Sprawozdanie z laboratorium nr 5

Kwadratury Gaussa 2D

Wojciech Matys

Wydział WIMiIP

Kierunek: Inżynieria obliczeniowa

Grupa lab nr 4

Data wykonania: 18.04.2024

Cel ćwiczenia

Celem laboratorium nr. 6 było zapoznanie się z pojęciem kwadratury Gaussa 2D służącej do obliczania całek podwójnych oraz implementacja tego zagadania w wybranym przez siebie języku programowania.

Wstęp

Całkowanie numeryczne to proces przybliżonego obliczania wartości całki oznaczonej funkcji na określonym przedziale. W przeciwieństwie do całkowania analitycznego, które opiera się na dokładnych metodach algebraicznych, całkowanie numeryczne korzysta z technik numerycznych i komputerowych do przybliżenia wyniku.

Kluczową kwestią podczas całkowania numerycznego jest kontrola błędów numerycznych, które mogą wynikać z ograniczeń komputerowych oraz przybliżeń użytych w metodach. Dlatego istotne jest zrozumienie, jakie błędy mogą wystąpić i jak je minimalizować.

Całkowanie numeryczne znajduje zastosowanie w różnych dziedzinach nauki i techniki, szczególnie tam, gdzie nie ma dostępu do rozwiązań analitycznych lub są one zbyt skomplikowane. Dzięki wykorzystaniu metod numerycznych, możemy szybko i efektywnie obliczać wartości całek dla szerokiego zakresu funkcji, co jest kluczowe w wielu aplikacjach inżynieryjnych, naukowych i matematycznych.

W niniejszym sprawozdaniu skoncentrujemy się na metodzie kwadratury Gaussa 2D, która jest jedną z efektywnych technik całkowania numerycznego.

Kwadratury Gaussa 2D

Metoda kwadratury Gaussa jest techniką numeryczną wykorzystywaną do przybliżonego obliczania wartości całek oznaczonych. Polega ona na zastąpieniu całki z funkcji sumą ważonych wartości tej funkcji w wybranych punktach. Ta metoda może być również stosowana do obliczania powierzchni.

Metodę możemy opisać za pomocą następującego wzoru:

$$\iint f(x,y)dxdy = \iint f(\xi,\eta)J_0 d\xi d\eta = \sum_{i=1}^{n}\sum_{j=1}^{n} w_i w_j \quad f(\xi,\eta)J_0 \qquad (1)$$

Gdzie:

- ξ, η są węzłami kwadratury.

- $w_i \; i \; w_j$ są odpowiadającymi tym węzłom wagami.

- J_0 - jakobian zawierający wszystkie pierwsze pochodne funkcji transformacji.

Bardzo istotnym krokiem metody jest przekształcenie figury na postać kwadratu o wymiarach 2 na 2. W tym celu obliczamy wyznacznik macierzy Jakobiego, zawierającej wszystkie pierwsze pochodne funkcji transformacji.

$$|J| = \frac{\partial x}{\partial \xi}\frac{\partial y}{\partial \tilde{\eta}} - \frac{\partial y}{\partial \xi}\frac{\partial x}{\partial \tilde{\eta}} \qquad (2)$$

Przy czym pochodne cząstkowe funkcji transformacji są obliczane z poniższych wzorów:

$$\frac{\partial x}{\partial \xi} = \sum_{i=1}^{4}\frac{\partial N_i}{\partial \xi}x_i \qquad (3)$$

$$\frac{\partial x}{\partial \tilde{\eta}} = \sum_{i=1}^{4}\frac{\partial N_i}{\partial \tilde{\eta}}x_i \qquad (4)$$

$$\frac{\partial y}{\partial \xi} = \sum_{i=1}^{4}\frac{\partial N_i}{\partial \xi}y_i \qquad (5)$$

$$\frac{\partial y}{\partial \tilde{\eta}} = \sum_{i=1}^{4}\frac{\partial N_i}{\partial \tilde{\eta}}y_i \qquad (6)$$

Funkcje transformacji mają postać:

$$N_{1(\xi,\eta)} = 0.25(1 - \xi)(1 - \eta) \quad (7)$$

$$N_{2(\xi,\eta)} = 0.25(1 + \xi)(1 - \eta) \quad (8)$$

$$N_{3(\xi,\eta)} = 0.25(1 + \xi)(1 + \eta) \quad (9)$$

$$N_{4(\xi,\eta)} = 0.25(1 - \xi)(1 + \eta) \quad (10)$$

Węzły oraz ich wagi są wartościami ustalonymi. W przypadku dwuwymiarowej kwadratury Gaussa, metoda ta gwarantuje dokładność dla całek z funkcji wielomianowych, których suma stopni wielomianów po zmiennych $\xi \; i \; \eta$ jest mniejsza lub równa $(2n - 1)$ i $(2m - 1)$ odpowiednio, gdzie (n) i (m) to liczby wykorzystanych węzłów wzdłuż każdego wymiaru.

W przypadku wykorzystania dwóch węzłów, ich wartości wynoszą:

$$[\xi_0 = \eta_0 = \backslash frac\left\{\sqrt{\{3\}}\right\}\{3\} = 0.57735026919\,] \quad (11)$$

$$[\xi_1 = \eta_1 = -\backslash frac\left\{\sqrt{\{3\}}\right\}\{3\} = -0.57735026919\,] \quad (12)$$

Wagi zaś są równe:

$$[\omega_0 = \omega_1 = 1\,] \quad (13)$$

Implementacja

Kwadratury Gaussa 2D implementuję w języku programowania c++, korzystając z środowiska Visual Studio

```
void kwadraturyGaussa() {
    double M[4][2]; // Przeniesione zmiennych do ciała funkcji

    // Odczyt współrzędnych z pliku
    ifstream read("MN5.txt");
    for (int i = 0; i < 4; i++) {
        for (int j = 0; j < 2; j++) {
            read >> M[i][j];
            cout << M[i][j] << " ";
        }
        cout << endl;
    }

    double punkt[2] = { -0.5773502692, 0.5773502692 };
    double waga = 1.0;
    double poch_ksi[2][4];
    double poch_ni[2][4];
```

(1. Fragment kodu)

Ten fragment kodu (1) jest odpowiedzialny za przeprowadzenie kwadratury Gaussa w dwóch wymiarach. Oto jego szczegółowy opis:

1. `void kwadraturyGaussa() {` : Definiuje funkcję o nazwie `kwadraturyGaussa`, która nie zwraca żadnej wartości (`void`).

2. `double M[4][2];` : Definiuje dwuwymiarową tablicę `M`, która przechowuje współrzędne punktów (x, y). Tablica ta ma rozmiar 4x2, co oznacza, że może pomieścić 4 pary współrzędnych.

3. `ifstream read("MN5.txt");` : Tworzy strumień wejściowy `read` do odczytu danych z pliku "MN5.txt".

4. `for (int i = 0; i < 4; i++) {` : Rozpoczyna pętlę iterującą od 0 do 3, która będzie wczytywać dane z pliku do tablicy `M`.

5. `for (int j = 0; j < 2; j++) {` : Rozpoczyna zagnieżdżoną pętlę iterującą od 0 do 1, która będzie wczytywać współrzędne (x, y) dla każdego punktu.

6. `read >> M[i][j];` : Wczytuje kolejne wartości z pliku i przypisuje je do odpowiedniego elementu tablicy `M`.

7. `cout << M[i][j] << " ";` : Wyświetla wczytane współrzędne na ekranie.

8. `cout << endl;` : Wyświetla znak nowej linii po wczytaniu wszystkich współrzędnych jednego punktu.

9. `double punkt[2] = { -0.5773502692, 0.5773502692 };` : Definiuje tablicę `punkt`, która zawiera dwa elementy: -0.5773502692 i 0.5773502692. Te wartości odpowiadają punktom kwadratury Gaussa.

10. `double waga = 1.0;` : Określa wartość wagi, która jest używana w metodzie kwadratury Gaussa.

11. `double poch_ksi[2][4];` : Definiuje dwuwymiarową tablicę `poch_ksi`, która będzie przechowywać pochodne cząstkowe względem ksi.

12. `double poch_ni[2][4];` : Definiuje dwuwymiarową tablicę `poch_ni`, która będzie przechowywać pochodne cząstkowe względem ni.

```
    for (int j = 0; j <= 1; j++) {
        poch_ksi[j][0] = -0.25 * (1.0 - punkt[j]);
        poch_ksi[j][1] = 0.25 * (1.0 - punkt[j]);
        poch_ksi[j][2] = 0.25 * (1.0 + punkt[j]);
        poch_ksi[j][3] = -0.25 * (1.0 + punkt[j]);
        poch_ni[j][0] = -0.25 * (1.0 - punkt[j]);
        poch_ni[j][1] = -0.25 * (1.0 + punkt[j]);
        poch_ni[j][2] = 0.25 * (1.0 + punkt[j]);
        poch_ni[j][3] = 0.25 * (1.0 - punkt[j]);
    }

    double dxdKsi, dydKsi, dxdNi, dydNi;
    double fun_detj[2][2];
    double powierzchnia = 0;

    // Obliczenie powierzchni
    for (int i = 0; i <= 1; i++) {
        dxdKsi = poch_ksi[i][0] * M[0][0] + poch_ksi[i][1] * M[1][0] + poch_ksi[i][2] * M[2][0] + poch_ksi[i][3] * M[3][0];
        dydKsi = poch_ksi[i][0] * M[0][1] + poch_ksi[i][1] * M[1][1] + poch_ksi[i][2] * M[2][1] + poch_ksi[i][3] * M[3][1];
        dxdNi = poch_ni[i][0] * M[0][0] + poch_ni[i][1] * M[1][0] + poch_ni[i][2] * M[2][0] + poch_ni[i][3] * M[3][0];
        dydNi = poch_ni[i][0] * M[0][1] + poch_ni[i][1] * M[1][1] + poch_ni[i][2] * M[2][1] + poch_ni[i][3] * M[3][1];
        fun_detj[i][0] = dxdKsi * dydNi - dxdNi * dydKsi;
        fun_detj[i][1] = dxdKsi * dydNi - dxdNi * dydKsi;
    }

    for (int i = 0; i <= 1; i++) {
        powierzchnia += abs(fun_detj[i][0] * waga * waga) + abs(fun_detj[i][1] * waga * waga);
    }
    cout << "Powierzchnia: " << powierzchnia << endl;
}

int main() {
    kwadraturyGaussa();
    return 0;
```

(2. Fragment kodu)

Ten fragment kodu (2) jest odpowiedzialny za obliczenie pochodnych funkcji transformacji oraz wyznaczenie powierzchni za pomocą metody kwadratury Gaussa. Oto szczegółowy opis:

1. Pochodne funkcji transformacji: W pętli `for` obliczane są pochodne cząstkowe funkcji transformacji względem ksi (poch_ksi) i ni (poch_ni) dla każdego z dwóch punktów (j=0, 1). Wartości te są obliczane dla każdego z 4 węzłów (0-3) i przechowywane w odpowiednich tablicach.

2. Obliczenia dla powierzchni:

- Zmienne `dxdKsi`, `dydKsi`, `dxdNi`, `dydNi` są używane do obliczenia pochodnych funkcji x i y względem ksi i ni.

- Następnie wyznaczane są wyznaczniki macierzy Jacobiego dla każdego punktu.

- Na koniec, dla każdego punktu (i=0, 1), obliczana jest powierzchnia przy użyciu wzoru kwadratury Gaussa 2D i sumowana do zmiennej `powierzchnia`.

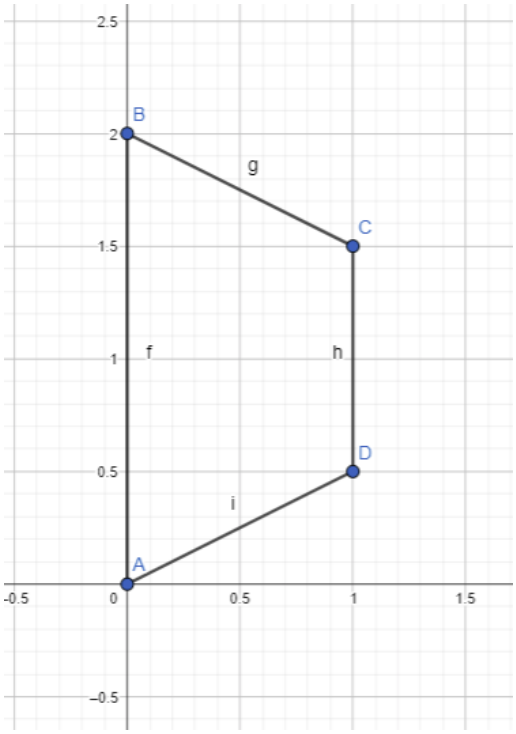
3. Wypisanie wyniku: Obliczona powierzchnia zostaje wyświetlona na ekranie za pomocą instrukcji `cout`.

4. Funkcja `main()`: W funkcji `main()` wywoływana jest funkcja `kwadraturyGaussa()` odpowiedzialna za przeprowadzenie całego procesu obliczeniowego. Następnie funkcja zwraca wartość 0, co oznacza poprawne zakończenie programu.

Testy jednostkowe

Test 1

Wizualizacja przypadku 1:



Wynik policzony przez kalkulator = **1.5**

Dane wejściowe programu:

0 0

0 2

1 1.5

1 0.5

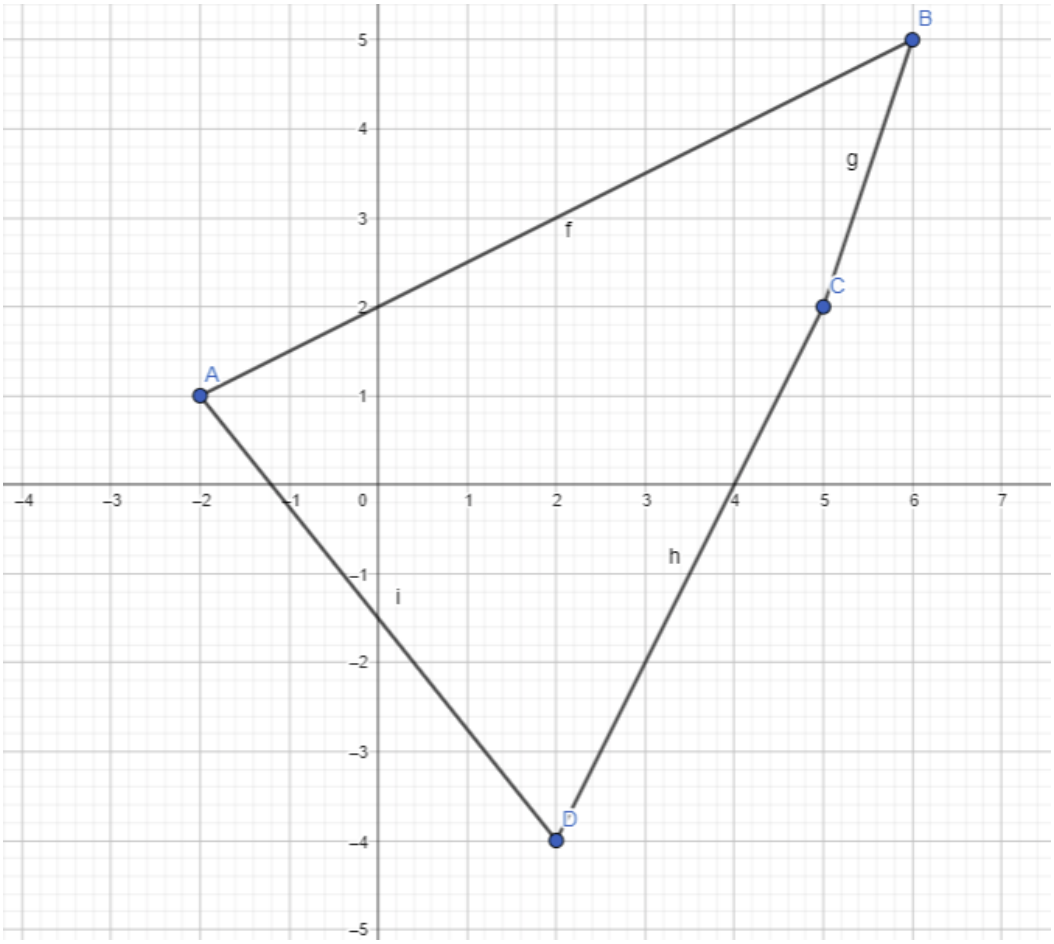
Wynik programu:

```
0 0
0 2
1 1.5
1 0.5
Powierzchnia: 1.5
```

Wynik programu = oczekiwany wynik

Test 2

Wizualizacja przypadku 2:



Wynik policzony przez kalkulator = **29.5**

Dane wejściowe programu:

-2 1

6 5

5 2

2 -4

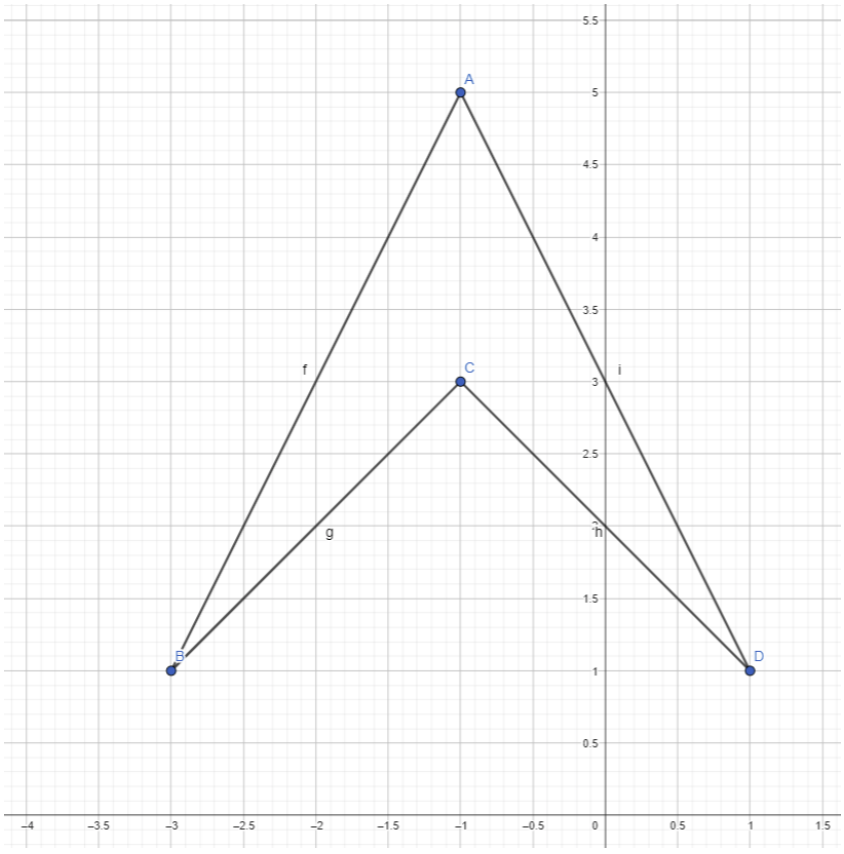
Wynik programu:

```
-2 1
6 5
5 2
2 -4
Powierzchnia: 29.5
```

Wynik programu = oczekiwany wynik

Test 3

Wizualizacja przypadku 3:



Wynik policzony przez kalkulator = **4**

Dane wejściowe programu:

-1 5

-3 1

-1 3

1 1

Wynik programu:

```
-1 5
-3 1
-1 3
1 1
Powierzchnia: 6.9282
```

Wynik programu ≠ oczekiwany wynik

Analiza wyników

Numer testu	Wynik programu	Wynik poprawny	Skala błędu
1	1,5	1,5	0%
2	29,5	29,5	0%
3	6,9282	4	42%

Podsumowanie

Metoda kwadratury Gaussa 2D jest zaawansowanym narzędziem matematycznym, umożliwiającym precyzyjne obliczenia całek dwuwymiarowych dla różnorodnych funkcji. Jej skomplikowane podejście, oparte na sumowaniu ważonych wartości funkcji w odpowiednio dobranych punktach, zapewnia wysoką dokładność nawet dla funkcji o złożonej strukturze.

Jedną z głównych zalet tej metody jest jej efektywność w porównaniu z tradycyjnymi technikami całkowania numerycznego, szczególnie w kontekście całek podwójnych. Dzięki właściwemu dobraniu węzłów i wag możliwe jest uzyskanie dokładnych wyników przy mniejszej liczbie operacji obliczeniowych.

Metoda kwadratury Gaussa 2D znajduje zastosowanie w różnych dziedzinach, od fizyki i inżynierii po geodezję i analizę struktur przestrzennych. Jej adaptacyjność sprawia, że jest niezwykle przydatna w rozwiązywaniu różnorodnych problemów matematycznych i inżynieryjnych wymagających obliczeń dwuwymiarowych całek.

Jednakże, podobnie jak w przypadku innych metod numerycznych, konieczne jest zrozumienie ograniczeń związanych z błędami numerycznymi oraz właściwe dobranie liczby węzłów w celu osiągnięcia oczekiwanej dokładności.

Mimo tych ograniczeń, metoda kwadratury Gaussa 2D jest niezastąpionym narzędziem w rozwiązywaniu problemów, których trudno jest dokładnie rozwiązać analitycznie. Otwiera ona drzwi do nowych możliwości w dziedzinie nauki i technologii, umożliwiając rozwiązanie bardziej złożonych problemów z wykorzystaniem obliczeń numerycznych.