

Assignment 3 (Due Friday 11th October):

Task: Extend your code from Lab 7, Q8 to integrate a function of three variables over a volume.

Details:

You should create a function `monty3d` that can be called in the following way (see **testfile1.py**):

```
7
8 import myquad as myq
9
10 def region(x,y,z):
11     return x**2+y**2+z**2<1**2
12
13 def f(x,y,z):
14     f = 1
15     return f
16
17 a,b = -1,1
18 M = 20
19 I = myq.monty3d(f,region,a,b,M)
20
21 print(I)
```

The function **monty3d** should therefore be contained within your own `myquad.py` module.

The inputs to `monty3d` are:

<code>f(x,y,z)</code>	a floating point function of three variables
<code>region(x,y,z)</code>	a function that specifies the integration region, returning “1” if the point (x,y,z) is in the integration region, and “0” otherwise, to
<code>a,b:</code>	the box in which the Monte-Carlo sampling takes place. For 3D the sample points should lie within a rectangular prism $x \in [a, b]$, $y \in [a, b]$, $z \in [a, b]$.
<code>M:</code>	the number of sample points for the Monty-Carlo method, given by 2^M .

The output should be a single real number that returns the value of the integral (see **output1.txt**):

```
In [2]: run testfile1
4.189048767089844
```

Important notes:

The module must work on the first try, and for an arbitrary function and region as an input. If it does not then marks will be deducted.

Try to make your code as robust as possible so that it does not miss any zeros. You may have to modify your bracketing procedure to do this.

Files to submit:

You should upload (only) the following files:

`myquad.py` - a python module that contains the **monty3d** function, as well as any other functions that it needs to run

Grading:

The code will be graded according to the following scale:

Compliance: This is whether the code is submitted as instructed	30%
Effectiveness (i.e. whether the code passes independent tests):	70%