



UNIVERSITATEA "TITU MAIORESCU" DIN BUCUREŞTI
FACULTATEA DE INFORMATICĂ

METODE NUMERICE PENTRU CALCULUL INTEGRALELOR DEFINITE
LUCRARE DE LICENȚĂ

Absolvent: **Andrei-Stefanel MURARIU**

Coordonator **Conf.Univ.Dr.Ing. Valentin GARBAN**
ştiinţific:

26 Iunie 2024 - 27 Iunie 2024

Cuprins

| | | |
|----------|---|-----------|
| 1 | Introducere | 1 |
| 1.1 | Motivație | 1 |
| 1.2 | Structură | 1 |
| 2 | Analiză și Bază Teoretică | 3 |
| 2.1 | Aproximări și Erori de Aproximare | 3 |
| 2.2 | Limite și Continuitate | 3 |
| 2.3 | Diferențiabilitate | 4 |
| 2.4 | Integrabilitate | 8 |
| 2.4.1 | Integrala Riemann | 9 |
| 2.4.2 | Integrale Improprii | 9 |
| 2.4.3 | Definiția Integrabilității | 10 |
| 3 | Metode Numerice | 11 |
| 3.1 | Metoda Dreptunghiurilor | 11 |
| 3.1.1 | Metoda Dreptunghiurilor la Stânga | 12 |
| 3.1.2 | Metoda Dreptunghiurilor la Mijloc | 16 |
| 3.1.3 | Metoda Dreptunghiurilor la Dreapta | 18 |
| 3.2 | Metoda Trapezelor | 20 |
| 3.3 | Metoda Simpson 1/3 | 24 |
| 3.4 | Cuadratura Legendre-Gaussiană | 28 |
| 3.4.1 | Găsirea Rădăcinilor și Ponderilor Polinoamelor Legendre | 29 |
| 3.4.2 | Schimbarea Intervalului | 29 |
| 3.4.3 | Regulă Gaussiană cu 1 Punct | 30 |
| 3.4.4 | Regulă Gaussiană cu 2 Puncte | 31 |
| 3.4.5 | Regulă Gaussiană cu 3 Puncte | 33 |
| 3.5 | Runge-Kutta 4 (RK4) | 36 |
| 3.5.1 | Algoritm | 37 |
| 4 | Aplicația IntegrX | 38 |
| 4.1 | Tehnologii și Securitatea lor | 38 |
| 4.1.1 | JAVA | 38 |
| 4.1.2 | MATLAB | 40 |
| 4.2 | JAVA | 41 |
| 4.2.1 | Clase și Obiecte | 41 |
| 4.2.2 | Declarație și Inițializare | 43 |
| 4.2.3 | Tipuri de Date | 43 |
| 4.2.4 | Control Flux | 44 |

| | | |
|-----------|--|-----------|
| 4.2.5 | Moștenire | 44 |
| 4.2.6 | Polimorfism | 45 |
| 4.2.7 | Encapsulare | 46 |
| 4.2.8 | Interfețe și Clase Abstracte | 46 |
| 4.2.9 | Excepții | 46 |
| 4.2.10 | JavaFX | 47 |
| 4.3 | MATLAB | 48 |
| 4.3.1 | Ecuății liniare | 48 |
| 4.3.2 | Functii Anonime | 48 |
| 4.3.3 | Integrare Numerică | 48 |
| 4.3.4 | Plot și Mesh | 49 |
| 4.4 | IntegrX | 50 |
| 4.4.1 | Structura Aplicației | 50 |
| 4.4.2 | Interfața Inițială | 52 |
| 4.4.3 | Interfața Rezultat | 56 |
| 4.4.4 | Interfața Derivate | 61 |
| 5 | Concluzii | 63 |
| 5.1 | Dezvoltare în viitor | 63 |
| 6. | Bibliografie | 65 |

Capitolul 1. Introducere

Calculul integralelor definite este esențial în multe discipline științifice și de inginerie, deoarece este un instrument vital pentru rezolvarea unei varietăți de probleme matematice. Chiar dacă există tehnici analitice disponibile pentru evaluarea integralelor simbolice, acestea întâmpină frecvent probleme sau devin inutile atunci când se ocupă de funcții complexe. În aceste circumstanțe, metodele numerice sunt esențiale, deoarece oferă alternative credibile pentru estimarea valorilor integrale cu o precizie ridicată.

1.1. Motivație

Am avut nevoie de o aplicație personalizată pentru a calcula integrala unei funcții și integrala folosind o metodă la alegere și pentru a compara rezultatele (verificând dacă rezultatele metodei sunt acceptabile pentru eroarea utilizatorului stabilită). Acest lucru a fost realizat printr-o aplicație care a folosit avantajele limbajelor de programare JAVA și MATLAB: JAVA a fost folosit pentru crearea corpului aplicativ, iar MATLAB a fost folosit pentru creierul acestei aplicații.

În plus, vreau ca această aplicație, împreună cu bazele teorice de analiza matematică și calcul numeric, să-mi ofere o explicație mai detaliată a problemei de calcul al distanței parcuse a unui vehicul. Spre exemplu, luând în considerare un caz în care viteza vehiculului este cunoscută în 2000 de momente și trebuie să fie parcursă în un an. Spre exemplu, această situație poate fi calculată. Cu toate acestea, există o problemă cu determinarea exactă a rezultatului, ceea ce înseamnă că aproximarea poate fi inexactă.

1.2. Structură

Pentru calcularea integralelor definite folosind metode numerice, există o mulțime de algoritme diferite. Aceste abordări se încadrează în două categorii principale: formula Newton-Cotes și quadratura Gaussiană. Există metode numerice suplimentare pentru calcularea integralelor definite în afară de aceste categorii. Ne vom concentra pe compararea următoarelor metode numerice clasice pentru calcularea integralelor definite în acest articol:

- Metodele dreptunghiulare (stânga, mijloc și dreapta) împart intervalul de integrare în subintervale mai mici și folosesc valorile funcției în punctele extreme sau centrale ale subintervalelor pentru a calcula integrala;
- Regula trapezoidală, care estimează integrala împărțind intervalul de integrare în subintervale mai mici și folosind tehnici de interpolare polinomică pentru a estima modul în care se creează fiecare subinterval;
- Regula lui Simpson 1/3, care se bazează pe interpolarea polinomială de gradul doi;
- Metoda Runge-Kutta de ordinul 4 (RK4), care este o abordare distinctă pentru rezolvarea ecuațiilor diferențiale, dar care poate fi folosită și pentru calcularea integralelor definite;
- Cuadratura Gauss-Legendre este o abordare a quadraturii Gauss care utilizează puncte de evaluare speciale pentru a obține o aproximare mai precisă a integralei.

Structură

- **Analiză și bază teoretică:** În capitolul acesta, va fi examinata baza teoretică și structurile analitice relevante pentru subiectul lucrării.
- **Metode Numerice:** În capitolul acesta, se va discuta despre subiectul principal a acestei lucrări.
- **Aplicația IntegrX:** în acest capitol vor fi abordate detaliile despre partea aplicativă a lucrării.

Capitolul 2. Analiză și Bază Teoretică

Acest capitol va prezenta elementele fundamentale ale studiului calcului numeric.

- Aproximări și erori de aproximare
- Limite și Continuitate
- Diferențiabilitate
- Integrabilitate

2.1. Aproximări și Erori de Aproximare

Când un număr este reprezentat cu mai puține cifre decât este necesar pentru a păstra exactitatea numerică completă, se produce o **eroare de rotunjire**. În schimb, **eroarea de trunchiere** apare atunci când precizia sau timpul de rulare al unei proceduri sunt limitate, ceea ce duce la aproximări care nu pot obține răspunsul complet corect. Acest lucru se datorează faptului că procedura necesită o cantitate enormă de timp, care poate fi chiar infinit, pentru a oferi un răspuns precis. [1]

Să presupunem că valoarea x este aproximată ca y . Atunci avem următoarele definiții în ceea ce privește x și y :

- Eroarea $E_t = x - y$
- Eroarea absolută $|E_t| = |x - y|$
- Eroarea relativă $e_t = \frac{x-y}{x} = 1 - \frac{y}{x}$
- Eroarea absolută relativă $|e_t| = \left| \frac{x-y}{x} \right| = \left| 1 - \frac{y}{x} \right|$

2.2. Limite și Continuitate

Pentru o funcție f definită pe un set X de numere reale, spunem că aceasta are limita L în x_0 , notată ca

$$\lim_{x \rightarrow x_0} f(x) = L,$$

dacă, pentru orice număr real $\varepsilon > 0$, există un număr real $\delta > 0$ astfel încât

$$|f(x) - L| < \delta, \quad \text{pentru orice } x \in X \quad \text{și} \quad 0 < |x - x_0| < \delta$$

Pentru o funcție f definită pe un set X de numere reale și x_0 din X , atunci f este continuă în x_0 dacă

$$\lim_{x \rightarrow x_0} f(x) = f(x_0).$$

Funcția f este **continuă** pe mulțimea X dacă este continuă la fiecare număr din X . [2]

O secvență de numere reale este considerată a fi convergentă la o limită x dacă, pentru fiecare număr real pozitiv s , există un întreg pozitiv $N(\varepsilon)$ astfel încât pentru toți indicii n mai mari decât N , diferența absolută în termenul x_n al secvenței și limita x este mai mică decât s .

$$\lim_{n \rightarrow \infty} x_n = x$$

Această notare indică faptul că atunci când indicele n se apropie de infinit, secvența $\{x_n\}_{n=1}^{\infty}$ converge la valoarea x . [2]

Astfel, atunci când vorbim despre o funcție, ea va fi considerată convergentă și continuă.

2.3. Diferențabilitate

Vom introduce aceasta secțiune cu un exemplu pentru a calcula dacă funcția $\cos(x)$ este diferențialabilă.

Formula pentru a determina derivata unei funcții la un anumit punct $x_0 \in [a, b]$ este urmatoarea unde $[a, b]$ este un interval continuu: [2]

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x) - f(x_0)}{x - x_0} = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}$$

Pentru funcția $f(x) = \cos(x)$, avem:

$$f'(x) = \lim_{h \rightarrow 0} \frac{\cos(x + h) - \cos(x)}{h}$$

Aplicând identitatea trigonometrică $\cos(a + b) = \cos(a)\cos(b) - \sin(a)\sin(b)$, se va obține:

$$f'(x) = \lim_{h \rightarrow 0} \frac{\cos(x)\cos(h) - \sin(x)\sin(h) - \cos(x)}{h}$$

$$f'(x) = \lim_{h \rightarrow 0} \frac{\cos(x)(\cos(h) - 1) - \sin(x)\sin(h)}{h}$$

$$f'(x) = \cos(x) \lim_{h \rightarrow 0} \frac{\cos(h) - 1}{h} - \sin(x) \lim_{h \rightarrow 0} \frac{\sin(h)}{h}$$

$$f'(x) = \cos(x) \cdot 0 - \sin(x) \cdot 1$$

$$f'(x) = -\sin(x)$$

Deci, derivata funcției $\cos(x)$ este $-\sin(x)$, ceea ce înseamnă că funcția $\cos(x)$ este diferențiabilă pentru orice x .

Derivata lui f în punctul x_0 este $f'(x_0)$. O funcție care are o derivată la fiecare valoare dintr-un set cunoscut cuprins între $[a, b]$ este diferențiabilă pe acest set. [2]

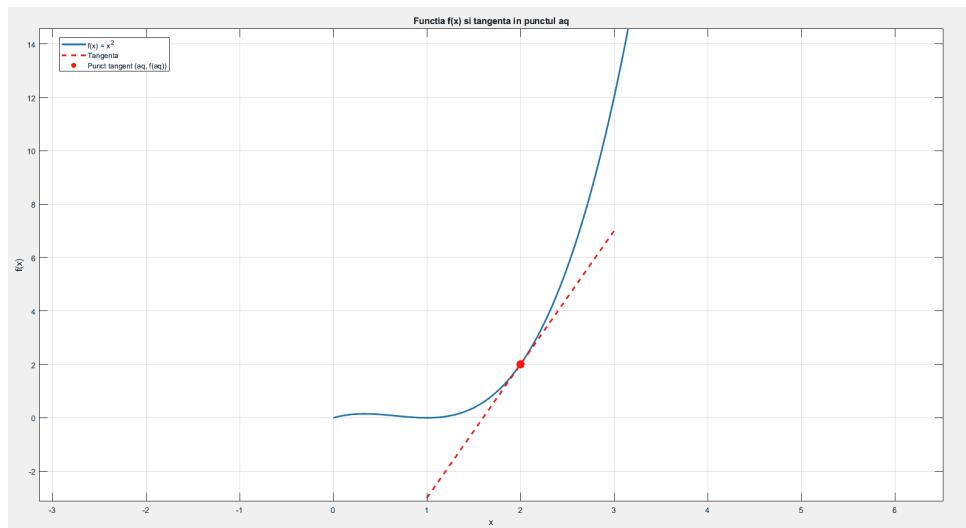


Figura 2.1: Derivata intr-un punct

În imaginea de mai sus, se poate observa că tangenta pantei față de graficul funcției $f(x)$ este rezultatul derivatei unei funcții în x_0 , ceea ce arată cât de crescătoare este funcția $f(x)$ într-un punct specific.[2]

Teorema lui Rolle: Fie f o funcție definită pe un interval I și $a < b$ două puncte din I . Dacă:

1. funcția f este continuă pe intervalul închis $[a, b]$;
2. funcția f este derivabilă pe intervalul deschis (a, b) ;
3. $f(a) = f(b)$,

atunci există cel puțin un punct $c \in (a, b)$, $a < c < b$, în care derivata se anulează, $f'(c) = 0$. [3]

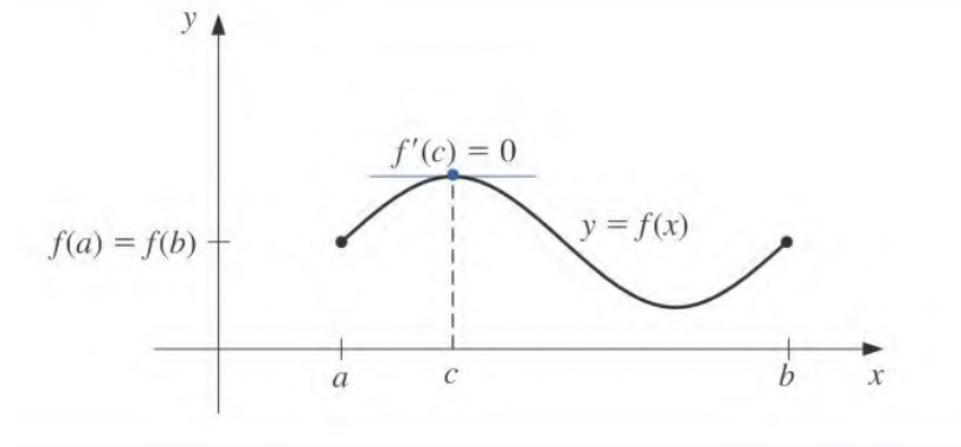


Figura 2.2: Punctul în care derivata este nula
[2, p. 4]

Teorema lui Lagrange: Fie f o funcție definită pe un interval I și $a < b$ două puncte din I . Dacă: [3]

1. f este continuă pe intervalul închis $[a, b]$;
2. f este derivabilă pe intervalul deschis (a, b) ,

atunci există cel puțin un punct $c \in (a, b)$, $a < c < b$, astfel încât să avem

$$f'(c) = \frac{f(b) - f(a)}{b - a}.$$

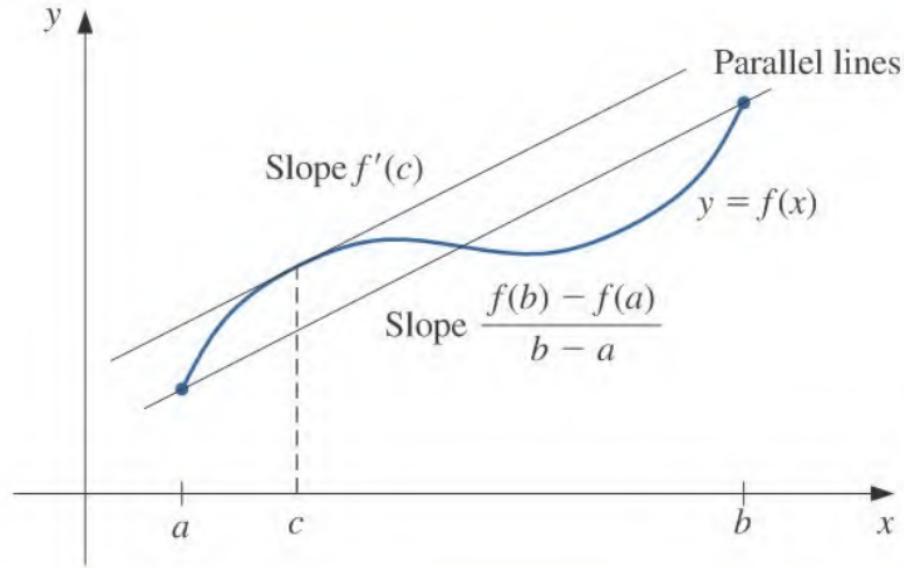


Figura 2.3: Teorema lui Lagrange
[2, p. 5]

Punctul a este $(a, f(a))$ și punctul b este $(b, f(b))$. Teorema valorilor intermediare garantă că există un punct c între a și b atunci când panta secantei care leagă punctele a și b este egală cu panta tangentei, ceea ce înseamnă că tangenta și secanta la c trebuie să fie paralele. [4]

Exemplu: Să se găsească toate valorile c care îndeplinesc condițiile teoremei valorii medii pentru funcția $f(x) = x^3 - 4x^2 + 6x$ cu $x \in [-1, 2]$.

$$f'(c) = \frac{f(2) - f(-1)}{2 - (-1)} \quad (1)$$

$$3c^2 - 8c + 6 = \frac{f(2) - f(-1)}{2 - (-1)} \quad (2)$$

$$3c^2 - 8c + 6 = \frac{4 - 1}{2 + 1} \quad (3)$$

$$3c^2 - 8c + 6 = 1 \quad (4)$$

$$3c^2 - 8c + 5 = 0 \quad (5)$$

$$c = \frac{8\sqrt{64 - 4(3)(5)}}{6} = \frac{82}{6} \rightarrow c_1 = 1.67, \quad c_2 = 1 \quad (6)$$

$$3f'(1) = 3 \rightarrow 3 = 3 \quad | \quad 3f'(\frac{10}{6}) = 3 \rightarrow 3 = 3 \quad (7)$$

Dupa cum se vede, ambele puncte c_1 și c_2 sunt valori care îndeplinesc condițiile teoremei valorii medii.

Teorema lui Fermat: Dacă funcția f are derivată într-un punct de extrem x_0 din interiorul intervalului I , atunci derivata sa este nulă în acest punct, $f'(x_0) = 0$. [3]

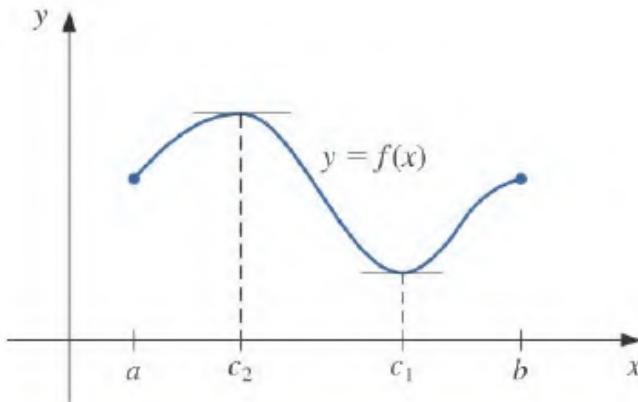


Figura 2.4: Teorema lui Fermat
[2, p. 5]

Exemplu: Să se găsească valourile extreme c pentru funcția $f(x) = x^3 - 3x$ cu $x \in (-1, 2)$.

$$f'(c) = 3c^2 - 3 \quad (1)$$

$$c = \frac{\sqrt{4(3)(3)}}{6} = \frac{6}{6} \quad (2)$$

$$c = \frac{6}{6} = 1 \quad (3)$$

$$c = \sqrt{1} \rightarrow c_1 = 1, \quad c_2 = -1 \quad (4)$$

$$3f'(1) = 0 \quad | \quad 3f'(-1) = 0 \quad (5)$$

Dacă derivata se anulează în ambele puncte c_1 și c_2 , atunci acest lucru implică că aceste puncte sunt puncte extreme ale funcției f pe intervalul (a, b) .

2.4. Integrabilitate

O integrală este o sumă de valori care se înmulțesc cu ponderi definite. Procesul implică înmulțirea unei părți mici dintr-un obiect cu o valoare numită „greutate” și apoi combinarea rezultatelor. Fiecare formă de integrală, fie că este simplă, dublă, triplă, linie sau suprafață, oferă o interpretare distinctă a acestei noțiuni.

2.4.1. Integrala Riemann

În cazul în care există, integrala Riemann a funcției (f) pe intervalul $[a, b]$ este următoarea limită: [2]

$$\int_a^b f(x) dx = \lim_{\Delta x \rightarrow 0} \sum_{i=1}^n f(z_i) \Delta x_i,$$

Numerele x_0, x_1, \dots, x_n sunt astfel alese încât $a = x_0 < x_1 < \dots < x_n = b$, unde $\Delta x_i = x_i - x_{i-1}$ pentru fiecare $i = 1, 2, \dots, n$, iar z_i este ales arbitrar în intervalul $[x_{i-1}, x_i]$.

O funcție f care este continuă pe intervalul $[a, b]$ este și Riemann integrabilă pe acest interval. Acest fapt ne permite să distribuim punctele x_i uniform în $[a, b]$ în scopuri de calcul, și pentru fiecare $i = 1, 2, \dots, n$ să alegem $z_i = x_i$. În această situație, integrala devine: [2]

$$\int_a^b f(x) dx = \lim_{n \rightarrow \infty} \frac{b-a}{n} \sum_{i=1}^n f(x_i) \quad \text{unde} \quad x_i = a + i \cdot \frac{(b-a)}{n}$$

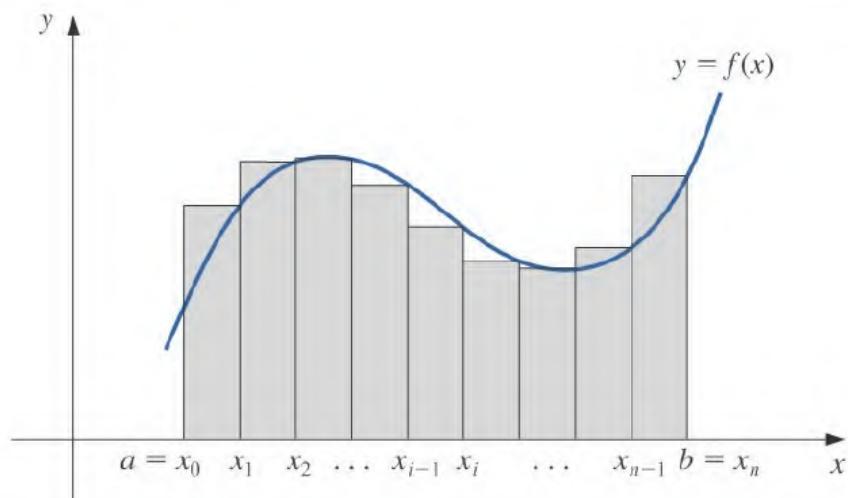


Figura 2.5: Integrala Riemann
[2, p. 7]

2.4.2. Integrale Improprii

Intervalul închis și mărginit $[a, b]$ este baza integralei Riemann, în care funcția trebuie să fie mărginită și integrabilă. Dar există situații în care aceste condiții nu sunt îndeplinite. Pentru a acoperi aceste cazuri, conceptul de integrală Riemann se extinde la

integralele improprii sau generalizate. În mod similar cu seriile numerice, putem determina convergența sau divergența seriei folosind limita șirului sumelor parțiale. Procesul de trecere la limită pentru integralele „parțiale” este utilizat pe domenii mai mici pentru a evita situațiile dificile în cazul integralelor inadecvate. [5]

Definiție: O integrală este numită *improprie* dacă una dintre limitele sale este infinită sau dacă funcția integrată este nemărginită într-un punct din intervalul de integrare. [5]

$$\int_a^\infty f(x)dx, \int_{-\infty}^a f(x)dx, \int_{-\infty}^\infty f(x)dx, \int_a^{b-0} f(x)dx, \int_{a+0}^b f(x)dx$$

Exemplu

Fie funcția $f : [0, 1] \rightarrow \mathbb{R}$, definită astfel:

$$f(x) = \begin{cases} \frac{1}{\sqrt{1-x}}, & \text{pentru } x \in [0, 1); \\ 0, & \text{pentru } x = 1. \end{cases}$$

Atunci, integrala improprie a funcției f este: [5]

$$\int_0^1 f(x)dx = \lim_{u \rightarrow 1} \int_0^u \frac{1}{\sqrt{1-x}} dx = \lim_{u \rightarrow 1} 2(1 - \sqrt{1-u}) = 2.$$

Rezultă că funcția f este integrabilă pe $[0, 1)$, dar nu este integrabilă pe $[0, 1]$, deoarece este nemărginită pe $[0, 1]$.

2.4.3. Definiția Integrabilității

Fie funcția $f : [a, b] \rightarrow \mathbb{R}$ (unde b este finit sau infinit), integrabilă pe orice interval compact $[a, u] \subset [a, b]$. Fie $F(u) = \int_a^u f(x)dx$. Funcția f este *integrabilă* pe $[a, b]$, dacă există $\lim_{u \rightarrow b} F(u)$ și este finită. [5]

Capitolul 3. Metode Numerice

Metodele numerice sunt tehnici eficiente pentru abordarea soluțiilor la problemele matematice. Aceste probleme ar putea apărea în orice domeniu de inginerie. Rezultatele obținute din aceste abordări sunt aproximative, mai degrabă decât exacte. Cu toate acestea, acestea oferă soluții mai rapide în comparație cu metodele convenționale. În plus, aceste metode sunt relativ ușor de programat.

În această lucrare, sunt tratate următoarele metode numerice pentru calculul integralelor definite:

- Metoda Dreptunghiurilor
- Metoda trapezelor
- Metoda Simpson 1/3
- Cuadratura Gaussiană *Q.G.*
 - *Q.G.* Formula cu 1 punct
 - *Q.G.* Formula cu 2 puncte
 - *Q.G.* Formula cu 3 puncte
- Runge-Kutta 4 (RK4)

Notă: Aplicația practică implementează metoda dreptunghiurilor, metoda Trapezelor, metoda Simpson 1/3 și metoda Runge-Kutta (RK4). Cuadratura Gaussiană va fi prezentată doar pe bază teorică.

3.1. Metoda Dreptunghiurilor

Considerați funcția f , care conectează intervalul $[a,b]$ la colecția numerelor reale \mathbb{R} . Pentru a determina o estimare aproximativă a suprafeței deasupra curbei, metoda rectangulară folosește definiția integrală a lui Riemann. Pentru a realiza acest lucru, trebuie făcute o serie de rectangulare cu o larghețe infinit de mică, așezate unul lângă celălalt. Un unghi este ales cu o lungime uniformă pentru a asigura simplitate. [6]

Se va adăuga numărul de intervale definite de valorile $a = x_0 < x_1 < x_2 < \dots < x_n = b$, în care distanța dintre fiecare interval este constantă și este reprezentată prin variabila $h = \frac{b-a}{n}$. [6]

Există trei variante a metodei:

1. Metoda Dreptunghiurilor la stânga
2. Metoda Dreptunghiurilor la mijloc
3. Metoda Dreptunghiurilor la dreapta

3.1.1. Metoda Dreptunghiurilor la Stânga

Pentru a aproxima integrala unei funcții f pe intervalul $[a, b]$, metoda dreptunghiului la stânga împarte intervalul în n subintervale egale, fiecare având lungimea $h = \frac{b-a}{n}$. Suma ariilor dreptunghiurilor formate de valorile funcției în punctele de la stânga ale fiecărui subinterval este dată de formula: [7]

$$I \approx h \sum_{i=0}^{n-1} f(i \times h + a)$$

Luăm în considerare funcția continuă $x^2 + 4x$ pe intervalul închis $[0, 5]$, cu $n = 10$ (unde n reprezintă numărul de intervale).

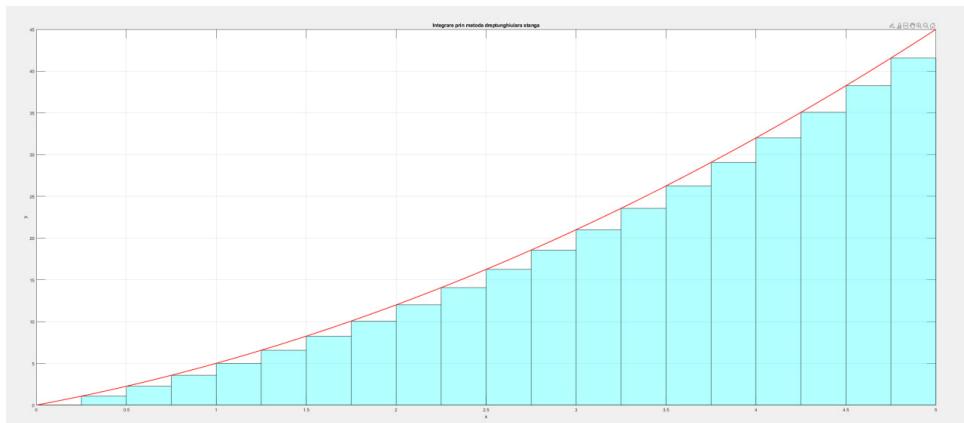


Figura 3.1: $f(x) = x^2 + 4x$ | Metoda Dreptunghiurilor la Stânga

Pasul 1: Calculăm lungimea fiecărui subinterval, h :

$$h = \frac{b-a}{n} = \frac{5-0}{10} = \frac{5}{10} = 0.5$$

Pasul 2: Calculăm punctele de evaluare ale funcției pe fiecare subinterval. Deoarece avem 10 subintervale, punctele de evaluare vor fi $x_0 = 0$, $x_1 = 0.5$, $x_2 = 1.0$, ...

Pasul 3: Evaluăm funcția $x^2 + 4x$ la fiecare punct de evaluare:

$$f(x_0), \quad f(x_1), \quad f(x_2), \quad \dots$$

Pasul 4: Folosind formula sumei Riemann, calculăm suma valorilor funcției în fiecare subinterval înmulțită cu lungimea subintervalului:

$$I \approx h \sum_{i=0}^{n-1} f(i \times h + a)$$

Pasul 5: Înlocuim fiecare $f(x_i)$ cu valoarea sa calculată și calculăm suma.

$$\begin{aligned} I &\approx h \sum_{i=0}^{n-1} f(i \times h + a) \\ &\approx 0.5 (f(0) + f(0.5) + f(1.0) + \dots + f(4.5)) \end{aligned}$$

Pasul 6: Calculăm fiecare termen din sumă:

$$f(0) = 0$$

$$f(0.5) \approx 2.25$$

$$f(1.0) \approx 5$$

⋮

$$f(4.5) \approx 38.25$$

Pasul 7: Calculăm suma valorilor și înlocuim în integrală:

$$\begin{aligned} I &\approx 0.5 \times (0 + 2.25 + 5 + 8.25 + 12 + 16.25 + 21 + 26.25 + 32 + 38.25) \\ &\approx 0.5 \times (161.25) \\ &\approx \color{blue}{80.625} \end{aligned}$$

Acum vom calcula integrala comparând rezultatul metodei dreptunghiului la stânga.

Pasul 1: Descompunem integrala inițială în două integrale mai simple:

$$\int_0^5 (x^2 + 4x) dx = \int_0^5 x^2 dx + \int_0^5 4x dx$$

Pasul 2: Calculăm prima integrală, $\int_0^5 x^2 dx$:

$$\int x^2 dx = \frac{x^3}{3} + C$$

Aplicăm limitele de integrare:

$$\left[\frac{x^3}{3} \right]_0^5 = \frac{5^3}{3} - \frac{0^3}{3} = \frac{125}{3}$$

Pasul 3: Calculăm a doua integrală, $\int_0^5 4x dx$:

$$\int 4x dx = 2x^2 + C$$

Aplicăm limitele de integrare:

$$[2x^2]_0^5 = 2 \cdot 5^2 - 2 \cdot 0^2 = 50$$

Pasul 4: Adunăm rezultatele celor două integrale:

$$\int_0^5 (x^2 + 4x) dx = \frac{125}{3} + 50 = \frac{125}{3} + \frac{150}{3} = \frac{275}{3} = 91.667$$

Calculul erorii absolute și relative:

- Rezultatul corect: 91.667
- Rezultatul metodei: 80.625

Calculăm erorile:

$$\begin{aligned} \text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |91.667 - 80.625| \\ &= 11.042 \end{aligned}$$

$$\begin{aligned}\text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{91.667 - 80.625}{91.667} \right| \times 100\% \\ &\approx 12.04\%\end{aligned}$$

Dacă pentru metoda numerică intervalele erau **100** decât 10, ar fi fost urmatoarele erori.

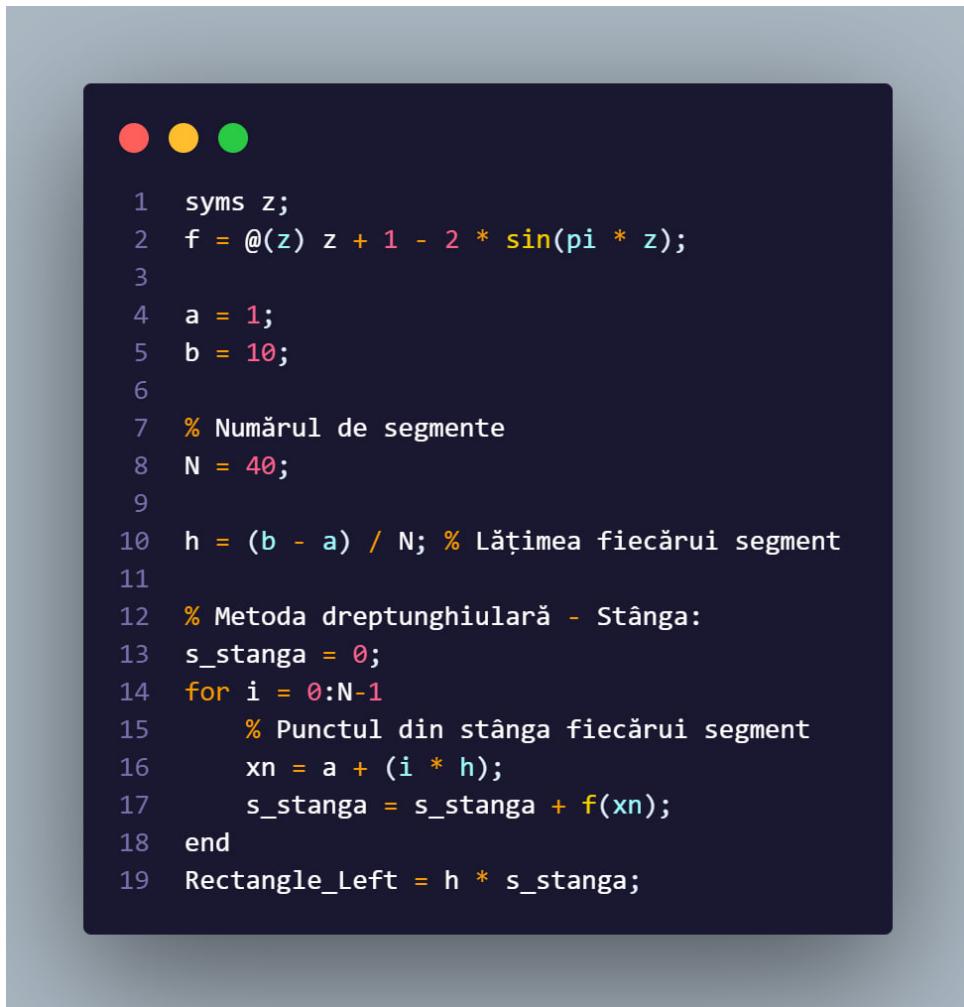
- Rezultatul corect: 91.667
- Rezultatul metodei: 90.544

Calculăm erorile:

$$\begin{aligned}\text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |91.667 - 90.544| \\ &= 1.123\end{aligned}$$

$$\begin{aligned}\text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{91.667 - 90.544}{91.667} \right| \times 100\% \\ &\approx 1.225\%\end{aligned}$$

Este prezent algoritmul în limbajul de programare MATLAB, care poate fi văzut mai jos:



```

1  syms z;
2  f = @(z) z + 1 - 2 * sin(pi * z);
3
4  a = 1;
5  b = 10;
6
7  % Numărul de segmente
8  N = 40;
9
10 h = (b - a) / N; % Lățimea fiecărui segment
11
12 % Metoda dreptunghiulară - Stânga:
13 s_stanga = 0;
14 for i = 0:N-1
15     % Punctul din stânga fiecărui segment
16     xn = a + (i * h);
17     s_stanga = s_stanga + f(xn);
18 end
19 Rectangle_Left = h * s_stanga;

```

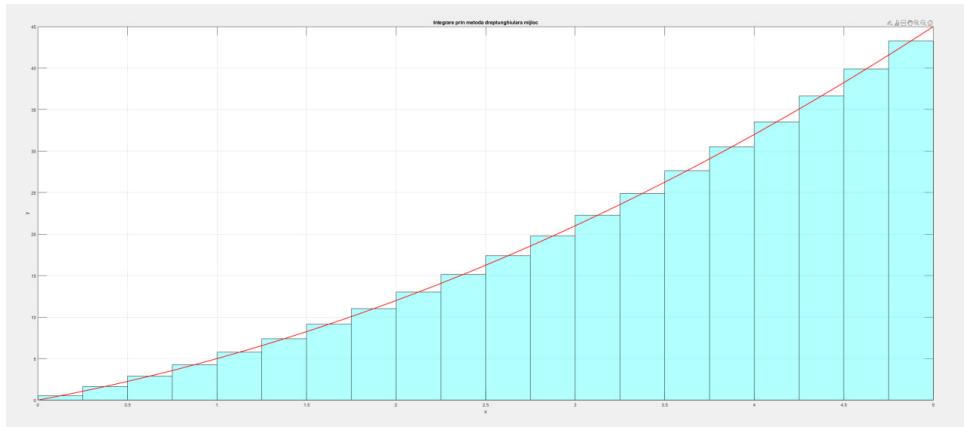
Figura 3.2: Algoritm | Metoda Dreptunghiurilor la Stânga

3.1.2. Metoda Dreptunghiurilor la Mijloc

Pentru a aproxima integrala unei funcții f pe intervalul $[a, b]$, metoda dreptunghiului la mijloc împarte intervalul în n subintervale egale, fiecare având lungimea $h = \frac{b-a}{n}$. Suma ariilor dreptunghiurilor formate de valorile funcției în punctele de la stânga ale fiecarui subinterval este dată de formula: [8]

$$I \approx h \sum_{i=0}^{n-1} f(i \times h + a + \frac{h}{2})$$

Luăm în considerare funcția continuă $x^2 + 4x$ pe intervalul închis $[0, 5]$, cu $n = 10$ (unde n reprezintă numărul de intervale).

Figura 3.3: $f(x) = x^2 + 4x$ | Metoda Dreptunghiurilor la Mijloc**Calculul erorii absolute și relative:**

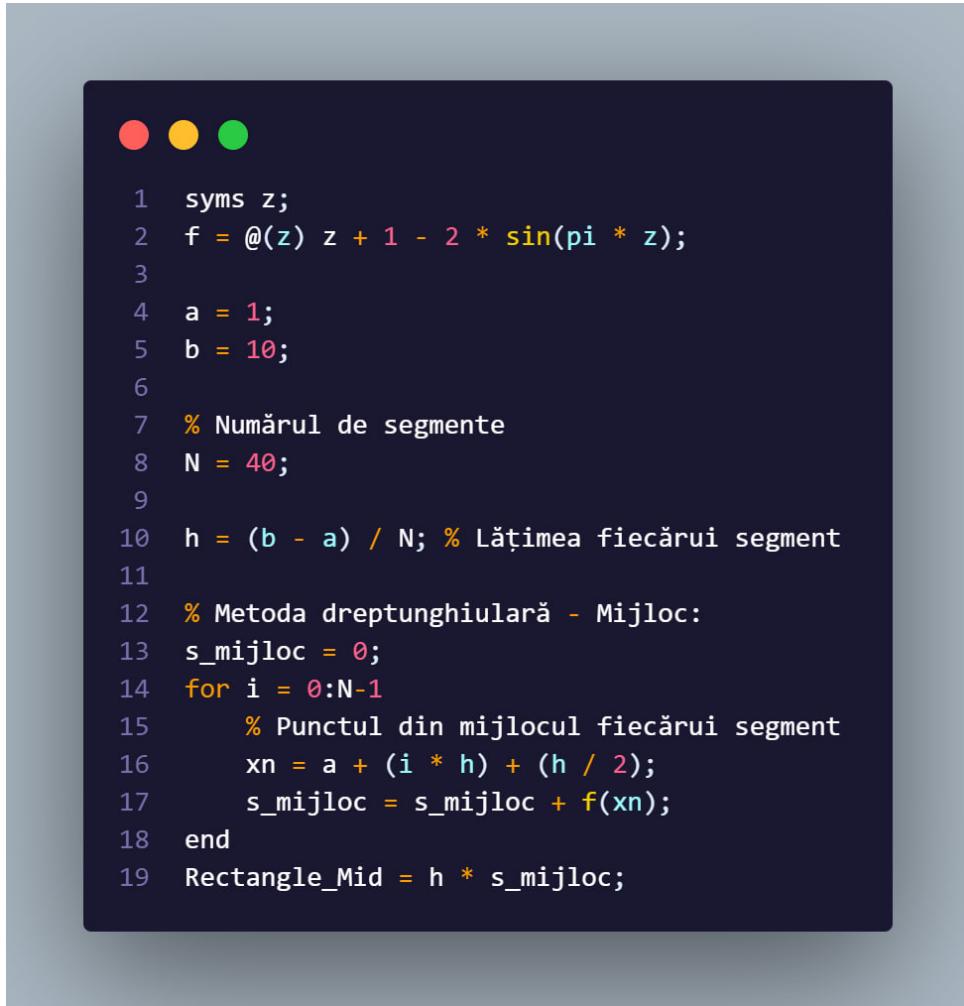
- Rezultatul corect: 91.667
- Rezultatul metodei: 91.563

Calculăm erorile:

$$\begin{aligned} \text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |91.667 - 91.563| \\ &= 0.104 \end{aligned}$$

$$\begin{aligned} \text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{91.667 - 91.563}{91.667} \right| \times 100\% \\ &\approx 0.113\% \end{aligned}$$

Este prezent algoritmul în limbajul de programare MATLAB, care poate fi văzut mai jos:



```

1 syms z;
2 f = @(z) z + 1 - 2 * sin(pi * z);
3
4 a = 1;
5 b = 10;
6
7 % Numărul de segmente
8 N = 40;
9
10 h = (b - a) / N; % Lățimea fiecărui segment
11
12 % Metoda dreptunghiulară - Mijloc:
13 s_mijloc = 0;
14 for i = 0:N-1
15     % Punctul din mijlocul fiecărui segment
16     xn = a + (i * h) + (h / 2);
17     s_mijloc = s_mijloc + f(xn);
18 end
19 Rectangle_Mid = h * s_mijloc;

```

Figura 3.4: Algoritm | Metoda Dreptunghiurilor la Mijloc

3.1.3. Metoda Dreptunghiurilor la Dreapta

Pentru a aproxima integrala unei funcții f pe intervalul $[a, b]$, metoda dreptunghiului la dreapta împarte intervalul în n subintervale egale, fiecare având lungimea $h = \frac{b-a}{n}$. Suma ariilor dreptunghiurilor formate de valorile funcției în punctele de la stânga ale fiecărui subinterval este dată de formula: [9]

$$I \approx h \sum_{i=0}^{n-1} f(i \times h + a + h)$$

Luăm în considerare funcția continuă $x^2 + 4x$ pe intervalul închis $[0, 5]$, cu $n = 10$ (unde n reprezintă numărul de intervale).

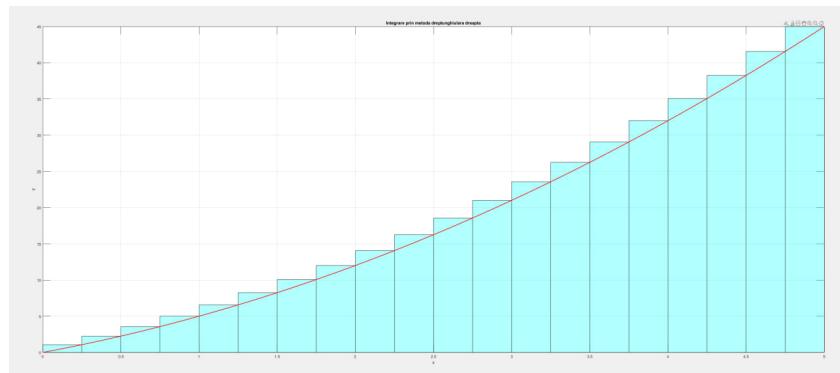


Figura 3.5: $f(x) = x^2 + 4x$ | Metoda Dreptunghiurilor la Dreapta

Calculul erorii absolute și relative:

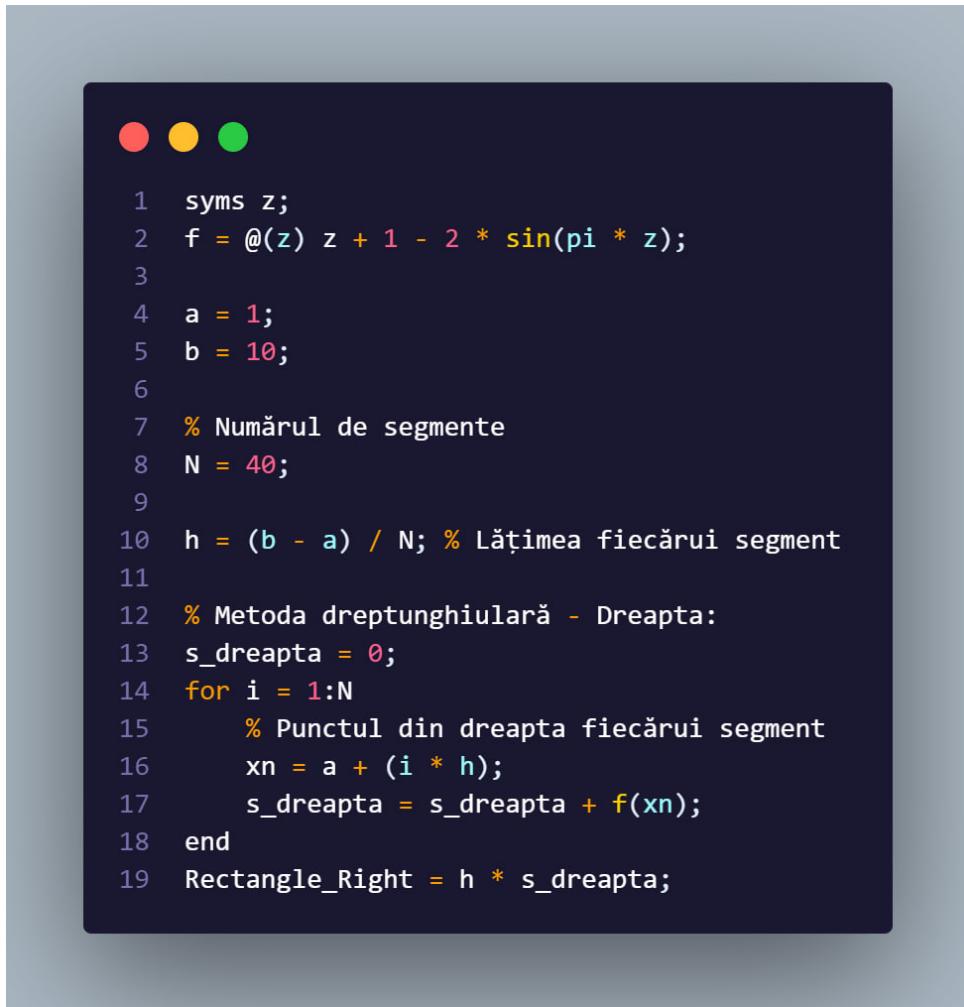
- Rezultatul corect: 91.667
- Rezultatul metodei: 91.563

Calculăm erorile:

$$\begin{aligned} \text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |91.667 - 103.125| \\ &= 11.458 \end{aligned}$$

$$\begin{aligned} \text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{91.667 - 103.125}{91.667} \right| \times 100\% \\ &\approx 12.5\% \end{aligned}$$

Este prezent algoritmul în limbajul de programare MATLAB, care poate fi văzut mai jos:



```

1  syms z;
2  f = @(z) z + 1 - 2 * sin(pi * z);
3
4  a = 1;
5  b = 10;
6
7  % Numărul de segmente
8  N = 40;
9
10 h = (b - a) / N; % Lățimea fiecărui segment
11
12 % Metoda dreptunghiulară - Dreapta:
13 s_dreapta = 0;
14 for i = 1:N
15     % Punctul din dreapta fiecărui segment
16     xn = a + (i * h);
17     s_dreapta = s_dreapta + f(xn);
18 end
19 Rectangle_Right = h * s_dreapta;

```

Figura 3.6: Algoritm | Metoda Dreptunghiurilor la Dreapta

3.2. Metoda Trapezelor

Pentru a aproxima integrala unei funcții f pe intervalul $[a, b]$, metoda trapezelor împarte intervalul în n subintervale egale, fiecare având lungimea $h = \frac{b-a}{n}$. Diferența față de metodele cu dreptunghi este că formula este acum utilizată pentru a determina suma ariilor Trapezelor formate de valorile funcției în punctele de la stânga ale fiecărui subinterval: [10]

$$I \approx \frac{h}{2} \sum_{i=0}^{n-1} f(i \times h + a) + f(i \times h + h + a)$$

Luăm în considerare acum o funcție continuă $\cos(x) - \log(x)$ pe intervalul închis $[1, 6]$, cu $n = 10$ (unde n reprezintă numărul de intervale).

Notă: Logaritmul lui 0 este nedefinit. Datorită faptului că este imposibil să se obțină zero creștând orice valoare la orice altă valoare, numărul în discuție nu poate fi considerat un număr real. În MATLAB $\log(x)$ se referă la $\ln(x)$.

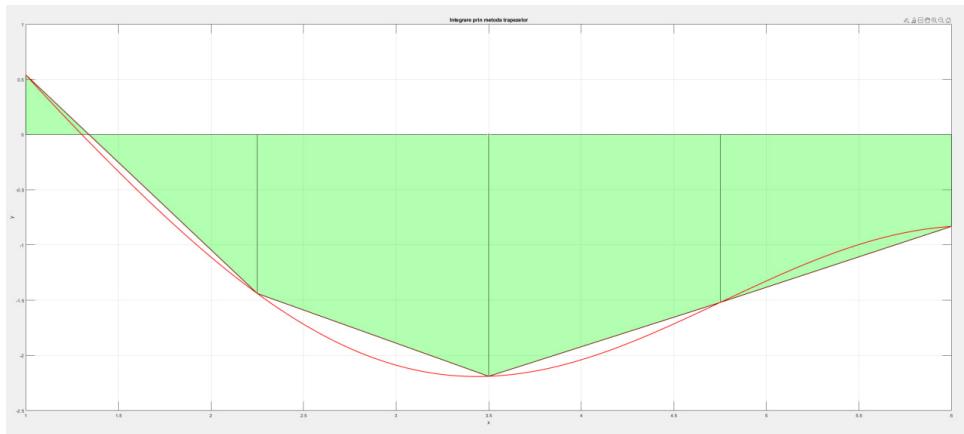


Figura 3.7: $f(x) = \cos(x) - \log(x)$ | Metoda trapezelor

Pasul 1: Calculăm lungimea fiecărui subinterval, h :

$$h = \frac{b-a}{n} = \frac{6-1}{10} = \frac{5}{10} = 0.5$$

Pasul 2: Calculăm punctele de evaluare ale funcției pe fiecare subinterval. Deoarece avem 10 subintervale, punctele de evaluare vor fi $x_0 = 1, x_1 = 1.5, x_2 = 2.0, \dots$

Pasul 3: Evaluăm funcția $\cos(x) - \log(x)$ la fiecare punct de evaluare:

$$f(x_0), \quad f(x_1), \quad f(x_2), \quad \dots$$

Pasul 4: Folosind formula, calculăm suma valorilor funcției în fiecare subinterval înmulțită cu lungimea subintervalului:

$$I \approx \frac{h}{2} \sum_{i=0}^{n-1} f(i \times h + a) + f(i \times h + h + a)$$

Pasul 5: Înlocuim fiecare $f(x_i)$ cu valoarea sa calculată și calculăm suma.

$$I \approx \frac{h}{2} \sum_{i=0}^{n-1} f(i \times h + a) + f(i \times h + h + a)$$

$$\approx 0.25 (f(1) + 2f(1.5) + \dots + 2f(5.5) + f(6))$$

Pasul 6: Calculăm fiecare termen din sumă:

$$f(1) = 0.5403$$

$$f(1.5) \approx -0.3347$$

$$f(2) \approx -1.1093$$

$$\vdots$$

$$f(6) \approx -0.8316$$

Pasul 7: Calculăm suma valorilor și înlocuim în integrală:

$$I \approx 0.25 \times (0.5403 - 0.6694 - 2.2186 - \dots - 2.6516 - 1.9922 - 0.8316)$$

$$\approx 0.25 \times (-27.3233)$$

$$\approx \color{blue}{-6.8308}$$

Acum vom calcula integrala comparând rezultatul metodei trapezelor.

Pasul 1: Descompunem integrala inițială în două integrale mai simple:

$$\int_1^6 \cos(x) - \log(x) dx = \int_1^6 \cos(x) dx - \int_1^6 \log(x) dx$$

Pasul 2: Calculăm prima integrală, $\int_1^6 \cos(x) dx$:

$$\int \cos(x) dx = \sin(x) + C$$

Aplicăm limitele de integrare:

$$[\sin(x)]_1^6 = \sin(6) - \sin(1) = -1.1209$$

Pasul 3: Calculăm a doua integrală, $\int_1^6 \log(x) dx$:

$$\int \log(x) dx = x \cdot (\log(x) - 1) + C$$

Aplicăm limitele de integrare:

$$[x \cdot (\log(x) - 1)]_1^6 = (6 \cdot (\log(6) - 1)) - (1 \cdot (\log(1) - 1)) = 5.7506$$

Pasul 4: Adunăm rezultatele celor două integrale:

$$\int_1^6 \cos(x) - \log(x) dx = -1.1209 - 5.7506 = \textcolor{blue}{-6.8715}$$

Calculul erorii absolute și relative:

- Rezultatul corect: -6.8715
- Rezultatul metodei: -6.8308

Calculăm erorile:

$$\begin{aligned} \text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |-6.8715 + 6.8308| \\ &= \textcolor{blue}{0.0407} \end{aligned}$$

$$\begin{aligned} \text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{-6.8715 + 6.8308}{-6.8715} \right| \times 100\% \\ &\approx \textcolor{blue}{0.59\%} \end{aligned}$$

Este prezent algoritmul în limbajul de programare MATLAB, care poate fi văzut mai jos:

```

1  syms x
2
3  % Limita inferioară
4  a = 0;
5
6  % Limita superioară
7  b = 1;
8
9  % Numărul de segmente
10 n = 10;
11
12 % Declararea funcției
13 f = @(x) 1 / (1 + x^2);
14
15 % h este dimensiunea segmentului
16 h = (b - a) / n;
17
18 % X stochează suma primului și ultimului segment
19 X = f(a) + f(b);
20
21 % Variabila R stochează suma tuturor termenilor
22 % de la 1 la n-1
23 R = 0;
24 for i = 1:n-1
25     xi = a + (i * h);
26     R = R + f(xi);
27 end
28
29 % Formula pentru a calcula integrarea numerică
30 % folosind regula trapezoidală
31 I = (h / 2) * (X + 2 * R);

```

Figura 3.8: Algoritm | Metoda Trapezelor

3.3. Metoda Simpson 1/3

Pentru a aproxima integrala unei funcții f pe intervalul $[a, b]$ folosind metoda Simpson, împărțim intervalul în n subintervale egale, unde n este un număr par. Fiecare subinterval are lungimea $h = \frac{b-a}{n}$. Formula pentru metoda Simpson este: [11]

$$\int_a^b f(x) dx \approx \frac{h}{3} \left[f(x_0) + 4 \sum_{\substack{i=1 \\ \text{impar}}}^{n-1} f(x_i) + 2 \sum_{\substack{i=2 \\ \text{par}}}^{n-2} f(x_i) + f(x_n) \right]$$

unde $x_i = a + ih$ pentru $i = 0, 1, 2, \dots, n$.

Luăm în considerare acum o funcție continuă $\cos(x) - \log(x)$ pe intervalul închis $[1, 10]$, cu $n = 6$ (unde n reprezintă numărul de intervale).

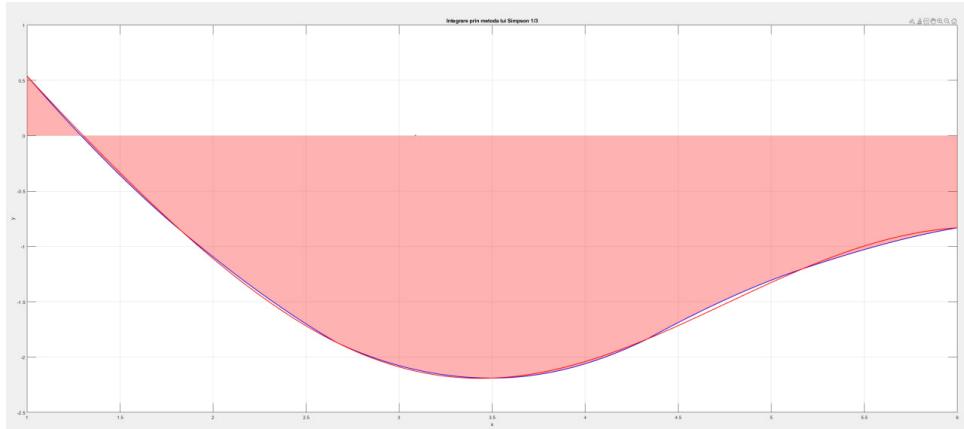


Figura 3.9: $f(x) = \cos(x) - \log(x)$ | Metoda Simpson 1/3

Pasul 1: Calculăm lungimea fiecărui subinterval, h :

$$h = \frac{b-a}{n} = \frac{10-1}{6} = \frac{9}{6} = \frac{3}{2}$$

Pasul 2: Calculăm punctele de evaluare ale funcției pe fiecare subinterval. Deoarece avem 10 subintervale, punctele de evaluare vor fi $x_0 = 1$, $x_1 = 2.5$, $x_2 = 4.0$, ...

Pasul 3: Evaluăm funcția $\cos(x) - \log(x)$ la fiecare punct de evaluare:

$$f(x_0), \quad f(x_1), \quad f(x_2), \quad \dots$$

Pasul 4: Folosind formula, calculăm suma valorilor funcției în fiecare subinterval înmulțită cu lungimea subintervalului:

$$I \approx \frac{h}{3} \left(f(a) + f(b) + 4 \sum_{\substack{i=1 \\ \text{impar}}}^{n-1} f(a + ih) + 2 \sum_{\substack{i=2 \\ \text{par}}}^{n-2} f(a + ih) \right)$$

Pasul 5: Înlocuim fiecare $f(x_i)$ cu valoarea sa calculată și calculăm suma.

$$\begin{aligned} I &\approx \frac{h}{3} \left(f(a) + f(b) + 4 \sum_{\substack{i=1 \\ \text{impar}}}^{n-1} f(a + ih) + 2 \sum_{\substack{i=2 \\ \text{par}}}^{n-2} f(a + ih) \right) \\ &\approx 0.5 (f(1) + 4f(2.5) + 2f(4) + \dots + f(10)) \end{aligned}$$

Pasul 6: Calculăm fiecare termen din sumă:

$$f(1) = 0.5403$$

$$f(2.5) \approx -6.8696$$

$$f(4) \approx -4.0798$$

⋮

$$f(10) \approx -3.1417$$

Pasul 7: Calculăm suma valorilor și înlocuim în integrală:

$$I \approx 0.25 \times (0.5403 - 6.8696 - 4.0798 - 3.9844 - 2.3840 - 10.9684 - 3.1417)$$

$$\approx 0.25 \times (-30.8876)$$

$$\approx \color{blue}{-15.4438}$$

Acum vom calcula integrala comparând rezultatul metodei trapezelor.

Pasul 1: Descompunem integrala inițială în două integrale mai simple:

$$\int_1^{10} \cos(x) - \log(x) dx = \int_1^{10} \cos(x) dx - \int_1^{10} \log(x) dx$$

Pasul 2: Calculăm prima integrală, $\int_1^{10} \cos(x) dx$:

$$\int \cos(x) dx = \sin(x) + C$$

Aplicăm limitele de integrare:

$$[\sin(x)]_1^{10} = \sin(10) - \sin(1) = -1.3855$$

Pasul 3: Calculăm a doua integrală, $\int_1^{10} \log(x) dx$:

$$\int \log(x) dx = x \cdot (\log(x) - 1) + C$$

Aplicăm limitele de integrare:

$$[x \cdot (\log(x) - 1)]_1^{10} = (10 \cdot (\log(10) - 1)) - (1 \cdot (\log(1) - 1)) = 14.0259$$

Pasul 4: Adunăm rezultatele celor două integrale:

$$\int_1^{10} \cos(x) - \log(x) dx = -1.3855 - 14.0259 = \textcolor{blue}{-15.4114}$$

Calculul erorii absolute și relative:

- Rezultatul corect: -15.4114
- Rezultatul metodei: -15.4438

Calculăm erorile:

$$\begin{aligned}\text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |-15.4114 + 15.4438| \\ &= \textcolor{blue}{0.0324}\end{aligned}$$

$$\begin{aligned}\text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{-15.4114 + 15.4438}{-15.4114} \right| \times 100\% \\ &\approx \textcolor{blue}{0.2102\%}\end{aligned}$$

Este prezent algoritmul în limbajul de programare MATLAB, care poate fi văzut mai jos:



```

1  syms x
2
3  % Limita inferioară
4  a = 4;
5
6  % Limita superioară
7  b = 5.2;
8
9  % Numărul de segmente
10 n = 6;
11
12 % Declarația funcției
13 f = @(x) log(x);
14
15 % h este dimensiunea segmentului
16 h = (b - a) / n;
17
18 % X stocă suma primului și ultimului segment
19 X = f(a) + f(b);
20
21 % Variabilele Odd și Even pentru a stoca
22 % suma termenilor impari și pari respectiv
23 Odd = 0;
24 Even = 0;
25
26 % Bucla pentru termenii impari
27 for i = 1:2:n-1
28     xi = a + (i * h);
29     Odd = Odd + f(xi);
30 end
31
32 % Bucla pentru termenii pari
33 for i = 2:2:n-2
34     xi = a + (i * h);
35     Even = Even + f(xi);
36 end
37
38 % Formula pentru a calcula integrarea numerică
39 % folosind regula 1/3 a lui Simpson
40 I = (h / 3) * (X + 4 * Odd + 2 * Even);

```

Figura 3.10: Algoritm | Metoda Simpson 1/3

3.4. Cuadratura Legendre-Gaussiană

Pentru a estima integrala unei funcții, se poate utiliza Cuadratura Gaussiană (sau Legendre-Gaussiană), o metodă de integrare numerică foarte eficientă. Estimarea integrală se realizează calculând suma ponderată a valorilor funcțiilor la anumite puncte specifice din domeniul de integrare, cunoscute sub numele de noduri. Nivelul de precizie al aproximării crește proporțional cu numărul de puncte utilizate. Pozițiile și greutățile acestor puncte sunt determinate folosind polinoamele Legendre.

Proprietate: Cuadratura Gauss-Legendre are o caracteristică semnificativă, deoarece

oferă valori exacte ale integralelor pentru polinoame de grad până la $2n - 1$, unde n este numărul de puncte sau noduri utilizate. [12]

3.4.1. Găsirea Rădăcinilor și Ponderilor Polinoamelor Legendre

Un set de polinoame ortogonale cu funcția de pondere $w(x) = 1$ sunt cunoscute sub numele de polinoame Legendre $P_n(x)$ pe intervalul $[-1, 1]$. Relația de recurență poate fi utilizată pentru a defini aceste polinoame: [13]

Notă: Un polinom ortogonal este perpendicular (ortogonal) față de toate polinoamele de grad mai mic decât el, adică produsul scalar al acestuia este zero.

$$P_0(x) = 1$$

$$P_1(x) = x$$

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} (x^2 - 1)^n, \quad n \in \mathbb{N}_0$$

sau

$$(n+1)P_{n+1}(x) = (2n+1)xP_n(x) - nP_{n-1}(x), \quad n = 1, 2, \dots$$

Rădăcinile x_i sunt găsite ca fiind rădăcinile polinomului Legendre $P_n(x)$. Greutățile w_i pot fi găsite folosind formula: [13]

$$w_i = \frac{2}{(1 - x_i^2)[P'_n(x_i)]^2}$$

unde $P'_n(x)$ este derivata polinomului Legendre de ordin n .

3.4.2. Schimbarea Intervalului

Dacă dorim să integrăm peste intervalul $[a, b]$, putem transforma integrala în intervalul standard $[-1, 1]$ folosind schimbarea de variabilă: [13]

$$t = \frac{b-a}{2}x + \frac{a+b}{2}$$

Prin urmare, integrala devine:

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

Luăm în considerare acum o funcție continuă $x^2 + 4x$ pe intervalul închis $[-1, 1]$, cu $n = 4$ (unde n reprezintă numărul de puncte).

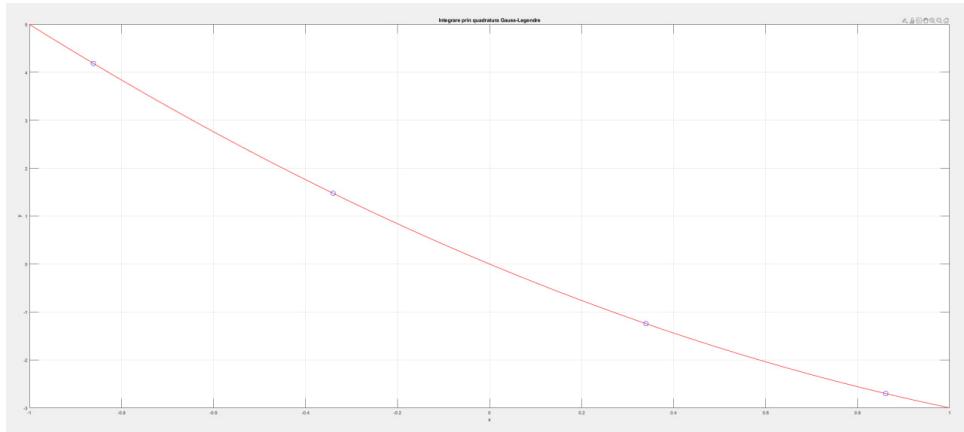


Figura 3.11: $f(x) = x^2 + 4x$ | Cuadratura Gauss-Legendre

3.4.3. Regulă Gaussiană cu 1 Punct

Pentru regula cu 1 punct, integrala este aproximată evaluând funcția în centrul intervalului $[-1, 1]$, adică $x_1 = 0$. [14]

$$\int_{-1}^1 f(x) dx \approx w_1 f(x_1)$$

Pentru a găsi ponderea w_1 , trebuie să calculăm ponderea asociată nodului x_1 . Ponderea asociată acestui nod este adesea dată de formula simplificată:

$$w_1 = \frac{2}{(1 - x_1^2)[P'_1(x_1)]^2} \quad (1)$$

$$w_1 = \frac{2}{(1 - 0)} \quad (2)$$

$$w_1 = 2 \quad (3)$$

- **Nod (x_1):** 0

- **Pondere (w_1):** 2

Astfel, formula de aproximare devine:

$$\int_{-1}^1 f(x) dx \approx 2f(0)$$

3.4.4. Regulă Gaussiană cu 2 Puncte

Pentru regula cu 2 puncte, integrala este aproximată evaluând funcția în două puncte simetrice în jurul centrului. [14]

$$\int_{-1}^1 f(x) dx \approx w_1 f(x_1) + w_2 f(x_2)$$

Considerăm polinomul Legendre de gradul 2:

$$P_2(x) = \frac{1}{2}(3x^2 - 1) \quad (0)$$

Rădăcinile x_i sunt găsite rezolvând ecuația $P_2(x) = 0$:

$$P_2(x) = \frac{1}{2}(3x^2 - 1) = 0 \quad (1)$$

$$2 \cdot \frac{1}{2}(3x^2 - 1) = 0 \cdot 2 \quad (2)$$

$$3x^2 - 1 = 0 \quad (3)$$

$$x^2 = \frac{1}{3} \quad (4)$$

$$x = \pm \frac{1}{\sqrt{3}} \quad (5)$$

Deci, rădăcinile sunt:

$$x_1 = -\frac{1}{\sqrt{3}}, \quad x_2 = \frac{1}{\sqrt{3}}$$

Greutățile w_i sunt calculate folosind formula:

$$w_i = \frac{2}{(1 - x_i^2)[P'_n(x_i)]^2} \quad (0)$$

În primul rând, calculăm derivata polinomului Legendre de gradul 2:

$$P'_2(x) = \frac{d}{dx} \left(\frac{1}{2}(3x^2 - 1) \right) = 3x \quad (1,2)$$

Pentru $x_1 = -\frac{1}{\sqrt{3}}$:

$$P'_2(x_1) = 3 \left(-\frac{1}{\sqrt{3}} \right) = -\sqrt{3} \quad (1.1)$$

$$w_1 = \frac{2}{\left(1 - \left(-\frac{1}{\sqrt{3}} \right)^2 \right) [-\sqrt{3}]^2} \quad (1.2)$$

$$w_1 = \frac{2}{\left(1 - \frac{1}{3} \right) 3} = \frac{2}{\frac{2}{3} \cdot 3} = \frac{2}{2} = 1 \quad (1.3)$$

Pentru $x_2 = \frac{1}{\sqrt{3}}$:

$$P'_2(x_2) = 3 \left(\frac{1}{\sqrt{3}} \right) = \sqrt{3} \quad (2.1)$$

$$w_2 = \frac{2}{\left(1 - \left(\frac{1}{\sqrt{3}} \right)^2 \right) [\sqrt{3}]^2} \quad (2.2)$$

$$w_2 = \frac{2}{\left(1 - \frac{1}{3} \right) 3} = \frac{2}{\frac{2}{3} \cdot 3} = \frac{2}{2} = 1 \quad (2.3)$$

- **Nod** (x_1): $\frac{1}{\sqrt{3}}$ **Nod** (x_2): $-\frac{1}{\sqrt{3}}$
- **Pondere** (w_1): 1 **Pondere** (w_2): 1

Astfel, formula devine:

$$\int_{-1}^1 f(x) dx \approx f \left(\frac{1}{\sqrt{3}} \right) + f \left(-\frac{1}{\sqrt{3}} \right)$$

Luăm în considerare funcția continuă $f(x) = x^2 + 4x$ pe intervalul închis $[-1, 1]$, cu $n = 2$ (unde n reprezintă numărul de puncte).

$$\int_{-1}^1 f(x) dx \approx f \left(\frac{1}{\sqrt{3}} \right) + f \left(-\frac{1}{\sqrt{3}} \right) \quad (0)$$

Înlocuim funcția $f(x)$ cu $x^2 + 4x$ și calculăm:

$$\int_{-1}^1 (x^2 + 4x) dx \approx \left(\frac{1}{\sqrt{3}} \right)^2 + 4 \left(\frac{1}{\sqrt{3}} \right) + \left(-\frac{1}{\sqrt{3}} \right)^2 + 4 \left(-\frac{1}{\sqrt{3}} \right) \quad (1)$$

$$= \frac{1}{3} + \frac{4}{\sqrt{3}} + \frac{1}{3} - \frac{4}{\sqrt{3}} \quad (2)$$

$$= \frac{2}{3} = 0.667 \quad (3)$$

Calculul erorii absolute și relative:

- Rezultatul corect: 0.667
- Rezultatul metodei: 0.667

Calculăm erorile:

$$\begin{aligned}\text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |0.667 - 0.667| \\ &= 0\end{aligned}$$

$$\begin{aligned}\text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{0.667 - 0.667}{0.667} \right| \times 100\% \\ &\approx 0\%\end{aligned}$$

3.4.5. Regulă Gaussiană cu 3 Puncte

Pentru regula cu 3 puncte, integrala este aproximată evaluând funcția în trei puncte specifice. [14]

$$\int_{-1}^1 f(x) dx \approx w_1 f(x_1) + w_2 f(x_2) + w_3 f(x_3)$$

Considerăm polinomul Legendre de gradul 3:

$$P_3(x) = \frac{1}{2}(5x^3 - 3x) \quad (0)$$

Rădăcinile x_i sunt găsite rezolvând ecuația $P_3(x) = 0$:

$$P_3(x) = \frac{1}{2}(5x^3 - 3x) = 0 \quad (1)$$

$$5x^3 - 3x = 0 \quad (2)$$

$$x(5x^2 - 3) = 0 \quad (3)$$

Astfel, avem trei rădăcini posibile: $x = 0$ sau $5x^2 - 3 = 0$. Pentru a găsi celelalte rădăcini, putem rezolva ecuația de gradul 2 $5x^2 - 3 = 0$:

$$5x^2 - 3 = 0 \quad (4)$$

$$x^2 = \frac{3}{5} \quad (5)$$

$$x = \pm \sqrt{\frac{3}{5}} \quad (6)$$

Deci, avem trei rădăcini:

$$x_1 = -\sqrt{\frac{3}{5}}, \quad x_2 = 0, \quad x_3 = \sqrt{\frac{3}{5}}$$

Greutățile w_i sunt calculate folosind formula:

$$w_i = \frac{2}{(1 - x_i^2)[P'_n(x_i)]^2} \quad (0)$$

În primul rând, calculăm derivata polinomului Legendre de gradul 3:

$$P'_3(x) = \frac{d}{dx} \left(\frac{1}{2}(5x^3 - 3x) \right) = \frac{15}{2}x^2 - \frac{3}{2} \quad (1,2,3)$$

Pentru $x_1 = -\sqrt{\frac{3}{5}}$:

$$P'_3(x_1) = \frac{15}{2} \left(-\sqrt{\frac{3}{5}} \right)^2 - \frac{3}{2} = \frac{15}{2} \cdot \frac{3}{5} - \frac{3}{2} = \frac{9}{2} - \frac{3}{2} = 3 \quad (1.1)$$

$$w_1 = \frac{2}{\left(1 - \left(-\sqrt{\frac{3}{5}} \right)^2 \right) \cdot 3^2} \quad (1.2)$$

$$w_1 = \frac{2}{\left(1 - \frac{3}{5} \right) \cdot 9} = \frac{2}{\frac{2}{5} \cdot 9} = \frac{2}{\frac{18}{5}} = \frac{10}{18} = \frac{5}{9} \quad (1.3)$$

Pentru $x_2 = 0$:

$$P'_3(x_1) = \frac{15}{2} \cdot 0^2 - \frac{3}{2} = -\frac{3}{2} \quad (2.1)$$

$$w_2 = \frac{2}{(1 - 0^2) \cdot (-\frac{3}{2})^2} = \frac{2}{1 \cdot \frac{9}{4}} = \frac{2}{\frac{9}{4}} = \frac{8}{9} \quad (2.2)$$

Pentru $x_3 = \sqrt{\frac{3}{5}}$:

$$P'_3(x_3) = \frac{15}{2} \left(\sqrt{\frac{3}{5}} \right)^2 - \frac{3}{2} = \frac{15}{2} \cdot \frac{3}{5} - \frac{3}{2} = \frac{9}{2} - \frac{3}{2} = 3 \quad (3.1)$$

$$w_3 = \frac{2}{\left(1 - \left(\sqrt{\frac{3}{5}} \right)^2 \right) \cdot 3^2} \quad (3.2)$$

$$w_3 = \frac{2}{\left(1 - \frac{3}{5} \right) \cdot 9} = \frac{2}{\frac{2}{5} \cdot 9} = \frac{2}{\frac{18}{5}} = \frac{10}{18} = \frac{5}{9} \quad (3.3)$$

- **Nod** (x_1): $-\sqrt{\frac{3}{5}}$ **Nod** (x_2): 0 **Nod** (x_3): $\sqrt{\frac{3}{5}}$
- **Pondere** (w_1): $\frac{5}{9}$ **Pondere** (w_2): $\frac{8}{9}$ **Pondere** (w_3): $\frac{5}{9}$

Astfel, formula devine:

$$\int_{-1}^1 f(x) dx \approx \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right) + \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right)$$

Luăm în considerare funcția continuă $f(x) = x^2 + 4x$ pe intervalul închis $[-1, 1]$, cu $n = 3$ (unde n reprezintă numărul de puncte).

$$\int_{-1}^1 f(x) dx \approx \frac{8}{9}f(0) + \frac{5}{9}f\left(\sqrt{\frac{3}{5}}\right) + \frac{5}{9}f\left(-\sqrt{\frac{3}{5}}\right) \quad (0)$$

Înlocuim funcția $f(x)$ cu $x^2 + 4x$ și calculăm:

$$\int_{-1}^1 (x^2 + 4x) dx \approx \frac{8}{9}(0^2 + 4 \cdot 0) + \frac{5}{9} \left(\left(\sqrt{\frac{3}{5}} \right)^2 + 4 \cdot \sqrt{\frac{3}{5}} \right) + \frac{5}{9} \left(\left(-\sqrt{\frac{3}{5}} \right)^2 + 4 \cdot (-\sqrt{\frac{3}{5}}) \right) \quad (1)$$

$$= \frac{8}{9} \cdot 0 + \frac{5}{9} \left(\frac{3}{5} + 4 \cdot \sqrt{\frac{3}{5}} \right) + \frac{5}{9} \left(\frac{3}{5} - 4 \cdot \sqrt{\frac{3}{5}} \right) \quad (2)$$

$$= \frac{5}{9} \left(\frac{3}{5} + 4 \cdot \sqrt{\frac{3}{5}} + \frac{3}{5} - 4 \cdot \sqrt{\frac{3}{5}} \right) \quad (3)$$

$$= \frac{5}{9} \cdot \frac{6}{5} \quad (4)$$

$$= \frac{2}{3} = \textcolor{blue}{0.667} \quad (5)$$

Calculul erorii absolute și relative:

- Rezultatul corect: 0.667
- Rezultatul metodei: 0.667

Calculăm erorile:

$$\begin{aligned}\text{Eroarea absolută} &= |\text{rezultat corect} - \text{rezultat metodei}| \\ &= |0.667 - 0.667| \\ &= 0\end{aligned}$$

$$\begin{aligned}\text{Eroarea relativă} &= \left| \frac{\text{rezultat corect} - \text{rezultat metodei}}{\text{rezultat corect}} \right| \times 100\% \\ &= \left| \frac{0.667 - 0.667}{0.667} \right| \times 100\% \\ &\approx 0\%\end{aligned}$$

3.5. Runge-Kutta 4 (RK4)

Metoda Runge-Kutta de ordinul al patrulea (RK4) este utilizată pentru rezolvarea ecuațiilor diferențiale ordinare (ODE) cu forma: [15]

$$\frac{dy}{dt} = f(t, y)$$

cu condiția inițială $y(t_0) = y_0$.

Metoda RK4 estimează soluția y la $t = t_0 + h$ folosind următorii pași: [15]

1. Calcularea primei pante intermediare:

$$k_1 = h f(t_n, y_n)$$

Această pantă se bazează pe valoarea funcției la începutul intervalului.

2. Calcularea celei de-a doua pante intermediare: [15]

$$k_2 = h f \left(t_n + \frac{h}{2}, y_n + \frac{k_1}{2} \right)$$

Această pantă folosește punctul de mijloc $(t_n + \frac{h}{2})$ și estimează valoarea funcției la acest

punct de mijloc folosind $y_n + \frac{k_1}{2}$.

3. Calcularea celei de-a treia pante intermediare: [15]

$$k_3 = hf \left(t_n + \frac{h}{2}, y_n + \frac{k_2}{2} \right)$$

Similar cu k_2 , această pantă folosește punctul de mijloc, dar îmbunătățește estimarea folosind $y_n + \frac{k_2}{2}$ în schimb.

4. Calcularea celei de-a patra pante intermediare: [15]

$$k_4 = hf(t_n + h, y_n + k_3)$$

Această pantă se bazează pe valoarea funcției la sfârșitul intervalului folosind estimarea $y_n + k_3$.

5. Combinarea pantelor intermediare pentru a actualiza soluția: [15]

$$y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

Această medie ponderată a pantelor oferă o estimare mai precisă a lui y la $t = t_n + h$. [15]

3.5.1. Algoritm

Dață o problemă de valoare initială:

$$\frac{dy}{dt} = f(t, y), \quad y(t_0) = y_0$$

1. Initializează: t_0 , y_0 , și alege pasul h .
2. Pentru $n = 0, 1, 2, \dots, N - 1$ (unde N este numărul de pași):
 - (a) Calculează prima pantă: $k_1 = hf(t_n, y_n)$.
 - (b) Calculează a doua pantă: $k_2 = hf(t_n + \frac{h}{2}, y_n + \frac{k_1}{2})$.
 - (c) Calculează a treia pantă: $k_3 = hf(t_n + \frac{h}{2}, y_n + \frac{k_2}{2})$.
 - (d) Calculează a patra pantă: $k_4 = hf(t_n + h, y_n + k_3)$.
 - (e) Actualizează soluția: $y_{n+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$.
3. Incrementează timpul: $t_{n+1} = t_n + h$.
4. Repetă până când se atinge timpul final dorit t .

Capitolul 4. Aplicația IntegrX

Acest capitol va prezenta tehnologiile utilizate pentru crearea aplicației **IntegrX**, împreună cu funcționalitățile sale. Aplicația a fost creată folosind limbajul de programare **Java**, cu integrarea motorului de calcul **MATLAB Engine**. Am folosit mediul de dezvoltare integrat (IDE) **IntelliJ IDEA 2024.1** pentru a dezvolta aplicația, acesta fiind un instrument potrivit pentru crearea de aplicații în Java. Pentru crearea interfeței grafice (GUI), s-a folosit biblioteca **JavaFX**, cunoscută pentru capacitatea sa de a crea interfețe moderne și plăcute estetic.

4.1. Tehnologii și Securitatea lor

Tehnologiile utilizate pentru implementarea aplicației destinate calculului integralelor definite prin metode numerice sunt următoarele:

- Java;
- MATLAB;

4.1.1. JAVA

Java este un limbaj de programare orientat pe obiecte care a fost dezvoltat în 1995 de **Sun Microsystems**. Deoarece este simplă, sigură și ușor de transferat de la un sistem la altul, platforma Java permite dezvoltatorilor să creeze software compatibil cu mai multe sisteme. [16]

Avantaje

Java are o serie de avantaje semnificative, inclusiv:

- **Portabilitate:** Aplicațiile Java pot fi rulate pe orice dispozitiv cu o Mașină Virtuală Java (JVM) instalată, indiferent de arhitectura hardware sau sistemul de operare; [17]
- **Orientarea pe obiecte:** Ușurează modularizarea și gestionarea codului și permite reutilizarea; [17]

- **Securitatea:** Datorită modelului său de securitate încorporat și a JVM-ului său, care verifică codul înainte de a fi executat, Java este considerat un limbaj sigur; [17]
- **Execuție Multiplă:** Deoarece Java suportă execuție multiplă (**multithreading**) nativ, se pot crea aplicații care pot îndeplini mai multe sarcini în același timp; [17]

Vulnerabilități

Chiar dacă Java este apreciat pentru modelul său de securitate integrat, nu poate fi considerat complet protejat de vulnerabilități.

- **Log4j:** CVE-2021-44228, CVE-2021-45046, CVE-2021-45105, Vulnerabilitatea Log4J sau Log4Shell a fost o breșă de securitate semnificativă în biblioteca de înregistrare a evenimentelor Java Apache Log4j. Vulnerabilitatea permite atacatorilor să execute cod de la distanță, compromițând securitatea sistemelor.

NIVEL RISC: CRITIC - 10 [18]

- **Oracle Java SE:** CVE-2023-22081, vulnerabilitatea afectează Oracle Java SE, permitând unui infractor neautentificat să provoace o denegare parțială a serviciului prin HTTPS.

NIVEL RISC - MEDIU 5.3 [19]

- **JavaFX:** CVE-2020-14664, vulnerabilitatea dificilă de exploatație permite unui atacator neautentificat să compromiță Java SE folosind mai multe protocoale de acces la rețea. Pentru ca un atac să funcționeze, el necesită o interacțiune umană dincolo de atacator.

NIVEL RISC - RIDICAT 8.3 [18]

Soluții

Soluțiile pentru vulnerabilitățile CVE-2021-44228, CVE-2021-45046 și CVE-2021-45105 sunt următoarele:

- Actualizarea bibliotecii Log4j la cea mai recentă versiune disponibilă (2.17.0 sau mai nouă);
- Să se asigure că fiecare componentă a software-ului și bibliotecilor externe utilizate este actualizată cu corecții (patch) de securitate.

Pentru a reduce riscul de exploatare a vulnerabilităților cunoscute, este necesar să se

mențină un proces regulat de gestionare a patch-urilor.

4.1.2. MATLAB

MATLAB, o platformă puternică dezvoltată de MathWorks, permite manipularea matricelor, trasarea funcțiilor și a datelor, implementarea algoritmilor, analiza datelor, dezvoltarea algoritmilor și crearea modelelor și aplicațiilor, inclusiv calculul integral.[20]

În MATLAB se pot face următoarele calcule: [20]

- Metode de rezolvare a ecuațiilor diferențiale;
- Calculul și diagrama funcțiilor matematice;
- Analiza și interpolarea datelor;
- Implementarea metodelor de optimizare numerică;
- Calcularea vectorilor și valorilor proprii ai unei matrice;
- Transformările;
- Integrarea numerică a funcțiilor și calculul de integrale definite;

Avantaje

MATLAB are multe avantaje, cum ar fi:

- **Capacitatea de a se integra:** MATLAB se integrează ușor cu Python sau Java și alte limbi și aplicații de programare; [20]
- **Performanță și Eficiență:** Prin utilizarea tehnicii avansate de optimizare și calcul paralel, MATLAB oferă performanță și eficiență ridicate în rezolvarea problemelor complex; [20]
- **Instrumente de analiză și vizualizare:** MATLAB oferă instrumente de analiză și vizualizare a datelor, inclusiv funcții de analiză statistică și trasee grafice personalizabile; [20]

Vulnerabilități

MATLAB are unele vulnerabilități, ca și toate limbajele de programare:

- **Jenkins:** CVE-2023-49656, pluginul Jenkins pentru MATLAB 2.11.0 și versiunea anterioară este vulnerabil la atacuri XXE din cauza configurației inadecvate a parser-ului XML.

NIVEL RISC: CRITIC - 9.8 [21]

Soluții

Soluția pentru vulnerabilitatea CVE-2023-49656:

- Să se asigure că fiecare componentă a software-ului și bibliotecilor externe utilizate este în conformitate cu ultimele corecții de securitate. Pentru a reduce riscul de utilizare a vulnerabilităților cunoscute, este esențial să se mențină un proces de gestionare a patchurilor în mod regulat.

În capituloarele următoare se vor prezenta tehnologiile mai în detaliu: JAVA și MATLAB.

Prezentarea va include fragmente de cod din aplicația **IntegrX**.

4.2. JAVA

In acest subcapitol se va discuta limbajul de programare Java. Vor fi discutate conceptele fundamentale de programare orientată pe obiecte (OOP), cum ar fi clasele și obiectele, moștenirea, polimorfismul și encapsularea.

4.2.1. Clase și Obiecte

O clasă servește ca model pentru crearea obiectelor și conține o definiție a unui tip de date specific, specificând atât atributele (variabilele de instanță) cât și comportamentele (metodele) pe care obiectele le vor avea.

```

public class TratareErori { 17 usages ± iamvorum+1*
    !
    // Atribut static pentru a stoca instanta clasei ErrorHandling
    private static TratareErori instance = null; 3 usages

    // Constructorul este privat pentru a preveni instantierea obiectelor in afara clasei
    private TratareErori() { 1 usage ± iamvorum
    }

    // Metodă statică pentru a obține o instanță a clasei ErrorHandling (Singleton pattern)
    public static TratareErori getInstance() { 7 usages ± Xorum+
        if (instance == null) {
            instance = new TratareErori();
        }
        return instance;
    }

    // Metodă statică pentru a gestiona cazurile în care câmpurile sunt goale
    public static boolean handleEmptyFields(String... fields) {...}

```

Figura 4.1: Exemplu Clasă Tratare Erori

Pentru a asigura că o clasă are o singură instanță, codul folosește un **singleton**. Acest lucru asigură coerența datelor din aplicație și economisește resurse.

- **Atribut:** Un atribut este o variabilă definită într-o clasă, care reprezintă o caracteristică a obiectului creat din acea clasă.
- **Constructor:** Un constructor este o metodă specială folosită pentru inițializarea unui nou obiect. Constructorul are același nume ca și clasa și nu are tip de return.
- **Metodă:** O metodă este o funcție definită într-o clasă, care descrie un comportament al obiectului și poate manipula atributele acestuia sau efectua acțiuni specifice.

Un obiect este o instanță a unei clase și reprezintă o entitate specifică care a fost creată în funcție de şablonul definit de clasă. Se poate accesa și manipula atributele și metodele definite în clasă prin obiecte.

```

TratareErori tratareErori = null;
try {
    tratareErori = TratareErori.getInstance();
} catch (Exception e) {
    // Afisează o alertă în caz de eroare
    tratareErori.showAlert(title: "Eroare", e.getMessage());
    return;
}

```

Figura 4.2: Exemplu Obiect Tratare Erori

Mai jos este funcția ”showAlert()” care afișează o alertă în cazul unei erori.

```

// Metodă statică pentru a afisa un dialog de avertizare
public static void showAlert(String title, String message) {
    // Creează un nou dialog de avertizare cu tipul WARNING
    Alert alert = new Alert(Alert.AlertType.WARNING);
    alert.setTitle(title); // Setează titlul dialogului
    alert.setHeaderText(null); // Nu se folosește antetul dialogului
    alert.setContentText(message); // Setează mesajul de continut al dialogului

    // Aplică stiluri CSS la panoul de dialog
    DialogPane dialogPane = alert.getDialogPane();

    // Initializează stilul dialogului
    alert.initStyle(StageStyle.UNDECORATED);

    // Încarcă fisierul CSS pentru utilizare
    URL cssURL = IntegrX.class.getResource( name: "/integrX/css/fullstyle.css"); // Asigură-te că calea este corectă
    if (cssURL != null) {
        dialogPane.getStylesheets().add(cssURL.toExternalForm());
    } else {
        System.out.println("Nu s-a putut găsi fisierul CSS.");
    }

    // Adaugă o clasă la panoul dialogului
    dialogPane.getStyleClass().add("dialog-pane");

    // Adaugă o clasă la textul continutului
    dialogPane.lookup(".content").getStyleClass().add("content-text");

    // Afisează dialogul și așteaptă până când utilizatorul interacționează cu acestea
    alert.showAndWait();
}

```

Figura 4.3: Funcția de afisare unei erori

4.2.2. Declarație și Inițializare

În Java, variabilele pot fi declarate și inițializate în același timp sau pot fi declarate mai întâi și apoi inițializate mai târziu.

```

// Obține valorile introduse în câmpuri
String function = "";
function = func_id.getText();
String min = min_id.getText();
String max = max_id.getText();
String interval = int_id.getText();
String plot_interval = plot_interval_id.getText();
String abs_err_usr = abs_err_input.getText();

```

Figura 4.4: Exemplu de Declarație și Inițializare

Variabilele sunt inițializate folosind valorile din interfața grafică de utilizator, aşa cum se arată în imaginea de mai sus.

4.2.3. Tipuri de Date

Java are două categorii principale de tipuri de date: primitive și obiecte.

- **Primitive:** Valorile individuale sunt stocate în programe Java folosind tipurile de date primitive, cum ar fi int pentru numere întregi, double pentru numere zecimale, boolean pentru valori de adevărat sau fals, char pentru caractere Unicode, byte și short pentru numere întregi mici și long pentru numere întregi mari.
- **Obiecte:** Acestea sunt instanțe ale claselor. Acestea pot stoca atât date, cât și metode pentru a lucra cu aceste date.

4.2.4. Control Flux

Controlul fluxului în Java se referă la modul în care programul ia decizii și execută acțiuni în funcție de anumite condiții. Aceasta poate fi realizat cu ajutorul structurilor de control cum ar fi instrucțiunile if-else, buclele for, while și switch.

- **IF:** Instrucțiunea „if” este utilizată pentru a executa un bloc de cod dacă o anumită condiție este evaluată ca fiind adevărată.
- **ELSE:** Pentru a executa un bloc de cod atunci când condiția specificată în instrucțiunea „if” este evaluată ca fiind falsă, instrucțiunea „else” este folosită împreună cu instrucțiunea „if”.
- **FOR:** Bucla „for” este utilizată pentru a executa un bloc de cod de un număr specificat de ori. Aceasta conține trei părți: inițializarea, condiția și expresia de incrementare/decrementare.
- **WHILE:** Un bloc de cod poate fi executat atâtă timp cât o anumită condiție este considerată adevărată prin utilizarea butonului „while”.
- **DO WHILE:** Bucla „do while” și bucla „while” sunt identice, dar condiția este verificată la sfârșitul ambelor bucle, asigurându-se că blocul de cod este executat cel puțin o dată.
- **SWITCH:** În funcție de valoarea unei expresii, un bloc de cod poate fi executat folosind instrucțiunea „switch”. Aceasta oferă o alternativă mai simplă și mai ușor de înțeles la un set de comenzi „if-else”.
- **TRY:** Blocul „try” este utilizat pentru a defini un bloc de cod în care pot apărea excepții. Aici este plasat codul care ar putea genera o excepție.
- **CATCH:** Blocul „catch” este utilizat pentru a trata (prinde) o excepție specificată. Este utilizat după blocul „try” și permite programatorului să gestioneze excepțiile care apar în timpul execuției codului din blocul „try”.

4.2.5. Moștenire

În JAVA (în general, în programarea orientată pe obiecte), principiul de moștenire permite unei clase să moștenească caracteristicile (câmpuri și metode) ale unei alte clase.

```

// Metoda pentru calculul integralei din clasa Integrare (parinte) implementata din InterfataIntegrare
// Metoda este suprascrisa in clasele copii pentru a implementa metode specifice de calcul al integralei
@Override // usage: & Xorun+1
public double integrate(String function, String min, String max) throws MatlabExecutionException, MatlabSyntaxException {
    try {
        engine.eval("syms x"); // Definiresa lui x ca simbol
        // Formatarea corecta a expresiei de integrare a passa functia si limitele
        String integral_expr = function + ", " + min + ", " + max;
        String integrationScript = "integralResult = integral(@(x) " + integral_expr + ");";
        engine.eval(integrationScript);
    } catch (InterruptedException | ExecutionException ex) {
        TratareErori.tratareErori = TratareErori.getInstance();
        tratareErori.showAlert("Eroare", message: "A apărut o eroare la calcularea integralei.\n" + ex.getMessage());
    }
    double integralResult;
    try {
        integralResult = ((Double) engine.getVariable("integralResult")).doubleValue(); // Obtinerea rezultatului integratiei
    } catch (InterruptedException | ExecutionException ex) {
        throw new RuntimeException("Nu s-a reusit obtinerea rezultatului integratiei: " + ex.getMessage(), ex);
    }
    return integralResult;
}

```

Figura 4.5: Exemplu de Moștenire

În aplicație, de exemplu, avem funcția ”integrate()” implementată în clasa părinte **Integrare** din interfața **InterfataIntegrare**. Aceasta este suprascrisă de către clasele **IntegrareRectangulara**, **IntegrareSimpson**, **IntegrareRungeKutta** și **IntegrareTrapezoidalala**. Pentru detalii suplimentare, puteți consulta codul din Anexă.

4.2.6. Polimorfism

Prin polimorfism, o interfață comună poate trata obiecte de diferite tipuri. Polimorfismul dinamic și polimorfismul static sunt cele două categorii de polimorfism.

```

// Metoda pentru calculul integral folosind metoda dreptunghiulară mijloc
public double integrate(String function, String min, String max, String plotInterval, String interval) throws MatlabExecutionException, MatlabSyntaxException {
    // Metoda pentru calculul integral folosind metoda dreptunghiulară stanga sau dreapta
    public double integrate(String function, String min, String max, String plotInterval, String interval, int type) throws MatlabExecutionException, MatlabSyntaxException {
}

```

Figura 4.6: Metoda de integrare supraîncărcată

```

int type = 2; // 2 = mijloc, 1 = stanga, 0 = dreapta
// Selectează și calculează integrala folosind metoda corespunzătoare
if(rectangularButton.isSelected() || rectangularButton_lft.isSelected() || rectangularButton_rgt.isSelected()){
    if(rectangularButton.isSelected()){
        method_result_name.setText("METODA DREPTUNGHIULARĂ MIJLOC");
    } else if(rectangularButton_lft.isSelected()){
        method_result_name.setText("METODA DREPTUNGHIULARĂ STÂNGĂ");
        type = 1;
    } else if(rectangularButton_rgt.isSelected()){
        method_result_name.setText("METODA DREPTUNGHIULARĂ DREAPTA");
        type = 0;
    }
    if(type == 2){
        result = ri.integrate(function, min, max, plot_interval, interval);
    } else {
        result = ri.integrate(function, min, max, plot_interval, interval, type);
    }
}

```

Figura 4.7: Utilizarea metodelor de integrare supraîncărcate

Metodele care au același nume și sunt din aceeași clasă sunt cunoscute sub numele de metode supraincarcate și pot fi utilizate în diferite contexte, în funcție de argumentele oferite.

Metodele suprascrisse sunt variante ale unei metode moștenite care au fost redefinite într-o clasă derivată pentru a oferi un comportament anume. În figura din capitolul

Moștenire a fost introdusă actiunea de suprascriere. 4.5

4.2.7. Encapsulare

Termenul ”encapsulare” se referă la procesul de ascundere a detaliilor implementării, în timp ce se dezvăluie o interfață publică. De obicei, atributele unei clase sunt private, iar accesul la ele se face prin metode publice de căutare și plasare (**getter și setter**). În aplicația IntegrX, s-au utilizat funcțiile **getText()** și **setText()** implicate din JavaFX pentru a obține și a seta textul în interfață grafică.

4.2.8. Interfețe și Clase Abstracte

O interfață dictează clasei implementarea unui set de metode, dar nu oferă implementarea. În aplicația IntegrX, interfața este implementată de clasa Integrare, unde atât atributele cât și metodele sunt utilizate și implementate de această clasă și de clasele moștenitoare.

```
public interface Integrare {
    // Constanta pentru pragul de eroare
    double THRESHOLD = 0.000001; no usages

    // Motorul Matlab folosit pentru calcul
    MatlabEngine engine = Matlab_Multithread.getEngine(); 38 usages

    public double integrate(String Function, String min, String max) throws ExecutionException, InterruptedException; 1 usage 1 implementation new *

    // Metoda pentru determinarea daca o functie este divergentă
    String isDivergent(String function) throws ExecutionException, InterruptedException; 1 usage 1 implementation 4 xorum

    // Metoda pentru realizarea unui grafic al unei functii
    void plotting(String function, String min, String max, String plot_interval) throws ExecutionException, InterruptedException; 1 usage 1 implementation 4 xorum
}
```

Figura 4.8: Exemplu de Interfață

O clasă abstractă este un tip de clasă care nu poate fi instantiată direct și este destinată să fie moștenită de alte clase, definind metode abstrakte care trebuie să fie suprascrisse de clasele derivate.

4.2.9. Excepții

Un mod de a gestiona erori în timpul execuției programului este tratarea excepțiilor. Pentru a gestiona excepțiile, Java folosește blocurile try, catch, finally și cuvântul cheie throw.

```

try {
    // Calculează integrala
    result = ni.integrate(function, min, max);
} catch (Exception e) {
    // Afisează o alertă în caz de eroare
    tratareErori.showAlert("Eroare", message:"A apărut o eroare la calcularea integralei.\n" + e.getMessage());
    // Tratează cazul în care apare o excepție și ascunde diviziunile
    integral_div.setVisible(false);
    method_div.setVisible(false);
    error_div.setVisible(false);
    return;
}

```

Figura 4.9: Exemplu de Exceptie

O secțiune din codul sursă al aplicației este afișată în exemplul de mai sus. Acest lucru permite ca o alertă să fie afișată și procesul de afișare a rezultatelor să fie întrerupt în cazul unei erori în calculul integralei.

4.2.10. JavaFX

JavaFX este o tehnologie de dezvoltare GUI (Graphical User Interface) utilizată în Java care este concepută pentru a crea aplicații cu interfețe utilizator atractive și interactive. Deoarece JavaFX oferă o gamă largă de controale grafice, suport pentru media, animații și stilizare CSS, este mai ușor să se creeze interfețe moderne și dinamice. Interfața aplicației a fost creată folosind JavaFX în programul de licență, oferind utilizatorilor o experiență vizuală îmbunătățită și interacțiuni ușor de înțeles. Acest lucru asigură un produs final profesional și user-friendly.

```

<application title="result.fxml" alignment="center" style="font-size: 1em; font-weight: bold; font-family: sans-serif; color: black; background-color: white; border: 1px solid black; padding: 10px; margin: auto; width: fit-content; height: fit-content;>
    <script>
        window.onload = function() {
            var button = document.getElementById("button");
            button.addEventListener("click", function() {
                var resultText = document.getElementById("resultText");
                resultText.textContent = "Resultat";
            });
        };
    </script>
    <graphical-element id="button" style="width: 100px; height: 50px; border: 1px solid black; border-radius: 5px; background-color: #f0f0f0; font-size: 1em; font-weight: bold; color: black; text-align: center; line-height: 50px; margin-bottom: 10px; cursor: pointer;">
        <text>Răsărit!</text>
    </graphical-element>
</application>

```

Figura 4.10: Buton in JavaFX

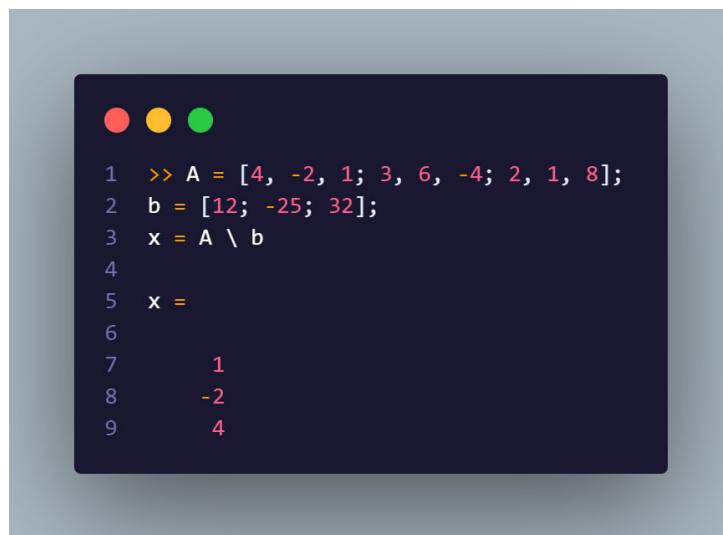
Codul pentru crearea butonului care este folosit de aplicația IntegrX pentru a afișa fereastra de rezultat este prezentat în imaginea de mai sus. Interfața rezultatului, inclusiv butonul rezultatului, este prezentată în figura din subcapitolul **Interfața de rezultat**. 4.27

4.3. MATLAB

În această subsecțiune, ne concentrăm pe prezentarea sintaxei și a anumitor comenzi în MATLAB, un limbaj de programare care se concentrează pe matematică.

4.3.1. Ecuății liniare

Sistemele de ecuații liniare de forma $Ax = b$, în care A este o matrice pătrată și b este un vector, pot fi rezolvate folosind operatorul \|. Rezultatul este stocat în x.



```

1 >> A = [4, -2, 1; 3, 6, -4; 2, 1, 8];
2 b = [12; -25; 32];
3 x = A \ b
4
5 x =
6
7     1
8     -2
9     4

```

Figura 4.11: Ecuății liniare

4.3.2. Funcții Anonime

MATLAB permite funcțiilor anonte să fie definite într-o singură linie, ceea ce înseamnă că nu este necesară crearea unui fișier separat. Acestea sunt utile pentru definirea rapidă și succintă a funcțiilor elementare.

Exemplu de funcție anonimă: $f = @(x)expresie$

4.3.3. Integrare Numerică

Integrarea numerică se referă la calculul aproximativ al unei integrale definite. În MATLAB, funcția *integral* este folosită pentru a calcula integrale definite pentru funcții continue.



```

1 >> a = 0;
2 >> b = 10;
3 >> f = @(x) cos(x);
4 >> integral(f,a,b)
5
6 ans =
7
8 -0.5440

```

Figura 4.12: Integrare Numerică

4.3.4. Plot și Mesh

Plot

MATLAB folosește funcția „plot” pentru a produce grafice de date în două dimensiuni care arată relația dintre două variabile pe axele x și y.

Mesh

MATLAB folosește funcția „mesh” pentru a produce grafice tridimensionale. Aceste grafice sunt formate din linii care conectează punctele generate pe o grilă bidimensională. Acest lucru oferă o reprezentare grafică a funcției $z = f(x, y)$. Acest grafic este util pentru a vedea variațiile și relațiile complicate dintre cele trei variabile.

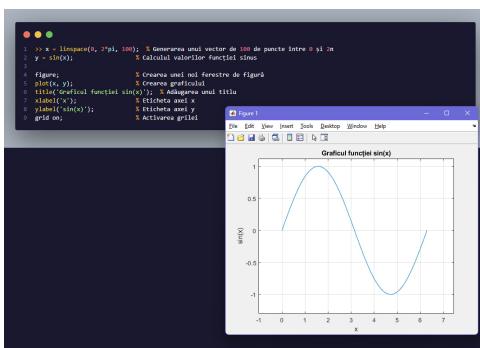


Figura 4.13: Exemplu plot

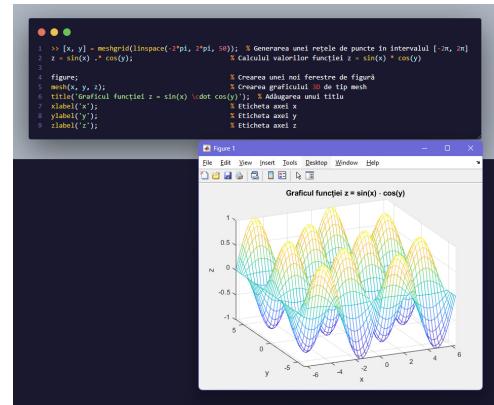


Figura 4.14: Exemplu mesh

4.4. IntegrX

În această secțiune vom prezenta aplicația IntegrX, dezvoltată în mediul de programare Java cu integrare Matlab, destinată metodelor numerice de calcul al integralelor definite. Pentru partea de interfață grafică s-a optat pentru utilizarea JavaFX.

4.4.1. Structura Aplicației

Această aplicație folosește clasele care urmează pentru a o permite să funcționeze:

- Clasa de incarcare: este activată la pornirea programului pentru a permite motorului Matlab să fie încărcat în aplicație;
- Clasa de integrare: este o clasă care utilizează funcția de integrare MATLAB „integral” pentru calcularea integralei simbolice și afișarea funcțiilor și graficelor în format LATEX.
- Clasa de integrare dreptunghiulară: permite integrarea prin metode dreptunghiulare (stanga, mijloc, dreapta) și afișează graficul integralei;
- Clasa de integrare trapezoidală: permite integrarea prin metoda cu trapezi și afișează graficul integralei;
- Clasa de integrare Simpson: permite integrarea prin metoda Simpson 1/3 și afișează graficul integralei;
- Clasa MATLAB MultiThread: reprezintă un fir de execuție Java care initializează și supraveghează o instanță a motorului MATLAB. Pentru a garanta o execuție eficientă și fără blocare a codului Java, aceasta utilizează metoda „startMatlab()” pentru a începe motorul MATLAB într-un canal separat;
- Clasa de tratare a erorilor: aceasta este folosită pentru a afisa erori în cazul în care o intervenție chirurgicală nu este posibilă;

- Clasa de control principal reprezintă nucleul programului, unde are loc întreaga sa logică și funcționare;
- Clasa IntegrX: aceasta este clasa de la care pornește programul;
- Interfata de Integrare: această interfață definește un set de metode și constante utilizate în procesul de integrare numerică.

Diagrama UML (Unified Modeling Language) a aplicației IntegrX este prezentată mai departe. Această diagramă prezintă structura și relațiile dintre diferitele componente și clase care formează aplicația.



Figura 4.15: Diagrama UML

4.4.2. Interfața Inițială

Fereastra de „incarcare” va fi afișată la pornirea programului. Această fereastră este rulată în mai multe thread-uri împreună cu motorul MATLAB, unde se așteaptă să funcționeze fără a bloca aplicatia Java.



Figura 4.16: Fereastra de Incarcare

Fereastră de ”Introducere” a MATLAB servește ca un ghid scurt care explică ce se poate face cu aplicația.

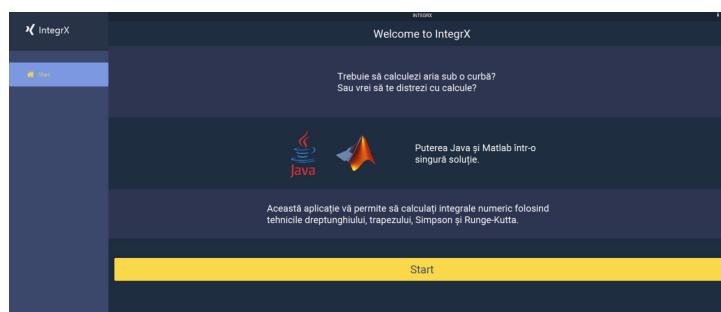


Figura 4.17: Fereastra de Start

După ce se tastează pe butonul clic pe Start, veți vedea o secțiune pentru introducerea datelor pentru calculul integralei unei funcții.

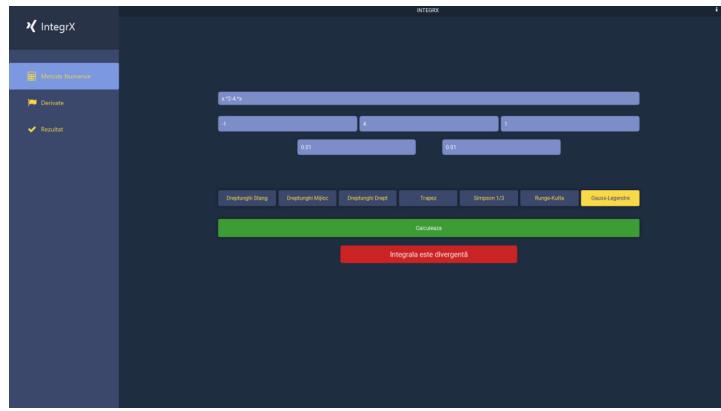


Figura 4.18: Formular

Vom găsi un formular în această fereastră în care să introducem următoarele:

- Functie: În acest camp se va introduce oricare functie dorîti pentru a calcula integrala definita. **De reținut este că funcția trebuie să fie în sintaxa MATLAB;** nu acceptă funcții complexe și acceptă doar funcții cu o singură variabilă.
 - $x^2 + 3x - 1$ se va scrie in următorul format $x.^2 + 3. \cdot x - 1$;

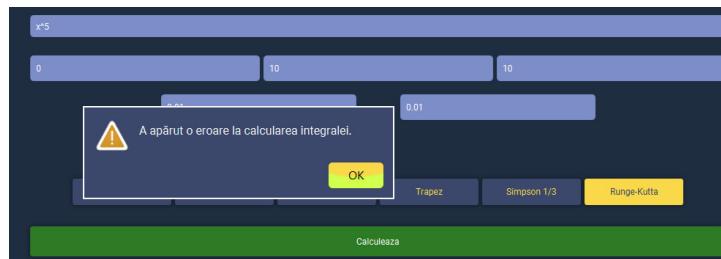


Figura 4.19: Fereastra de Eroare Functie

- $\sin(x)$ este corect și in sintaxa MATLAB;
- e^{-x} se va scrie in următorul format $e.^{-x}$
- $x^2 + y^2$ nu va funcționa, deoarece aplicația se ocupă doar de funcții care implică o singură variabilă;

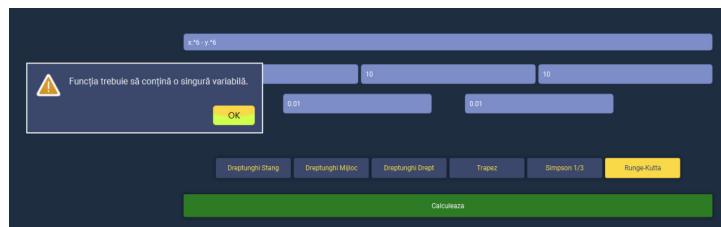


Figura 4.20: Fereastra de Eroare Singură Variabilă

- $z^2 + 3i$, unde $z = x + yi$ și i este unitatea imaginară. Nu va funcționa, deoarece are două variabile și numere complexe.



Figura 4.21: Fereastra de Eroare Numere Complexe

- Minim (Min): Marginea stângă a intervalului integralei va fi introdusă în acest camp.
- Maxim (Max): Marginea dreaptă a intervalului integralei va fi introdusă în acest camp.

Notă: Valorile speciale „pi”, „e”, „phi” și „Inf” sunt acceptate în câmpurile Minim și Maxim. Aplicația va realiza conversia acestora în valori de tip double, cu excepția valorii „Inf”, care va rămâne neschimbată și va genera o eroare deoarece aplicația va gestiona calculul integralelor definite.

```
// Obține valorile introduse în câmpuri
String function = func_id.getText();
String min = min_id.getText();
String max = max_id.getText();
String interval = int_id.getText();
String plot_interval = plot_interval_id.getText();
String abs_err_usr = abs_err_input.getText();

// Definirea unui map pentru a stoca conversiile
Map<String, String> conversionMap = new HashMap<>();
conversionMap.put("pi", String.valueOf(Math.PI));
conversionMap.put("e", String.valueOf(Math.E));
conversionMap.put("-pi", String.valueOf(-Math.PI));
conversionMap.put("phi", String.valueOf((1 + Math.sqrt(5)) / 2));
conversionMap.put("-phi", String.valueOf(-(1 + Math.sqrt(5)) / 2));
conversionMap.put("Inf", "Inf");
conversionMap.put("-Inf", "-Inf");

// Actualizarea lui min și max dacă sunt egale cu o cheie din map
if (conversionMap.containsKey(min)) {
    min = conversionMap.get(min);
}
if (conversionMap.containsKey(max)) {
    max = conversionMap.get(max);
```

Figura 4.22: Valori Speciale

- Intervale: Numărul de subintervale care vor fi folosite trebuie introdus în acest câmp.
- Interval Plot: Este folosit pentru a crea graficul funcției; Se folosește ca valoare incrementală de la minim la maxim (min:input:max); cu cât valoarea este mai mică, cu atât graficul va fi mai precis. Mai jos puteți vedea exemple între valorile 1 și 0.01:

```
@Override 1 usage + Xorum+1
public void plotting(String function, String min, String max, String plot_interval) throws ExecutionException, InterruptedException {
    String latexExpr;
    try {
        engine.eval("clear"); // Sterge orice obiecte sau grafice existente in mediul de lucru MATLAB
        engine.eval("x = " + min + ":" + plot_interval + ":" + max + ";"); // Definirea punctelor x
        engine.eval("y = " + function + ";"); // Evaluarea functiei pentru punctele x date
        engine.eval(MATLAB_PLOT_SKELETON); // Plasarea graficului

        engine.eval("syms x"); // Definirea lui x ca simbol

        // Convertirea functiei in LaTeX
        String latexScript = "latexFunction = latex(" + function + ");";
        engine.eval(latexScript);

        // Obtinerea expresiei LaTeX
        Object latexFunction = engine.getVariable("latexFunction");
        latexExpr = "\\int_{" + min + "}^{{" + max + "}} " + latexFunction; // Construirea expresiei LaTeX pentru integrala definită

        TexFormula formula = new TexFormula(latexExpr);
        Color newBlue = new Color(31, 94, 164);
        Color smokeWhite = new Color(250, 250, 250);
        formula.createPNG(TeXConstants.STYLE_DISPLAY,
                size,
                out + "/src/main/resources/Integrix/plots/funct_latex.png", // Salveaza imaginile LaTeX
                newBlue,
                smokeWhite);
    } catch (MatlabExecutionException | MatlabSyntaxException ex) {
        TratareErori.tratareErori = TratareErori.getInstance();
        tratareErori.showAlert("Eroare", "A apărut o eroare la calcularea integralei.\n" + ex.getMessage());
    }
}
```

Figura 4.23: Funcție pentru plot

```
// Sablonul pentru graficul MATLAB
private static String MATLAB_PLOT_SKELETON = 1 usage
    "fig = figure('Visible', 'off');\n" +
    "plot(x,y)\n" +
    "xlabel('x')\n" +
    "ylabel('y')\n" +
    "set(gcf, 'color', [0.1216 0.1765 0.2510]);\n" +
    "saveas(gcf, './src/main/resources/Integrix/plots/funct_plot_1s.png')";
```

Figura 4.24: řablon pentru plot

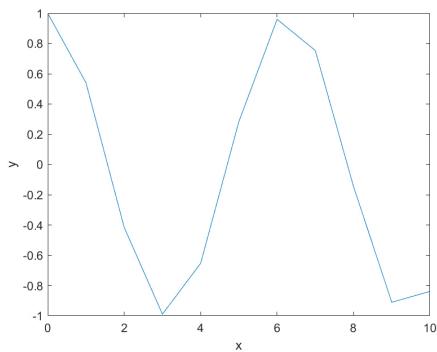


Figura 4.25: Exemplu cu 1

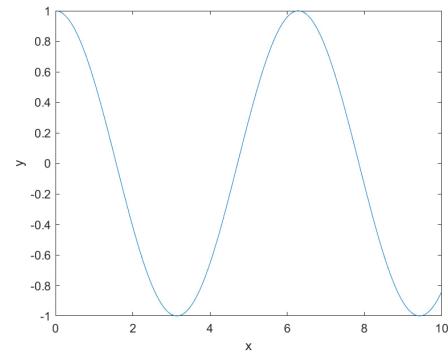


Figura 4.26: Exemplu cu 0.01

- Eroare Absolută: În acest câmp se va introduce limita erorii, astfel încât dacă rezultatul este mai mare decât această limită, nu va fi acceptat; dacă este mai mic, va fi considerat acceptabil.
- Grup de butoane care ne va permite alegerea metodei numerice de integrare, și anume:
 - Metoda dreptunghiului stânga: Calculează aria dreptunghiului folosind valoarea funcției la începutul fiecărui interval.
 - Metoda dreptunghiului mijloc: Calculează aria dreptunghiului folosind valoarea funcției la mijlocul fiecărui interval.
 - Metoda dreptunghiului dreapta: Calculează aria dreptunghiului folosind valoarea funcției la sfârșitul fiecărui interval.
 - Metoda trapezului: Utilizează trapeze pentru a aproxima aria sub graficul funcției. La capetele fiecărui interval, calculează media valorilor funcției.
 - Metoda Simpson 1/3: Utilizează parabole pentru a aproxima aria sub graficul funcției, combinând valori ale funcției la capete și la mijlocul intervalului.
 - Metoda Runge-Kutta (RK4): Este o abordare de ordinul patru avansată pentru rezolvarea ecuațiilor diferențiale care folosește evaluări multiple ale funcției pentru o precizie mai mare.

Formularul are cazuri de tratare a erorilor pentru următoarele situații:

- Câmpurile nu sunt toate completate și selectate;
- Intervalele nu sunt valide;
- Funcția nu este acceptată.

4.4.3. Interfața Rezultat

După ce se introduc datele în formular pentru a utiliza metoda preferată pentru calculul integralei, se va apăsa butonul „Calculează”. Acest lucru va începe procesul de calculare a integralei, care va genera un grafic pentru funcție și va evidenția aria specifică metodei integralei selectate. În plus, va calcula eroarea absolută și relativă a metodei pe baza limitei de eroare introdusă în formular. În plus, o altă fereastră va prezenta derivata funcției introduse de ordinul întâi și de ordinul doi, precum și faptul dacă funcția este convergentă sau divergentă.

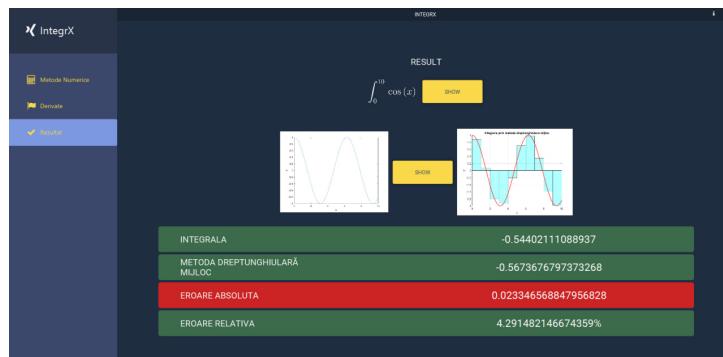


Figura 4.27: Fereastra Rezultat

Această fereastra are două butoane „SHOW”. Primul buton va afișa funcția în format .PNG (Portable Network Graphics), care vă va permite să o vizualizați mai detaliat și mai ușor. Această funcție este utilă în cazul funcțiilor mai lungi și complicate.

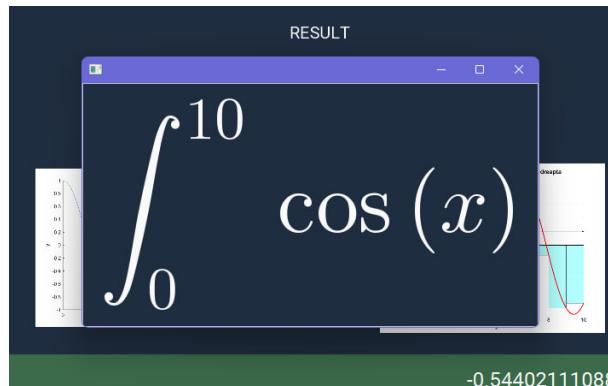


Figura 4.28: Vizualizare Functie

Pentru a vedea integrala și valorile de a lungul functiei mai detaliat, puteți apăsa al doilea buton de vizualizare „SHOW” și va deschide o fereastra MATLAB „figure”. Următoarele sunt câteva imagini ale integralei folosind toate cele șase metode disponibile pentru funcția $\cos(x)$:

- Calcularea integralei folosind funcția "integral" a Matlab și metoda dreptunghiulară la mijloc sunt prezentate în imaginea de mai jos. Se vede o comparație între rezultatele și eroarea calculată.

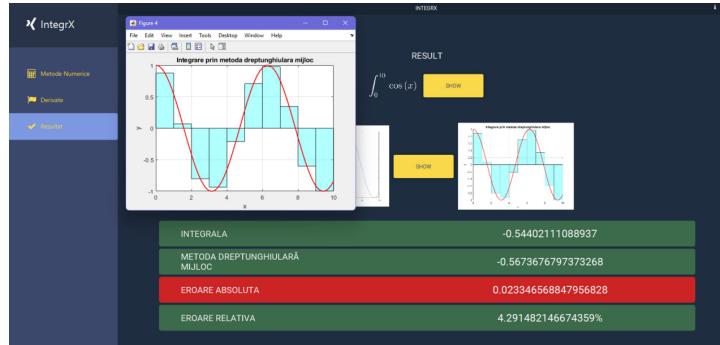


Figura 4.29: Metoda Dreptunghiurilor la Mijloc

- Calcularea integralei prin metodele dreptunghiulare la stânga și dreapta este prezentată în imaginea de mai jos. Acest lucru subliniază comparația între rezultatele și eroarea calculată.

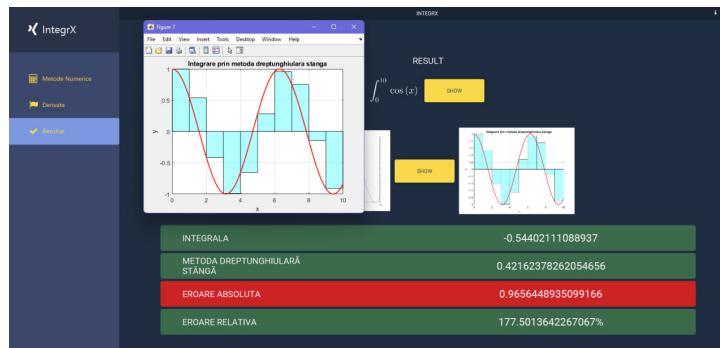


Figura 4.30: Metoda Dreptunghiurilor la Stânga

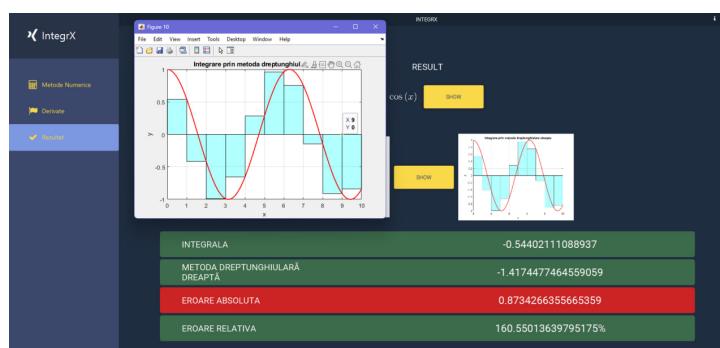


Figura 4.31: Metoda Dreptunghiurilor Dreapta

- Calcularea integralei cu metoda trapezelor este prezentată în imaginea de mai jos. Acest lucru subliniază comparația între rezultatele și eroarea calculată.



Figura 4.32: Metoda Trapezelor

- Calcularea unei integrale cu metoda numerică Simpson 1/3 și funcția Matlab integral sunt ilustrate în imaginile. Compararea vizuală între metoda Simpson și funcția integrată din Matlab este prezentată în grafic. În plus, codul include crearea unui grafic care poate fi folosit pentru a prezenta integrala calculată.

```
// Metoda pentru calculul integralei folosind metoda lui Simpson
public void integrate(String function, String min, String max, String plot_interval, String segments) throws MatlabExecutionException, MatlabSyntaxException {
    try {
        // Se evaualează simbolul 'x' în internal MATLAB
        engine.eval("syms x");
        // Se definește expresia integrală și scriptul de integrare
        String integralExpr = "(" + function + ") * " + min + " : " + max;
        String integralScript = "int(" + integralExpr + ", x);";
        String integralResult = eval(integralScript);
        double integralValue = Double.parseDouble(integralResult);

        // Se calculează segmentele
        double h = (max - min) / segments;
        double a = min;
        double b = max;
        double even = 0;
        double odd = 0;
        for (int i = 0; i < segments; i++) {
            if (i % 2 == 0) { // Even
                a += h;
                odd += f(a);
            } else { // Odd
                b -= h;
                even += f(b);
            }
        }

        // Se calculează integrala numerică
        double simpsonResult = (h / 3) * (integralValue + 2 * even + 4 * odd);
        double error = Math.abs(simpsonResult - integralValue);

        // Se generează graficul
        String plotScript = "f(x) = " + function;
        String plotInterval = min + ":" + max;
        String plotSegments = "segments = " + segments;
        String plotH = "h = (" + max + "-" + min + ")/" + segments;
        String plotEven = "even = " + even;
        String plotOdd = "odd = " + odd;
        String plotIntegral = "integralResult = " + integralResult;
        String plotError = "error = " + error;

        String plotString = plotScript + ";" + plotInterval + ";" + plotSegments + ";" + plotH + ";" + plotEven + ";" + plotOdd + ";" + plotIntegral + ";" + plotError;
        engine.eval(plotString);

        // Se salvează fișierul
        String saveFig = "savefig(fig, '" + figName + ".png');";
        engine.eval(saveFig);
    } catch (InterruptedException | ExecutionException ex) {
        Thread.currentThread().interrupt();
        Thread.currentThread().getStackTrace();
        ex.printStackTrace();
    }
}
```

Figura 4.33: Funcția Simpson 1/3



Figura 4.34: Rezultat Simpson 1/3

- Calcularea integralei folosind funcția ”integral” a Matlab și metoda de extrapolare Runge-Kutta (RK4) sunt prezentate în imaginea de mai jos. Se poate observa comparația între rezultatele și eroarea calculată.

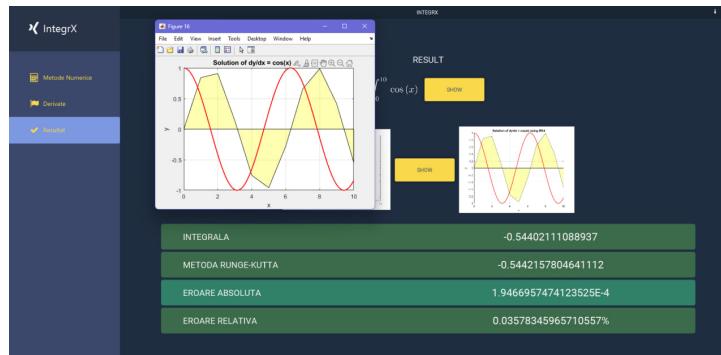


Figura 4.35: Metoda Runge-Kutta (RK4)

4.4.4. Interfață Derivate

Această interfață suplimentară afișează derivatele orfinului prim și al doilea, precum și dacă funcția este convergentă sau divergentă (în plus, pentru Cuadratura Gauss-Legendre, polinomul de gradul n unde n este numărul de puncte va fi afișat.).

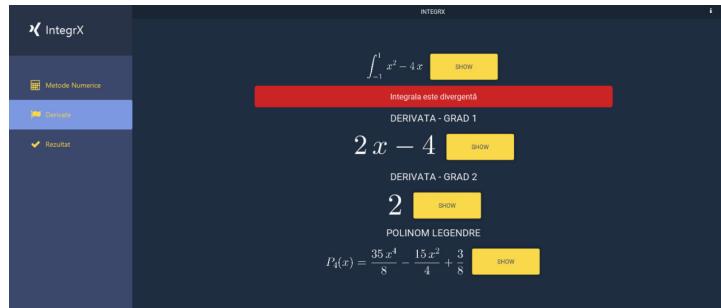


Figura 4.36: Fereastra Derivate

Convergență și Divergență

O integrală care suma ei este infinită este considerată divergentă dacă nu converge la o valoare finită. Dacă suma integralei tinde spre o sumă finită, integrala este convergentă. Mai jos aveți imaginea cu funcția care verifică dacă o integrală este convergentă sau divergentă în intervalul $(-\infty, \infty)$:

```
@Override Usage Xorum*
public String isDivergent(String function) throws ExecutionException, InterruptedException {
    // Initializează simbolul 'x' în motorul MATLAB
    engine.eval("$\"syms x\"");

    // Elimină punctele din funcție pentru a evita eroare
    String cleanedFunction = function.replaceAll("\\.", " ");

    // Evaluază funcția în motorul MATLAB
    engine.eval("$\"f = " + cleanedFunction + ";\"");

    // Calculă integrala indefinită a funcției pe intervalul (-inf, inf)
    engine.eval("$\"result = int(f, x, -inf, inf);\"");

    // Verifică dacă integrala este infinită
    engine.eval("$\"isDivergent = isnan(result);\"");

    // Obține rezultatul
    Object isDivergent = engine.getVariable("$\"isDivergent\"");

    // Returnează rezultatul ca string
    if (isDivergent.toString().equals("true")) {
        return "Integrala funcției este divergentă.";
    } else {
        return "Integrala funcției este convergentă.";
    }
}
```

Figura 4.37: Funcția de verificare a divergenței

Notă: MATLAB, unele functii care în realitate sunt divergente, le consideră convergente, deoarece există posibilitatea ca funcția integrata *integral()* să nu calculeze integrala corect dacă există o valoare în interval în care funcția are valoare +Inf sau -Inf.

Integrală Convergentă

Pe intervalul $[2, Inf)$, să luăm în considerare integrala funcției e^{-x} . Pe măsură ce x crește spre infinit, exponentiala negativă se apropie de zero, ceea ce demonstrează că integrala converge.

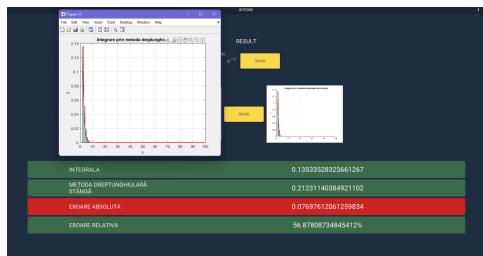


Figura 4.38: Exemplu convergentă interval $[2,100]$

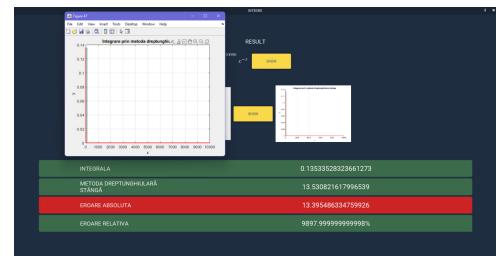


Figura 4.39: Exemplu convergentă interval $[2,10000]$

Integrală Divergentă

Acum luăm în considerare integrala funcției $\ln(x)$ pe intervalul $[0, Inf)$. Aceasta arată o integrală care diverge, adică nu se apropie de o valoare finită pe măsură ce limita superioară a intervalului de integrare se apropie de infinit.



Figura 4.40: Exemplu divergentă interval $[0,100]$

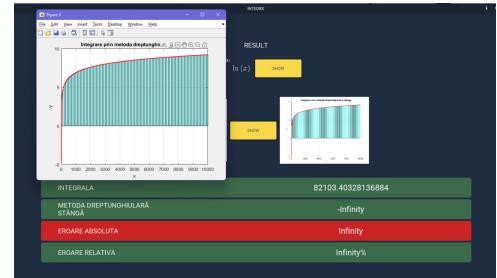


Figura 4.41: Exemplu convergentă interval $[0,10000]$

Capitolul 5. Concluzii

Utilizatorii pot efectua calculul integralelor definite continue cu o singură variabilă cu aplicația dezvoltată, care integrează funcționalitățile MATLAB într-o interfață JavaFX. Utilizarea aplicației nu numai că facilitează calculul integralelor definite precise, ci și facilitează analiza comparativă a performanțelor diferitelor metode numerice. Metode precum metoda Trapezelor, metoda Dreptunghiurilor, metoda Simpson, metoda RK4 și quadratura Gauss-Legendre sunt doar câteva metode implementate dintre metodele numerice care pot fi dezvoltate în viitor în acest program.

Interfața grafică simplă creata prin utilizarea JavaFX face ca aplicația să fie mai ușor de utilizat și de accesat. Acest lucru ajută la transformarea proceselor de calcul și analiză într-unul simplu și eficient.

5.1. Dezvoltare în viitor

Pentru viitor, am intenția de a implementa noi metode numerice, cum ar fi Quadratura Gauss-Konrod sau Formula Gauss-Cebisev. De asemenea, am intenția de a implementa logica matematică direct în Java, renunțând la proprietățile și instrumentele oferite de MATLAB, precum și de a realiza o posibilă remodernizare a interfetei grafice. În plus, dacă programul va reuși să integreze logica matematică în JAVA, are potențialul de a se extinde la alte domenii, mai degrabă decât calculul integralelor. De exemplu, aplicația poate dezvolta algoritme de criptare, cum ar fi ECC (Criptografia cu Curbe Eliptice), care este o tehnică de criptografie cu chei publice care utilizează proprietățile matematice ale curbelor eliptice pentru a crea o schimbare sigură a cheii.

Bibliografie

- [1] California State University, “Approximation and Errors.” [Online]. Available: <https://home.csusb.edu/~tebert/teaching/lectures/271/approx/approx.pdf>
- [2] Annette M. Burden, J. Douglas Faires, and Richard L. Burden, *Numerical Analysis*, 10th ed. Cengage Learning, 2015.
- [3] Ministerul Educatiei si Invatamantului - Universitatea Bucuresti, *Analiza Matematica*, 5th ed. Editura Didactica si Pedagogica Bucuresti, vol. 1.
- [4] Paul’s Online Notes, “The Mean Value Theorem.” [Online]. Available: <https://tutorial.math.lamar.edu/classes/calci/MeanValueTheorem.aspx>
- [5] Conf. Univ. Dr. Ing. Garban Valentin, *Analiza Matematica*. Editura Universitatii Titu Maiorescu, 2010.
- [6] Engineering at Alberta, “Introduction to Numerical Analysis for Engineers.” [Online]. Available: <https://engcourses-uofa.ca/books/numericalanalysis/>
- [7] Dummies, “Left Rectangular Method.” [Online]. Available: <https://www.dummies.com/article/academics-the-arts/math/calculus/how-to-approximate-area-with-left-rectangles-192249/>
- [8] —, “Midpoint Rectangular Method.” [Online]. Available: <https://www.dummies.com/article/academics-the-arts/math/calculus/how-to-approximate-area-with-midpoint-rectangles-192261/>
- [9] —, “Right Rectangle Method.” [Online]. Available: <https://www.dummies.com/article/academics-the-arts/math/calculus/how-to-approximate-area-with-right-rectangles-192144/>
- [10] —, “Trapezoid Method.” [Online]. Available: <https://www.dummies.com/article/academics-the-arts/math/calculus/how-to-approximate-area-with-the-trapezoid-rule-192248/>

- [11] Math24, “Simpson 1/3 Rule.” [Online]. Available: <https://math24.net/simpsons-rule.html>
- [12] Wikipedia, “Gauss–Legendre quadrature.” [Online]. Available: https://en.wikipedia.org/wiki/Gauss%E2%80%93Legendre_quadrature
- [13] Dumitru Ebanca, *Metode de Calcul Numeric*. Editura Sitech Craiova, 1994.
- [14] LibreText Mathematics, “Lesson 1: Theory of Gauss Quadrature Rule.” [Online]. Available: [https://math.libretexts.org/Workbench/Numerical_Methods_with_Applications_\(Kaw\)/7%3A_Integration/7.05%3A_Gauss_Quadrature_Rule_of_Integration](https://math.libretexts.org/Workbench/Numerical_Methods_with_Applications_(Kaw)/7%3A_Integration/7.05%3A_Gauss_Quadrature_Rule_of_Integration)
- [15] Swarthmore, “Fourth Order Runge-Kutta.” [Online]. Available: <https://lpsa.swarthmore.edu/NumInt/NumIntFourth.html>
- [16] IBM, “What is Java?” [Online]. Available: <https://www.ibm.com/topics/java>
- [17] Trung Tran, “Java Advantages.” [Online]. Available: <https://www.orientsoftware.com/blog/java-advantages/>
- [18] NIST, “CVE-2020-14664.” [Online]. Available: <https://nvd.nist.gov/vuln/detail/CVE-2020-14664>
- [19] ——, “CVE-2023-22081.” [Online]. Available: <https://nvd.nist.gov/vuln/detail/CVE-2023-22081>
- [20] TutorialSpoint, “MATLAB Numerical Computing.” [Online]. Available: https://www.tutorialspoint.com/matlab/matlab_tutorial.pdf
- [21] NIST, “CVE-2023-49656.” [Online]. Available: <https://nvd.nist.gov/vuln/detail/CVE-2023-49656>