NUMERICAL METHODS 2

Mini-project Task 1 - Report

$\boldsymbol{Int2D(func, a, b, nx, c, d, ny)}$

By: Noman Noor

(Student ID: 302343)

Introduction

❖ Goal of the task:

Calculation of 2D Integrals of the form $\int_c^d \int_a^b \boldsymbol{f(x,y)dx\,dy}$ using composite Newton $\frac{3}{8}$ rule *(i.e., composite Simpson's $\frac{3}{8}$)* with respect to x, and the composite Trapezoidal rule with respect to y. Formulas for each rule are present in the appendix.

❖ Approach to the task:

Keeping in mind the formulae for the composite Simpson's $\frac{3}{8}$ quadrature and composite Trapezoidal quadrature (provided in the appendix), for each quadrature we take out the common scalar multiplier of each term's coefficients and a vector containing the coefficients of all the terms. Let their labels be c_S,v_S and c_T,v_T for each rule respectively. A matrix fxy containing values of f(x,y) for each (x,y) combination is computed. Then, due to the nature of Matrix Multiplication, the expression $c_S c_T sum(sum(v_T * fxy * v_S{}^T))$ is the same as : $\left(\boldsymbol{Q}_{s_{\frac{3}{8}}} \times \boldsymbol{Q}_T\right)(\boldsymbol{f}) = \sum_{k=0}^{ny} \boldsymbol{A}_k \sum_{j=0}^{nx} \boldsymbol{B}_j \boldsymbol{f}(\boldsymbol{x}_j, \boldsymbol{y}_k)$

❖ Usage:

$$\int_c^d \int_a^b \boldsymbol{f(x,y)dx\,dy} \approx \mathbf{Int2D(func, a, b, nx, c, d, ny)}$$

where [a,b] and nx are Interval and value of n for using Simpson's 3/8 rule, and [c,d] and ny are Interval and value of n for using Trapezoidal rule.

Numerical Tests

❖ For $\boldsymbol{I = \int_1^2 \int_1^3 x^3 y\, dx\; dy}$:

We compute the integral in a way that avoids numerical errors. We find: $\int \boldsymbol{x^3 y dx} = \frac{\boldsymbol{yx^4}}{\boldsymbol{4}} + \boldsymbol{C}$. Applying the bounds [1 to 3], we get $\frac{\mathbf{(81-1)}}{\mathbf{4}}\boldsymbol{y} = \mathbf{20}\boldsymbol{y}$. Finally: $\boldsymbol{I} = \mathbf{20} \int_1^2 \boldsymbol{ydy} = \mathbf{20}\left[\frac{\boldsymbol{y^2}}{\mathbf{2}}\right]_1^2 = \mathbf{20}\left(\frac{\mathbf{3}}{\mathbf{2}}\right) = \mathbf{30.}$

Comparing with our implementation, we get the following table:

nx	ny	Int2D Approximation	Actual Value (I)	Error
3	1	30	30	0
3	7	30		0
9	1	30		0
6	10	30		0

Simpson's 3/8 rule is exact for polynomials with degree<=3, and Trapezoidal rule is exact for polynomials of degree <=1 regardless of nx or ny are, therefore, we get the exact result (no error).

When nx=3, the first rule becomes simple Simpson's 3/8 (*not* composite) (refer to the formulas in the appendix). Similarly, when ny=1, the second rule becomes simple Trapezoidal rule (*not* composite). From the table it can be seen we have all kinds of such composite and non-composite variants tested.

- For $I = \int_{1}^{0}\int_{0}^{1}(e^{x} - e^{y})\,dx\,dy$

We have I=0 (calculation of accurate I is done in the Appendix). Comparing with Int2D (our implementation):

nx	ny	Int2D approximation	Actual Value (I)	Error
3	10	0.0011733	0	0.0011733
18	30	0.00015889		0.00015889
600	500	0.00000057276		0.00000057276
6000	5000	0.0000000057276		0.0000000057276

We aren't able to get a perfectly accurate value, but we get better approximations nx and/or ny increase. This comes with the cost of computational time and resource usage increases.

Comparison of "Int2D" with MatLab Implementation "integral2"

Consider $\int_{1}^{3}\int_{1}^{2}(y^{3}x)\,dx\,dy$. It's equivalent to the first function we tested (i.e., $\int_{1}^{2}\int_{1}^{3} x^{3}y\,dx\ dy$). So, I=30. Here, Simpson's rule is exact because degree(x)<1. So, let nx = 3. For changing ny, we get:

nx	ny	Int2D Approximation	MATLAB integral2 aprx	Actual Value (I)
3	10	42	30	30
3	50	30.0048		
3	100	30.0012		
3	200	30.0003		
3	300	30.0001		
3	490	30		

Conclusion

We conclude that our implementation can be as accurate as we want depending on the values of nx and ny. It is also exactly accurate when polynomials of x and y where x have degree <=3 and y have degree <=1. Since we can take nx=3 and ny=1 in that case, it is also hugely efficient. Of course, in other cases, computational resources serve as limiting factor. Clearly then, our implmentation is just as viable as MATLAB's default implementation *integral2* for non-mission-critical use-cases, if we have proper nx and ny pairs and computational resources or can accept the extra hit in accuracy. If not, it's recommended to use integral2 in MATLAB instead.

The data also serves as further verification of correctness of the implementation as it shows our implementation shares the same characteristic traits (degree of exactness and increasing accuracy for increasing nx,ny for non-exact functions) as applying the required quadratures in order.

Appendix

- **Actual (accurate) solution to $I = \int_{1}^{0}\int_{0}^{1}(e^{x} - e^{y})\,dx\,dy$:**
  - We find, $\int_{0}^{1} e^{x} - e^{y}\,dx = [-e^{y}x + e^{x} + C]_{0}^{1} = -e^{y} + e - 1$
  - $\Rightarrow \boldsymbol{I} = \int_{1}^{0}(-e^{y} + e - 1)dy = [-e^{y} + ey - y + C]_{1}^{0} = \mathbf{0}$
- **Composite Simpson's 3/8 rule formula:** Let n be a multiple of 3,
  - $$\int_{a}^{b} f(x)\,dx \approx \frac{3h}{8}\left[f(x_0) + 3f(x_1) + 3f(x_2) + 2f(x_3) + 3f(x_4) + 3f(x_5) + 2f(x_6) + \cdots + 3f(x_{n-2}) + 3f(x_{n-1}) + f(x_n)\right]$$
    $$= \frac{3h}{8}\left[f(x_0) + 3\sum_{i \neq 3k}^{n-1} f(x_i) + 2\sum_{j=1}^{n/3-1} f(x_{3j}) + f(x_n)\right] \quad \text{for } k \in \mathbb{N}_0.$$

- **Composite Trapezoidal rule:**
    - $$\int_a^b f(x)\,dx \approx \frac{\Delta x}{2}\left(f(x_0)+2f(x_1)+2f(x_2)+2f(x_3)+2f(x_4)+\cdots+2f(x_{N-1})+f(x_N)\right).$$