# Numerical Integration

In this lesson, we will learn about single and multiple integrals using numerical integration.

Numerically integrating a function is a common engineering task. The SciPy module provides a number of functions for integration. A general-purpose tool used to solve integrals like this"

$$I = \int_a^b f(x)dx$$

is the `quad()` function. SciPy provides a series of functions for different kinds of integrations, for example, `dblquad` for double integration and `tplquad` for triple integration.

We use the following command to import the relevant functions from the integrate package:

```
from scipy.integrate import quad, dblquad, tplquad
```

💡 The `quad` function takes a large number of optional arguments that can be seen using the `help(quad)` command.

The input arguments of the `quad()` function include the integrand `f(x)` and the limits of integration `a` and `b`. `quad()` returns a tuple with the first value being the numerical result and the second being the estimation of the numerical error in the result.

```
res, err = quad(f, a, b)
```

## Single integration #

Let's learn about the basic usage of the `quad()` function by solving a basic integration problem:

$$I = \int_0^{\pi/2} 2sin(x)cos(x)\ dx$$

```
from scipy import *
from scipy.integrate import quad

# function to be integrated
def f(x):
    return (2 * sin(x) * cos(x))

# call quad to integrate the function f from 0 to pi/2
val, err = quad(f, 0, pi/2)
print("Value of integral:", val)
print("Error in integral:", err)
```

The answer we retrieved from the `quad()` function is quite close to the actual answer, which is 1.

For simple functions, we can use a lambda function instead of explicitly defining a function for the integrand. Let's look at an example of this below:

```
from scipy import *
from scipy.integrate import quad

# call quad to integrate the function f from -Inf to Inf
val, err = quad(lambda x: exp(-x**2) , -Inf, Inf)
print("Value of integral:", val)
print("Error in integral:", err)
```

In the SciPy namespace, we use `Inf` instead of `oo` to refer to infinity.

## Multiple integration #

Multiple integration works in the same way as single integration. Let’s look at some examples of double and triple integration.

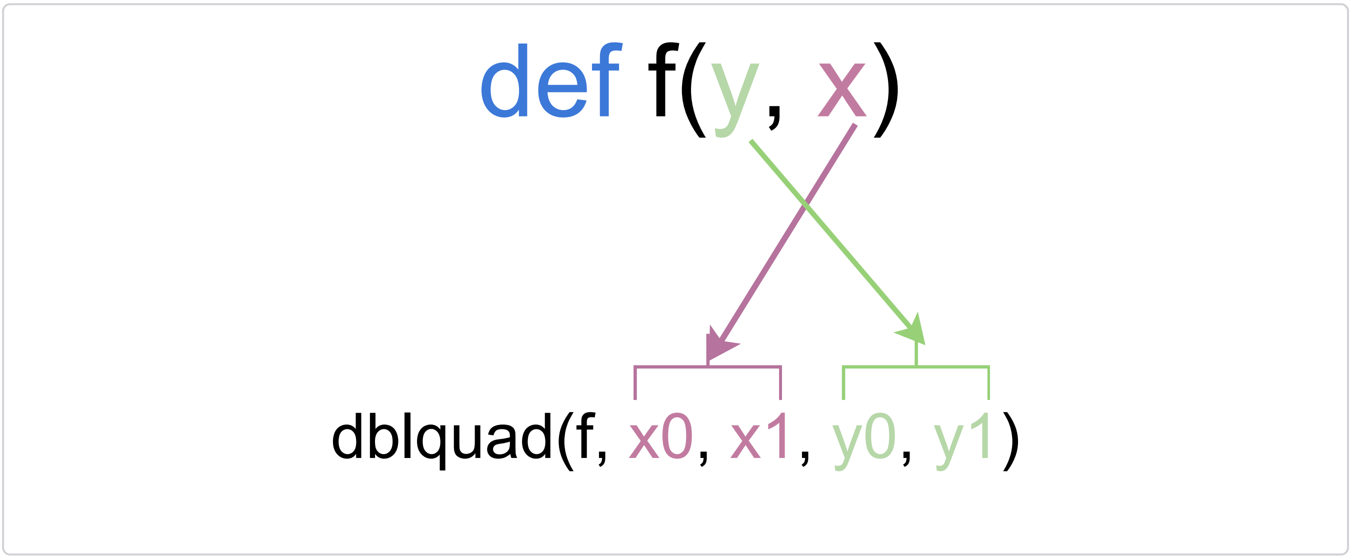
> 💡 For `dblquad()` and `tplquad()`, the integration limits of all inner integrals need to be defined as functions.

### Double integration #

We will use the `dblquad()` function to compute the double integral.

$$I = \int_0^2 \int_0^3 x^2 y \, dx dy$$

One important thing to notice is the integration limits. The order of arguments for `f` must correctly correspond to the order of the integration in `dblquad()`. This is given below:



Let’s look at an example of double integration below:

```
from scipy import *
from scipy.integrate import quad, dblquad

def f(y, x):
  return x**2 * y

# call dblquad to integrate the function f
# x goes from 0 to 3
# y goes from 0 to 2
val, err = dblquad(f, 0, 3, lambda x: 0, lambda x: 2)
```

```
print("Actual Value:", 18)      # calculated by hand

print("Value of integral:", val)
print("Error in integral:", err)
```

For the integration of `y`, we passed lambda functions as the limits, since these, in general, can be functions of `x`. As you can see, the value of the actual integral and the integral computed by `dblquad()` has a minimal margin of error, which is the order of $10^{-13}$.

## Triple integration

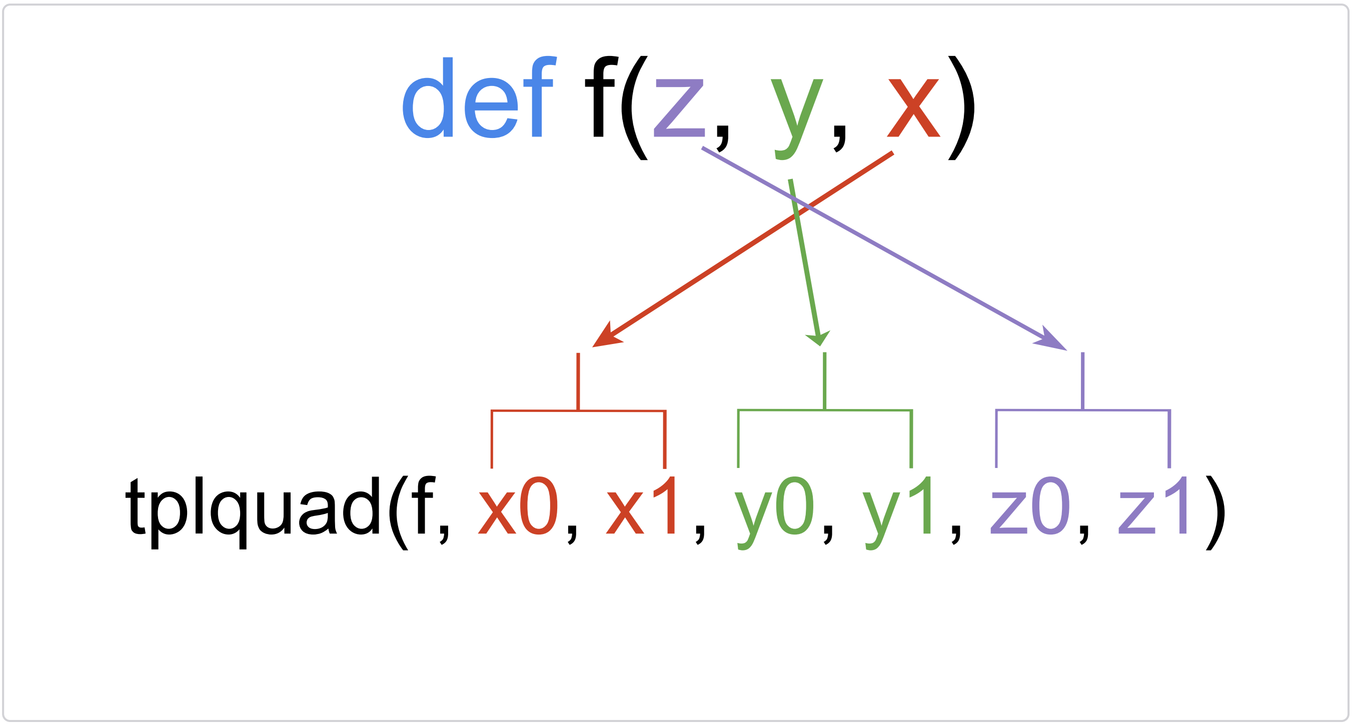
We will use the `tplquad()` function to compute the triple integral. Let's use an interesting example: the volume of a sphere. The integral for the volume of sphere is:

$$V = \int_0^{2\pi} \int_0^{\pi} \int_0^{R} r^2 sin\theta \; dr d\theta d\phi$$

where $R$ is the radius of the sphere.

> 💡 **Important Note:** The documentation of `tplquad()` states that the integration limits of the inner integrals should be provided as functions of the outer integration variables even if they happen to be constants. This makes sure that the limits are always written in general terms.

The order of the arguments in the definition of the function should correspond to the order of integrations.



Let’s compute the volume for a sphere with a radius 3:

```
from scipy import *
from scipy.integrate import quad, dblquad, tplquad

def f(phi, theta, r):
  return (r**2 * sin(theta))

# call dblquad to integrate the function f
# r goes from 0 to 3
# theta goes from 0 to pi
# phi goes from 0 to 2pi

val, err = tplquad(f, 0, 3, lambda r: 0, lambda r: pi, lambda r, theta: 0, lambda r, theta: 2

print("Actual Volume:", (pi) * (3**3) * (4 / 3)) # computed by formula
print("Value of integral:", val)
print("Error in integral:", err)
```

- We passed lambda functions for the limits of the `theta` integration, since these, in general, can be functions of `r`.
- We passed lambda functions for the limits of the `phi` integration, since these, in general, can be functions of `r` and `theta`.

As you can see, the value of the actual volume and the volume computed by the `tplquad()` has a minimal error, which is the order of $10^{-12}$.

In the next lesson, we will learn about interpolation.