

Integration: Riemann Sums

Michael Penna, Indiana University – Purdue University, Indianapolis

Objective

To illustrate how to approximate an area using Riemann sums.

Narrative

Finding the area under a curve using Riemann sums, while being conceptually straightforward, can be computationally challenging. In this project you will see how Maple can be used to simplify computations.

In this project we introduce the commands:

<code>sum(E(i), i=1..n)</code>	The sum of the expression $E(i)$ from $i = 1$ to $i = n$.
<code>leftbox(f(x), x=a..b, n)</code>	Draw a left-hand Riemann sum.
<code>rightbox(f(x), x=a..b, n)</code>	Draw a right-hand Riemann sum.
<code>middlebox(f(x), x=a..b, n)</code>	Draw a midpoint Riemann sum.
<code>int(f(x), x=a..b)</code>	The definite integral $\int_{x=a}^b f(x) dx$.

(The commands `leftbox`, `rightbox`, and `middlebox` are in Maple's `Student` package.)

Tasks

1. Type the command lines in the left-hand column below into Maple in the order in which they are listed. These commands are aimed at finding the area under the graph of $f(x) = x^2 + x$ from $x = 1$ to $x = 3$. The effect of each command is described in the right-hand column for your reference. Your lab report will be a hard copy of your typed input and Maple's responses (both text and graphics).

<code>> # Your name, today's date</code>	
<code>> # Integration: Riemann Sums</code>	
<code>> restart;</code>	Clear Maple's memory.
<code>> f := x -> x^2+x;</code>	Let $f(x) = x^2 + x$.
<code>> a := 1; b := 3;</code>	Let $a = 1$ and $b = 3$.
<code>> dx := (b-a)/n;</code>	Let $dx = (b - a)/n$.
<code>> LHSum := sum(f(a+i*dx)*dx, i=0..n-1);</code>	Let LHSum denote the left-hand Riemann sum.
<code>> RHSum := sum(f(a+i*dx)*dx, i=1..n);</code>	Let RHSum denote the right-hand Riemann sum.
<code>> MPSum := sum(f(a+(i-0.5)*dx)*dx, i=1..n);</code>	Let MPSum denote the midpoint Riemann sum.

Observe that since n has not yet been specified, `RHSum`, `LHSum`, and `MPSum` are functions of n .

Continue by typing the following command lines into Maple.

<code>> n := 4;</code>	Let $n = 4$.
<code>> with(Student):</code>	Load the package <code>Student</code> .
<code>> leftbox(f(x), x=a..b, n);</code>	Draw the left-hand Riemann sum.
<code>> rightbox(f(x), x=a..b, n);</code>	Draw the right-hand Riemann sum.
<code>> middlebox(f(x), x=a..b, n);</code>	Draw the midpoint Riemann sum.
<code>> evalf(LHSum);</code>	Evaluate LHSum.
<code>> evalf(RHSum);</code>	Evaluate RHSum.
<code>> evalf(MPSum);</code>	Evaluate MPSum.
<code>> n := 'n';</code>	Redefine n as a variable.
<code>> evalf(limit(LHSum, n=infinity));</code>	Compute the limit of LHSum as n goes to ∞ .
<code>> evalf(limit(RHSum, n=infinity));</code>	Compute the limit of RHSum as n goes to ∞ .
<code>> evalf(limit(MPSum, n=infinity));</code>	Compute the limit of MPSum as n goes to ∞ .
<code>> Int(f(x), x=a..b) = int(f(x), x=a..b);</code>	Find $\int_{x=a}^b f(x) dx$.
<code>> evalf(rhs(%));</code>	Write $\int_{x=a}^b f(x) dx$ in decimal form.

At this time make a hard-copy of your typed input and Maple's responses. Then:

2. On the graphs you produced in Task 1, plot and label the sample points $(x_i^*, f(x_i^*))$, used to compute LHSum, RHSum, and MPSum for $n = 4$.

Your lab report will be a hard-copy of your typed input and Maple's responses (both text and hand-labeled graphics).

Comments

1. Over the interval $[1, 3]$, f is increasing; thus the right-hand Riemann sum is associated with *circumscribed* rectangles, and the left-hand Riemann sum is associated with *inscribed* rectangles. Over the interval $[-3, -1]$, on the other hand, f is decreasing; thus here the right-hand Riemann sum is associated with *inscribed* rectangles, and the left-hand Riemann sum is associated with *circumscribed* rectangles.
2. Observe that the values of LHSum, RHSum and MPSum are not necessarily the same for any finite n , but they get closer and closer to each other as n gets larger and larger, and their limits (as n goes to ∞) are the same.
3. In addition to the command `sum(E(i), i=1..n)` there is a command `Sum(E(i), i=1..n)`. The difference between these commands is that the former automatically expands and simplifies the sum, while the later does not: after using the later, you must expand and simplify it yourself if you want it done. (The value of the later command is that the former does not always work, so if you want to do your own simplification, you can.) Similarly, in addition to the command `int(f(x), x=a..b)` there is a command `Int(f(x), x=a..b)`.