COLORADO COLLEGE

Colorado College

MA418 Numerical Analysis

Investigating Numerical Integration

Sohair Abdullah, Hanbo Shao

I have neither given nor received any unauthorized aid in this project.

Signature:

May 16, 2018

1. Introduction

In this project, we start by investigating the visualization of double integral into a 3D graph. This approximation of the volume integral function is then carried onto polar coordinates to see if it gives a better approximation for circles. We also compare Gaussian Double Integral and Simpson's Double Integral. At the end, we applied Gaussian Triple Integral to solve triple integral problems.

2. Theory (Visualization)

The purpose of this section is to visualize the volume integral function of functions that cannot be computed analytically. Let's start with the simple one dimensional case, and we will then generalize the model to two dimensions. The integral shown below cannot be computed analytically:

$$\Gamma(x) = \int_0^x t^{4.1} e^{-t} dt. \tag{2.1}$$

Using Simpson's method, however, we can visualize the graph of $\Gamma(x)$. The graph of this integral function is shown below in figure 1.

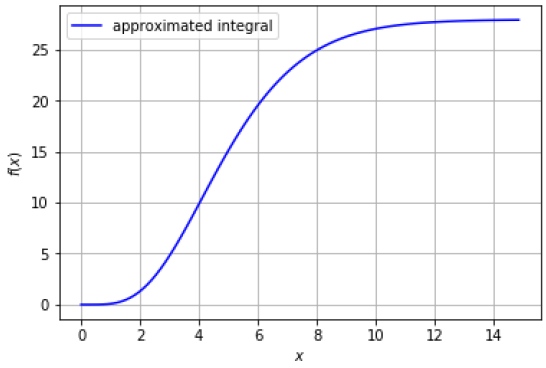


Figure 1. Approximated solution to Euler's Gamma Function (Equation (2.1))

However, now we would like to generalize our model to a volume integral of the form:

$$\int_{x_0}^{x_1} \int_{y_0}^{y_1} f(x, y)\ dydx. \tag{2.2}$$

The way we approach this integral is using a Simpson's double method. Simpson's double algorithm is designed to give an approximation to the integral, to an arbitrary high number of accuracy. It takes in the parameters of the start point of the integral, end point of the integral, and the number of intervals. Naturally, increasing the number of intervals decreases the step size at which point the function is evaluated. The number of intervals are two in Simpson's double, one for the x domain, and one for the y domain. For a start, let's take a simplest example to solve an integral of the form given by:

$$\int_0^1 \int_0^1 (x + y)\ dydx. \tag{2.3}$$

The purpose of this section is to plot the graph of the resulting function given by:

$$F(p, q) = \int_0^p \int_0^q (x + y)\ dydx. \tag{2.4}$$

where $0 \leq p \leq 1$ and $0 \leq q \leq 1$. How does our numerical approximation algorithm achieve this result?

First step: We first recognize the domain. The domain in this specific case is rectangular: $[0, 1] \times [0, 1]$.

Second Step: We then fix the number of intervals we want our domain to be divided into. Let's say the number of intervals is (20, 20). This means that our x coordinates will be

divided into 20 parts from 0 to 1. Similarly, our y coordinates will be divided into 20 parts also, from 0 to 1. Therefore, the step sizes for the x and y domains are given by:

$$h_x = h_y = \frac{b-a}{n} = \frac{1}{20} = 0.05. \tag{2.5}$$

The domain now has been divided into what we call mesh points, given by $x = [0, 0.05, 0.1, ..., 1]$ and $y = [0, 0.05, 0.1, ..., 1]$.

Third step: Apply Simpson's double integral approximation to the area generated by every mesh point. This is a little more complicated than it sounds. What the algorithm does in this step is the following: it takes a step in the x direction first. So, x goes from 0 to 0.05. This x is fixed for a while now. Then, it takes a step in the y direction. Therefore, y goes from 0 to 0.05. We are at the point $[0.05, 0.05]$ on our domain right now. Then, what our algorithm does next is that it computes, using Simpson's double rule, the value of the integral given by:

$$\int_0^{0.05} \int_0^{0.05} (x+y)\, dydx. \tag{2.6}$$

The next thing the algorithm will do is that it will take a step further in the y direction. Therefore, our y goes from $y = 0.05$ to $y = 0.1$ now. Note that this time there will be no step taken in the x direction. It is fixed at $x = 0.05$. And, the algorithm will compute the value of the integral given by:

$$\int_0^{0.05} \int_0^{0.1} (x+y)\, dydx. \tag{2.7}$$

Note something amazing: we have the value of the function $F(p, q)$ at two different mesh points, namely: [0.05, 0.05] and [0.05, 0.1]. The algorithm will keep on proceeding in the same manner, with $x = 0.05$ until we reach a value of $y = 1$. We are then going to take a step in the x direction, where x now goes from $x = 0.05$ to $x = 0.01$ and y will go back all the way to 0. Similarly, x will be fixed to $x = 0.1$ while y increments from $y = 0$ to $y = 1$, computing the value of the integral at every mesh point. The algorithm will move in a similar manner, until we have reached $x = 1$ and $y = 1$, at which point the algorithm stops and we have covered the domain over which we wanted our integral to be computed. Figures $1 - 4$ below show the procedure in a more geometric manner.

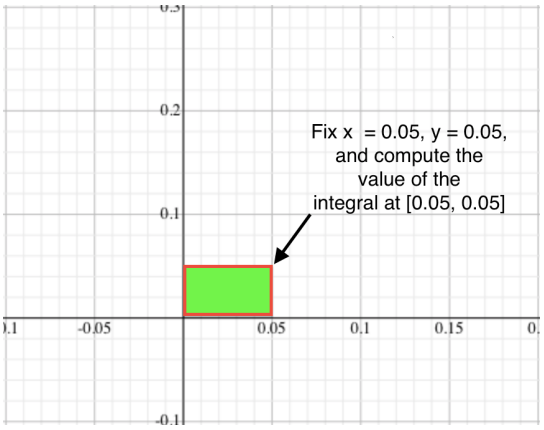


FIGURE 2. $x, y = 0.05$

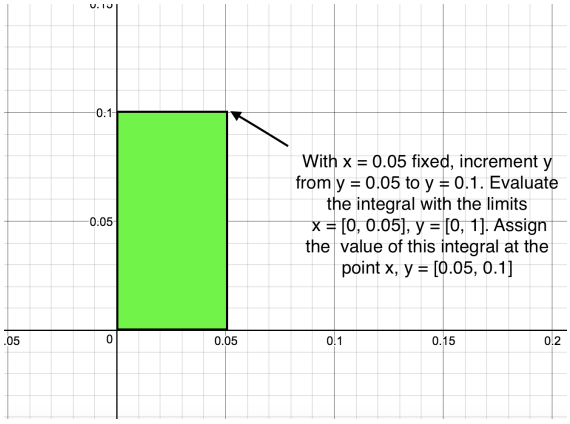


FIGURE 3. $x = 0.05$, y = 0.1

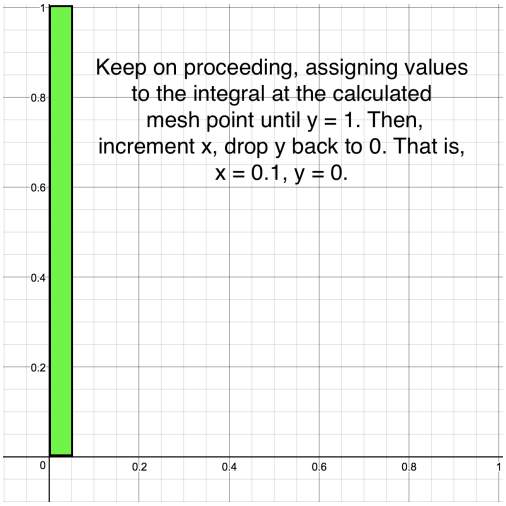


FIGURE 4. $x = 0.05$, $y = 0, 0.05, \ldots, 1$.

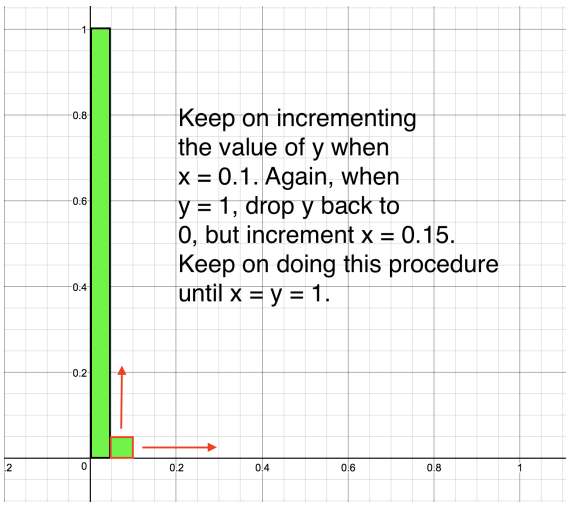


FIGURE 5. $x = 0.05, ..., 1$, $y = 0.05, \ldots, 1$

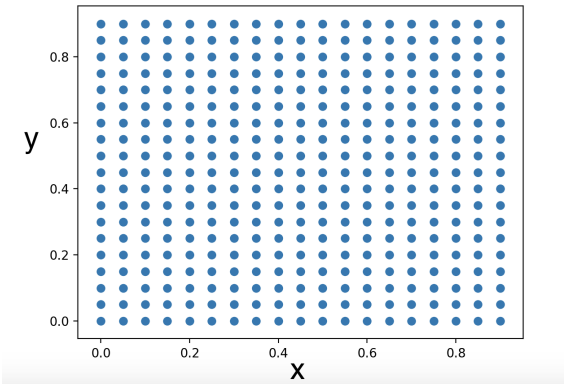


FIGURE 6. Algorithm stops when value of Simpson's double integral is computed at every mesh point, 0.05 points apart.

Figure 6 shows Simpson's double integrated over the entire domain. So at every mesh point, we will have the double integral evaluated. For example, at $[x, y] = [0.5, 0.5]$ we will have the Simpson's double approximate:

$$F(0.5, 0.5) = \int_0^{0.5} \int_0^{0.5} (x + y)\, dydx. \tag{2.8}$$

$F(0.5, 0.5)$ is the value of the function at $[x, y] = [0.5, 0.5]$, one point in the graph of our approximated volume function. Similarly, at every mesh point we have the value of the volume integral. As a result, we can plot the volume function because we know the value of $F(p, q)$ at a range of mesh points (p, q). The plot yielded by our algorithm is shown below in figure 7. Figure 8 shows the plot of the analytical solution given by $\frac{x^2y}{2} + \frac{xy^2}{2}$. Note: the complete analytical solution is $\frac{x^2y}{2} + \frac{xy^2}{2} + Cx$, where C is some constant. But, we have assumed this constant of integration to be 0.

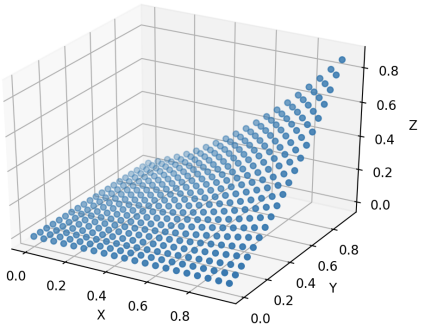


FIGURE 7. Approximation

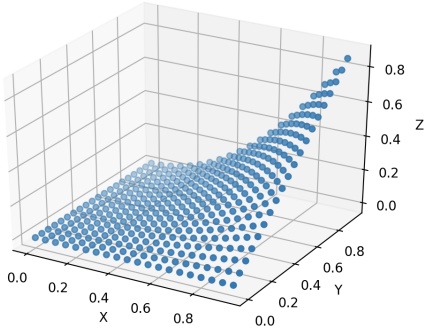


FIGURE 8. Analytical Solution

It can be seen that the plots look quite similar. On this rectangular domain, our algorithm has yielded fairly good results. Perhaps a table of errors, between actual and approximate values, at some selected mesh points will be even more illustrative. Table 1 illustrates these results below:

Table 1: Comparison Between Polar Coordinates and Cartesian Coordinates

mesh points	Approximate	Actual	Error
$(0.1, 0.1)$	0.00100	0.00100	0.00000
$(0.2, 0.2)$	0.00800	0.00800	0.00000
$(0.3, 0.3)$	0.02700	0.02700	0.00000
$(0.4, 0.4)$	0.06400	0.06400	0.00000
$(0.5, 0.5)$	0.12500	0.12500	0.00000

We see from this table that the error between the actual and the approximate solution is essentially zero. In fact, we found the values to differ only on the order of 10^{-17} decimal place. And, the total error, which is the total sum of the errors (absolute difference between the actual value and the approximated value at every mesh point), is 6.7×10^{-15}. The question is, we did not divide the grid into many mesh points (only 20 parts in 1 in each direction. Or, a unit square divided into only 400 smaller squares). Then how does the algorithm still yield us such good results? The answer to this question is rather subtle. As mentioned before, at every mesh point, a Simpson double is evaluated. However, the Simpson double also has to divide the grid further into many other points. So, for example, if we were trying to compute the value of our integral at $[x, y] = [0.05, 0.05]$, the Simpson double algorithm will 'see' this as a domain of $[0, 0.05] \times [0, 0.05]$, and will further divide it into $30 \times 30 = 900$ smaller squares (we put the number of intervals in our Simpson's algorithm to be $(30, 30)$), whose sum will give us a highly accurate value of the integral at $[x, y] = [0.05, 0.05]$. We now turn our attention to a different, non rectangular, domain. That is, we are going to apply polar coordinates to a circle domain in the next section.

3. Polar Coordinates

In calculus, for certain shapes such as circles, we tend to use polar coordinates to evaluate the integral. Let us take the unit circle as an example. If we stick to Cartesian coordinates, we will need to evaluate two parts, i.e. two half circles. Also, the upper bound for y is $\pm\sqrt{x^2 - 1}$, respectively and x goes from -1 to 1. When it comes to polar coordinates, we simply set the radius r to go from 0 to 1 and the angle θ from 0 to 2π. In this section, we are investigating the if using one coordinate system will be more accurate than the other for certain shapes such as circles, with the same number of grids.

Now consider the following example. We want to calculate $\iint_D (x^2 + y^2)^{-2} dA$, where D is the region described in Figure 9 ([1]).

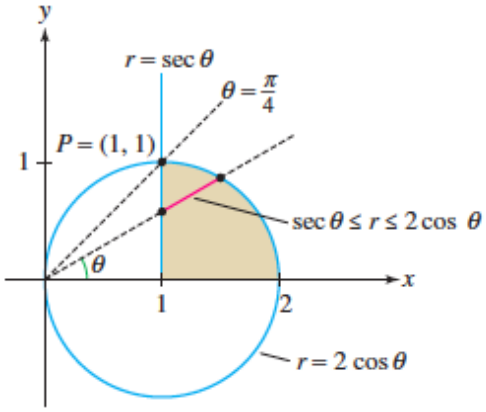


FIGURE 9. Polar Coordinates

As this is a circle centered at $x = 1$, the circle can then be represented by $(x-1)^2 + y^2 = 1$. The upper bound for y in the desired region would then become $\sqrt{1-(x-1)^2}$. If we use Cartesian coordinates, we are integrating

$$\int_1^2 \int_0^{\sqrt{1-(x-1)^2}} (x^2 + y^2)^{-2} dy dx, \tag{3.1}$$

if we integrate in polar coordinates, we are integrating

$$\int_0^{\frac{\pi}{4}} \int_{r=\sec\theta}^{2\cos\theta} r^{-3} dr d\theta. \tag{3.2}$$

As the mechanism behind two methods are essentially the same, so there is nearly no change when coding up the double integral using polar coordinates. Again, we are applying Simpson's Double method as it is so far the most accurate one we learned compared to Trapezoid method and Midpoint method. With the same number of grids for two methods (Cartesian coordinates and polar coordinates), we form the following table.

Table 2: Comparison Between Polar Coordinates and Cartesian Coordinates

n	Error (Polar)	Error (Cartesian)
10	7.2×10^{-6}	2.0×10^{-4}
100	7.4×10^{-10}	7.2×10^{-6}
500	1.2×10^{-12}	6.4×10^{-7}
1000	7.5×10^{-14}	2.3×10^{-7}
3000	9.2×10^{-16}	4.4×10^{-8}
5000	3.1×10^{-16}	2.0×10^{-8}
10000	1.9×10^{-16}	7.2×10^{-9}

From the table, we can observe that evaluating double integral using polar coordinates is always more accurate than the using the double integral in terms of Cartesian coordinates. However, when $n = 10000$, i.e. there exists one hundred million grids, a breakdown occurs. This is most probably due to computing limit caused by the hardware.

4. Approximating Double Integral: Simpson and Gauss

We learned to apply Simpson's Double Method to evaluate double integral in class. There is another method called Gaussian Double Integral method which is based on Gaussian quadrature ([2]). In this section, we will compare between these two methods.

Before moving into Gaussian Integral method, we need some background knowledge about Gaussian quadrature. In calculating Gaussian quadrature, we use a specific group of polynomials, Legendre polynomials, which is named after the famous French mathematician. The first few polynomials are $P_0(x) = 1$, $P_1(x) = x$, $P_2(x) = x^2 - \frac{1}{3}$, and so forth. For a Legendre polynomial of order n, $P_n(x)$. It has n roots, $x_1, \ldots, x_n$, by fundamental theorem of algebra. By theorem, we calculate the corresponding coefficients by $c_i = \displaystyle\int_{-1}^{1} \prod_{\substack{j=1 \\ j \neq i}}^{n} \frac{x - x_i}{x_i - x_j} dx$. The first few Gaussian quadrature is laid out in the following table where n stands for the order ([2]).

x

Table 3: First Few Terms of Gaussian Quadrature

n	Roots $r_{n,i}$	Coefficients $c_{n,i}$
2	-0.5773502691896257	1.0000000000000000
	0.5773502691896257	1.0000000000000000
3	0.0000000000000000	0.8888888888888888
	-0.7745966692414834	0.5555555555555556
	0.7745966692414834	0.5555555555555556
4	-0.3399810435848563	0.6521451548625461
	0.3399810435848563	0.6521451548625461
	-0.8611363115940526	0.3478548451374538
	0.8611363115940526	0.3478548451374538

The basic idea of Gaussian Double Integral method is to change the upper bound and lower bound of integration to be 1 and -1, respectively. We then apply quadrature rule to approximate the integral. For arbitrary interval, we perform the following calculation,

$$\int_a^b f(x)dx = \int_{-1}^{1} f\left(\frac{(b-a)t + (b+a)}{2}\right)\frac{(b-a)}{2}dt. \tag{4.1}$$

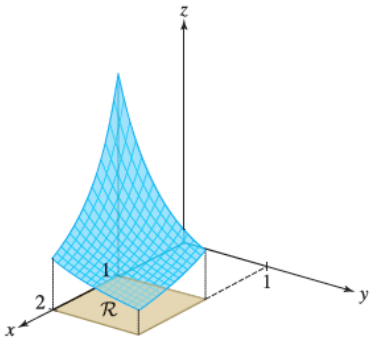
With the correct interval, we use the following formula to approximate the integral,

$$\int_{-1}^{1} P(x)dx = \sum_{i=1}^{n} c_i P(x_i), \tag{4.2}$$

where c_i and x_i corresponds to specific weight associated to a root in Table 3, respectively. Let's now consider the following integral

$$\iint_{\mathcal{R}} \frac{dxdy}{(x+y)^2}, \tag{4.3}$$

where $\mathcal{R} = [1,2] \times [0,1]$. This double integral is shown in the figure below [1]. Through calculation, we obtained that the result of this double integral is $\ln(\frac{4}{3})$.



$\frac{1}{(x+y)^2}$ over $\mathcal{R}$

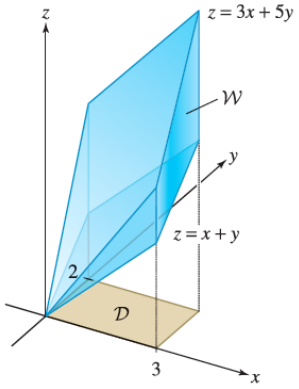
Table 4: Comparison Between Simpson and Gaussian Integral Method

n	error(Gaussian)	error(Simpson)
1	0.037682072451781	0.108669726772769
2	0.001525047658392	0.002503112733404
3	0.000049493897754	0.039809322528766
4	0.000001511841346	0.000202140564106
5	0.000000045271619	0.023412864485344
6	0.000000001345410	0.000042754312220
7	0.000000000039849	0.016514363451853
8	0.000000000001178	0.000013899605129
9	0.000000000000035	0.012743272142899
10	0.000000000000001	0.000005769449753

We can see from the table above that Gaussian Double Integral Method always gives a better approximation with same n. In other words, we need much bigger n for Simpson's Double Integral Method if we require same accuracy for both methods. Gaussian Integral method is similar to the way Romberg Method works. It converges to the solution much faster and thus reduce the number of function evaluations. However, calculating the roots and their corresponding weights for Gaussian quadrature takes a long time. Simpson's Double Integral method doesn't require some pre-processing as required in Gaussian Double Integral Method. As we increase n to be sufficiently, we are still able to obtain a fairly close approximation.

5. Triple Integral

After investigating Gaussian Double Integral in the last section, it is natural to extend the code to evaluate Triple Integral. This comes with the cost of adding a for loop in an already nested for loop. We consider the following triple integral in this section: $\iiint_{\mathcal{W}} z dV$, where $\mathcal{W}$ is the region between the planes $z = x + y$ and $z = 3x + 5y$ over the rectangle $[0, 3] \times [0, 2]$. A graph demonstration is shown below [1] and a table showing the error of Gaussian Triple Integral approximation is on the right.



Graph of $\mathcal{W}$

Table 5: Error (Gaussian Triple Integral)

n	error(Gaussian)
1	0.16666666666666674068
2	0.00000000000000022204
3	0.00000000000000022204
4	0.00000000000000066613
5	0.00000000000000000000
6	0.00000000000000022204
7	0.00000000000000088818
8	0.00000000000000022204
9	0.00000000000000000000
10	0.00000000000000022204

From the table, we can clearly how powerful the Gaussian Triple Integral is. Even at $n = 2$, it is within 10^{-16} to the true value. As we add one loop to obtain the code for Gaussian Triple Integral. It is entirely possible to be able to generalize this method to calculate higher dimensional integrals. This integration technique is powerful in its own right. Consider, for example, if we are trying to compute the total mass of some substance. We will proceed by taking a triple volume integral of density function of that substance. Since the substance is arbitrary, the domain will be arbitrary, too. The resulting function, when evaluated with the Gaussian triple integral, will give us the total mass of the substance.

6. Conclusion

We write down the algorithm to compute volume integrals of functions that cannot be computed analytically. We also give a way to plot such volume integral functions so they can be visualized easily in three dimensions. This will allow us to compute an visualize volume integrals of functions–such as two dimensional Gaussian functions–over any arbitrary domain. After this, we investigate polar coordinates and find that for circles, the polar coordinates offer a better approximation. We then compare two methods that approximate double integral, Gaussian and Simpson's method. It turns out that Gaussian is more efficient.

However, it takes a much longer time to pre-process if we are going to calculate roots and weights of Gaussian quadrature from scratch. At the end, we generalize Gaussian Double Integral Method to calculate triple integrals. It is quite accurate even with small n values.

References

[1] Rogawski, Jon. "Calculus: Early Transcendentals". *Macmillan* (2011).

[2] Burden, R., J. Faires, and A. M. Burden. "Numerical Analysis". *Cengage Learning* (2010).

[3] "Gaussian Quadrature Weights and Abscissae" DOI:

https://pomax.github.io/bezierinfo/legendre-gauss.html