

## Approximating Definite Integrals

```
> restart;
```

The `restart` command is used to reset the entire Maple system and clears all variables. Next, using the `with` statement, we load two packages of extra commands: the `student` package giving the extra calculus commands shown below, and the `plots` package giving extra graphing commands.

```
> with(student);
```

```
[D, Diff, Doubleint, Int, Limit, Lineint, Product, Sum, Tripleint, changevar, completesquare, distance, equate, integrand, intercept, intparts, leftbox, leftsum, makeproc, middlebox, middlesum, midpoint, powsubs, rightbox, rightsum, showtangent, simpson, slope, summand, trapezoid]
```

```
> with(plots);
```

Warning, the name `changecoords` has been redefined

```
[animate, animate3d, animatecurve, changecoords, complexplot, complexplot3d, conformal, contourplot, contourplot3d, coordplot, coordplot3d, cylinderplot, densityplot, display, display3d, fieldplot, fieldplot3d, gradplot, gradplot3d, implicitplot, implicitplot3d, inequal, listcontplot, listcontplot3d, listdensityplot, listplot, listplot3d, loglogplot, logplot, matrixplot, odeplot, pareto, pointplot, pointplot3d, polarplot, polygonplot, polygonplot3d, polyhedra_supported, polyhedraplot, replot, rootlocus, semilogplot, setoptions, setoptions3d, spacecurve, sparsematrixplot, sphereplot, surfdata, textplot, textplot3d, tubeplot]
```

### **Our functions and their integrals.**

We define two functions as Maple `expressions`, both of which we intend to integrate from 1 to 3. We choose the first,  $f$ , to be increasing and concave up on the interval, and the second,  $g$ , to be decreasing and concave down.

```
> f:=sqrt(4+x^4);
```

$$f := \sqrt{4 + x^4}$$

```
> g:=-sqrt(4+x^4)+12;
```

$$g := -\sqrt{4 + x^4} + 12$$

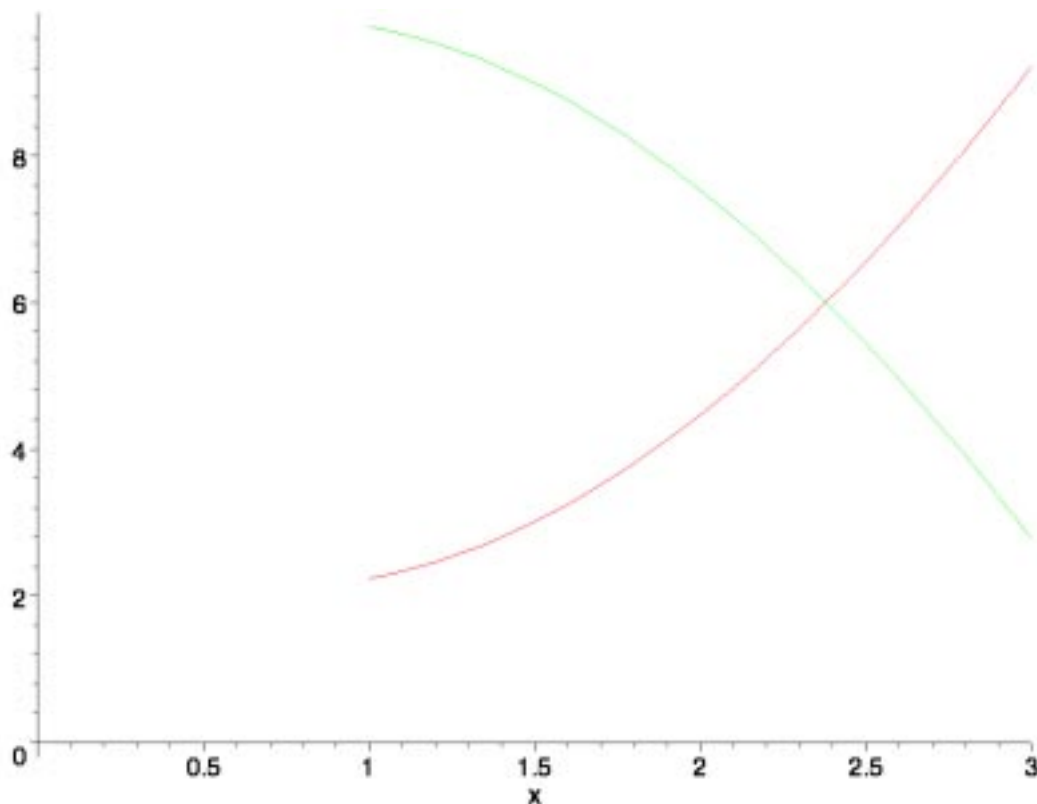
We plot the functions over the interval from 0 to 3, with  $f$  red and  $g$  green.

```
> p0:=plot(0,x=0..3):
```

```
> p1:=plot(f,x=1..3,color=red):
```

```
> p2:=plot(g,x=1..3,color=green):
```

```
> display(p0,p1,p2);
```



Next we integrate the functions from 1 to 3, with  $f$  first.

```
> int(f,x=1..3);
```

$$\sqrt{85} - \frac{1}{3}\sqrt{5} + \frac{4}{3}\sqrt{2} \operatorname{EllipticK}\left(\frac{1}{2}\sqrt{2}\right) - \frac{2}{3}\sqrt{2} \operatorname{EllipticF}\left(\frac{2}{3}\sqrt{2}, \frac{1}{2}\sqrt{2}\right) - \frac{2}{3}\sqrt{2} \operatorname{EllipticF}\left(\frac{6}{11}\sqrt{2}, \frac{1}{2}\sqrt{2}\right)$$

Looks kind of crazy. We approximate this integral to 9 decimal places using the `evalf` command.

```
> actualf:=evalf(%);
```

```
actualf := 9.782635677
```

We next integrate  $g$ .

```
> int(g,x=1..3);
```

$$24 - \sqrt{85} + \frac{1}{3}\sqrt{5} - \frac{4}{3}\sqrt{2} \operatorname{EllipticK}\left(\frac{1}{2}\sqrt{2}\right) + \frac{2}{3}\sqrt{2} \operatorname{EllipticF}\left(\frac{2}{3}\sqrt{2}, \frac{1}{2}\sqrt{2}\right) + \frac{2}{3}\sqrt{2} \operatorname{EllipticF}\left(\frac{6}{11}\sqrt{2}, \frac{1}{2}\sqrt{2}\right)$$

```
> actualg:=evalf(%);
```

```
actualg := 14.21736432
```

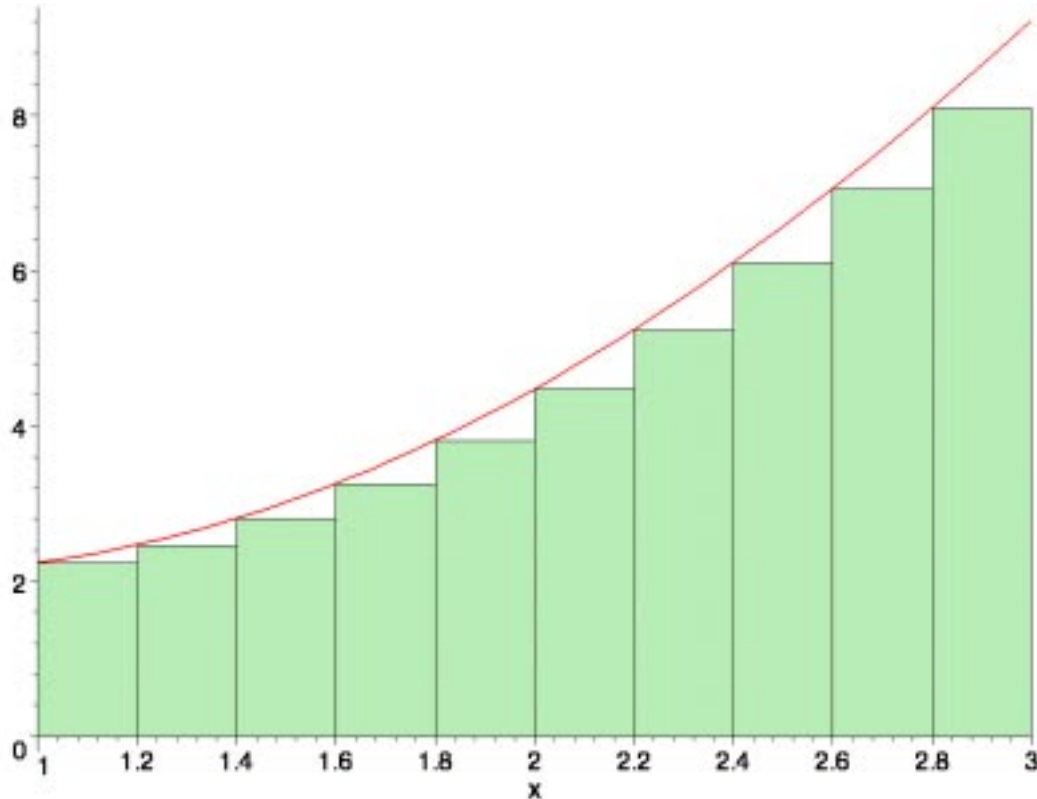
### Approximating using the left and right rules.

The reason we need to learn how to approximate integrals is because some functions, such as  $e^{(x^2)}$ , do not have an elementary antiderivative, thus denying us the use of the first fundamental theorem of calculus. In other cases, there may be an elementary antiderivative, but it may be very difficult to

obtain. Now suppose we are approximating  $\int_a^b f(x) dx$ . In all of the methods discussed here, we

partition the interval  $[a, b]$  into  $n$  equal subintervals and let  $\Delta x = \frac{b-a}{n}$ . The partition points are then labeled  $a = x_1, x_2, \dots, x_n = b$ . The **left rule** uses the left hand endpoint of each interval to determine the height of each of the  $n$  rectangles. With  $n = 10$ , lets see what this looks like graphically by using [leftbox](#).

```
> leftbox(f,x=1..3,10);
```



Our approximation is the area of the 10 rectangles. The general formula is  $\sum_{i=0}^{n-1} f(x_i) \Delta x$ . In Maple, we implement this sum with [leftsum](#). We also approximate this answer to 9 decimal places.

```
> leftsum(f,x=1..3,10);
```

$$\frac{1}{5} \left( \sum_{i=0}^9 \sqrt{4 + \left(1 + \frac{1}{5}i\right)^4} \right)$$

```
> left10:=evalf(%);
```

```
left10 := 9.100836470
```

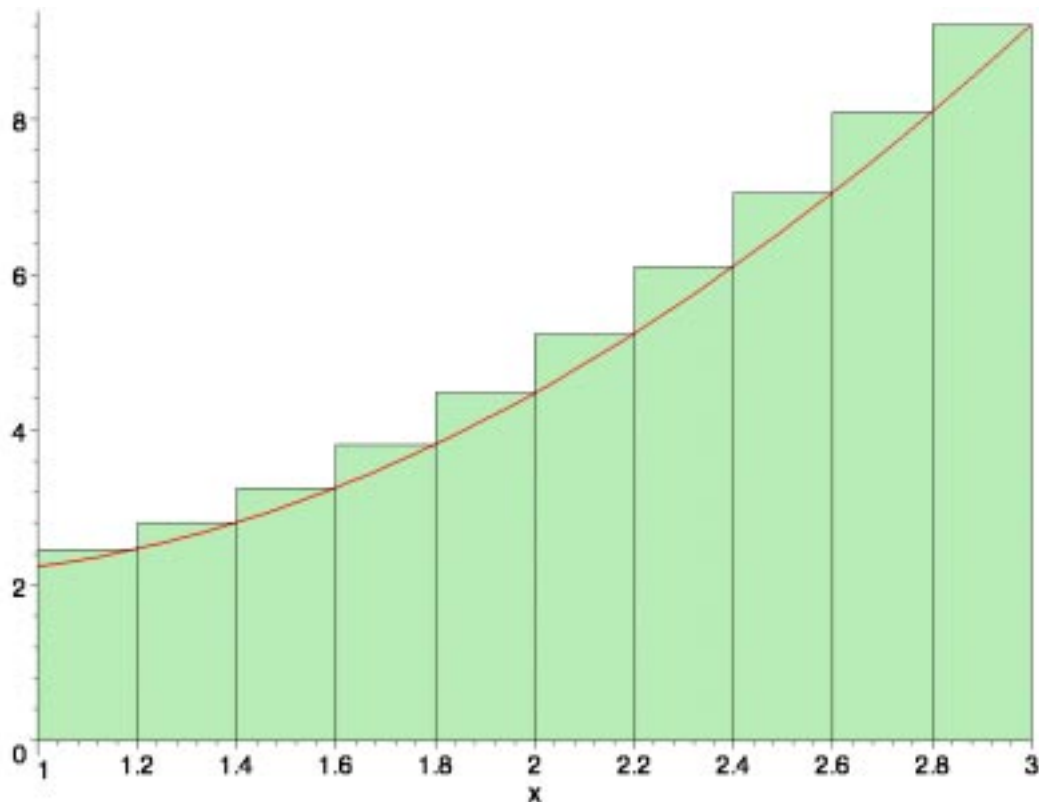
In general, we define the error by **error = actual value - approximation**. With an increasing function, it is clear that the left rule will give an under-approximation.

```
> error_l10:=actualf-left10;
```

```
error_l10 := .681799207
```

Next we consider the **right rule**, where we use the right hand endpoint of each interval to determine the height of each of the  $n$  rectangles. We again use  $n = 10$ , and use [rightbox](#) to view the situation graphically.

```
> rightbox(f,x=1..3,10);
```



The general formula is  $\sum_{i=1}^n f(x_i) \Delta x$ . In Maple, we implement this sum with [rightsum](#). We again approximate this answer to 9 decimal places and find the error, which for the right rule will always be an over-approximation for an increasing function..

```
> rightsum(f,x=1..3,10);
```

$$\frac{1}{5} \left( \sum_{i=1}^{10} \sqrt{4 + \left(1 + \frac{1}{5}i\right)^4} \right)$$

```
> right10:=evalf(%);
```

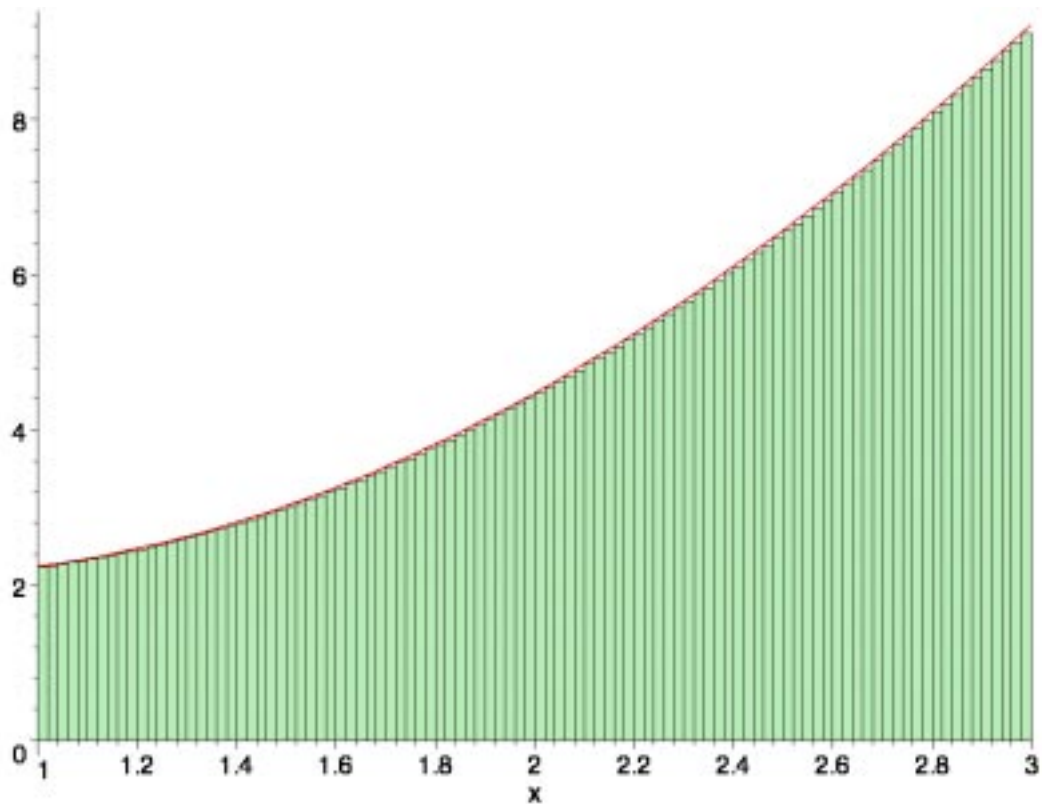
```
right10 := 10.49753177
```

```
> error_r10:=actualf-right10;
```

```
error_r10 := -.714896093
```

Now let's redo the above with  $n = 100$  rectangles. First, the **left sum**.

```
> leftbox(f,x=1..3,100);
```



```
> leftsum(f,x=1..3,100);
```

$$\frac{1}{50} \left( \sum_{i=0}^{99} \sqrt{4 + \left(1 + \frac{1}{50}i\right)^4} \right)$$

```
> left100:=evalf(%);
```

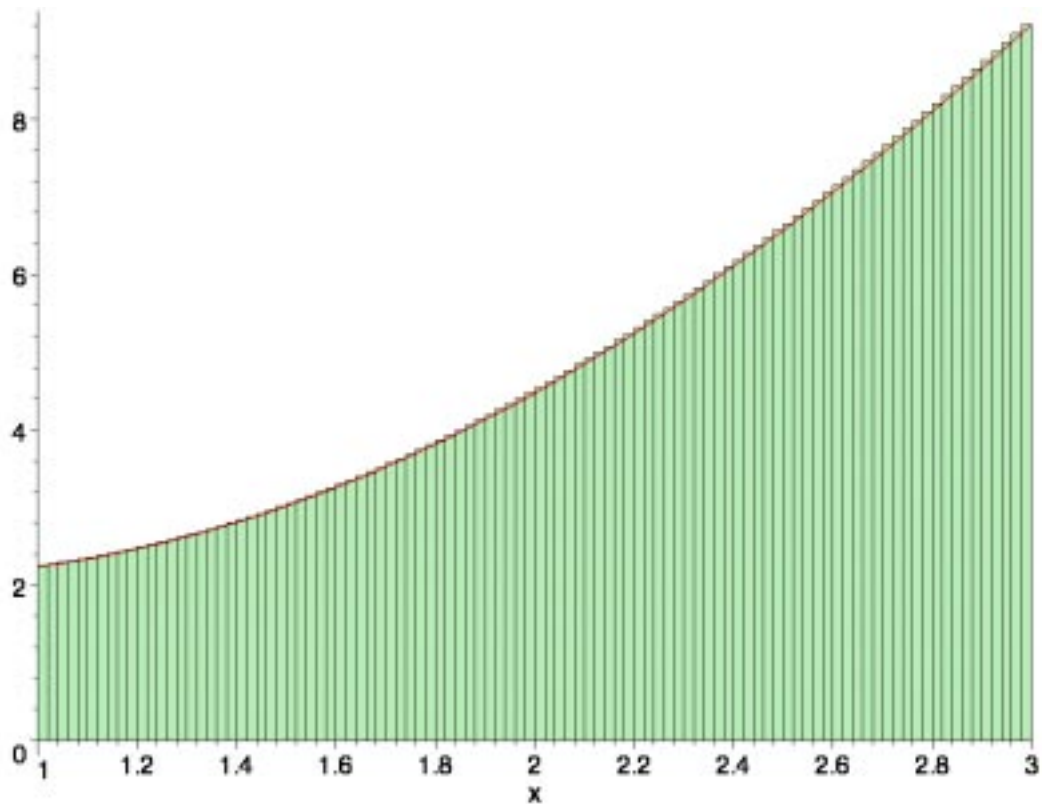
*left100 := 9.712966334*

```
> error_l100:=actualf-left100;
```

*error\_l100 := .069669343*

We see we get much less error since the area of the rectangles is much closer to the area under the curve. Next we see the same thing with the **right sum**.

```
> rightbox(f,x=1..3,100);
```



```
> rightsum(f,x=1..3,100);
```

$$\frac{1}{50} \left( \sum_{i=1}^{100} \sqrt{4 + \left(1 + \frac{1}{50}i\right)^4} \right)$$

```
> right100:=evalf(%);
```

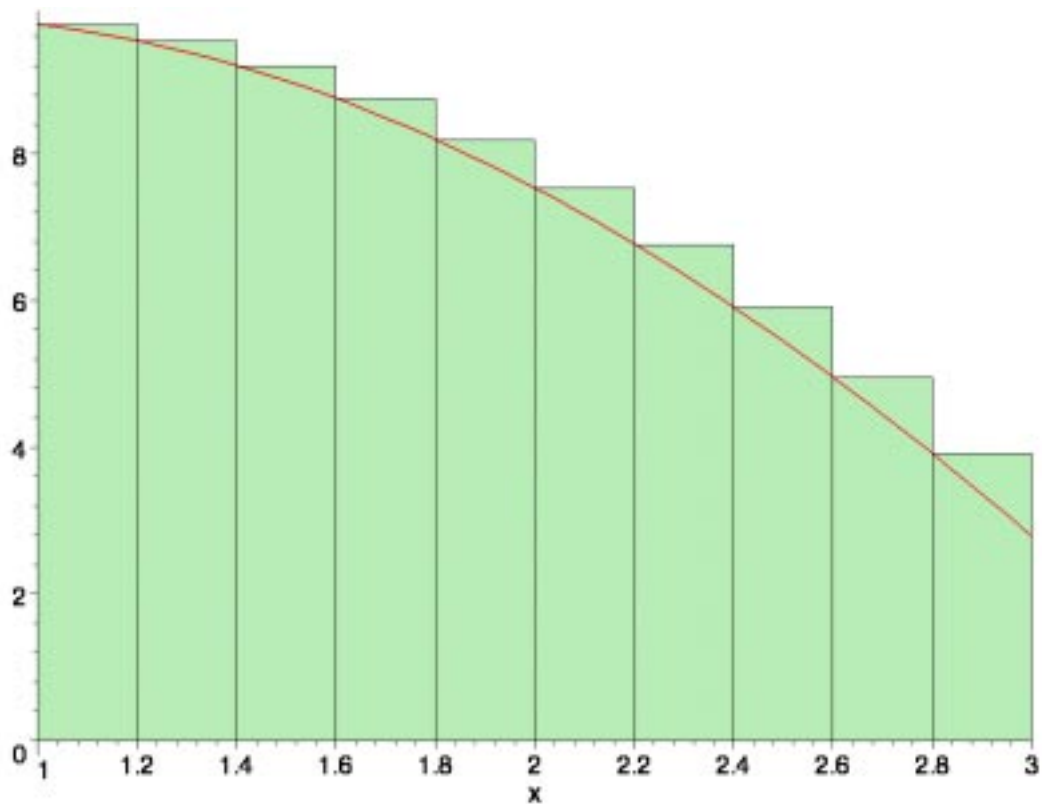
```
right100 := 9.852635864
```

```
> error_r100:=actualf-right100;
```

```
error_r100 := -.070000187
```

We see that increasing the number of rectangles gives us a better approximation. Now, let's look at  $g$  with the **left rule** and  $n = 10$ .

```
> leftbox(g,x=1..3,10);
```



It is clear that, with a decreasing function, the left rule will give us an over-approximation.

```
> leftsum(g,x=1..3,10);
```

$$\frac{1}{5} \left( \sum_{i=0}^9 \left( -\sqrt{4 + \left(1 + \frac{1}{5}i\right)^4} + 12 \right) \right)$$

```
> left10:=evalf(%);
```

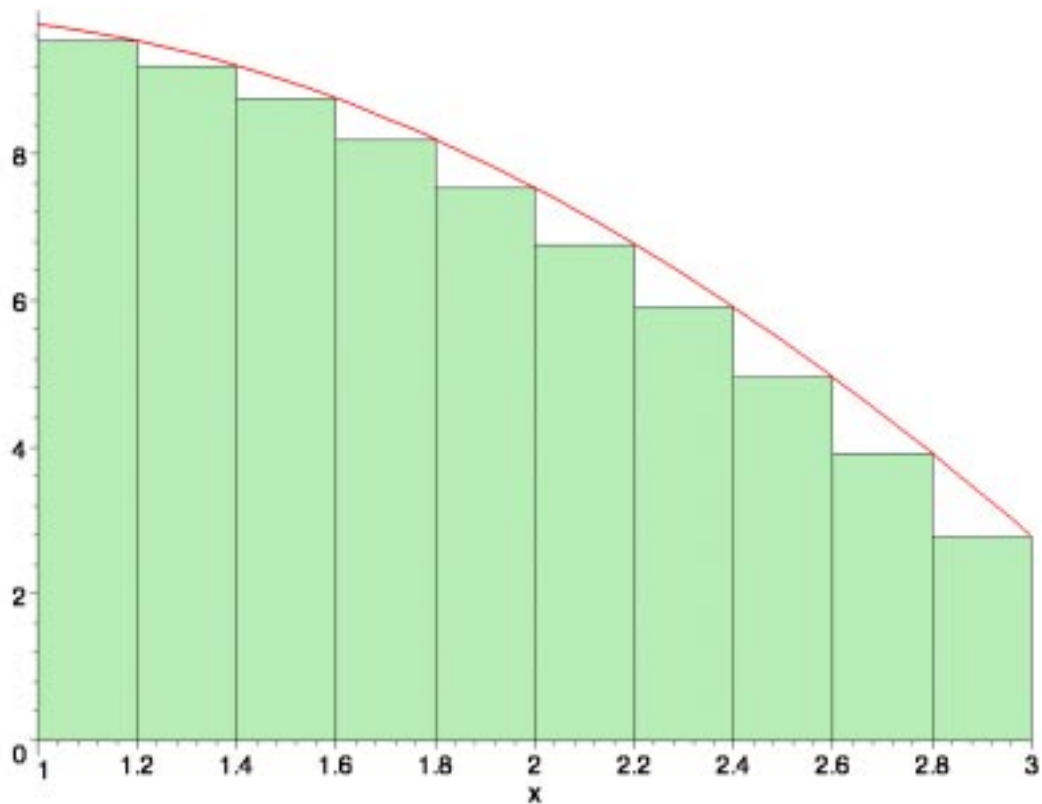
```
left10 := 14.89916353
```

```
> errorg_l10:=actualg-left10;
```

```
errorg_l10 := -.68179921
```

Similarly, the **right rule** will give us an under-approximation.

```
> rightbox(g,x=1..3,10);
```



```
> rightsum(g,x=1..3,10);
```

$$\frac{1}{5} \left( \sum_{i=1}^{10} \left( -\sqrt{4 + \left(1 + \frac{1}{5}i\right)^4} + 12 \right) \right)$$

```
> right10:=evalf(%);
```

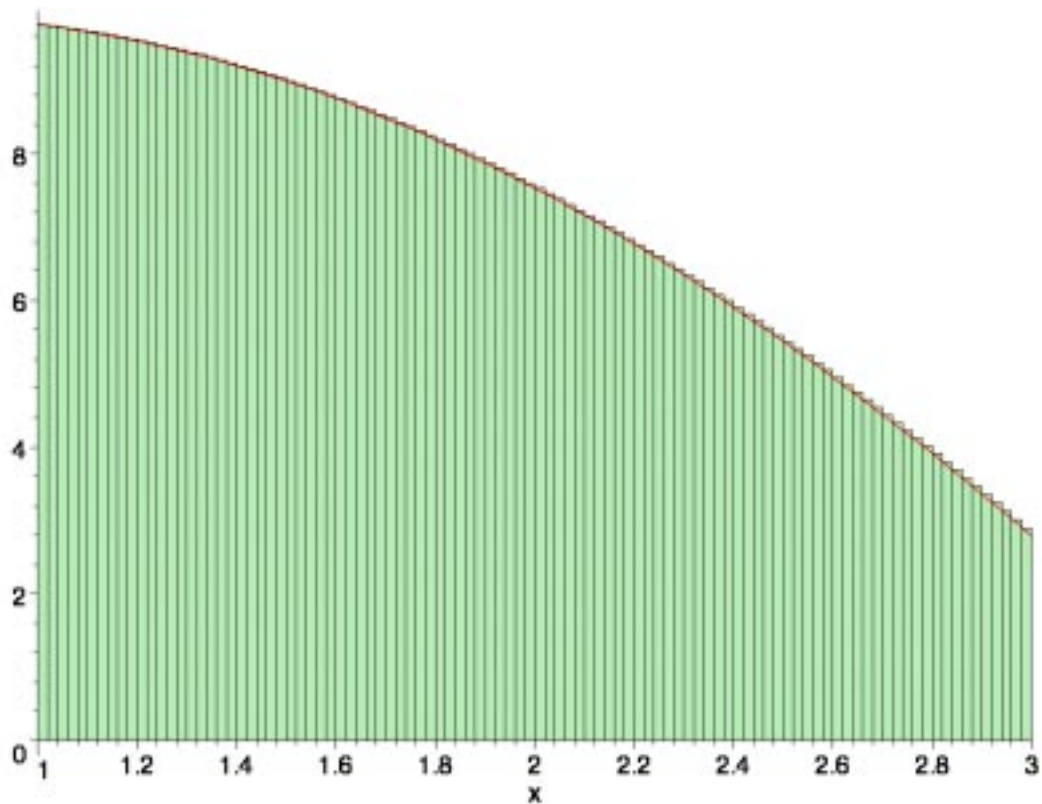
```
right10 := 13.50246823
```

```
> error_r10:=actualf-right10;
```

```
error_r10 := -3.719832553
```

Again, we increase the number of rectangles to  $n = 100$ , using the **left rule** first.

```
> leftbox(g,x=1..3,100);
```



```
> leftsum(g,x=1..3,100);
```

$$\frac{1}{50} \left( \sum_{i=0}^{99} \left( -\sqrt{4 + \left(1 + \frac{1}{50}i\right)^4} + 12 \right) \right)$$

```
> left100:=evalf(%);
```

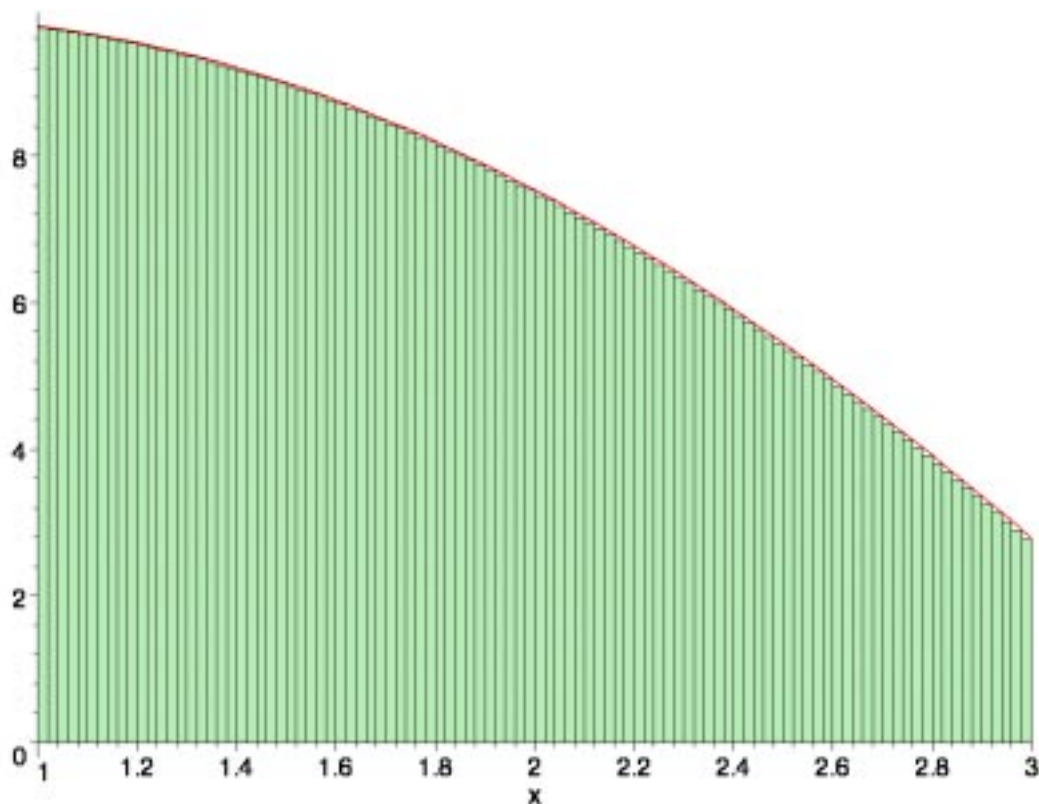
*left100 := 14.28703367*

```
> error_1100:=actualg-left100;
```

*error\_1100 := -.06966935*

Then the **right rule**.

```
> rightbox(g,x=1..3,100);
```



```
> rightsum(g,x=1..3,100);
```

$$\frac{1}{50} \left( \sum_{i=1}^{100} \left( -\sqrt{4 + \left(1 + \frac{1}{50}i\right)^4} + 12 \right) \right)$$

```
> right100:=evalf(%);
```

```
right100 := 14.14736414
```

```
> error_r100:=actualg-right100;
```

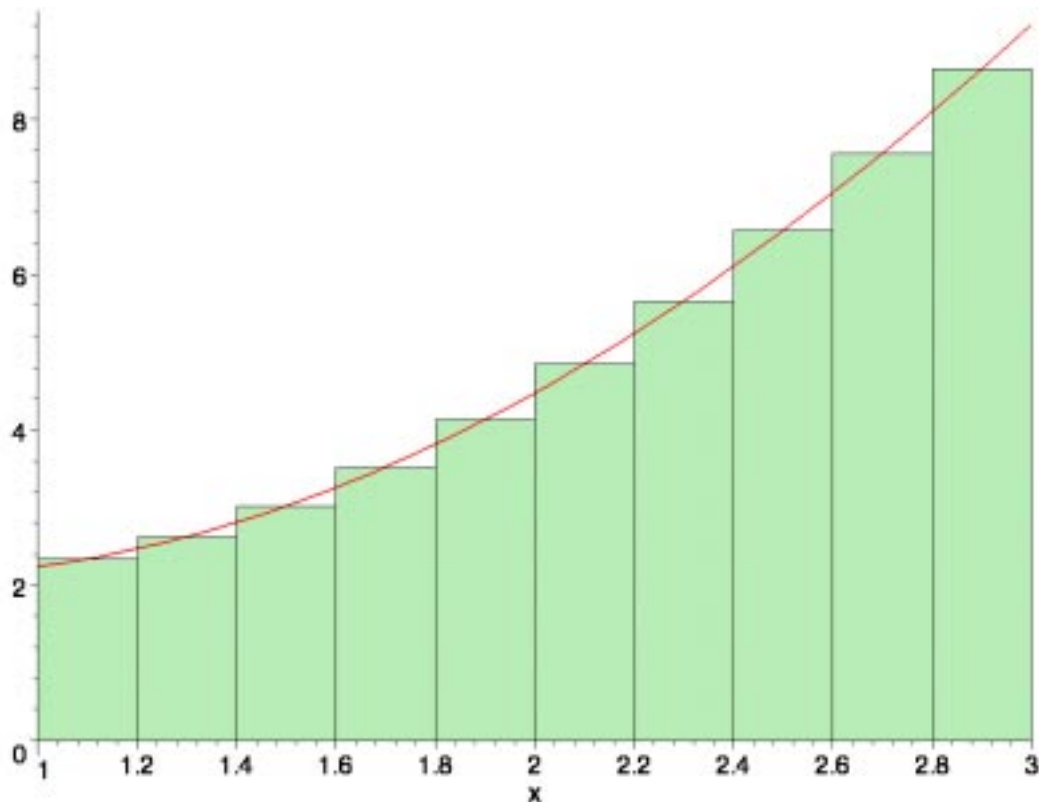
```
error_r100 := .07000018
```

Again, more rectangles give us better approximations.

### Approximating using the midpoint rule.

An alternate method uses the midpoint of each interval to determine the height of the rectangles to be summed. This is the **midpoint rule**. We view this rule graphically using  $f$  and [middlebox](#).

```
> middlebox(f,x=1..3,10);
```



The general formula is  $\sum_{i=1}^n f(m_i) \Delta x$  where  $m_i = \frac{x_i + x_{i-1}}{2}$  is the midpoint of  $[x_{i-1}, x_i]$ . In Maple, we implement this rule with [middlesum](#).

```
> middlesum(f,x=1..3,10);
```

$$\frac{1}{5} \left( \sum_{i=0}^9 \sqrt{4 + \left( \frac{11}{10} + \frac{1}{5} i \right)^4} \right)$$

```
> mid10:=evalf(%);
```

```
mid10 := 9.774359154
```

```
> error_m10:=actualf-mid10;
```

```
error_m10 := .008276523
```

We see that we have an over-approximation here. That is because our function is concave up over the interval of integration. To see this, let's look at the midpoint rule applied to the single subinterval [2, 2.2].

```
> middlebox(f,x=2..2.2,1);
```



For the computations we need to do, we rewrite our function, currently a Maple expression, as a Maple function using the `unapply` command.

```
> ff:=unapply(f,x);
```

$$ff := x \rightarrow \sqrt{4 + x^4}$$

This allows for easy evaluation, such as

```
> ff(2);
```

$$\sqrt{20}$$

This is in contrast to the to the following, which is necessary to find  $f(2)$ .

```
> subs(x=2,f);
```

$$\sqrt{20}$$

We use the `D` command to find the derivative.

```
> ffprime:=D(ff);
```

$$ffprime := x \rightarrow 2 \frac{x^3}{\sqrt{4 + x^4}}$$

Then we use the point-slope formula to find the tangent line to the curve at the point where the top of the rectangle meets the curve.

```
> tanline:=ff(2.1)+ffprime(2.1)*(x-2.1);
```

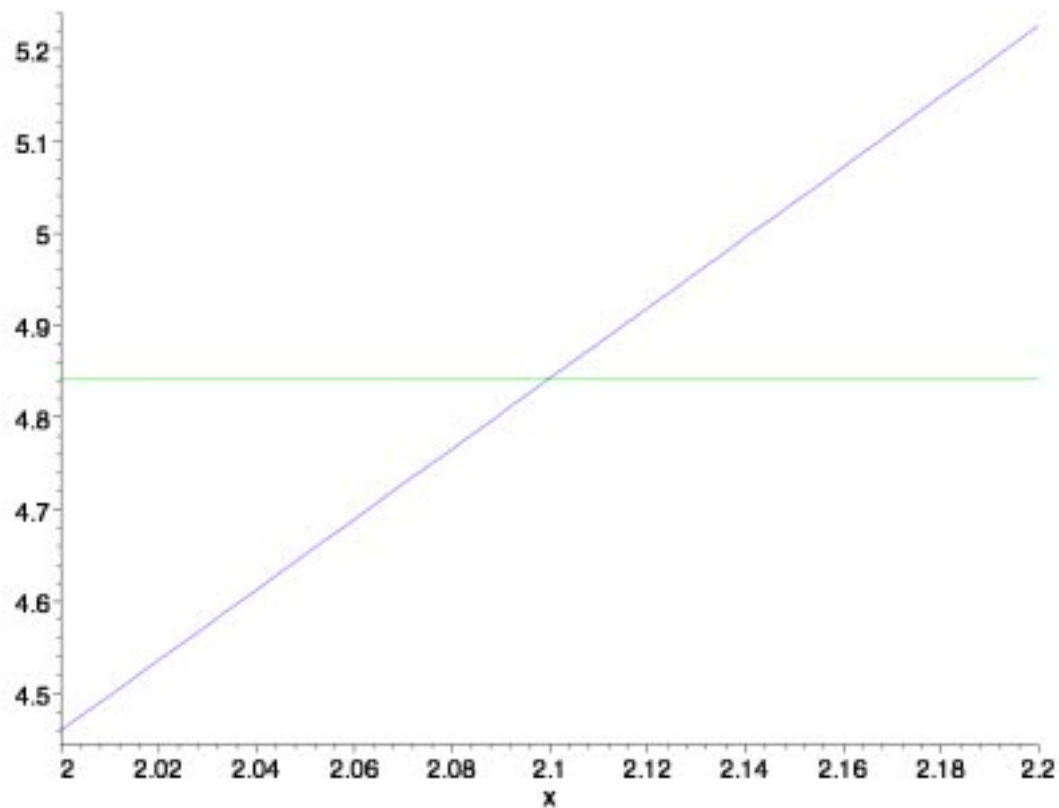
$$tanline := -3.190224478 + 3.825023000 x$$

We first compare the area under the rectangle (green) with the area under the tangent line (blue).

```
> p3:=plot(ff(2.1),x=2..2.2,color=green):
```

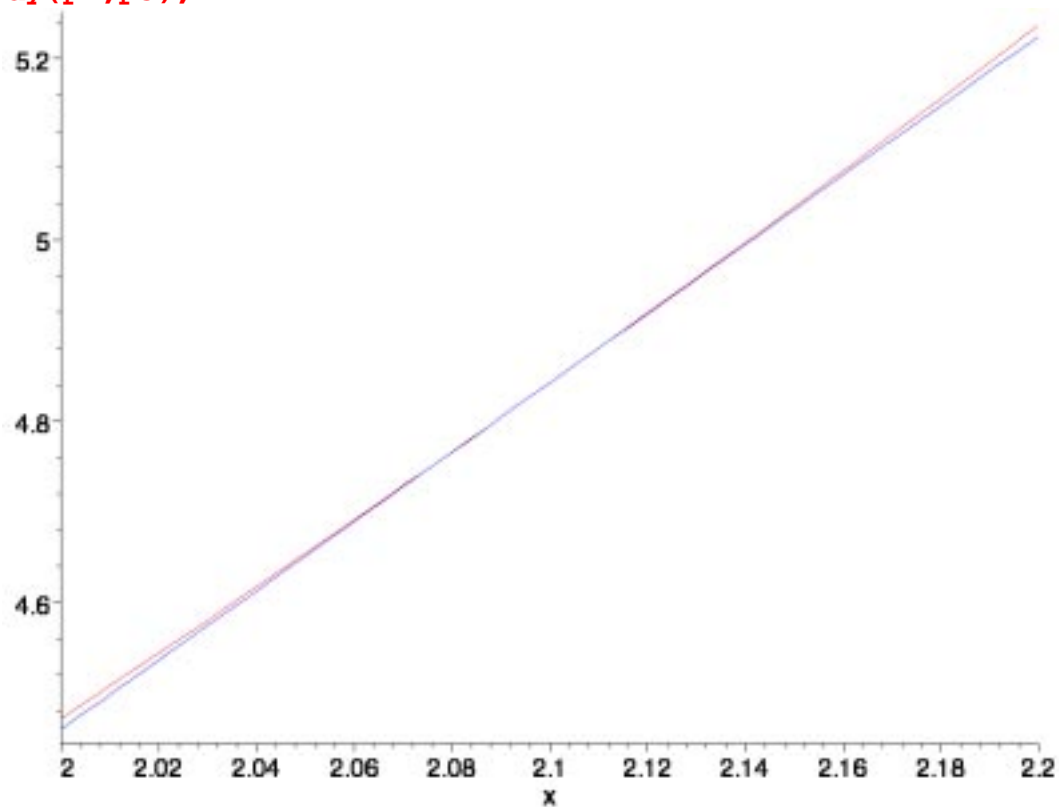
```
> p4:=plot(tanline,x=2..2.2,color=blue):
```

```
> display(p3,p4);
```



Using congruent triangles, we see that the areas are the same. So then we compare the area under the curve (red) with the area under the tangent line.

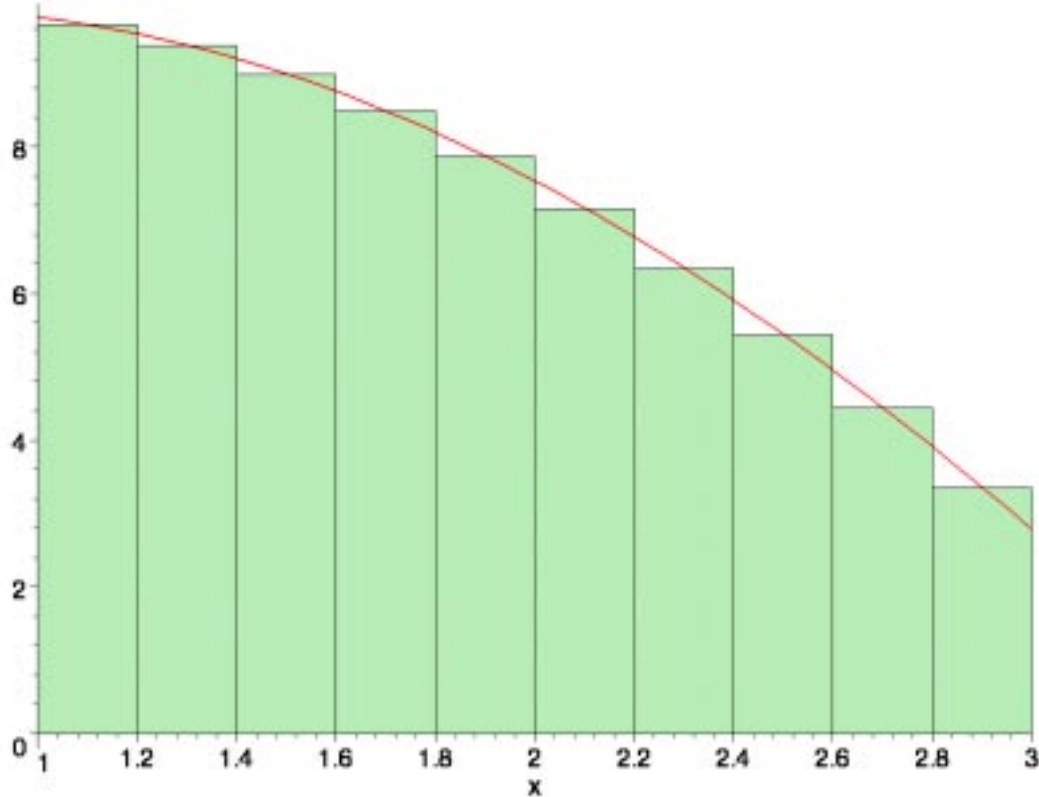
```
> p5:=plot(f,x=2..2.2,color=red):
> display(p4,p5);
```



We conclude that for a curve that is concave up, the midpoint approximation gives an

under-approximation. Similarly, the midpoint approximation with a function that is concave down, such as  $g$ , is an over-approximation.

```
> middlebox(g,x=1..3,10);
```



```
> middlesum(g,x=1..3,10);
```

$$\frac{1}{5} \left( \sum_{i=0}^9 \left( -\sqrt{4 + \left( \frac{11}{10} + \frac{1}{5}i \right)^4} + 12 \right) \right)$$

```
> mid10:=evalf(%);
```

```
mid10 := 14.22564085
```

```
> error_m10:=actualg-mid10;
```

```
error_m10 := -.00827653
```

### Approximating using the trapezoid rule.

An alternate approach is to use the **trapezoid rule**, which for a given value of  $n$  is the average of the left rule and right rule, i.e.,  $\text{trap}(n) = \frac{\text{left}(n) + \text{right}(n)}{2}$ . To see this graphically with  $n = 2$ , we first define [lists](#) of our  $x$  and  $f(x)$  (or  $y$ ) values using the [seq](#) command.

```
> X:=[seq(1+1*i,i=0..2)];
```

```
X := [1, 2, 3]
```

```
> Y:=[seq(ff(X[i]),i=1..3)];
```

```
Y := [√5, √20, √85]
```

We plot the graph with the trapezoids.

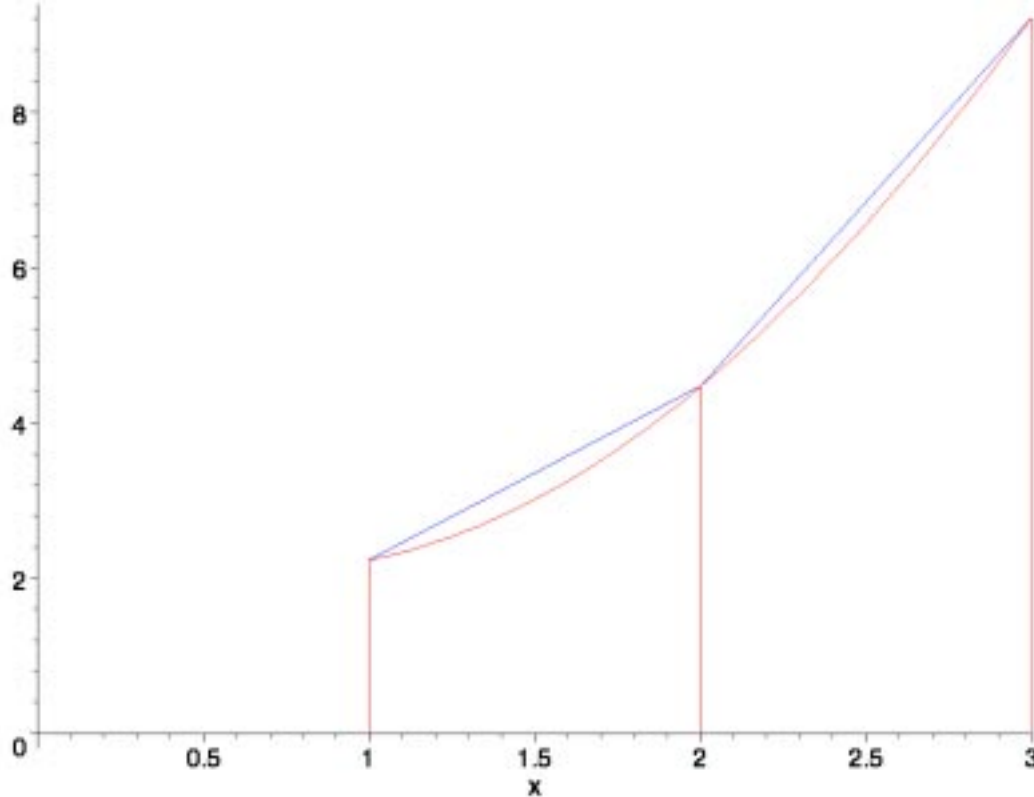
```
> p6:=pointplot([seq([X[i],Y[i]],i=1..3)],style=line,color=blue):
```

```
> p7:=pointplot({[1,0],[1,ff(1)]},style=line,color=red):
```

```
> p8:=pointplot({[2,0],[2,ff(2)]},style=line,color=red):
```

```
> p9:=pointplot({[3,0],[3,ff(3)]},style=line,color=red):
```

```
> display(p0,p1,p6,p7,p8,p9);
```



We see that for a function that is concave up, the trapezoid rule will give an over-approximation. Similarly, for a function that is concave down, the trapezoid rule gives an under-approximation. The general formula for the trapezoid rule is  $\frac{\Delta x}{2} [f(x_0) + 2 f(x_1) + 2 f(x_2) + \dots + 2 f(x_{n-1}) + f(x_n)]$ . It is implemented in Maple with the `trapezoid` command. We use the trapezoid rule with  $n = 10$ .

```
> trapezoid(f,x=1..3,10);
```

$$\frac{1}{10}\sqrt{5} + \frac{1}{5} \left( \sum_{i=1}^9 \sqrt{4 + \left(1 + \frac{1}{5}i\right)^4} \right) + \frac{1}{10}\sqrt{85}$$

```
> trap10:=evalf(%);
```

```
trap10 := 9.799184118
```

```
> error10:=actualf-trap10;
```

```
error10 := -.016548441
```

We see we get the expected over-approximation, and that it is considerably better than either the left rule or right rule, but not as good as the midpoint rule.

### Approximating using Simpson's rule.

Since the trapezoidal error is usually about twice the midpoint error and opposite in sign, Simpson's rule, which uses a weighted average of these two, can be expected to give even more accurate approximations. The rule is  $\text{simp}(n) = \frac{2 \text{mid}(n) + \text{trap}(n)}{3}$ . The geometric effect of this weighted average is to use quadratic polynomials rather than straight lines to approximate curves. We illustrate this with the function  $F(x) = x \sin(x)$  over the interval  $[0, 4]$ . We will take at first  $n = 2$  to give us the three points 0, 2, 4 on the  $x$ -axis. Now three nonlinear points will determine a unique parabola, and the three points we use are the three points on the graph of  $F$  with  $x$ -coordinates 0, 2, 4. To find the formula for this parabola, we use the `interp` command, which takes as its arguments the lists of  $x$  and  $y$

values along with the variable chosen, here  $x$ . First, we directly define  $F$  as a Maple function. Notice the syntax.

```
> restart:with(student):with(plots):  
Warning, the name changecoords has been redefined
```

```
> F:=x->x*sin(x);  
F := x → x sin(x)
```

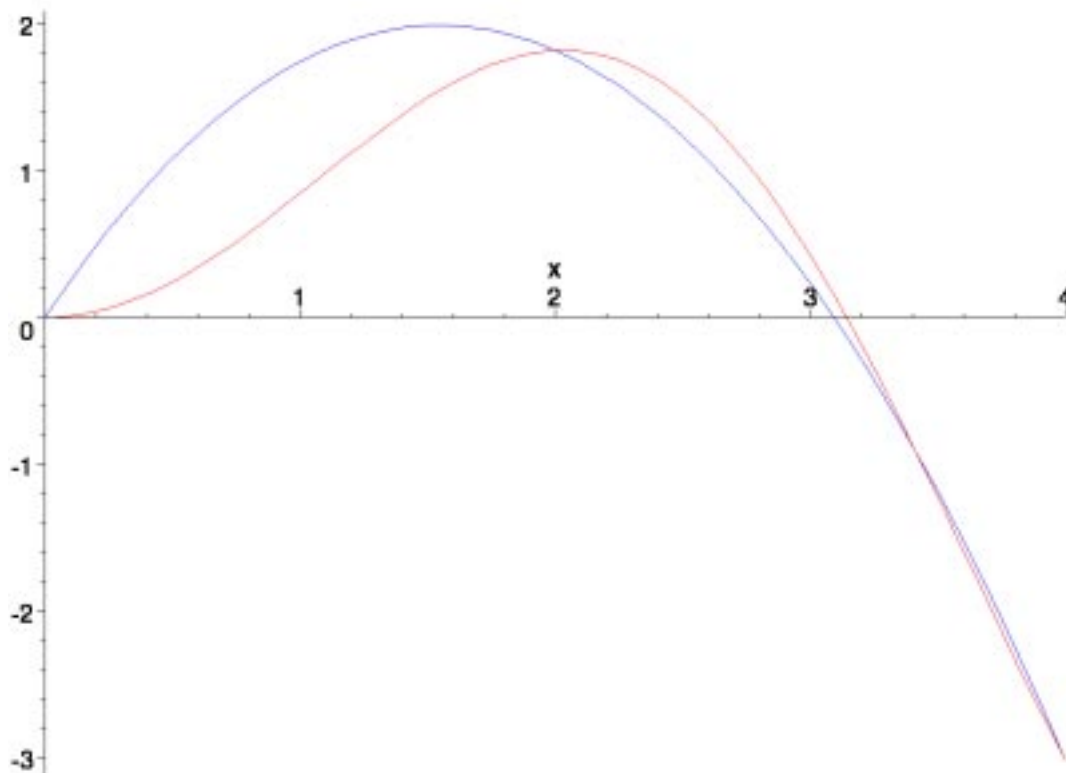
```
> X:=seq(2*i,i=0..2);  
X := [0, 2, 4]
```

```
> Y:=seq(F(X[i]),i=1..3);  
Y := [0, 2 sin(2), 4 sin(4)]
```

```
> parabola:=interp(X,Y,x);  
parabola :=  $\frac{1}{2} \sin(4) x^2 - \sin(4) x - \frac{1}{2} \sin(2) x^2 + 2 \sin(2) x$ 
```

Now that we have a formula for the parabola (blue), we graph it along with the function (red).

```
> p10:=plot(F(x),x=0..4,color=red):  
p11:=plot(parabola,x=0..4,color=blue):  
display(p10,p11);
```



The parabola follows the curve pretty accurately here. Let's take the integral of the parabola and get its decimal approximation.

```
> int(parabola,x=0..4);  
 $\frac{8}{3} \sin(4) + \frac{16}{3} \sin(2)$ 
```

```
> evalf(%);  
2.831446288
```

Now let's approximate the integral of  $F$  by Simpson's rule by using the command [simpson](#) with  $n = 2$ .

Because a parabola is determined by three points covering two subintervals,  $n$  must always be even when using Simpson's rule.

```
> simpson(F(x),x=0..4,2);
```

$$\frac{8}{3} \sin(4) + \frac{8}{3} \left( \sum_{i=1}^1 (4i-2) \sin(4i-2) \right) + \frac{4}{3} \left( \sum_{i=1}^0 (4i) \sin(4i) \right)$$

```
> simp2:=evalf(%);
```

```
simp2 := 2.831446290
```

So we see that Simpson's approximation is just the integral of the parabola (with a bit of round-off error). Now let's find the integral of  $F$  and find the error in our approximation.

```
> int(F(x),x=0..4);
```

```
sin(4) - 4 cos(4)
```

```
> actualF:=evalf(%);
```

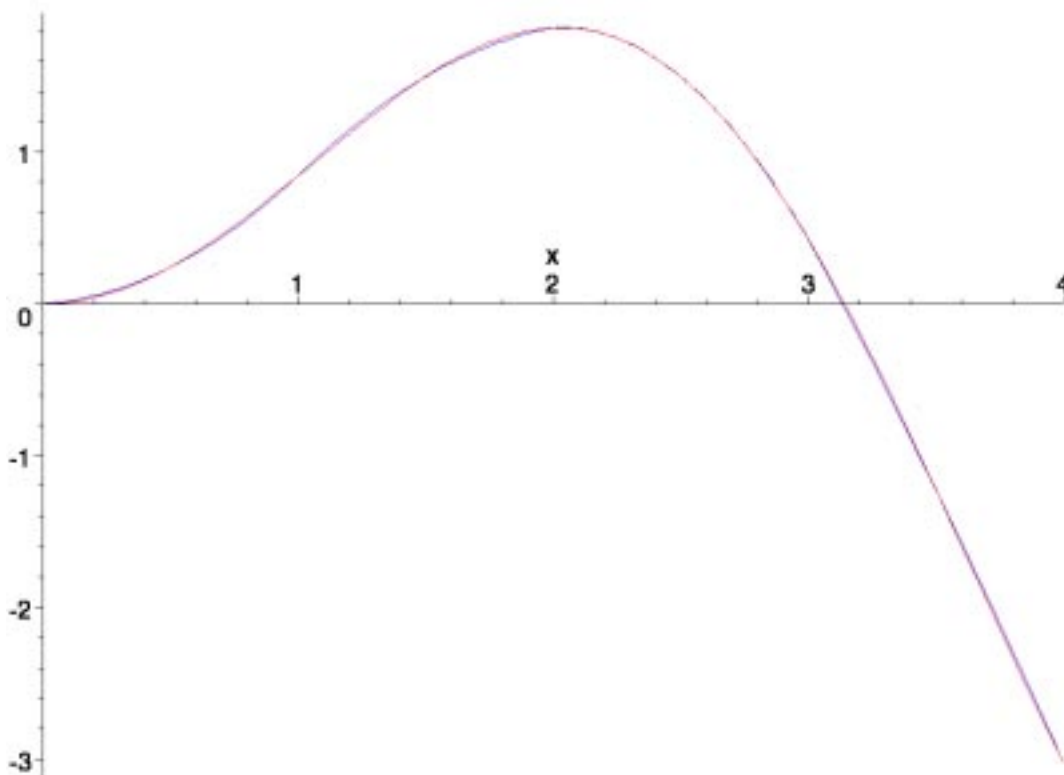
```
actualF := 1.857771989
```

```
> error2:=actualF-simp2;
```

```
error2 := -.973674301
```

OK, this is not a real good approximation, but then our  $n$  is as small as it can possibly be. Let's see what happens with  $n = 8$ . We first create a graph of the function with the four parabolas we will use.

```
> for j from 0 to 3 do
  X:=[seq(j+i*.5,i=0..2)];
  Y:=[seq(F(X[i]),i=1..3)];
  parabola:=interp(X,Y,x);
  pl[j]:=plot(parabola,x=j..(j+1),color=blue):
od:
display(pl0,seq(pl[j],j=0..3));
```



We see that the four parabolas fit the graph of the function very well, giving hopes for a very accurate

approximation. The general formula for the trapezoid rule is  $\frac{\Delta x}{3} [f(x_0) + 4 f(x_1) + 2 f(x_2) + 4 f(x_3) + \dots + 2 f(x_{n-2}) + 4 f(x_{n-1}) + f(x_n)]$ . We now find Simpson's approximation for  $n = 8$  and the error in that approximation.

```
> simpson(F(x), x=0..4, 8);
```

$$\frac{2}{3} \sin(4) + \frac{2}{3} \left( \sum_{i=1}^4 \left( i - \frac{1}{2} \right) \sin \left( i - \frac{1}{2} \right) \right) + \frac{1}{3} \left( \sum_{i=1}^3 i \sin(i) \right)$$

```
> simp8:=evalf(%);
```

```
simp8 := 1.859536499
```

```
> error8:=actualF-simp8;
```

```
error8 := -.001764510
```

Pretty good!

### Spreadsheet summary.

We can use the Maple spreadsheet below to see the approximations and errors given by the various methods for approximating the integral of a given function over the interval  $[a, b]$ . To use the spreadsheet, enter the function  $f$ , the limits of integration  $a$  and  $b$ , and three values of  $n$ , labeled as  $n1$ ,  $n2$ , and  $n3$ , as shown in the next line.

```
> f:=sqrt(4+x^4);a:=0;b:=3;n1:=10;n2:=100;n3:=1000;
```

$$f := \sqrt{4 + x^4}$$

```
a := 0
```

```
b := 3
```

```
n1 := 10
```

```
n2 := 100
```

```
n3 := 1000
```

With these values, click the mouse on the blank gray rectangle in the upper left corner of the spreadsheet to select all the cells of the spreadsheet and then hit the **Enter** key. You can see how the spreadsheet was created by clicking on the individual cells and observing the formulas at the top of the page.



[ >

[ >

[ Let's try another function.

[ > **f:=x\*sin(x);a:=0;b:=2;n1:=10;n2:=100;n3:=1000;**

*f := x sin(x)*

*a := 0*

*b := 2*

*n1 := 10*

*n2 := 100*

*n3 := 1000*

[ >



[ >

[ **Error in Simpson's rule: choosing  $n$  to control error.**

[ >

[ The error for Simpson's Rule is  $-\frac{(b-a)^5}{180 n^4} f''''(\mu)$ , where  $\mu$  is some number between  $a$  and  $b$ . Thus, if  $|f''''(x)| \leq M$  for  $x = a \dots b$ ,

$$E \leq \frac{(b-a)^5}{180 n^4} M \quad (*)$$

[ where  $E$  is the absolute value of the error. For  $F(x) = \sqrt{x} \sin(x)$ , used above, we have  $a = 0$ ,  $b = 4$ , and  $n = 8$ . We find the fourth derivative of  $F$ .

[ > **a:=0;b:=4;n:=8;**

$a := 0$

$b := 4$

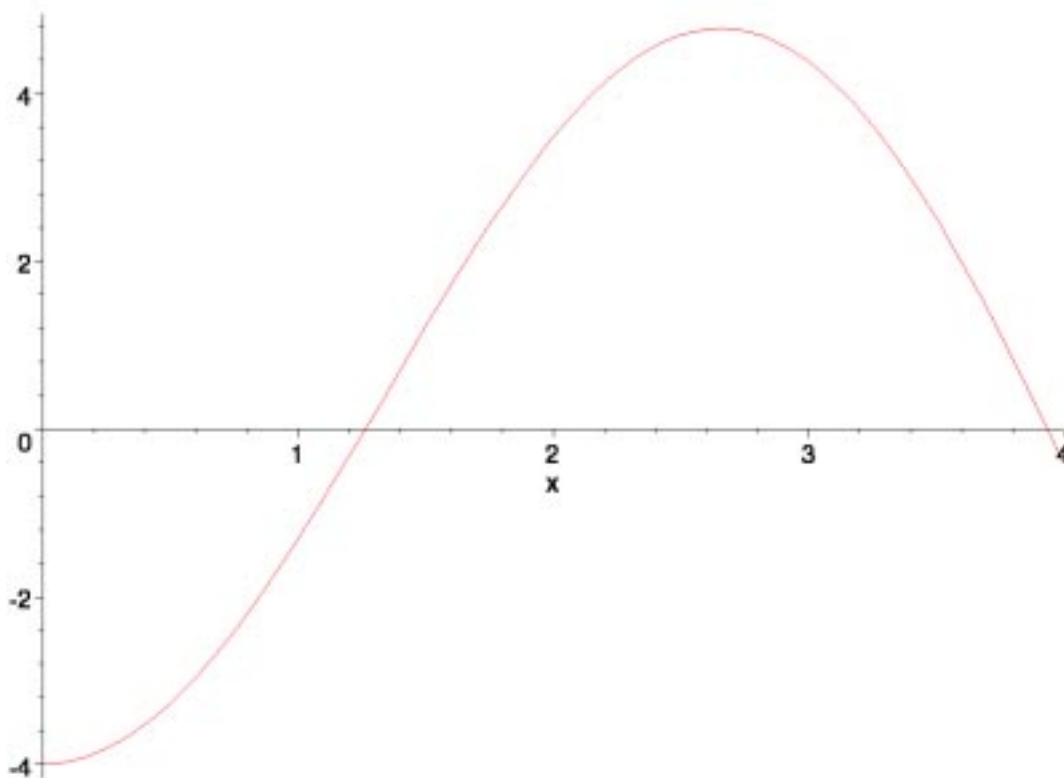
$n := 8$

[ > **F4p:=(D@@4)(F);**

$F4p := x \rightarrow -4 \cos(x) + x \sin(x)$

[ We look at the graph of this fourth derivative between 0 and 4.

[ > **plot(F4p(x),x=0..4);**



[ By viewing the graph, we could simply choose  $M = 5$ . But we can be more exact, as follows:

[ > **M:=evalf(maximize(F4p(x),x=0..4));**

$M := 4.777253597$

[ Then we compute an upper bound for the absolute error in our approximation.

[ > **Error:=(b-a)^5/(180\*n^4)\*M;**

$Error := .006635074439$

We note from above that the absolute value of the error was less than this. We can also rewrite the inequality (\*) above as

$$n \geq \left( \frac{(b-a)^5 M}{180 E} \right)^{\frac{1}{4}}$$

so as to choose an appropriate  $n$  so that the absolute error will be less than a chosen value  $E$ . Suppose we wish the error to be less than  $E = .0001$ .

**> E:=.0001;**

$E := .0001$

**> n:=evalf(((b-a)^5\*M/(180\*E))^(1/4));**

$n := 22.83239331$

Thus, since  $n$  must be even and at least as large as this number, choose  $n = 24$  and use Simpson.

**> simpson(F(x),x=0..4,24);**

$$\frac{2}{9} \sin(4) + \frac{2}{9} \left( \sum_{i=1}^{12} \left( \frac{1}{3}i - \frac{1}{6} \right) \sin \left( \frac{1}{3}i - \frac{1}{6} \right) \right) + \frac{1}{9} \left( \sum_{i=1}^{11} \left( \frac{1}{3}i \sin \left( \frac{1}{3}i \right) \right) \right)$$

**> simp24:=evalf(%);**

$simp24 := 1.857793020$

**> error24:=actualF-simp24;**

$error24 := -.000021031$

We are certainly within the required tolerance.

**>**