Approximate Integration

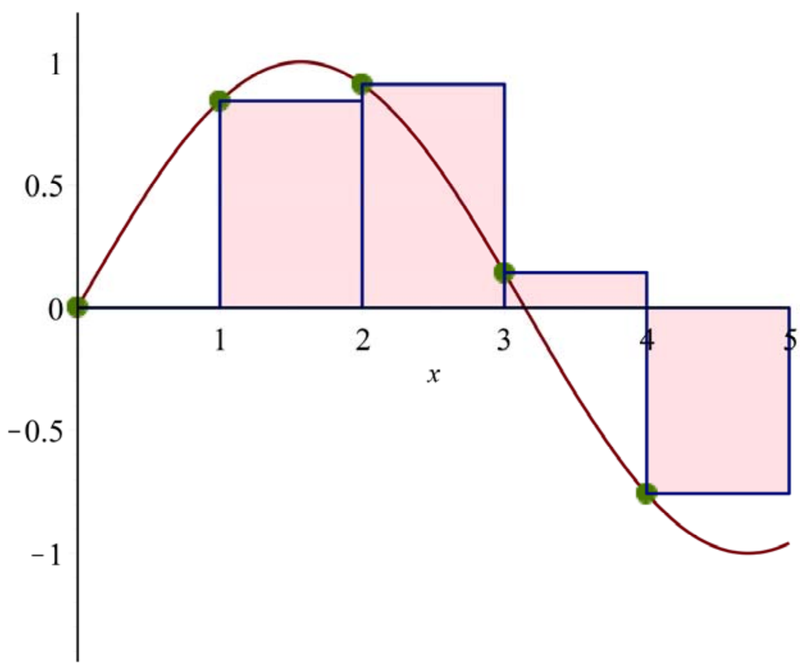
James Rohal

First we load the necessary packages and use an example function *f* with interval bound [*a*, *b*]. There will be *n* subintervals.

```
> with(Student[Calculus1]):
  f := sin(x):
  a := 0:
  b := 5:
  n := 5:
```

Regardless of the method used, we can always find the area to higher precision by shrinking the partition. *Click the image below and click the play button in the toolbar above to play an animation.*

```
> ApproximateInt(f, x=a..b, method = left, partition = n, output =
  animation, pointoptions=[symbolsize=20,symbol=solidcircle],
  boxoptions=[filled=[color=pink,transparency=.5]]);
```



An animated left Riemann sum approximation of $\int_0^5 f(x)\,dx$, where $f(x) = \sin(x)$ and the partition is uniform. The approximate value of the integral is 1.135085925. Number of subintervals used: 5.

Rectangular Approximations

The first three methods approximate the area under the curve by using rectangles. The height is determined by where we pick a sample point.

Left Endpoints

The sample point is chosen as the left endpoint of each subinterval.

```
> # gives the series we will be computing for N subintervals
  ApproximateInt(f, x=a..b, method = left, partition = N);
```

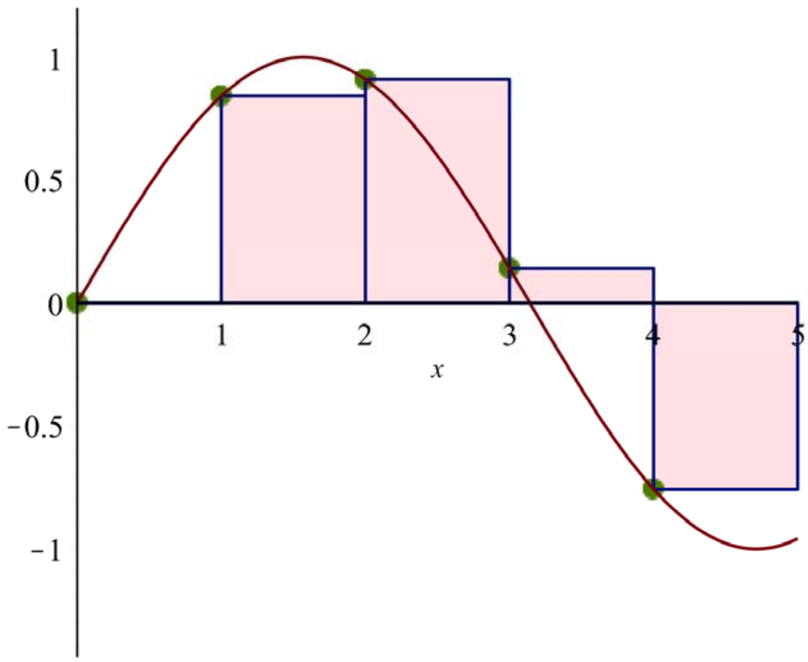
$$\frac{5\left(\sum_{i=0}^{N-1}\sin\left(\frac{5\,i}{N}\right)\right)}{N} \qquad \textbf{(1.1.1)}$$

```
> # gives the approximate integral using n subintervals
  left_approx := ApproximateInt(f, x=a..b, method = left,
  partition = n);
  evalf(left_approx);
```

$$left\_approx := \sin(1) + \sin(2) + \sin(3) + \sin(4)$$

$$1.135085925 \qquad \textbf{(1.1.2)}$$

```
> # plots the function and the approximating rectangles for the
  given method
  ApproximateInt(f, x=a..b, method = left, partition = n,
  output = plot, pointoptions=[symbolsize=20,symbol=
  solidcircle], boxoptions=[filled=[color=pink,transparency=.5]
  ]);
```



A left Riemann sum approximation of $\int_0^5 f(x)\,dx$, where $f(x) = \sin(x)$ and the partition is uniform. The approximate value of the integral is 1.135085925. Number of subintervals used: 5.

Right Endpoints

The sample point is chosen as the right endpoint of each subinterval.

```
> ApproximateInt(f, x=a..b, method = right, partition = N);
```

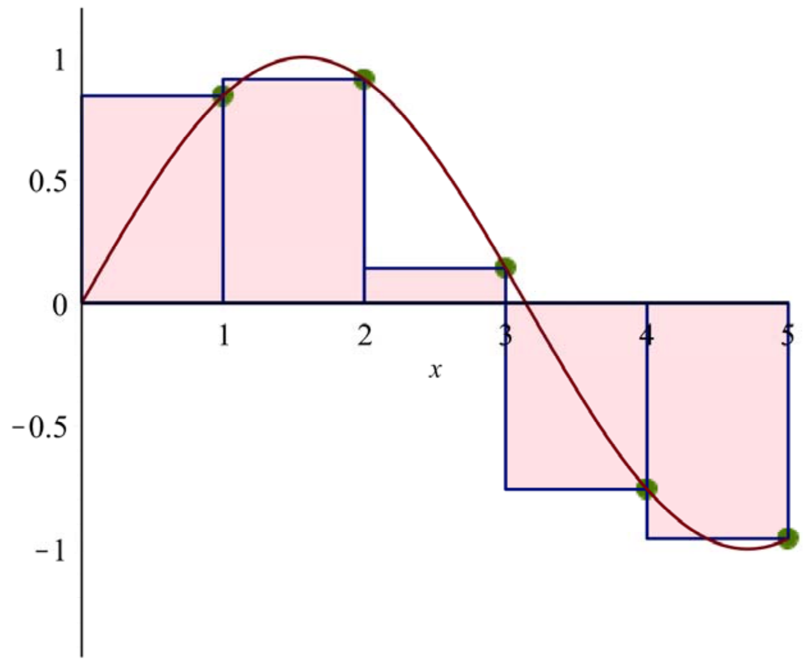
$$\frac{5\left(\sum_{i=1}^{N}\sin\left(\frac{5\,i}{N}\right)\right)}{N} \tag{1.2.1}$$

```
> right_approx := ApproximateInt(f, x=a..b, method = right,
  partition = n);
  evalf(right_approx);
```

$$right\_approx := \sin(1) + \sin(2) + \sin(3) + \sin(4) + \sin(5)$$

$$0.1761616503 \tag{1.2.2}$$

```
> ApproximateInt(f, x=a..b, method = right, partition = n,
  output = plot, pointoptions=[symbolsize=20,symbol=
  solidcircle], boxoptions=[filled=[color=pink,transparency=.5]
  ]);
```



A right Riemann sum approximation of $\int_0^5 f(x)\,dx$, where $f(x) = \sin(x)$ and the partition is uniform. The approximate value of the integral is 0.1761616503. Number of subintervals used: 5.

Midpoint Rule

The sample point is chosen as the midpoint of each subinterval.

```
> ApproximateInt(f, x=a..b, method = midpoint, partition = N);
```

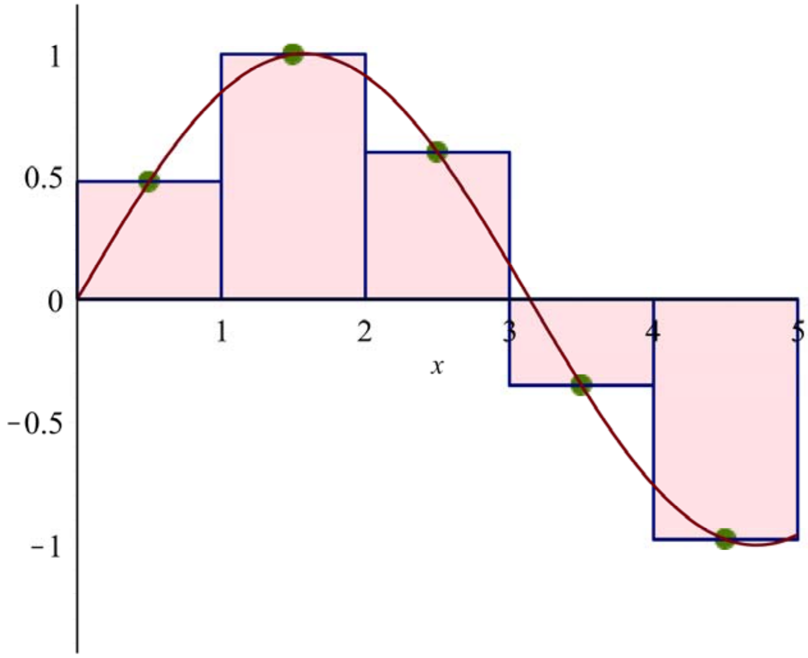
$$\frac{5\left(\sum_{i=0}^{N-1}\sin\left(\frac{5\left(i+\frac{1}{2}\right)}{N}\right)\right)}{N} \qquad \textbf{(1.3.1)}$$

```
> mid_approx := ApproximateInt(f, x=a..b, method = midpoint,
  partition = n);
  evalf(mid_approx);
```

$$mid\_approx := \sin\left(\frac{1}{2}\right)+\sin\left(\frac{3}{2}\right)+\sin\left(\frac{5}{2}\right)+\sin\left(\frac{7}{2}\right)+\sin\left(\frac{9}{2}\right)$$

$$0.7470793233 \qquad \textbf{(1.3.2)}$$

```
> ApproximateInt(f, x=a..b, method = midpoint, partition = n,
  output = plot, pointoptions=[symbolsize=20,symbol=
  solidcircle], boxoptions=[filled=[color=pink,transparency=.5]
  ]);
```



A midpoint Riemann sum approximation of $\int_0^5 f(x)\,dx$, where $f(x) = \sin(x)$ and the partition is uniform. The approximate value of the integral is 0.7470793233. Number of subintervals used: 5.

Non-Rectangular Approximations

Trapezoid Rule

```
> ApproximateInt(f, x=a..b, method = trapezoid, partition = N);
```

$$\frac{5}{2}\,\frac{\sum_{i=0}^{N-1}\left(\sin\left(\frac{5\,i}{N}\right)+\sin\left(\frac{5\,(i+1)}{N}\right)\right)}{N} \quad \textbf{(2.1.1)}$$

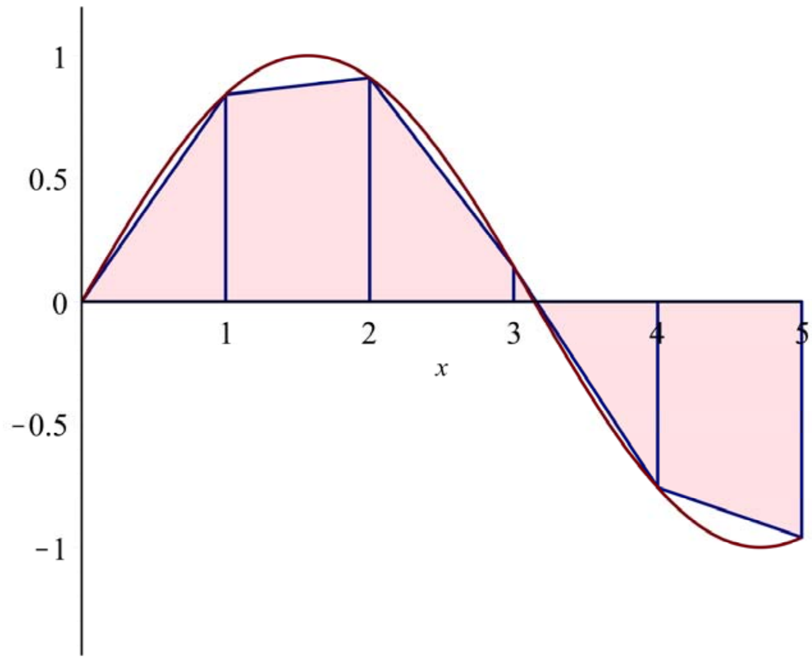
```
> trap_approx := ApproximateInt(f, x=a..b, method = trapezoid,
  partition = n);
  evalf(trap_approx);
```

$$trap\_approx := \sin(1)+\sin(2)+\sin(3)+\sin(4)+\frac{1}{2}\sin(5)$$

$$0.6556237876 \quad \textbf{(2.1.2)}$$

```
> ApproximateInt(f, x=a..b, method = trapezoid, partition = n,
  output = plot, boxoptions=[filled=[color=pink,transparency=
```

```
.5]]);
```



An approximation of $\int_0^5 f(x)\,dx$ using trapezoid rule, where $f(x) = \sin(x)$ and the partition is uniform. The approximate value of the integral is 0.6556237871. Number of subintervals used: 5.

Simpson's Rule

```
> ApproximateInt(f, x=a..b, method = simpson, partition = N);
```

$$\frac{5}{6}\,\frac{\displaystyle\sum_{i=0}^{N-1}\left(\sin\left(\frac{5\,i}{N}\right)+4\sin\left(\frac{5\left(i+\frac{1}{2}\right)}{N}\right)+\sin\left(\frac{5\,(i+1)}{N}\right)\right)}{N} \tag{2.2.1}$$

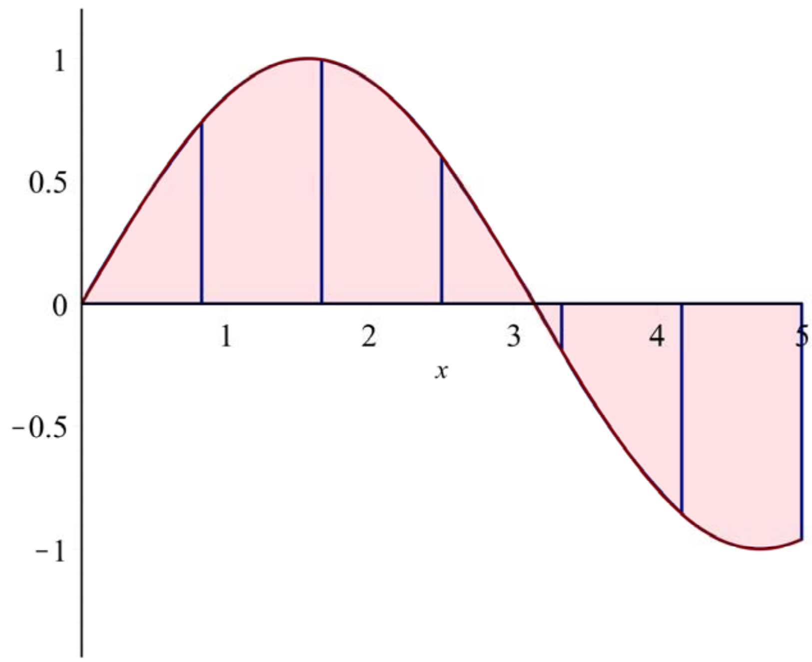
```
> # I had to use 6 subintervals here rather than 5. This is
  because n must be even to use Simpson's rule
  simp_approx := ApproximateInt(f, x=a..b, method = simpson,
  partition = 6);
  evalf(simp_approx);
```

$$simp\_approx := \frac{5}{9}\sin\left(\frac{5}{12}\right)+\frac{5}{18}\sin\left(\frac{5}{6}\right)+\frac{5}{9}\sin\left(\frac{5}{4}\right)+\frac{5}{18}\sin\left(\frac{5}{3}\right)+\frac{5}{9}\sin\left(\frac{25}{12}\right)+\frac{5}{18}\sin\left(\frac{5}{2}\right)+\frac{5}{9}\sin\left(\frac{35}{12}\right)+\frac{5}{18}\sin\left(\frac{10}{3}\right)$$

$$+\frac{5}{9}\sin\left(\frac{15}{4}\right)+\frac{5}{18}\sin\left(\frac{25}{6}\right)+\frac{5}{9}\sin\left(\frac{55}{12}\right)+\frac{5}{36}\sin(5)$$

$$0.7164602895 \qquad \textbf{(2.2.2)}$$

```
> ApproximateInt(f, x=a..b, method = simpson, partition = 6,
  output = plot, boxoptions=[filled=[color=pink,transparency=
  .5]]);
```



An approximation of $\int_0^5 f(x)\,dx$ using Simpson's rule, where $f(x)=\sin(x)$ and the partition is uniform. The approximate value of the integral is 0.7164602897. Number of subintervals used: 6.

Absolute Error

First we need the exact area.

```
> exact := int(f, x=a..b);
  evalf(exact);
```

$$exact := 1-\cos(6)$$

$$0.0398297133 \qquad \textbf{(3.1)}$$

The absolute error is `abs(exact_solution - approximate_solution)`.

```
> evalf(abs(exact - left_approx));
  evalf(abs(exact - right_approx));
  evalf(abs(exact - mid_approx));
  evalf(abs(exact - trap_approx));
  evalf(abs(exact - simp_approx));
```

$$0.1363319363$$
$$0.1430835619$$
$$0.0017092847$$
$$0.0033758128$$
$$0.00001425209 \qquad (3.2)$$

Homework

This is an extension of 5.9 #17. **DO NOT USE `ApproximateInt` UNLESS TOLD TO DO SO**. Part (a) - (g) should be done as if you were writing them on a sheet of paper. We are simply using Maple to help us do the messy computations. Be sure to execute the commands below.

```
> with(Student[Calculus1]):
  f := cos(x^2):
  a := 0:
  b := 1:
  n := 4:
  dx := (b-a)/n:
  exact := int(cos(x^2), x=0..1):
```

(a) Write out the approximation L_4 for $\int_0^1 \cos(x^2)\, dx$ by hand. Give the exact value for L_4 and use `evalf` to give a decimal approximation for L_4. Assign the exact answer for L_4 to the variable `L4`.

```
> # define the subinterval endpoints
  x0 := 0/4:
  x1 := 1/4:
  x2 := 2/4:
  x3 := 3/4:

  # write out L4 "by hand" and assign it to the variable L4
  L4 := dx * ( cos(x0^2) + cos(x1^2) + cos(x2^2) + cos(x3^2) );

  # give the decimal approximation
  evalf(L4);
```

$$L4 := \frac{1}{4} + \frac{1}{4}\cos\left(\frac{1}{16}\right) + \frac{1}{4}\cos\left(\frac{1}{4}\right) + \frac{1}{4}\cos\left(\frac{9}{16}\right)$$
$$0.9532211079 \qquad (4.1.1)$$

(b) Write out the approximation R_4 for $\int_0^1 \cos(x^2)\, dx$ by hand. Give the exact value for R_4 and use `evalf` to give a decimal approximation for R_4. Assign the exact answer for R_4 to the variable `R4`.

(c) Write out the approximation M_4 for $\int_0^1 \cos(x^2)\,dx$ by hand. Give the exact value for M_4 and use `evalf` to give a decimal approximation for M_4. Assign the exact answer for M_4 to the variable `M4`.

(d) Write out the approximation T_4 for $\int_0^1 \cos(x^2)\,dx$ by hand. Give the exact value for T_4 and use `evalf` to give a decimal approximation for T_4. Assign the exact answer for T_4 to the variable `T4`.

(e) Write out the approximation S_4 for $\int_0^1 \cos(x^2)\,dx$ by hand. Give the exact value for S_4 and use `evalf` to give a decimal approximation for S_4. Assign the exact answer for S_4 to the variable `S4`.

(f) Estimate the absolute error in approximation for part (c). Assign this value to the variable `E_M4`. Is this an overestimation or an underestimation?

```
> E_M4 := abs(exact - M4):
  evalf(E_M4);
```

$$0.0043825527 \quad (4.6.1)$$

```
> evalf(exact - M4);
```

$$-0.0043825527 \quad (4.6.2)$$

This is an overestimation since the difference above is negative.

(g) Estimate the absolute error in approximation for part (d). Assign this value to the variable `E_T4`. Is this an overestimation or an underestimation?

(h) Use `ApproximateInt` with 8 subintervals to calculate a decimal approximation M_8 for $\int_0^1 \cos(x^2)\,dx$. Assign your approximation to

the variable `M8`.

```
> M8 := evalf(ApproximateInt(f, x=a..b, method = midpoint,
  partition = 8));
```

$$M8 := 0.9056199570 \tag{4.8.1}$$

(i) Use `ApproximateInt` with 8 subintervals to calculate a decimal approximation T_8 for $\int_0^1 \cos(x^2)\, \mathrm{d}x$. Assign your approximation to the variable `T8`.

(j) Estimate the absolute error in the approximation for part (h). Assign this value to the variable `E_M8`.

(k) Estimate the absolute error in the approximation for part (i). Assign this value to the variable `E_T8`.

(l) As *n* increases by a factor of 2, by what whole number factor do the errors for M_n drop? *Hint: Use (f) and (j).*

```
> evalf(E_M4/E_M8);
```

$$3.999704943 \tag{4.12.1}$$

It decreases by a factor of 4.

(m) As *n* increases by a factor of 2, by what whole number factor do the errors for T_n drop? *Hint: Use (g) and (k).*