Approximate Integration

Your Name Goes Here

Introduction

The goal of this notebook is to introduce you to the process of computing intergrals numerically in *Mathematica*. We will introduce you to the **`NIntegrate`** command, which will allow you to compute integrals approximately. We then provide methods for computing approximations that were discussed in class such as left/right/midpoint Riemann sum approximiations, Trapezoid approximations, and Simpson approximations.

Execute This Cell Before Beginning

Evaluate the cell below every time you begin working on this notebook. It initializes certain commands that you will be using. You do not need to understand the commands in this section.

```
LeftRiemannSum[f_, {var_, a_, b_}, n_] := Module[{dx, heights, Ln},
      b - a
  dx = ─────;
        n
  heights = Most@Table[f /. var → xi, {xi, a, b, dx}];
  Ln = dx * Total@heights;
  Return[Ln];
 ]
```

```
RightRiemannSum[f_, {var_, a_, b_}, n_] := Module[{dx, heights, Rn},
      b - a
  dx = ─────;
        n
  heights = Rest@Table[f /. var → xi, {xi, a, b, dx}];
  Rn = dx * Total@heights;
  Return[Rn];
 ]
```

```
MidpointRiemannSum[f_, {var_, a_, b_}, n_] := Module[{dx, heights, Mn},
      b - a
  dx = ─────;
        n
                                  xi + (xi + dx)
  heights = Table[f /. var → ──────────────, {xi, a, b - dx, dx}];
                                        2
  Mn = dx * Total@heights;
  Return[Mn];
 ]
```

```
TrapezoidSum[f_, {var_, a_, b_}, n_] := Module[{dx, heights, Tn},
        b - a
   dx = ─────;
          n
   heights = Table[f /. var → xi, {xi, a, b, dx}];
        dx
   Tn = ── * Sum[heights⟦i⟧ + heights⟦i + 1⟧, {i, n}];
        2
   Return[Tn];
  ]
```

```
SimpsonSum[f_, {var_, a_, b_}, n_] := Module[{dx, heights, Sn, coefficient},
        b - a
   dx = ─────;
          n
   coefficient[i_?EvenQ] := 4;
   coefficient[i_?OddQ] := 2;
   heights = Table[f /. var → xi, {xi, a, b, dx}];
   Sn =
    dx
    ── * (heights⟦1⟧ + Sum[coefficient[i] * heights⟦i⟧, {i, 2, n}] + heights⟦n + 1⟧);
    3
   Return[Sn];
  ]
```

```
DrawLeftRiemannSum[f_, {var_, a_, b_}, n_] := Module[{fplot, deltax, rectangles},
            b - a
   deltax = ─────;
              n
   fplot = Plot[f, {var, a, b},
     FrameTicks →
      {{Table[{f /. var → (a + i deltax), Style[f /. var → (a + i deltax), Gray]},
          {i, 0, n}], None}, {Table[a + i deltax, {i, 0, n}], None}},
     Frame → {{True, False}, {True, False}},
     Axes → None, PlotStyle → Directive[Black, Thick]];
   rectangles = Table[Rectangle[{a + i deltax, 0},
      {a + (i + 1) deltax, f /. var -> a + i deltax}], {i, 0, n - 1}];
   Show[fplot, Graphics[{LightBlue, EdgeForm[Gray], rectangles}], fplot]
  ]
```

```
DrawRightRiemannSum[f_, {var_, a_, b_}, n_] := Module[{fplot, deltax, rectangles},
            b - a
   deltax = ─────;
              n
   fplot = Plot[f, {var, a, b},
     FrameTicks →
      {{Table[{f /. var → (a + i deltax), Style[f /. var → (a + i deltax), Gray]},
          {i, 0, n}], None}, {Table[a + i deltax, {i, 0, n}], None}},
     Frame → {{True, False}, {True, False}},
    Axes → None, PlotStyle → Directive[Black, Thick]];
  rectangles = Table[Rectangle[{a + i deltax, 0},
     {a + (i + 1) deltax, f /. var -> a + (i + 1) deltax}], {i, 0, n - 1}];
  Show[fplot, Graphics[{LightBlue, EdgeForm[Gray], rectangles}], fplot]
 ]

DrawMidpointRiemannSum[f_, {var_, a_, b_}, n_] :=
 Module[{fplot, deltax, rectangles},
  deltax = (b - a)/n;
  fplot = Plot[f, {var, a, b},
    FrameTicks →
     {{Table[{f /. var → (a + (i + 1/2) deltax), Style[f /. var → (a + (i + 1/2) deltax),
          Gray]}, {i, 0, n}], None}, {Table[a + i deltax, {i, 0, n}], None}},
    Frame → {{True, False}, {True, False}},
    Axes → None, PlotStyle → Directive[Black, Thick]];
  rectangles = Table[Rectangle[{a + i deltax, 0},
     {a + (i + 1) deltax, f /. var → a + (i + 1/2) deltax}], {i, 0, n - 1}];
  Show[fplot, Graphics[{LightBlue, EdgeForm[Gray], rectangles}], fplot]
 ]

DrawTrapezoidSum[f_, {var_, a_, b_}, n_] := Module[{fplot, deltax, trapezoids},
  deltax = (b - a)/n;
  fplot = Plot[f, {var, a, b},
    FrameTicks →
     {{Table[{f /. var → (a + i deltax), Style[f /. var → (a + i deltax), Gray]},
        {i, 0, n}], None}, {Table[a + i deltax, {i, 0, n}], None}},
    Frame → {{True, False}, {True, False}},
    Axes → None, PlotStyle → Directive[Black, Thick]];
  trapezoids = Table[
    Polygon[{{a + i deltax, 0}, {a + i deltax, f /. var → a + i deltax}, {a + (i + 1) deltax,
        f /. var → a + (i + 1) deltax}, {a + (i + 1) deltax, 0}}], {i, 0, n - 1}];
  Show[fplot, Graphics[{LightBlue, EdgeForm[Gray], trapezoids}], fplot]
 ]

DrawSimpsonSum[f_, {var_, a_, b_}, n_] :=
 Module[{fplot, deltax, i, a2, a1, a0, coefficients, parabola, parabolaplots,
   verticallines, x0, y0, x1, y1, x2, y2, eqn1, eqn2, eqn3, solns},
  deltax = (b - a)/n;
```

```
fplot = Plot[f, {var, a, b},
   FrameTicks →
    {{Table[{f /. var → (a + i deltax), Style[f /. var → (a + i deltax), Gray]},
        {i, 0, n}], None}, {Table[a + i deltax, {i, 0, n}], None}},
   Frame → {{True, False}, {True, False}},
   Axes → None, PlotStyle → Directive[Black, Thick]];
parabolaplots = {};
verticallines = {};
For[i = 0, i < n, i = i + 2,
 x0 = a + i deltax;
 y0 = f /. var → x0;
 x1 = a + (i + 1) deltax;
 y1 = f /. var → x1;
 x2 = a + (i + 2) deltax;
 y2 = f /. var → x2;
 eqn1 = y0 == a2 x^2 + a1 x + a0 /. x → x0;
 eqn2 = y1 == a2 x^2 + a1 x + a0 /. x → x1;
 eqn3 = y2 == a2 x^2 + a1 x + a0 /. x → x2;
 solns = First@Solve[{eqn1, eqn2, eqn3}, {a2, a1, a0}];
 parabola = a2 x^2 + a1 x + a0 /. solns;
 AppendTo[parabolaplots, Plot[parabola, {x, x0, x2},
    PlotStyle → Gray, Filling → 0, FillingStyle → LightBlue]];
 AppendTo[verticallines, Line[{{x0, 0}, {x0, y0}}]];
 AppendTo[verticallines, Line[{{x2, 0}, {x2, y2}}]];
];
Show[fplot, parabolaplots,
 Graphics[{Gray, Thickness[0.001], verticallines}], fplot]
]
```

Mathematica Commands

NIntegrate

To compute integrals numerically in *Mathematica*, we use the **NIntegrate** command. **NIntegrate[f, {x, a, b}]** gives an approxmation of the definite integral $\int_a^b f\,dx$. For example, $\int_0^\pi \sin(x)\,dx$ can be computed as:

```
NIntegrate[Sin[x], {x, 0, Pi}]
```

```
2.
```

Notice the output is 2. (two with a decimal) as opposed to 2 (two without a decimal). This is done to indicate the answer is an approximate answer.

Mathematica by default uses a “global adaptive” strategy when computing the numerical approximation of an integral via **NIntegrate**. We have built several functions to compute the numerical approximations we have done in class.

Computing L_n

To compute L_n, the Riemann sum with n subintervals using left end points, for the integral $\int_a^b f\,dx$, we use `LeftRiemannSum[f, {x, a, b}, n]`. For example, to compute L_4 for the integral $\int_0^\pi \sin(x)\,dx$:

```
LeftRiemannSum[Sin[x], {x, 0, Pi}, 4]
```

$$\frac{1}{4}\left(1+\sqrt{2}\right)\pi$$

Let’s think about for a moment how to compute this by hand so we can check our answer. First we define a, b, n, Δx and the integrand function $f(x)$:

```
a = 0;
b = Pi;
n = 4;
deltax = (b - a) / n;
f[x_] := Sin[x];
```

We then need to split the interval $[0, \pi]$ in to 4 subintervals. To quickly do this we can use the `Range` command. `Range[start, end, dx]` gives all the numbers between `start` and `end` taking steps of `dx`. For example, to get all the numbers between 0 and π taking steps of $\frac{\pi}{4}$ (which is our Δx), we do:

```
Range[0, Pi, deltax]
```

$$\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi\right\}$$

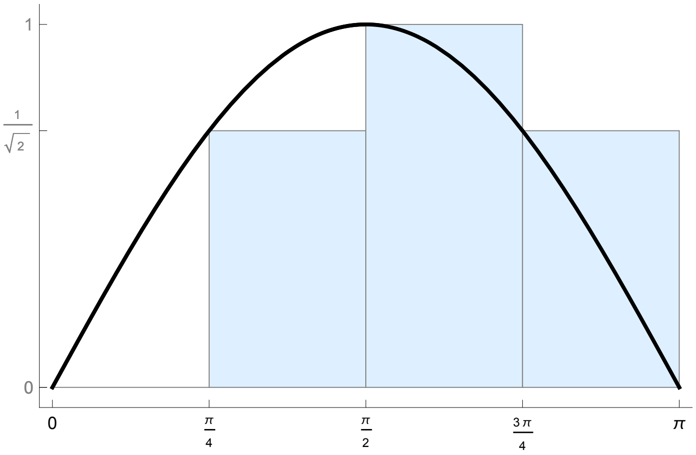
From the above we see the four subintervals are $[0, \pi/4]$, $[\pi/4, \pi/2]$, $[\pi/2, 3\pi/4]$, $[3\pi/4, \pi]$. To compute the left sum approximation, we use the four left endpoints of our four subintervals:

```
deltax * (f[0] + f[Pi / 4] + f[Pi / 2] + f[3 Pi / 4])
```

$$\frac{1}{4}\left(1+\sqrt{2}\right)\pi$$

To draw the corresponding rectangles, you can use the `DrawLeftRiemannSum` command. It works just like the `LeftRiemannSum` command:

```
DrawLeftRiemannSum[Sin[x], {x, 0, Pi}, 4]
```



Computing R_n

To compute R_n, the Riemann sum with n subintervals using right end points, for the integral $\int_a^b f \, dx$, we use `RightRiemannSum[f, {x, a, b}, n]`. For example, to compute R_4 for the integral $\int_0^\pi \sin(x) \, dx$:

```
RightRiemannSum[Sin[x], {x, 0, Pi}, 4]
```

$$\frac{1}{4}\left(1+\sqrt{2}\right)\pi$$

Let’s think about for a moment how to compute this by hand so we can check our answer. First we define a, b, n, Δx and the integrand function $f(x)$:

```
a = 0;
b = Pi;
n = 4;
deltax = (b - a) / n;
f[x_] := Sin[x];
```

We then need to split the interval [0, π] in to 4 subintervals. To quickly do this we can use the `Range` command. `Range[start, end, dx]` gives all the numbers between `start` and `end` taking steps of `dx`. For example, to get all the numbers between 0 and π taking steps of $\frac{\pi}{4}$ (which is our Δx), we do:

```
Range[0, Pi, deltax]
```

$$\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi\right\}$$

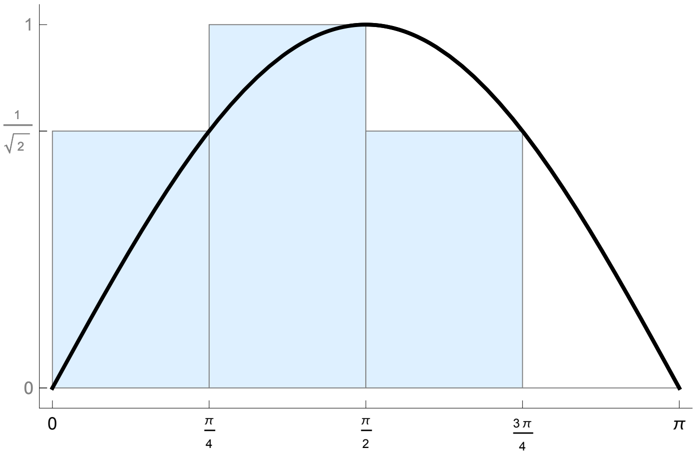
From the above we see the four subintervals are $[0, \pi/4]$, $[\pi/4, \pi/2]$, $[\pi/2, 3\pi/4]$, $[3\pi/4, \pi]$. To compute the right sum approximation, we use the four right endpoints of our four subintervals:

```
deltax * (f[Pi / 4] + f[Pi / 2] + f[3 Pi / 4] + f[Pi])
```

$$\frac{1}{4}\left(1+\sqrt{2}\right)\pi$$

To draw the corresponding rectangles, you can use the `DrawRightRiemannSum` command. It works just like the `RightRiemannSum` command:

```
DrawRightRiemannSum[Sin[x], {x, 0, Pi}, 4]
```



Computing M_n

To compute M_n, the Riemann sum with n subintervals using midpoints, for the integral $\int_a^b f\,\text{dx}$, we use `MidpointRiemannSum[f, {x, a, b}, n]`. For example, to compute M_4 for the integral $\int_0^\pi \sin(x)\,\text{dx}$:

```
MidpointRiemannSum[Sin[x], {x, 0, Pi}, 4]
```

$$\frac{1}{4}\pi\left(2\,\text{Cos}\left[\frac{\pi}{8}\right]+2\,\text{Sin}\left[\frac{\pi}{8}\right]\right)$$

Let’s think about for a moment how to compute this by hand so we can check our answer. First we define a, b, n, Δx and the integrand function $f(x)$:

```
a = 0;
b = Pi;
n = 4;
deltax = (b - a) / n;
f[x_] := Sin[x];
```

We then need to split the interval [0, π] in to 4 subintervals. To quickly do this we can use the `Range` command. `Range[start, end, dx]` gives all the numbers between `start` and `end` taking steps of `dx`. For example, to get all the numbers between 0 and π taking steps of $\frac{\pi}{4}$ (which is our Δx), we do:

```
Range[0, Pi, deltax]
```

$$\left\{0,\frac{\pi}{4},\frac{\pi}{2},\frac{3\pi}{4},\pi\right\}$$

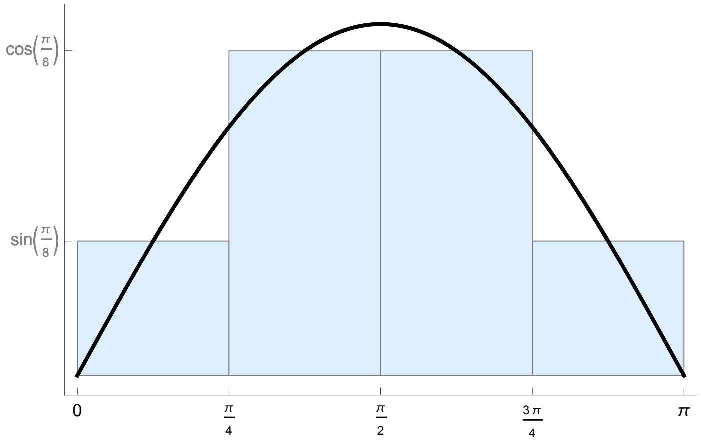
From the above we see the four subintervals are $[0, \pi/4]$, $[\pi/4, \pi/2]$, $[\pi/2, 3\pi/4]$, $[3\pi/4, \pi]$. To compute the midpoint sum approximation, we use the four midpoints of our four subintervals. I will label these as `midpoint1`, `midpoint2`, `midpoint3`, and `midpoint4` for ease of presentation:

```
midpoint1 = (0 + Pi / 4) / 2;
midpoint2 = (Pi / 4 + Pi / 2) / 2;
midpoint3 = (Pi / 2 + 3 Pi / 4) / 2;
midpoint4 = (3 Pi / 4 + Pi) / 2;
deltax * (f[midpoint1] + f[midpoint2] + f[midpoint3] + f[midpoint4])
```

$$\frac{1}{4}\pi\left(2\,\mathrm{Cos}\left[\frac{\pi}{8}\right]+2\,\mathrm{Sin}\left[\frac{\pi}{8}\right]\right)$$

To draw the corresponding rectangles, you can use the **DrawMidpointRiemannSum** command. It works just like the **MidpointRiemannSum** command:

```
DrawMidpointRiemannSum[Sin[x], {x, 0, Pi}, 4]
```



Computing T_n

To compute T_n, the approximiation to the integral $\int_a^b f\,dx$ using n trapezoids, we use **TrapezoidSum[f, {x, a, b}, n]**. For example, to compute T_4 for the integral $\int_0^\pi \sin(x)\,dx$:

```
TrapezoidSum[Sin[x], {x, 0, Pi}, 4]
```

$$\frac{1}{8}\left(2+2\sqrt{2}\right)\pi$$

Let’s think about for a moment how to compute this by hand so we can check our answer. First we define a, b, n, Δx and the integrand function $f(x)$:

```
a = 0;
b = Pi;
n = 4;
deltax = (b - a) / n;
f[x_] := Sin[x];
```

We then need to split the interval [0, π] in to 4 subintervals. To quickly do this we can use the **Range** command. **Range[start, end, dx]** gives all the numbers between **start** and **end** taking steps of **dx**. For example, to get all the numbers between 0 and π taking steps of $\frac{\pi}{4}$ (which is our Δx), we do:

```
Range[0, Pi, deltax]
```

$$\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\,\pi}{4}, \pi\right\}$$

From the above we see the four subintervals are $[0, \pi/4]$, $[\pi/4, \pi/2]$, $[\pi/2, 3\pi/4]$, $[3\pi/4, \pi]$. To compute the trapezoid sum approximation, we calculate the area of each trapezoid using the endpoints of each subinterval:

```
areaOfTrapezoid1 = (1 / 2) * deltax * (f[0] + f[Pi / 4]);
areaOfTrapezoid2 = (1 / 2) * deltax * (f[Pi / 4] + f[Pi / 2]);
areaOfTrapezoid3 = (1 / 2) * deltax * (f[Pi / 2] + f[3 Pi / 4]);
areaOfTrapezoid4 = (1 / 2) * deltax * (f[3 Pi / 4] + f[Pi]);
areaOfTrapezoid1 + areaOfTrapezoid2 + areaOfTrapezoid3 + areaOfTrapezoid4
```

$$\frac{\pi}{4\sqrt{2}} + \frac{1}{4}\left(1 + \frac{1}{\sqrt{2}}\right)\pi$$

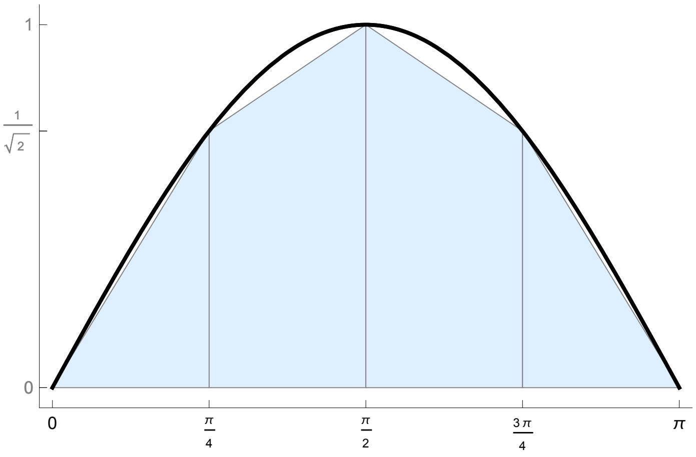
Alternatively, we can use the trapezoid rule formula where the coefficient order is {1, 2, ..., 2, 1}:

```
deltax / 2 * (f[0] + 2 * f[Pi / 4] + 2 * f[Pi / 2] + 2 * f[3 Pi / 4] + f[Pi])
```

$$\frac{1}{8}\left(2 + 2\sqrt{2}\right)\pi$$

These answers are identical although they are written differently. You can check by using the **N** command to compute a decimal approximation. To draw the corresponding trapezoids, you can use the **DrawTrapezoidSum** command. It works just like the **TrapezoidSum** command:

```
DrawTrapezoidSum[Sin[x], {x, 0, Pi}, 4]
```



Computing S_n

To compute S_n, the approximiation to the integral $\int_a^b f\,dx$ using Simpson's rule with n subintervals (where n is even), we use **SimpsonSum[f, {x, a, b}, n]**. For example, to compute S_4 for the integral $\int_0^{\pi} \sin(x)\,dx$:

```
SimpsonSum[Sin[x], {x, 0, Pi}, 4]
```

$$\frac{1}{12}\left(2+4\sqrt{2}\right)\pi$$

Let’s think about for a moment how to compute this by hand so we can check our answer. First we define *a*, *b*, *n*, Δx and the integrand function *f*(*x*):

```
a = 0;
b = Pi;
n = 4;
deltax = (b - a) / n;
f[x_] := Sin[x];
```

We then need to split the interval [0, π] in to 4 subintervals. To quickly do this we can use the **`Range`** command. **`Range[start, end, dx]`** gives all the numbers between **`start`** and **`end`** taking steps of **`dx`**. For example, to get all the numbers between 0 and π taking steps of $\frac{\pi}{4}$ (which is our Δx), we do:

```
Range[0, Pi, deltax]
```

$$\left\{0, \frac{\pi}{4}, \frac{\pi}{2}, \frac{3\pi}{4}, \pi\right\}$$

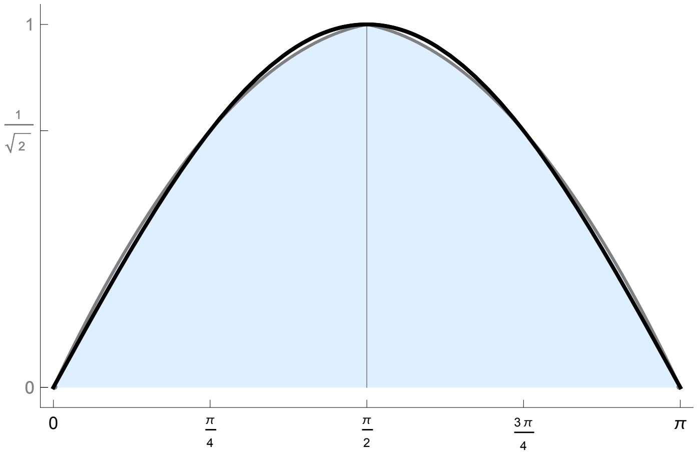
From the above we see the four subintervals are $[0, \pi/4]$, $[\pi/4, \pi/2]$, $[\pi/2, 3\pi/4]$, $[3\pi/4, \pi]$. To compute the Simpson sum approximation, we use the Simpson’s rule formula where the coefficient order is {1, 4, 2, 4, ..., 4, 2, 4, 1}:

```
deltax / 3 * (f[0] + 4 * f[Pi / 4] + 2 * f[Pi / 2] + 4 * f[3 Pi / 4] + f[Pi])
```

$$\frac{1}{12}\left(4+2\sqrt{2}\right)\pi$$

To draw the corresponding approximating parabolas, you can use the **`DrawSimpsonSum`** command. It works just like the **`SimpsonSum`** command:

```
DrawSimpsonSum[Sin[x], {x, 0, Pi}, 4]
```



Computing Error

To compute the (absolute) error, we compute error = | exact – approximate |. For example, to compute

E_{T_4}, the error in using the trapezoid sum approximation for the integral $\int_0^{\pi} \sin(x)\, dx$:

```
exact = Integrate[Sin[x], {x, 0, Pi}];
approximate = TrapezoidSum[Sin[x], {x, 0, Pi}, 4];
error = N[Abs[exact - approximate]]
0.103881
```

Note that we used the N command to get a decimal approximation of the error. As we increase *n*, we expect the error to decrease.

```
exact = Integrate[Sin[x], {x, 0, Pi}];
approximate = TrapezoidSum[Sin[x], {x, 0, Pi}, 100];
error = N[Abs[exact - approximate]]
0.000164496
```

Problems

We will observe that *Mathematica* helps us perform these approximate integrations to high degees of accuracy.

(a) Compute the left Riemann sum approximation using 10 rectangles for $\int_0^1 \cos(x^2)\, dx$. Plot the approximating rectangles.

(b) Compute the right Riemann sum approximation using 10 rectangles for $\int_0^1 \cos(x^2)\, dx$. Plot the approximating rectangles.

(c) Compute the midpoint Riemann sum approximation using 10 rectangles for $\int_0^1 \cos(x^2)\, dx$. Plot the approximating rectangles.

(d) Compute the trapezoid sum approximation using 10 rectangles for $\int_0^1 \cos(x^2)\, dx$. Plot the approximating trapezoids.

(e) Compute the Simpson sum approximation using 10 rectangles for $\int_0^1 \cos(x^2)\, dx$. Plot the approximating parabolas.

(f) Estimate the absolute error in approximation for part (a). Is this an overestimation or an underestimation?

Answer:

(g) Estimate the absolute error in approximation for part (b). Is this an overestimation or an underestimation?

Answer:

(h) Estimate the absolute error in approximation for part (c). Is this an overestimation or an underestimation?

Answer:

(i) Estimate the absolute error in approximation for part (d). Is this an overestimation or an underestimation?

Answer:

(j) Estimate the absolute error in approximation for part (e). Is this an overestimation or an underestimation?

Answer:

(k) Use Theorem 5.20 (pp. 351) to determine a value of n such that the Trapezoidal Rule will aproximate the value of $\int_0^1 \cos(x^2)\,\mathrm{dx}$ with an error that is less than 0.000001. Use the fact that $|f''(x)| \leq 6$.

Answer:

(l) What is the smallest value of n such that the Trapezoidal Rule will aproximate the value of $\int_0^1 \cos(x^2)\,\mathrm{dx}$ with an error that is less than 0.000001?

Answer: