02-06-Integration

September 1, 2018

1 Introduction to Integration

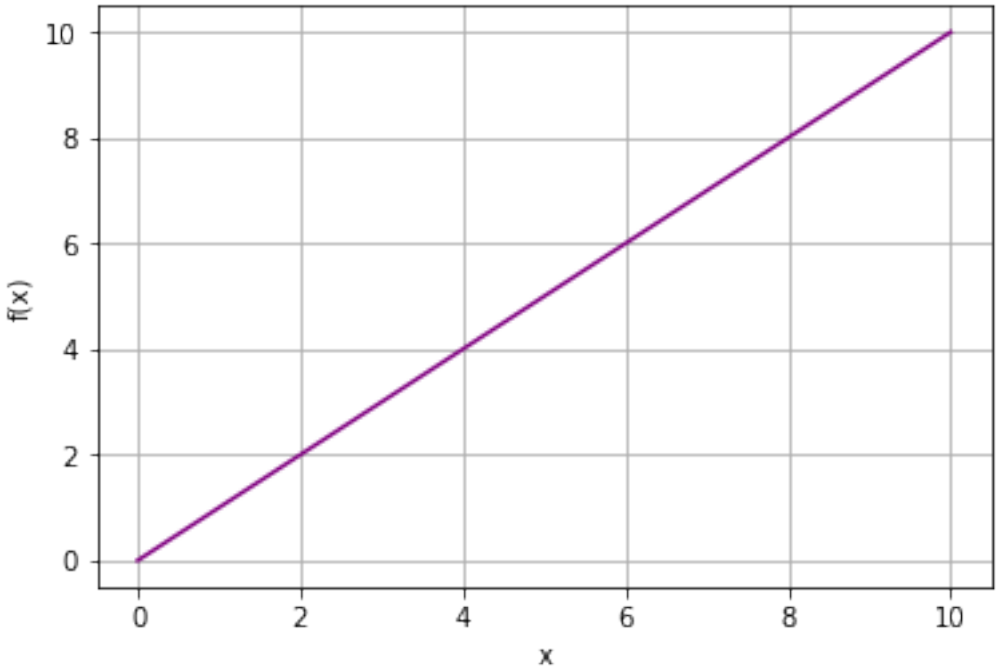
Integrals are the inverses of derivatives. More importantly, using integration provides a way to compute the area under the curve of most any function. There are many applications for integration. For example, if you need to compute a probability of some occurance between limits (which we'll discuss later in this course), then you will use an integral.

Let's start with a simple function:

$$f(x) = x$$

We can plot this function as a line. Run the code below to plot the function for the range 0 to 10:

```
In [5]: import numpy as np
        from matplotlib import pyplot as plt
        %matplotlib inline

        # Define function f
        def f(x):
            return x

        # Create an array of x values from 0 to 10
        x = range(0, 11)

        # Get the corresponding y values from the function
        y = [f(a) for a in x]

        # Set up the plot
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.grid()

        # Plot x against f(x)
        plt.plot(x,y, color='purple')

        plt.show()
```



1.1 Performing Integration

The *integral* of a function is the area under it - in this case, the area under the purple diagonal line down to the x-axis.

So how do you find the integral of a function? well, for our simple function $f(x) = x$, the formula for an integral is written as follows:

$$\int f(x)\, dx$$

The $\int$ symbol shows that this formula is an integral. The dx indicates that the integration is with respect to the x variable. Note that since $f(x) = x$, we could also write this integral formula as $\int x\, dx$

So, what is the integral of $x\, dx$? To answer this question, we need the *antiderivative* of $\boldsymbol{f}$ - in other words we need to find a function which has a derivative matching the output of f, which is just x. Using the power rule in reverse, a function that has the derivative x would be $\frac{1}{2}x^2$

So, the *unbound* integral formula for f with respect to x can be written as:

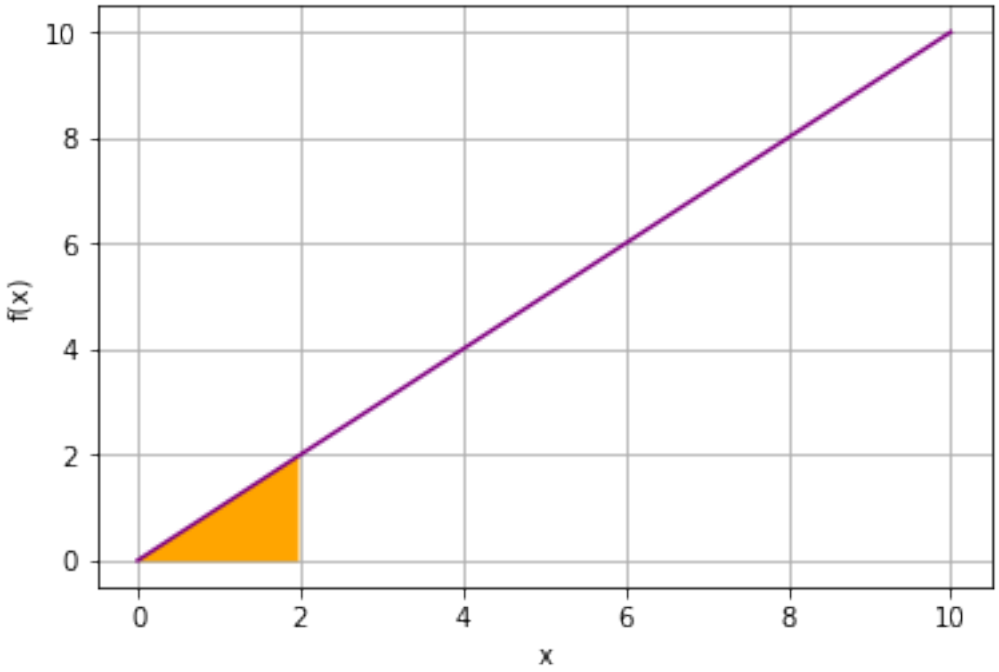
$$\int f(x)\, dx = \frac{1}{2}x^2$$

1.2 Integration between Limits

Now that we have the unbound integral formula, we can use it to find the integral between specific start and end points. Let's suppose we want to find the area under the function between the x values 0 and 2. In other words, the *integral* of f for the range 0 to 2 with respect to x.

Run the following code to re-plot the function and show the area we're interested in:

```
In [6]: # Set up the plot
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.grid()

        # Plot x against f(x)
        plt.plot(x,y, color='purple')

        # show area for integral
        section = np.arange(0, 2, 1/20)
        plt.fill_between(section,f(section), color='orange')

        plt.show()
```



We call the start and end point the **limits** of the integral. The lower limit is placed as a subscript of the integral sign. The upper limit is placed as a superscript of the integral sign. Using this notation the integral of $f(x)$ from 0 to 2 is written as follows:

$$\int_0^2 f(x)\, dx$$

The integral is evaluated by subtracting the value of the integrand at the lower limit from the integrand at the upper limit; and since we know the formula based on our antiderivative function, the integral can be evaluated in the following manner.

$$\int_0^2 f(x)\,dx = \frac{1}{2}x^2\,|_0^2 = \frac{1}{2}2^2 - \frac{1}{2}0^2 = \frac{4}{2} - \frac{0}{2}x^2 = 2$$

Execute the code in the cell below and verify that the result returned by the ***scipy.integrate.quad*** function in Python is approximately the same as we computed analytically.

```
In [2]: import scipy.integrate as integrate
        i, e = integrate.quad(lambda x: f(x), 0, 2)
        print (i)
```

```
2.0
```

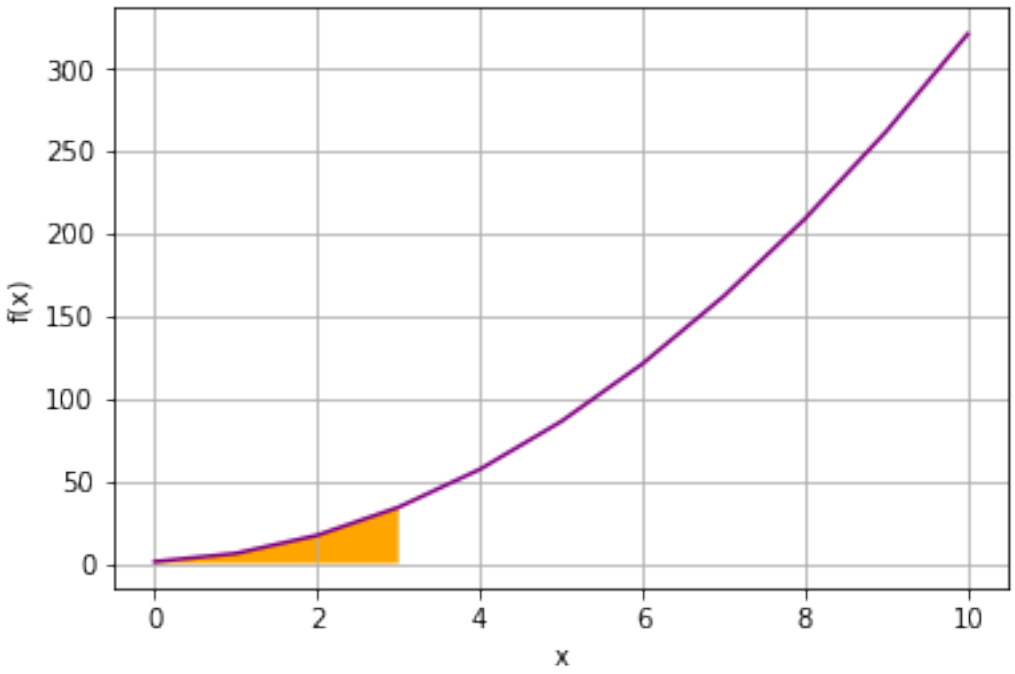
1.3 Another Integral

Here is another example for a slightly more complex function. What is the area under the curve of the function $3x^2 + 2x + 1$ between 0 and 3?

let's look at that function and the area in question:

```
In [3]: import numpy as np
        from matplotlib import pyplot as plt
        from matplotlib.patches import Polygon
        %matplotlib inline

        # Define function g
        def g(x):
            return 3 * x**2 + 2 * x + 1

        # Create an array of x values from 0 to 10
        x = range(0, 11)

        # Get the corresponding y values from the function
        y = [g(a) for a in x]

        # Set up the plot
        fig, ax = plt.subplots()
        plt.xlabel('x')
        plt.ylabel('f(x)')
        plt.grid()

        # Plot x against g(x)
        plt.plot(x,y, color='purple')

        # Make the shaded region
        ix = np.linspace(0, 3)
        iy = g(ix)
        verts = [(0, 0)] + list(zip(ix, iy)) + [(3, 0)]
        poly = Polygon(verts, facecolor='orange')
```

```
ax.add_patch(poly)

plt.show()
```



We can evaluate this integral just as before, this time using function:

$$\int_0^3 3x^2 + 2x + 1 \, dx$$

We can calculate the antiderivative of $3x^2 + 2x + 1 \, dx$ as $\frac{3}{3}x^3 + \frac{2}{2}x^2 + x$, so:

$$\int_0^3 = \frac{3}{3}x^3 + \frac{2}{2}x^2 + x \,|_0^3 = \frac{3}{3}3^3 + \frac{2}{2}3^2 + 3 - \frac{3}{3}0^3 - \frac{2}{2}0^2 + 0 = 27 + 9 + 3 + 0 + 0 + 0 = 39$$

Now, execute the code in the cell below to verify the result:

```
In [4]: i, e = integrate.quad(lambda x: 3 * x**2 + 2 * x + 1, 0, 3)
        print(i)
```

```
38.99999999999999
```

Note that the result from the ***scipy.integrate.quad*** function is approximate - the function actually returns an estimated integral (*i* in this case) and also a measure of absolute error (*e*). Run the following code to see what the absolute error was in this case:

```
In [5]: print(e)
```

```
4.3298697960381095e-13
```

The absolute error in this case is extremely small (around 4.3×10^{-13}).

1.4 Infinite limits

In many cases the limits of an integral can be $+/-\infty$. Perhaps suprisingly, this situation is not really a problem if the function being integrated converges to 0 at the infinite limit.

Here is an example. The function $e^{-5x} \to 0$ as $x \to \infty$. Therefore, the integral of this function from some limit to ∞. This integral can be written as follows:

$$\int_0^\infty e^{-5x} dx$$

The code in the cell below computes this integral numerically.

```
In [6]: import numpy as np
        i, e = integrate.quad(lambda x: np.exp(-x*5), 0, np.inf)

        print('Integral: ' + str(i))
        print('Absolute Error: ' + str(e))
```

```
Integral: 0.20000000000000004
Absolute Error: 1.5606666959324066e-11
```

This integral converges to a small number with a much smaller error estimate.

Here is another example that illustrates why having infinite integration limits is so useful. When computing probabilities it is often necessary to have infinite limits. Don't worry too much about the details of probability theory. This is covered in a later lesson.

A Normal distribution with zero mean and a standard deviation of 1 has the following density function:

$$\frac{1}{2\pi} e^{\frac{-x^2}{\sqrt{(2\pi)}}}$$

It makes sense that the integral of this probability density function from $-\infty$ to ∞ must be 1.0. In other words the probability of a Normally distributed event ocurring at all possible values must be 1.0.

The code in the cell below computes the following integral:

$$\int_{-\infty}^\infty \frac{1}{2\pi} e^{\frac{-x^2}{\sqrt{(2\pi)}}} dx$$

Execute this code and verify that the result is approximately 1.0.

```
In [7]: import numpy as np
        norms = lambda x: np.exp(-x**2/2.0)/np.sqrt(2.0 * 3.14159)
        i, e = integrate.quad(norms, -np.inf, np.inf)

        print('Integral: ' + str(i))
        print('Absolute Error: ' + str(e))
```

```
Integral: 1.0000004223321999
Absolute Error: 1.0178195695613028e-08
```