \left[\frac{2-0}{2}\right][f(0) + 2f(2) + f(4)] = 12,142.22. \tag{41}$$

Romberg Integration

Systematic Refinement: $T_1, T_2, \cdots, T_8$: $O(h^2)$

No Intervals	h	Integral T_n
1	4.0	$R_{11} = T_1 = 23{,}847.66$
2	2.0	$R_{21} = T_2 = 12{,}142.22$
4	1.0	$R_{31} = T_4 = 7{,}288.79$
8	0.5	$R_{41} = T_8 = 5{,}764.76$

Python Source Code:

```
1   import math;
2   import Integration;
3
4   # Define mathematical functions ...
5
6   def f2(x):
7       return x*math.exp(2 * x)
8
9   # main method ...
10
11  def main():
12      # Initialize problem setup ...
```

Romberg Integration

Python Source Code: Continued ...

```
13
14      a = 0.0;
15      b = 4.0
16      nointervals = 4
17
18      print("--- Inputs:")
19      print("---   a = {:9.4f} ...".format(a) )
20      print("---   b = {:9.4f} ...".format(b) )
21      print("---   no intervals = {:d} ...".format(nointervals) )
22
23      # Compute numerical solution to integral ..
24
25      print("--- Execution:")
26      xi = Integration.romberg( f2, a, b, nointervals )
27
28      # Summary of computations ...
29
30      print("--- Output:")
31      print("---   integral = {:14.8e} ...".format( xi ) )
32
33  # call the main method ...
34
35  main()
```

Romberg Integration

Abbreviated Output:

```
--- Inputs:
---   a = 0.0000, b = 4.0000 ...
---   no intervals = 4 ...
--- Execution:
---   Initialize Romberg Integration Table ...
---   Compute trapezoid rule for first column ...
---   Iterate over levels of refinement ...

Matrix: Romberg Integration Table (instantiated)
  2.38476639e+04   0.00000000e+00   0.00000000e+00   0.00000000e+00
  1.21422245e+04   8.24041143e+03   0.00000000e+00   0.00000000e+00
  7.28878771e+03   5.67097543e+03   5.49967970e+03   0.00000000e+00
  5.76476205e+03   5.25675350e+03   5.22913871e+03   5.22484441e+03

--- Output:
---   integral = 5.22484441e+03 ...

      Absolute error = 7.564
      Relative Error = 0.0014
```

Python Code Listings

Code 1: Gaussian Quadrature

```
1
2   # ================================================================================
3   # Integration.gaussquadrature(): Two-point Gauss Quadrature.
4   #
5   # Args: f (function): the equation f(x).
6   #       a: the initial point ...
7   #       b: the end point ...
8   # ================================================================================
9
10  def gaussquadrature(f, a, b):
11      x = (b - a)/2
12      y = (b + a)/2
13      return x * (f(x * (-1 / math.sqrt(3)) + y) + f(x * (1 / math.sqrt(3)) + y))
```

Code 2: Romberg Integration

```
1   # ================================================================
2   # Function to print two-dimensional matrices ...
3   # ================================================================
4
5   def PrintMatrix(name, a):
6       print("Matrix: {:s} ".format(name) );
7       for row in a:
8           for col in row:
9               print("{:16.8e}".format(col), end=" ")
10          print("")
11
12  # ================================================================
13  # Integration.romberg(): Compute integral with Rhomberg Integration.
14  #
15  # Args: f (function): the equation f(x).
16  #       a (float): the initial point.
17  #       b (float): the final point.
18  #       n (int): number of intervals.
19  #
20  # Returns:
21  #       xi (float): numerical approximation of the definite integral.
22  # ================================================================
23
24  def romberg(f, a, b, n):
25
26      # Check that the function changes sign ....
27
28      print("---   Initialize Romberg Integration Table ... ")
```

Code 2: Romberg Integration

```
29
30      # Initialize the Romberg integration table
31
32      r = np.zeros((n, n))
33
34      PrintMatrix("Romberg Integration Table (empty)", r)
35
36      # Compute the trapezoid rule for the first column (h = b - a)
37
38      print("---   Compute trapezoid rule for first column ... ")
39
40      h = b - a
41      r[0, 0] = 0.5 * h * (f(a) + f(b))
42
43      # Iterate for each level of refinement
44
45      print("---   Iterate over levels of refinement ... ")
46
47      for i in range(1, n):
48          h = 0.5 * h  # Halve the step size
49
50          # Compute the composite trapezoid rule
51
52          sum_f = 0
53          for j in range(1, 2**i, 2):
54              x = a + j * h
55              sum_f += f(x)
56          r[i, 0] = 0.5 * r[i - 1, 0] + h * sum_f
```

Code 2: Romberg Integration

```
57
58              # Richardson extrapolation for higher order approximations
59
60              for k in range(1, i + 1):
61                  r[i, k] = r[i, k - 1] + \
62                      (r[i, k - 1] - r[i - 1, k - 1]) / ((4**k) - 1)
63
64          # Print details of romberg integration table ...
65
66          PrintMatrix("Romberg Integration Table (instantiated)", r)
67
68          # Return integral ...
69
70          return float(r[n - 1, n - 1])
```