高斯積分程式

01057033 洪銘均

Thursday 24th October, 2024

Contents

1 檔案與功能

以下是每個原始碼檔案的簡要描述：

1.1 C++ 標頭檔案

- **func.h**：定義函數 $f(x)$，並新增以下常數：
  - `PI`：定義 PI 常數，使用 cmath.h 的 $aros()$，傳入參數-1。
  - `FUNCSTR`：定義函數 $f(x)$ 的字串表示，用於 GnuPlot 繪圖。
- **tabulation.h**：定義計算高斯積分的權重和採樣點

1.2 C++ 實作檔案

- **func.cpp**：實作函數 $f(x)$
- **tabulation.cpp**：實作高斯積分的權重和採樣點。
- **main.cpp**：主程式，呼叫 compute_weights 計算權重及採樣點，並計算積分值。

2 函式功能描述

以下是主要函式的功能說明：

2.1 func.cpp

- `double F(double, double)`: 計算函數 $F(x, y)$ 的值。

2.2 tabulation.cpp

使用 Pomax 所提供的方法計算 [1]

- `double legendre(int, doulbe)`: 計算勒讓德多項式的值。
- `double legendre_derivative(int, double)`: 計算勒讓德多項式的導數值。
- `std::vector<double> legendre_roots(int, int, double)`: 計算勒讓德多項式的根。
- `std::vector<double> compute_weights(int, const std::vector<double>)`: 計算高斯積分的權重。

2.3 main.cpp

- `main()`: 主程式，呼叫 compute_weights 計算權重及採樣點，並計算積分值，最後使用 gnuplot 繪圖。
- `gauss_quadrature_2D(int, double(*)(double, double), double, double, double, double)`: 計算二維高斯積分。
- `gauss_quadrature_2D_grid(int, double(*)(double, double), double, double, double, double)`: 計算二維高斯積分，並使用網格量分段積分的方式。

3 運行結果

3.1 積分正確答案與函數圖形

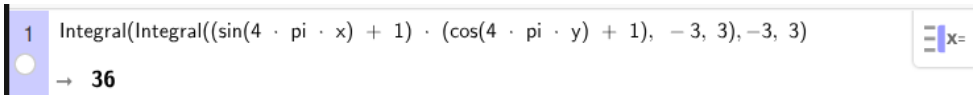

Figure 1: 積分正確答案-使用 Geogebra 計算

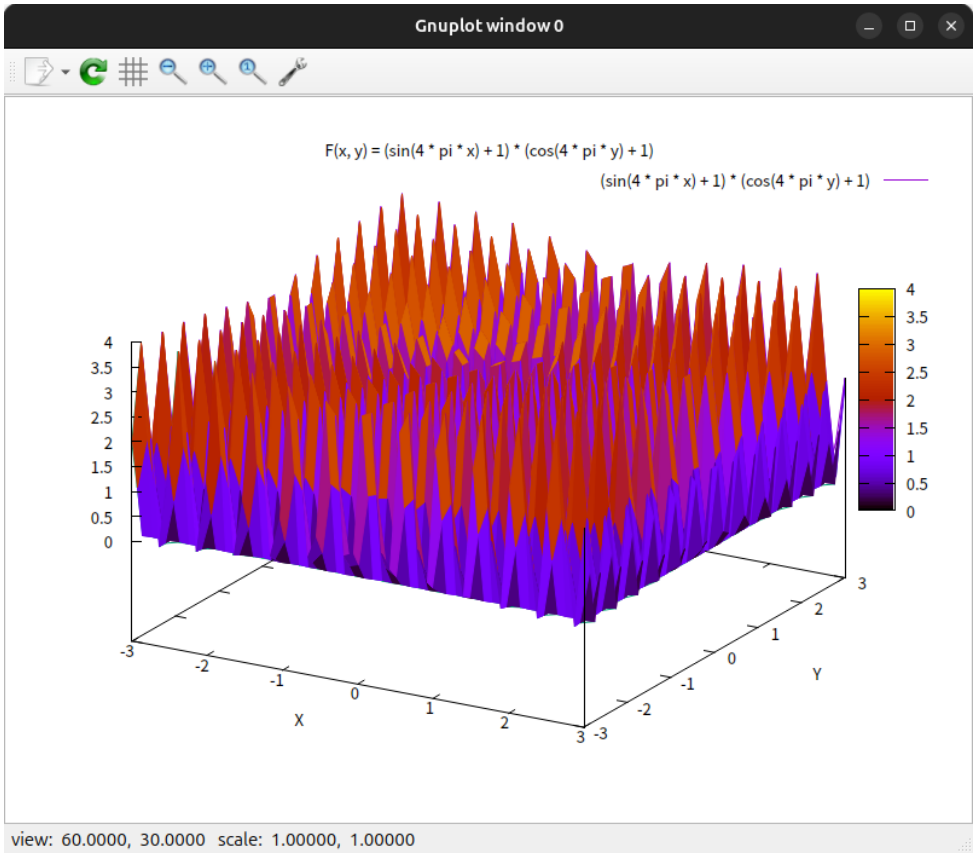

Figure 2: $f(x,y) = (sin(4 * pi * x) + 1) * (cos(4 * pi * y) + 1)$

3.2 P-Refinement

N 為 Legendre 多項式次數，$Result$ 為積分值，$Error$ 為誤差。

N	Result	Error	N	Result	Error
2	9.1188491146e-01	3.5088115089e+01	18	1.8413133260e+01	1.7586866740e+01
3	3.9999637870e+01	3.9996378703e+00	19	4.6208395774e+01	1.0208395774e+01
4	6.4994893227e+01	2.8994893227e+01	20	3.1395581187e+01	4.6044188129e+00
5	4.0440463347e+01	4.4404633474e+00	21	3.7696229819e+01	1.6962298190e+00
6	2.8290249855e+01	7.7097501451e+00	22	3.5473701296e+01	5.2629870408e-01
7	1.9756218686e+01	1.6243781314e+01	23	3.6140478205e+01	1.4047820528e-01
8	5.4755662235e+01	1.8755662235e+01	24	3.5967235045e+01	3.2764954577e-02

9	5.6083021904e+01	2.0083021904e+01	25	3.6006759412e+01	6.7594119433e-03
10	4.6797642332e+01	1.0797642332e+01	26	3.5998754483e+01	1.2455171449e-03
11	3.4682639395e+01	1.3173606046e+00	27	3.6000206651e+01	2.0665102727e-04
12	2.4753251334e+01	1.1246748666e+01	28	3.5999968916e+01	3.1083542787e-05
13	3.5058279612e+01	9.4172038778e-01	29	3.6000004264e+01	4.2635055522e-06
14	5.0119759982e+01	1.4119759982e+01	30	3.5999999464e+01	5.3598760275e-07
15	3.1362156500e+01	4.6378434996e+00	31	3.6000000062e+01	6.2035162784e-08
16	2.2474694991e+01	1.3525305009e+01	32	3.5999999993e+01	6.6364975737e-09
17	5.7341100010e+01	2.1341100010e+01	33	3.6000000001e+01	6.5857364007e-10

3.3 H-Refinement

$G * G$ 為 Grid 數量，$Result$ 為積分值，$Error$ 為誤差，$N = 8$。

G	Result	Error	G	Result	Error
1	5.4755662235e+01	1.8755662235e+01	17	3.6000000000e+01	7.1054273576e-15
2	1.0081204098e+01	2.5918795902e+01	18	3.6000000000e+01	1.4210854715e-14
3	3.4499874630e+01	1.5001253702e+00	19	3.6000000000e+01	2.1316282073e-14
4	3.6043462043e+01	4.3462043139e-02	20	3.6000000000e+01	0.0000000000e+00
5	3.6000000000e+01	0.0000000000e+00	21	3.6000000000e+01	7.1054273576e-15
6	3.5999864294e+01	1.3570560129e-04	22	3.6000000000e+01	4.9737991503e-14
7	3.6000000000e+01	7.1054273576e-15	23	3.6000000000e+01	2.8421709430e-14
8	3.6000000000e+01	2.8421709430e-14	24	3.6000000000e+01	4.9737991503e-14
9	3.6000000000e+01	1.4210854715e-14	25	3.6000000000e+01	2.1316282073e-14
10	3.6000000000e+01	1.4210854715e-14	26	3.6000000000e+01	3.5527136788e-14
11	3.6000000000e+01	1.4210854715e-14	27	3.6000000000e+01	2.8421709430e-14
12	3.6000000003e+01	3.1435547498e-09	28	3.6000000000e+01	7.1054273576e-15
13	3.6000000000e+01	7.1054273576e-15	29	3.6000000000e+01	1.4210854715e-14
14	3.6000000000e+01	7.1054273576e-15	30	3.6000000000e+01	7.1054273576e-15
15	3.6000000000e+01	4.2632564146e-14	31	3.6000000000e+01	3.5527136788e-14
16	3.6000000000e+01	9.9475983006e-14	32	3.6000000000e+01	1.2789769244e-13

3.4 Refinement

橫軸為網格數量，縱軸為 N，表格內容為誤差值。

	3	4	5	6	7	8	9	10
2	2.03e+01	2.40e+01	7.11e-15	3.18e+01	2.13e-14	1.42e-14	1.42e-14	1.42e-14
3	3.05e+00	2.65e+01	2.84e-14	1.91e+01	3.55e-14	6.39e-14	1.42e-14	2.84e-14
4	1.22e+01	2.67e+01	7.11e-15	4.53e+00	7.11e-15	4.26e-14	0.00e+00	2.13e-14
5	2.88e+01	1.09e+01	7.11e-15	6.00e-01	7.11e-15	2.13e-14	7.11e-15	7.11e-15
6	1.83e+01	2.56e+00	1.42e-14	5.12e-02	1.42e-14	7.11e-15	0.00e+00	2.84e-14
7	6.45e+00	3.94e-01	7.11e-15	3.06e-03	1.42e-14	0.00e+00	2.13e-14	2.13e-14
8	1.50e+00	4.35e-02	0.00e+00	1.36e-04	7.11e-15	2.84e-14	1.42e-14	1.42e-14
9	2.52e-01	3.62e-03	7.11e-15	4.65e-06	1.42e-14	1.42e-14	7.11e-15	3.55e-14
10	3.22e-02	2.36e-04	1.42e-14	1.27e-07	4.26e-14	1.42e-14	7.11e-15	4.26e-14
11	3.26e-03	1.24e-05	0.00e+00	2.82e-09	4.26e-14	2.13e-14	7.11e-15	2.13e-14

3.5 Visualize

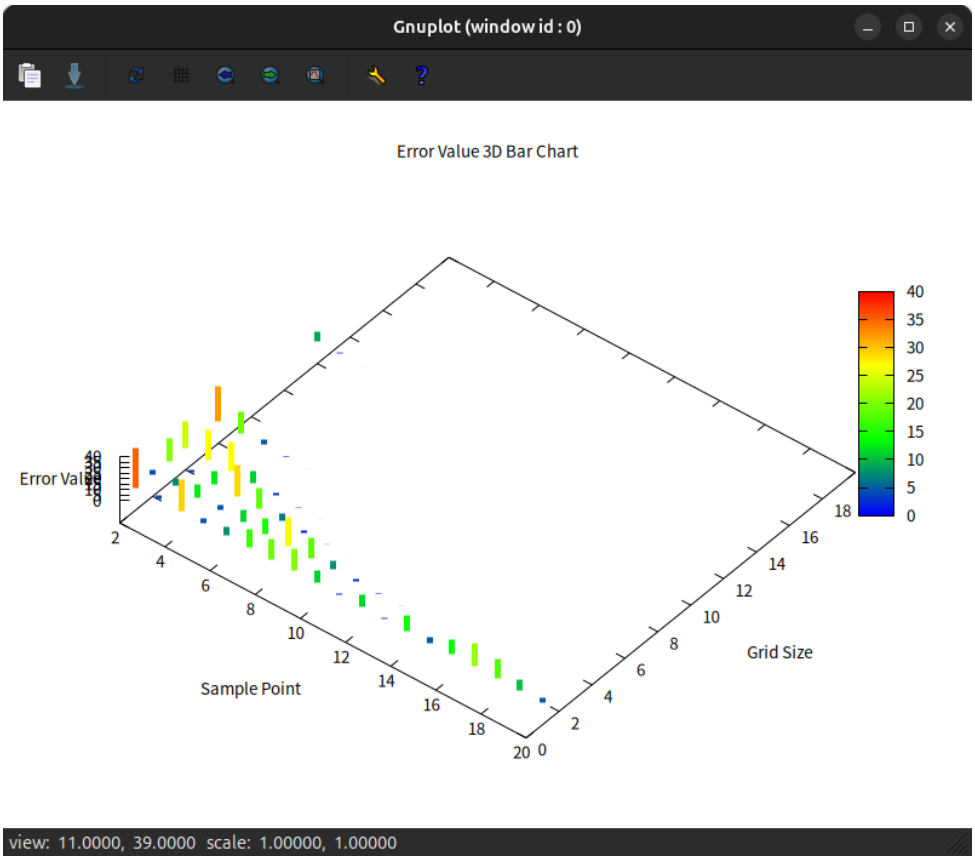

Figure 3: Visualize

4 心得

這次作業實做了高斯積分，並透過網格量分段積分的方式 (H-Refinement) 和多項式次數的增加 (P-Refinement) 來進行誤差分析，最後透過 Refinement 來找比較哪個方法的誤差較小。
在實做過程中，原本是打算使用建表的方式來計算高斯積分的權重和採樣點，但在上網尋找資源時看到有人提供計算權重和採樣點的 psuedo code，因此改用這個方法來計算，得到相比建表的方法更精確且彈性的結果。
在結果視覺化中，原本打算使用 OpenGL，但後來想說嘗試看看使用 gnuplot，因此最後使用 gnuplot 來繪製函數圖形以及誤差。

5 參考資料

References

[1] Pomax, “Legendre-Gauss Quadrature Nodes and Weights”, https://pomax.github.io/bezierinfo/legendre-gauss.html