Imię i nazwisko Meg Paskowski	Temat laboratorium Całkowanie numeryczne – Kwadratura Gaussa 2D	Data oddania 17.04.2024r.	Data ćwiczeń 12.04.2024r.
Prowadzący dr hab. inż. Marcin Hojny		Grupa laboratoryjna 4	

1. 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.

2. Wstęp teoretyczny

Kwadratura Gaussa 2D to metoda numeryczna używana do obliczania całek podwójnych, czyli całek dwuwymiarowych, wykorzystywana w analizie numerycznej i metodach numerycznych rozwiązywania równań różniczkowych. Polega na przybliżonym obliczaniu całki podwójnej funkcji dwuwymiarowej *f(x,y)* poprzez sumowanie odpowiednio ważonych wartości funkcji w wybranych punktach w obszarze całkowania przy czym nie są one rozłożone równomiernie oraz punkty krańcowe nie są brane pod uwagę.

Podstawową ideą tej metody jest to, że dla odpowiednio dobranej liczby punktów (węzłów) oraz odpowiednich wag, można uzyskać dokładne lub bardzo dokładne przybliżenie całki podwójnej. W kontekście kwadratury Gaussa 2D, stosuje się wielomiany ortogonalne dwóch zmiennych, a kluczowe punkty (węzły) oraz ich wagi są wybierane tak, aby zminimalizować błąd przybliżenia całki.

Postać ogólna Kwadratury Gaussa dla jednowymiarowej całki w granicach [-1;1]:

$$\int_{-1}^{1} f(x)dx \approx \sum_{i=1}^{n} c_i f(x_i) = c_1 f(x_1) + c_2 f(x_2) + \ldots + c_n f(x_n) \quad (1)$$

Gdzie:

- c_i to wagi kwadratury Gaussa. Liczby rzeczywiste, które określają, jak bardzo każdy punkt (węzeł) kwadratury Gaussa przyczynia się do całkowitej sumy w przybliżeniu całki.
- $f(x_i)$ to wartość funkcji w punkcji x_i

Wzór ogólny dla obliczania całki podwójnej przy użyciu kwadratury Gaussa 2D.

$$\iint (x,y)dxdy = \int_{-1}^{1}\int_{-1}^{1} f(\xi_i,\eta_i)\cdot|J_0|\cdot d\xi\cdot d\eta = \sum_{i=1}^{n}\sum_{j=1}^{n}\omega_i\cdot\omega_j\cdot f(\xi_i,\eta_i)\cdot J_0 \quad (3)$$

Gdzie:

- $f(\xi_i,\eta_i)$ → wartość funkcji w punkcie (ξ_i,η_i), gdzie ξ_i *i* η_i to punkty kwadratury Gaussa.
- $|J_0|$ → moduł wyznacznika Jacobiego J_0.
- ω_{ij} → wagi kwadratury Gaussa
- n → liczba punktów kwadratury w jednym kierunku.

Bardzo istotnym etapem w tym algorytmie jest wyznaczenie przekształcenia figury na postać kwadratu o wymiarach 2 na 2. W tym celu obliczany jest wyznacznik macierzy Jacobiego, który zawiera odpowiednie pochodne funkcji transformacji.

$$|J| = \frac{\partial x}{\partial \xi}\cdot\frac{\partial y}{\partial \tilde{\eta}} - \frac{\partial y}{\partial \xi}\cdot\frac{\partial x}{\partial \tilde{\eta}} \quad (4)$$

Poszczególne pochodne funkcji transformacji możemy obliczyć przy pomocy wzorów (5-8).

$$\frac{\partial x}{\partial \xi} = \sum_{i=1}^{4}\frac{\partial N_i}{\partial \xi}x_i \quad (5)$$

$$\frac{\partial x}{\partial \tilde{\eta}} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \tilde{\eta}} x_i \quad (6)$$

$$\frac{\partial y}{\partial \xi} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \xi} y_i \quad (7)$$

$$\frac{\partial y}{\partial \tilde{\eta}} = \sum_{i=1}^{4} \frac{\partial N_i}{\partial \tilde{\eta}} y_i \quad (8)$$

Po dokonaniu obliczeń uzyskujemy następujące funkcje transformacji (9-12).

$$N_1(\xi, \tilde{\eta}) = 0.25(1 - \xi)(1 - \tilde{\eta}) \quad (9)$$
$$N_2(\xi, \tilde{\eta}) = 0.25(1 + \xi)(1 - \tilde{\eta}) \quad (10)$$
$$N_3(\xi, \tilde{\eta}) = 0.25(1 + \xi)(1 + \tilde{\eta}) \quad (11)$$
$$N_4(\xi, \tilde{\eta}) = 0.25(1 - \xi)(1 + \tilde{\eta}) \quad (12)$$

Głównym celem tych funkcji kształtu jest interpolacja tzn. przybliżenie wartości funkcji wewnątrz obszaru na podstawie wartości znanych w punktach – węzłach kwadratury Gaussa. Każda z funkcji określa wpływ i zakres oddziaływania danego węzła na obszar, który jest analizowany.

Wagi w naszym algorytmie wynoszą (13).

$$\omega_0 = \omega_1 = 1 \quad (13)$$

Wartości wykorzystywanych dwóch węzłów (14-15).

$$\xi_0 = \tilde{\eta}_0 = \frac{\sqrt{3}}{3} = 0.57735026919 \quad (14)$$

$$\xi_1 = \tilde{\eta}_1 = \frac{\sqrt{3}}{3} = -0.57735026919 \quad (15)$$

3. Implementacja

W ramach ćwiczeń zaimplementowano obliczanie całek podwójnych – kwadraturę Gaussa 2D w języku C++. Najważniejszą rolę w tym programie pełni funkcja „*gaussQuadrature()*” przyjmująca tablice punktów „*points*”. Funkcja zaczyna od zainicjowania zmiennych „*surface*” – przechowującej obliczoną powierzchnię, „*scale*” – tablice dwuwymiarową wartości wagi, które są ustawione na, „*nodes*” – tablice zawierającą węzły kwadratury Gaussa. W dalszej części obliczane są pochodne cząstkowe (transformacji) względem ξ(x) i η(y) dla każdego punktu kwadratury (9-12), wartości współczynnika Jacobiego J_0 (4) oraz powierzchni – całki (3). Na sam koniec po zsumowaniu wartości bezwzględnych wyznaczników Jacobiego, skalowaniu i dodaniu do powierzchni zwracany jest wynik „*surface*”.

```
//Funkcja obliczająca całke podwujną dla ustalonych wag i wezłów (punktów)
double gaussQuadrature(double (*points)[4]) {
    double surface = 0.0, dxdKSI, dydKSI, dxdETA, dydETA;
    double scales[2] = { 1, 1 };
    double nodes[2] = { -0.5773502692, 0.5773502692 };
    double derivative_ksi[2][4], derivative_eta[2][4], FUN_DETJ[2][2];

    for (int i = 0; i < 2; i++) {
        derivative_ksi[i][0] = -0.25 * (1.0 - nodes[i]);
        derivative_ksi[i][1] = 0.25 * (1.0 - nodes[i]);
        derivative_ksi[i][2] = 0.25 * (1.0 + nodes[i]);
        derivative_ksi[i][3] = -0.25 * (1.0 + nodes[i]);

        derivative_eta[i][0] = -0.25 * (1.0 - nodes[i]);
        derivative_eta[i][1] = -0.25 * (1.0 + nodes[i]);
        derivative_eta[i][2] = 0.25 * (1.0 + nodes[i]);
```

```
        derivative_eta[i][3] = 0.25 * (1.0 - nodes[i]);
    }

    // Obliczanie powierzchni (całki)
    for (int i = 0; i < 2; i++) {
        dxdKSI = derivative_ksi[i][0] * points[0][0] + derivative_ksi[i][1] * points[1][0] +
        derivative_ksi[i][2] * points[2][0] + derivative_ksi[i][3] * points[3][0];
        dydKSI = derivative_ksi[i][0] * points[0][1] + derivative_ksi[i][1] * points[1][1] +
        derivative_ksi[i][2] * points[2][1] + derivative_ksi[i][3] * points[3][1];

        dxdETA = derivative_eta[i][0] * points[0][0] + derivative_eta[i][1] * points[1][0] +
        derivative_eta[i][2] * points[2][0] + derivative_eta[i][3] * points[3][0];
        dydETA = derivative_eta[i][0] * points[0][1] + derivative_eta[i][1] * points[1][1] +
        derivative_eta[i][2] * points[2][1] + derivative_eta[i][3] * points[3][1];

        // Obliczenie wartości wyznacznika Jacobiego dla każdego punktu kwadratury
        FUN_DETJ[i][0] = dxdKSI * dydETA - dydKSI * dxdETA;
        FUN_DETJ[i][1] = dxdKSI * dydETA - dydKSI * dxdETA;
    }

    // Sumowanie wartości bezwzględnych wyznaczników Jacobiego, skalowanie i dodanie do
powierzchni
    for (int i = 0; i < 2; i++) {
        surface += fabs(FUN_DETJ[i][0]) * scales[0] * scales[1];
        surface += fabs(FUN_DETJ[i][1]) * scales[0] * scales[1];
    }
    return surface;
}
```

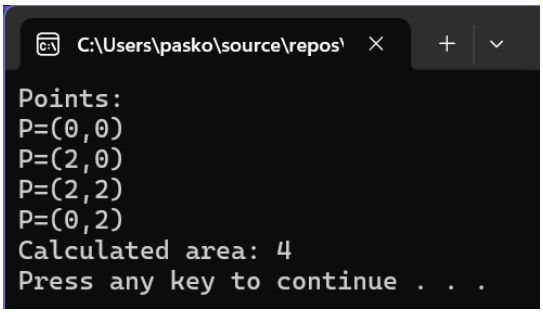
W programie głównym (*main*) dokonywany jest odczyt danych punktów z pliku „*MN6.txt*”. Punkty są odczytywane oraz zapisywane do tablicy dwuwymiarowej „*points*” za pomocą pętli for. Następnie do funkcji „*gaussQuadrature*” przekazywana jest tablica „*points*” i uzyskany wynik policzonego obszaru całkowania jest wyświetlany w terminalu użytkownika. Na sam koniec zamykany jest odczyt pliku.

```
//Odczyt danych punktów dla których szukamy funkcji
    fstream read("MN6.txt");
    double points[4][4]; // Tablica punktów x, y

    if (read.is_open()) {
        // Odczytanie danych do tablicy dwuwymiarowej i wypisanie
        cout << "Points:" << endl;

        for (int i = 0; i < 4; i++) {
            read >> points[i][0] >> points[i][1];
            cout << "P=(" << points[i][0] << "," << points[i][1] << ")" << endl;
        }
        // Obliczanie powierzchni - całki
        cout << "Calculated area: " << gaussQuadrature(points) << endl;

    }
    else
        cout << "Failed to open the file :<" << endl;

    read.close();
```

4. Testy jednostkowe

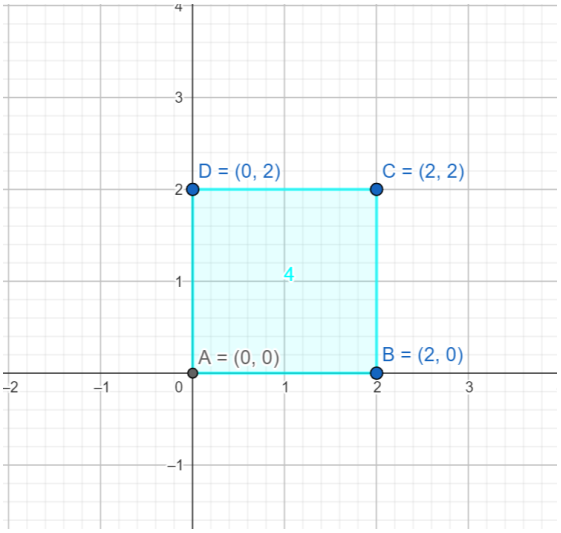
Testy dla tej metody zostały przeprowadzone dla pięciu różnych przykładów, a wyniki zostały porównane do uzyskanych na stronie „*geogebra.com*”.

I. Test 1 dla kwadratu
   - Punkty: (0,0), (2,0), (2,2), (0,2)
   - Wynik programu (Rysunek 1): 4



Rysunek 1 Wyniki działania programu dla testu 1

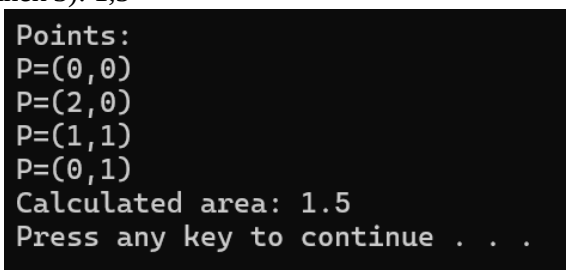
- Wynik uzyskany na stronie: 4



Rysunek 2 Interpretacja graficzna wraz z wynikiem pola dla figury testu 1

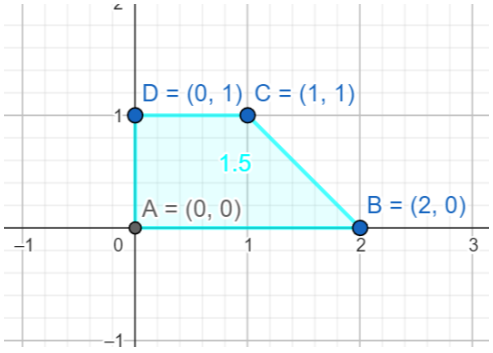
II. Test 2 dla trapezu równoległego

- Punkty: (0,0), (2,0), (1,1), (0,1)
- Wynik programu (Rysunek 3): 1,5



Rysunek 3 Wyniki działania programu dla testu 2

- Wynik uzyskany na stronie: 1,5



Rysunek 4 Interpretacja graficzna wraz z wynikiem pola dla figury testu 2

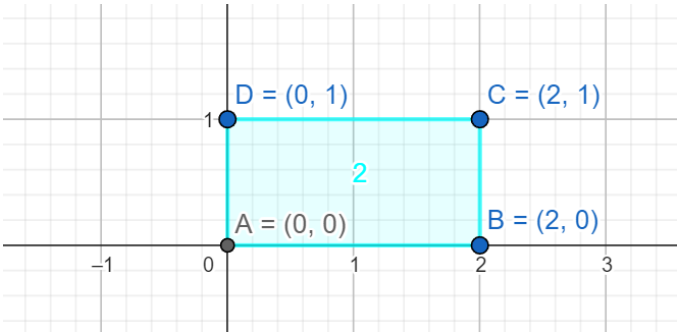
III. Test 3 dla prostokąta

- Punkty: (0,0), (2,0), (2,1), (0,1)
- Wynik programu: 2

```
Points:
P=(0,0)
P=(2,0)
P=(2,1)
P=(0,1)
Calculated area: 2
Press any key to continue . . .
```

Rysunek 5 Wyniki działania programu dla testu 3

- Wynik uzyskany na stronie: 2



Rysunek 6 Interpretacja graficzna wraz z wynikiem pola dla figury testu 3

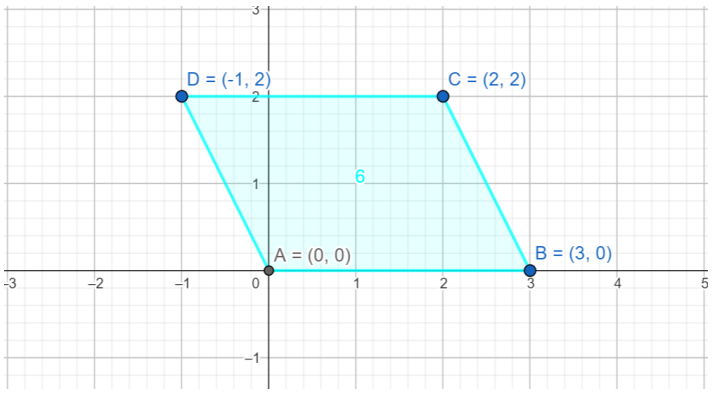
IV. Test 4 dla równoległoboku

- Punkty: (0,0), (3,0), (2,2), (-1,2)
- Wynik programu:6

```
Points:
P=(0,0)
P=(3,0)
P=(2,2)
P=(-1,2)
Calculated area: 6
Press any key to continue . . .
```

Rysunek 7 Wyniki działania programu dla testu 4

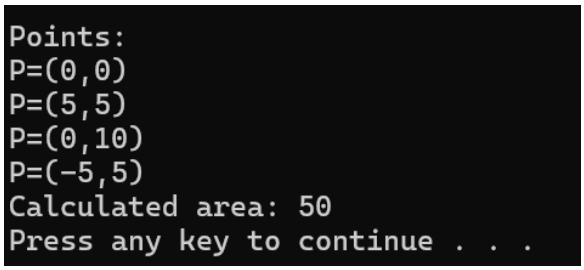
- Wynik uzyskany na stronie:6



Rysunek 8 Interpretacja graficzna wraz z wynikiem pola dla figury testu 4

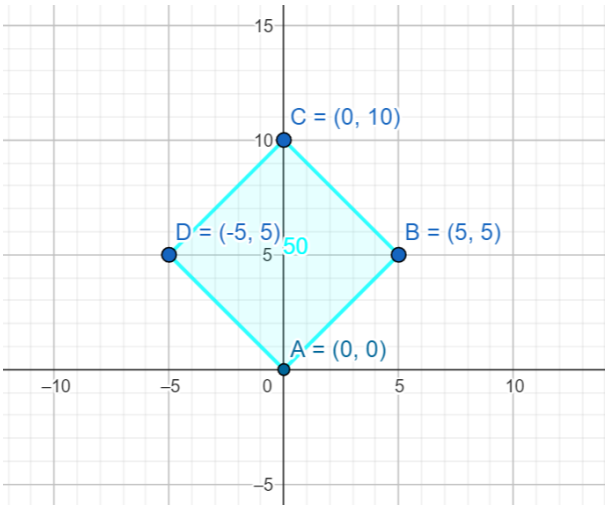
V. Test 5 dla rombu

- Punkty: (0,0), (1,1), (0,2), (-1,1)
- Wynik programu: 50



Rysunek 9 Wyniki działania programu dla testu 5

- Wynik uzyskany na stronie: 50



Rysunek 10 Interpretacja graficzna wraz z wynikiem pola dla figury testu 5

5. Opracowanie wyników

Zestawienie zawarte w poniższej tabeli 1 przedstawia wszystkie wyniki całek oznaczonych dla konkretnych funkcji *f(x,y)* porównane do wyników uzyskanych na stronie „*geogebra.com*".

Tabela 1. Opracowanie uzyskanych wyników testów

Numer Testu	Wynik programu kwadratury Gaussa 2D	Wartość oczekiwana (Geogebra)
1	4	4
2	1.5	1.5
3	2	2
4	6	6
5	50	50

Wyniki analizy przedstawione w tabeli 1 jednoznacznie potwierdzają, że metoda kwadratury Gaussa 2D skutecznie odwzorowuje wartości powierzchni, zgodnie z danymi uzyskanymi na stronie. Warto podkreślić, że niezależnie od umiejscowienia figur w różnych ćwiartkach układu współrzędnych i ich wielkości, metoda ta zachowuje swoją skuteczność.

6. Wnioski

Po analizie wyników można stwierdzić, że testy przeprowadzone na metodzie kwadratury Gaussa w celu obliczania powierzchni różnych kształtów geometrii potwierdzają jej niezawodność i dokładność. Przyjęta implementacja okazała się skuteczna nie tylko dla podstawowych kształtów takich jak kwadraty, ale także dla bardziej złożonych form, takich jak trapezy oraz figury o różnych rozmiarach.

Równocześnie, zgodność wyników dla figur rozmieszczonych w różnych ćwiartkach układu współrzędnych wyraża wszechstronność tej metody.

7. Źródła

- Prezentacja autorstwa dr hab. inż. Marcina Hojnego „Całkowanie numeryczne –kwadratura Gaussa 2D”.
- https://www.youtube.com/watch?v=GgC2UJhNrbE – „Metody numeryczne. Wykład nr 7: Całkowanie”