

5.9 Project

Trapezoidal and Simpson Approximations

In the Section 5.4 project we illustrated the calculator/computer calculation of Riemann sum approximations to the integral $\int_a^b f(x) dx$ of a given function $f(x)$ from $x = a$ to $x = b$. Here we illustrate the combination of Riemann sums to form trapezoidal and Simpson sums.

Any Riemann sum approximating the given integral corresponds to subdivision of the interval $[a, b]$ into a selected number n of equal-length subintervals with successive endpoints $a = x_0 = x_1 = x_2 = \dots = x_n = b$. Each such subinterval $[x_{i-1}, x_i]$ has length $h = \Delta x = (b-a)/n$. We start by calculating the **left-endpoint sum**

$$L_n = \sum_{i=1}^n f(x_{i-1})h \quad (1)$$

the **right-endpoint sum**

$$R_n = \sum_{i=1}^n f(x_i)h \quad (2)$$

and the **midpoint sum**

$$M_n = \sum_{i=1}^n f(m_i)h, \quad (3)$$

where $m_i = (x_{i-1} + x_i)/2$ is the midpoint of the i th subinterval $[x_{i-1}, x_i]$. We can then proceed immediately to calculate first the **trapezoidal sum**

$$T_n = \frac{L_n + R_n}{2} \quad (4)$$

and then the **Simpson sum**

$$S_{2n} = \frac{T_n + 2M_n}{3} = \frac{L_n + 4M_n + R_n}{6}. \quad (5)$$

Note that the left-endpoint, right-endpoint, and midpoint sums with n subintervals yield the Simpson sum corresponding to a subdivision of $[a, b]$ into *twice* as many subintervals. We illustrate below the calculation of the sums in (1)–(5) using the particular integral

$$\int_1^4 3\sqrt{x} dx = \left[2x^{3/2} \right]_1^4 = 14 \quad (6)$$

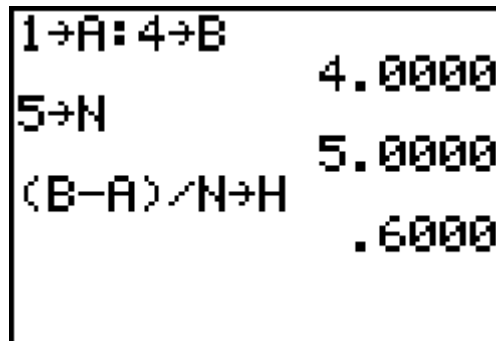
with known exact value.

Using a Graphing Calculator

In order to calculate sums with a TI graphing calculator, we need only know the command

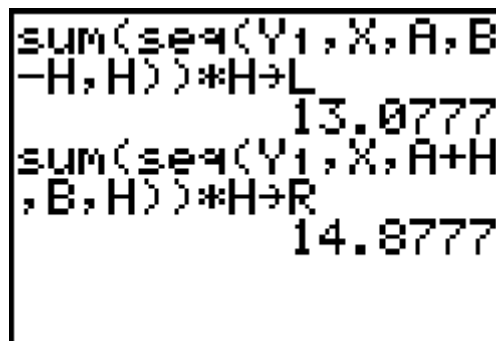
sum(seq(expression, variable, begin, end, increment))

for calculating the sum of the "sequence" (or list) of numbers obtained by substituting the successive values begin , $\text{begin}+\text{increment}$, $\text{begin}+2*\text{increment}$,, end for the "variable" in the "expression". On the TI-83 calculator the **sum** command is found in the **2nd LIST MATH** menu and the **seq** command in the **2nd LIST OPS** menu. On the TI-86 calculator both are found in the **2nd MATH MISC** menu.



```
1→A:4→B      4.0000
5→N           5.0000
(B-A)/N→H    .6000
```

This TI-83 screen shows a calculator prepared to calculate Riemann sums for the **Y=** menu function $Y1 = 3*X^{(1/2)}$ on the interval $[1, 4]$ with $n = 5$ subintervals each of length $h = (b - a)/n$. The following screen now shows the calculation of the left-endpoint and right-endpoints sums **L** and **R**. Note how we get the left-hand sum by starting the sum at the left endpoint **A** of the first subinterval, and the right-hand sum by starting at the right endpoint **A+H**.



```
sum(seq(Y1,X,A,B
-H,H))*H→L    13.0777
sum(seq(Y1,X,A+H
,B,H))*H→R    14.8777
```

We calculate the midpoint sum M similarly, and then the averages that define the trapezoidal sum T with $n = 5$ subintervals and the Simpson sum S with $2n = 10$ subintervals:

```
sum(seq(Y1,X,A+H
/2,B-H/2,H))*H→M
14.0111
(L+R)/2→T
13.9777
(T+2*M)/3→S
14.0000
```

To whether we've actually got the exact value of the integral, we set our calculator to **MODE Float 8** and re-execute the final command:

```
/2,B-H/2,H))*H→M
14.0111
(L+R)/2→T
13.9777
(T+2*M)/3→S
14.0000
13.99995489
```

Thus the best we can say at this point is that Simpson's approximation with 10 subintervals is 13.999955 accurate to 6 decimal places. But knowing that our commands are "working", we can quickly double the number of subintervals and repeat the calculations. (Indeed, no further typing is necessary, as each command above can be retrieved with the **2nd Enter** key.)

```
10→N
10.00000000
(B-A)/N→H
.30000000
sum(seq(Y1,X,A,B
-H,H))*H→L
13.54438704
```

Our final screen looks like this:

```
sum(seq(Y1,X,A+H
/2,B-H/2,H)*H→M
14.00280199
(L+R)/2→T
13.99438704
(T+2*M)/3→S
13.99999700
```

Thus the Simpson sum with 20 subintervals is 13.999997 accurate to 6 decimal places. Finally, we can ask our calculator to use its own more sophisticated internal numerical integration method:

```
fnInt(Y1,X,A,B)
14.00000000
```

The **fnInt** command is found in the **MATH** menu on the TI-83 and in the **2nd CALC** menu on the TI-86.

In order to approximate a different integral, you need only define the desired integrand function **Y1=f(x)** and re-enter the sequence of commands shown above, starting with the desired limits *a* and *b* and number *n* of subintervals.

Using Maple

For an illustrative example, we consider numerical approximation of the integral

```
Int(3*sqrt(x), x=1..4 ) = int(3*sqrt(x), x=1..4 );
```

$$\int_1^4 3\sqrt{x} dx = 14$$

with known exact value 14, given integrand function

```
f := x -> 3*sqrt(x):
```

and limits

```
a := 1:      b := 4:
```

and with 15-digit precision

```
Digits := 15:
```

If we start with

```
n := 5:
```

subintervals each having length

```
h := (b - a)/n:
```

then the i th subdivision point is defined by

```
x := i -> a + i*h:
```

and the endpoints of the i th subinterval are $\mathbf{x(i-1)}$ and $\mathbf{x(i)}$.

With this setup, the following computations will result in the Simpson sum S corresponding to a subdivision of the interval $[a, b]$ into the doubled number $2n$ of subintervals. We calculate first the left-endpoint sum L and the right-endpoint sum R .

```
L := evalf(sum( f(x(i))*h, i=0..n-1 ));
```

```
L := 13.0776834708238
```

```
R := evalf(sum( f(x(i))*h, i=1..n ));
```

```
R := 14.8776834708238
```

The midpoint of the i th subinterval is given by

```
m := i -> (x(i-1)+x(i))/2:
```

so the midpoint sum with n subintervals is

```
M := evalf(sum( f(m(i))*h, i=1..n ));
```

```
M := 14.0110906009057
```

Next we average the left-endpoint and right-endpoint sums to calculate the trapezoidal sum

```
T := (L + R)/2;
```

```
T := 13.9776834708238
```

with n subintervals. Finally we calculate the weighted average of the midpoint and trapezoidal sums that yields the Simpson sum with $2n$ subintervals.

```
S := (T + 2*M)/3;
```

```
S := 13.9999548908784
```

Rounded off to 6 decimal places, Simpson's approximation with 10 subintervals is 13.999955.

The command

```
evalf( Int(f(x), x=1..4), 25);
```

```
14.000000000000000000000000
```

asks Maple to do its best, carrying 25 digits internally. It uses a numerical technique that is somewhat more sophisticated and accurate than Simpson's method.

Using Mathematica

For an illustrative example, we consider numerical approximation of the integral

```
Integrate[3*Sqrt[x], {x, 1, 4}]
```

```
14
```

with known exact value 14, given integrand function

```
f[x_] := 3 Sqrt[x]
```

and limits

```
a = 1;      b = 4
```

If we start with

```
n = 5;
```

subintervals each having length

```
h = (b - a)/n;
```

then the i th subdivision point is defined by

$$\mathbf{x}[i_] := \mathbf{a} + i*\mathbf{h}$$

and the endpoints of the i th subinterval are $\mathbf{x}[i-1]$ and $\mathbf{x}[i]$.

With this setup, the following computations will result in the Simpson sum S corresponding to a subdivision of the interval $[a, b]$ into the doubled number $2n$ of subintervals. We calculate first the left-endpoint sum L and the right-endpoint sum R with 20-digit accuracy.

$$\text{Sum}[\mathbf{f}[\mathbf{x}[i-1]]*\mathbf{h}, \{i,1,n\}];$$
$$\mathbf{L} = \mathbf{N}[\%, 20]$$

13.077683470823783185

$$\text{Sum}[\mathbf{f}[\mathbf{x}[i]]*\mathbf{h}, \{i,1,n\}];$$
$$\mathbf{R} = \mathbf{N}[\%, 20]$$

14.877683470823783185

The midpoint of the i th subinterval is given by

$$\mathbf{m}[i_] := (\mathbf{x}[i-1] + \mathbf{x}[i])/2$$

so the midpoint sum with n subintervals is

$$\text{Sum}[\mathbf{f}[\mathbf{m}[i]]*\mathbf{h}, \{i,1,n\}];$$
$$\mathbf{M} = \mathbf{N}[\%, 20]$$

14.011090600905693381

Next we average the left-endpoint and right-endpoint sums to calculate the trapezoidal sum

$$\mathbf{T} = (\mathbf{L} + \mathbf{R})/2$$

13.977683470823783185

with n subintervals. Finally we calculate the weighted average of the midpoint and trapezoidal sums that yields the Simpson sum with $2n$ subintervals.

$$\mathbf{S} = (\mathbf{T} + 2 \mathbf{M})/3$$

13.999954890878389982

Rounded off to 6 decimal places, Simpson's approximation with 10 subintervals is 13.999955.

The command

```
NIntegrate[f[x], {x, a, b}, WorkingPrecision -> 20]  
14.000000000
```

asks Mathematica to do its best, carrying 20 digits internally. It uses a numerical technique that is somewhat more sophisticated and accurate than Simpson's method.

Using MATLAB

First we define the integrand function

```
f = inline('3*sqrt(x)');
```

of our integral $\int_1^4 3\sqrt{x} dx$ with known exact value 14 and limits given by

```
a = 1;    b = 4;
```

If we now start with

```
n = 5;
```

subintervals each having length

```
h = (b - a)/n;
```

then the successive endpoints of these n subintervals are the $n+1$ elements of the vector

```
x = a : h : b  
x =  
    1.0000    1.6000    2.2000    2.8000    3.4000    4.0000
```

The left endpoints of these subintervals are

```
xL = x(1:n)  
xL =  
    1.0000    1.6000    2.2000    2.8000    3.4000
```

The right endpoints are

```
xR = x(2:n+1)  
xR =  
    1.6000    2.2000    2.8000    3.4000    4.0000
```

And the midpoints of these n subintervals are


```

xM = (xL + xR)/2
xM =
    1.3000    1.9000    2.5000    3.1000    3.7000

```

Using the MATLAB function `sum(v)` that calculates the sum of the elements of the vector `v`, we can now calculate immediately the left-endpoint, right-endpoint, and midpoint sums for the function $f(x)$ on the interval $[a, b]$:

```

L = sum(f(xL)*h)
L =
    13.0777

```

```

R = sum(f(xR)*h)
R =
    14.8777

```

```

M = sum(f(xM)*h)
M =
    14.0111

```

We can now proceed to the trapezoidal sum

```

T = (L + R)/2
T =
    13.9777

```

with n subintervals and the Simpson sum

```

S = (T + 2*M)/3
S =
    14.0000

```

To check whether we've really got the exact result 14, we can ask for full double precision:

```

format long
S
S =
    13.99995489087839

```

Rounded off to 6 decimal places, Simpson's approximation with 10 subintervals is 13.999955.

The MATLAB command `quad(f,a,b,tol)` employs a more sophisticated numerical method to approximate the desired integral of f from $x=a$ to $x=b$ with a "relative error" of `tol`. For instance, we get

```

quad(f,1,4,1e-4)
ans =
    13.99999013131382

```

```
quad(f,1,4,1e-8)
ans =
    13.99999999990862
```

This gives the value 14.000000000 accurate to 9 decimal places.