

UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
DEPARTAMENTUL CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI

PROIECT
CALCUL NUMERIC

PROFESOR COORDONATOR:
BUCATARU MIHAI

STUDENȚI:
LITU BOGDAN
BĂLAN MIHAI

BUCUREȘTI
2025

UNIVERSITATEA DIN BUCUREȘTI
FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ
DEPARTAMENTUL CALCULATOARE ȘI TEHNOLOGIA INFORMAȚIEI

**REZOLVAREA NUMERICĂ A ECUAȚIEI DE DIFUZIE A
CĂLDURII PENTRU UN DREPTUNGHI ÎN DOUĂ
DIMENSIUNI**

PROFESOR COORDONATOR:
BUCATARU MIHAI

STUDENȚI:
LITU BOGDAN
BĂLAN MIHAI

BUCUREȘTI

2025

Cuprins

1. Introducere.....	4
1.1 Aplicație	4
1.2 Motivație	4
2. Aspecte teoretice	4
2.1 Modelul generic.....	4
2.2 Prelucrarea primei ecuații.....	5
2.3 Alegerea domeniului Ω	5
2.4 Alegerea porțiunii pe care se aplică data Dirichlet.....	6
2.5 Alegerea porțiunii pe care se aplică data Neumann	6
2.6 Alegerea conductivității termice a materialului.....	6
2.7 Forma finală a primei ecuații.....	7
2.8 Reuniunea datelor sub același sistem	8
2.9 Atribuirea valorilor reale particulare t , v , s . Particularizarea funcției f	9
2.10 Discretizarea domeniului.....	9
2.11 Discretizarea ecuației principale (*) pentru nodurile interioare	10
2.12 Prelucrarea ecuației Neumann.....	10
2.13 Construirea sistemului de ecuații pentru un exemplu arbitrar	12
2.14 Rezolvarea efectivă a sistemului	14
3. Implementarea practică	15
3.1 Discretizarea domeniului.....	15
3.2 Funcția pentru data Dirichlet.....	16
3.3 Funcția pentru data Neumann.....	16
3.4 Funcția pentru conductivitatea materialului	16
3.5 Componenta F a sistemului de rezolvat.....	17
3.6 Componenta A a sistemului de rezolvat	18
3.7 Rezolvarea sistemului.....	19
3.8 Vizualizarea grafică în cadrul unui heatmap	20
3.9 Evaluarea erorii	21
4 Concluzii	21

1. Introducere

1.1 Aplicație

Proiectul “Rezolvarea numerică a ecuației de difuzie a căldurii pentru un dreptunghi în două dimensiuni” a fost realizat întâi conceptual, prin prelucrarea și alegerea datelor modelului generic, iar apoi utilizând programul software “JupyterLab” din distribuția “Anaconda”.

1.2 Motivație

Se dorește implementarea în limbajul Python a metodei diferențelor finite pentru rezolvarea unei ecuații cu derivate parțiale cu date la frontieră. Astfel, respectând modelul generic explicitat pe larg în continuare și alegând date specifice, vom documenta rezolvarea numerică a ecuației difuziei căldurii, pentru o complexitate fizică și geometrică dată.

2. Aspecte teoretice

2.1 Modelul generic

Modelul generic care stă la baza modelării difuziei căldurii are în componență următoarele ecuații interdependente:

- (1) $-\nabla[k(x, y)\nabla u(x, y)] = f(x, y)$ în Ω
- (2) $u(x, y) = g_D(x, y)$ pe $\Gamma_D \subset \partial\Omega$
- (3) $k(x, y)\frac{\partial u}{\partial n}(x, y) = g_N(x, y)$ pe $\Gamma_N = \partial\Omega \setminus \Gamma_D$

2.2 Prelucrarea primei ecuații

$$\begin{aligned} (*) - \nabla[k(x, y)\nabla u(x, y)] &= f(x, y) \Leftrightarrow \\ \Leftrightarrow - \left[\frac{\partial}{\partial x} \left(k(x, y) \frac{\partial u}{\partial x} \right) + \frac{\partial}{\partial y} \left(k(x, y) \frac{\partial u}{\partial y} \right) \right] &= f(x, y) \end{aligned}$$

Luăm pe rând cei doi termeni ai sumei și desfacem parantezele derivând cu formula produsului pe direcțiile x respectiv y:

$$\frac{\partial}{\partial x} \left(k(x, y) \frac{\partial u}{\partial x} \right) = \frac{\partial k}{\partial x} (x, y) \frac{\partial u}{\partial x} + k(x, y) \frac{\partial^2 u}{\partial x^2}$$

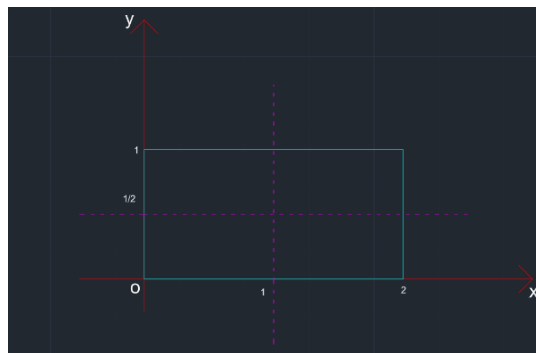
$$\frac{\partial}{\partial y} \left(k(x, y) \frac{\partial u}{\partial y} \right) = \frac{\partial k}{\partial y} (x, y) \frac{\partial u}{\partial y} + k(x, y) \frac{\partial^2 u}{\partial y^2}$$

Ecuația principală (*) devine :

$$- \left[\frac{\partial k}{\partial x} (x, y) \frac{\partial u}{\partial x} + k(x, y) \frac{\partial^2 u}{\partial x^2} + \frac{\partial k}{\partial y} (x, y) \frac{\partial u}{\partial y} + k(x, y) \frac{\partial^2 u}{\partial y^2} \right] = f(x, y)$$

2.3 Alegerea domeniului Ω

Alegem domeniul Ω un dreptunghi în 2D pe intervalul $[0, 2] \times [0, 1]$:



Figură 1- Domeniul Ω

2.4 Alegerea porțiunii pe care se aplică data Dirichlet

Alegem $\Gamma_D \subset \partial\Omega$ porțiunea de frontieră pe care se aplică data Dirichlet (pentru a specifica valorile exacte ale temperaturii pe anumite părți ale frontierei domeniului studiat).

$$\Gamma_D = \{(x, y) \in \partial\Omega \mid y = 0 ; y = 1 \text{ sau } x = 2\}$$

În acest caz, $g_D : \partial\Omega \rightarrow \mathbb{R}$ și:

$$g_D(x, y) = \begin{cases} v_1 ; & \text{dacă } x = 2 \\ v_2 ; & \text{dacă } y = 0 \\ v_3 ; & \text{dacă } y = 1 \end{cases}$$

2.5 Alegerea porțiunii pe care se aplică data Neumann

Alegem $\Gamma_N = \partial\Omega \setminus \Gamma_D$ porțiunea de frontieră pe care se aplică data Neumann (pentru a specifica fluxul de căldură pe anumite părți ale domeniului studiat).

$$\Gamma_N = \{(x, y) \in \partial\Omega \setminus \Gamma_D \mid x = 0\}$$

Pentru $x = 0 \Rightarrow n = (-1, 0) \Rightarrow \frac{\partial u}{\partial n} = -\frac{\partial u}{\partial x}$.

Explicitând:

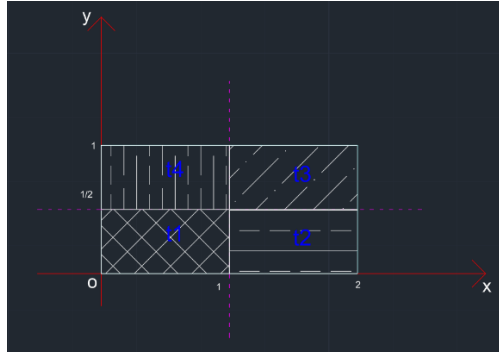
$$\frac{\partial u}{\partial n}(x, y) = -\frac{\partial u}{\partial x}(x, y) ; \text{dacă } x = 0$$

Atunci, în acest caz, $g_N : \partial\Omega \setminus \Gamma_D \rightarrow \mathbb{R}$ și :

$$g_N(x, y) = -k(x, y) \frac{\partial u}{\partial x} = s ; \text{dacă } x = 0.$$

2.6 Alegerea conductivității termice a materialului

În continuare, alegem funcția care explicităză conductivitatea termică a materialului (proprietate a materialului care influențează viteza și eficiența cu care căldura se transferă prin el). Particular, alegem patru zone de interes, reprezentate grafic în figura de mai jos:



Figură 2- Partiționarea conductivității termice

Astfel, fie funcția k definită astfel:

$$k(x, y) : [0, 2] \times [0, 1] \rightarrow \mathbb{R},$$

$$k(x, y) = \begin{cases} t1 ; \text{dacă } x \in [0, 1] \text{ și } y \in [0, \frac{1}{2}] \\ t2 ; \text{dacă } x \in (1, 2] \text{ și } y \in [0, \frac{1}{2}] \\ t3 ; \text{dacă } x \in (1, 2] \text{ și } y \in (\frac{1}{2}, 1] \\ t4 ; \text{dacă } x \in [0, 1] \text{ și } y \in (\frac{1}{2}, 1] \end{cases}$$

2.7 Forma finală a primei ecuații

Întrucât funcția k ia valori constante pentru oricare punct infinitesimal din domeniul Ω , derivatele parțiale de ordin 1 pe direcțiile x respectiv y din scrierea relației (*) vor fi nule.

Atunci, relația (*) devine:

$$-\left[k(x, y) \frac{\partial^2 u}{\partial x^2} + k(x, y) \frac{\partial^2 u}{\partial y^2} \right] = f(x, y)$$

2.8 Reuniunea datelor sub același sistem

Sub același sistem, modelele teoretice anterioare arată astfel:

$$\left\{ \begin{array}{l} (4) - \left[k(x, y) \frac{\partial^2 u}{\partial x^2} + k(x, y) \frac{\partial^2 u}{\partial y^2} \right] = f(x, y) \\ (5) k(x, y) = t1 ; \text{dacă } x \in [0,1] \text{ și } y \in [0, \frac{1}{2}] \\ (6) k(x, y) = t2 ; \text{dacă } x \in (1,2] \text{ și } y \in [0, \frac{1}{2}] \\ (7) k(x, y) = t3 ; \text{dacă } x \in (1,2] \text{ și } y \in (\frac{1}{2}, 1] \\ (8) k(x, y) = t4 ; \text{dacă } x \in [0,1] \text{ și } y \in (\frac{1}{2}, 1] \\ (9) u(x, y) = v1 ; \text{dacă } x = 2 \\ (10) u(x, y) = v2 ; \text{dacă } y = 0 \\ (11) u(x, y) = v3 ; \text{dacă } y = 1 \\ (12) - k(x, y) \frac{\partial u}{\partial x} = s ; \text{dacă } x = 0 \end{array} \right.$$

Legendă:

(4) - ecuația principală ; (5)..(8) – ecuațiile conductivității

(9)..(11) – date Dirichlet ; (12) – data Neumann

Prelucrăm în continuare și obținem:

$$\left\{ \begin{array}{l} - \left[k(x, y) \frac{\partial^2 u}{\partial x^2} + k(x, y) \frac{\partial^2 u}{\partial y^2} \right] = f(x, y) \\ k(x, y) = t1 ; \text{dacă } x \in [0,1] \text{ și } y \in [0, \frac{1}{2}] \\ k(x, y) = t2 ; \text{dacă } x \in (1,2] \text{ și } y \in [0, \frac{1}{2}] \\ k(x, y) = t3 ; \text{dacă } x \in (1,2] \text{ și } y \in (\frac{1}{2}, 1] \\ k(x, y) = t4 ; \text{dacă } x \in [0,1] \text{ și } y \in (\frac{1}{2}, 1] \\ u(x, y) = v1 ; \text{dacă } x = 2 \\ u(x, y) = v2 ; \text{dacă } y = 0 \\ u(x, y) = v3 ; \text{dacă } y = 1 \\ (13) - \frac{\partial u}{\partial x}(x, y) = \frac{s}{t1} \text{ dacă } x = 0 \text{ și } y \in (0, \frac{1}{2}] \\ (14) - \frac{\partial u}{\partial x}(x, y) = \frac{s}{t4} \text{ dacă } x = 0 \text{ și } y \in (\frac{1}{2}, 1) \end{array} \right.$$

Legendă:

(13),(14) – prelucrări ale ecuației (12)

2.9 Atribuirea valorilor reale particulare t , v , s . Particularizarea funcției f .

Alegem valori arbitrare considerate cunoscute pentru modelul ales. În plus, pentru dependența conductivității termice vor fi alese date apropiate pentru evitarea erorilor computaționale :

$$\left\{ \begin{array}{l} t1 = \dots (\text{conductivitatea pentru materialul 1}) \\ t2 = \dots (\text{conductivitatea pentru materialul 2}) \\ t3 = \dots (\text{conductivitatea pentru materialul 3}) \\ t4 = \dots (\text{conductivitatea pentru materialul 4}) \\ v1 = \dots (\text{valoare a primei datei Dirichlet}) \\ v2 = \dots (\text{valoare a celei de - a doua date Dirichlet}) \\ v3 = \dots (\text{valoare a celei de - a treia date Dirichlet}) \\ s = \dots (\text{valoarea datei Neumann}) \end{array} \right.$$

De asemenea, funcția f va fi aleasă în particular :

$$f : \mathbb{R} \times \mathbb{R} \rightarrow [-1, 1]$$
$$f(x, y) = \sin(2xy) \text{ (spre exemplu)}$$

2.10 Discretizarea domeniului

Pentru a discretiza dreptunghiul $\Omega = [0, 2] \times [0, 1]$ vom discretiza cele două axe, \mathcal{O}_x și \mathcal{O}_y , folosind $n+1$ noduri pentru abscisa respectiv $m+1$ pentru ordonată, astfel:

$$\rightarrow x_i = ih_1, \forall i \in [0, n+1]$$

$$\rightarrow y_j = jh_2, \forall j \in [0, m+1]$$

Observație: Datele se vor alege astfel încât $n+1 = 2 * (m+1)$.

Cele $[0, n+1] \times [0, m+1]$ noduri vor fi numerotate crescător, de la stânga la dreapta și de jos în sus, astfel încât punctului de coordonate (x_i, y_j) îi va corespunde un nod notat $N_{i+j(n+1)}$.

2.11 Discretizarea ecuației principale (*) pentru nodurile interioare

Fiecare derivată parțială de ordin 2 se va aproxima folosind diferențele finite centrale în 1D, întrucât sunt aplicate unei singure direcții în parte:

$$\begin{aligned}
 -\frac{\partial^2 u}{\partial x^2}(x_i, y_j) &= \frac{-u(x_{i-1}, y_j) + 2u(x_i, y_j) - u(x_{i+1}, y_j))}{h_1^2} + \mathcal{O}(h^2) \\
 -\frac{\partial^2 u}{\partial y^2}(x_i, y_j) &= \frac{-u(x_i, y_{j-1}) + 2u(x_i, y_j) - u(x_i, y_{j+1}))}{h_2^2} + \mathcal{O}(h^2) \\
 \Rightarrow k(x, y) \left[-\frac{\partial^2 u}{\partial x^2}(x_i, y_j) - \frac{\partial^2 u}{\partial y^2}(x_i, y_j) \right] &= \\
 = k(x, y) \left[-\frac{1}{h_1^2} u(x_{i-1}, y_j) + \frac{2}{h_1^2} u(x_i, y_j) - \frac{1}{h_1^2} u(x_{i+1}, y_j) - \frac{1}{h_2^2} u(x_i, y_{j-1}) + \right. \\
 \left. + \frac{2}{h_2^2} u(x_i, y_j) - \frac{1}{h_2^2} u(x_i, y_{j+1}) \right] + \mathcal{O}(h^2) \quad \forall (x_i, y_j)_{1 \leq i \leq n-1; 1 \leq j \leq m-1}.
 \end{aligned}$$

2.12 Prelucrarea ecuației Neumann

Vom deriva formula pentru condiția Neumann pe partea de frontieră pe care se aplică data ($x=0$) folosind o aproximare cu diferențe finite de ordin 2 prin metoda interpolării Lagrange. Întâi ne vom aminti forma ecuației Neumann:

$$k(x, y) \frac{\partial u}{\partial n} = s \quad (1)$$

$$\frac{\partial u}{\partial n} = -\frac{\partial u}{\partial x} \quad (2)$$

$$\text{Din (1) și (2)} \Rightarrow k(x, y) \left(-\frac{\partial u}{\partial x} \right) = s \Rightarrow \frac{\partial u}{\partial x} = -\frac{s}{k(x, y)}$$

Fie $x_0 = 0, x_1 = h_1, x_2 = 2h_1$.

Polinomul Lagrange specific este:

$$p(x) = u(x_0, y_j)L_{2,0}(x) + u(x_1, y_j)L_{2,1}(x) + u(x_2, y_j)L_{2,2}(x)$$

Formula de interes:

$$L_{n,k}(x) = \prod_{\substack{i=0 \\ i \neq k}}^n \frac{x - x_i}{x_k - x_i} \quad \forall 0 \leq k \leq n$$

Prelucrăm polinoamele Lagrange date având la bază formula de interes:

$$L_{2,0}(x) = \frac{(x - x_1)(x - x_2)}{(x_0 - x_1)(x_0 - x_2)} = \frac{(x - h_1)(x - 2h_1)}{(-h_1)(-2h_1)} = \frac{x^2 - 3xh_1 + 2h_1^2}{2h_1^2}$$

$$L_{2,1}(x) = \frac{(x - x_0)(x - x_2)}{(x_1 - x_0)(x_1 - x_2)} = \frac{-x(x - 2h_1)}{-h_1(-2h_1)} = -\frac{x^2 - 2xh_1}{h_1^2}$$

$$L_{2,2}(x) = \frac{(x - x_0)(x - x_1)}{(x_2 - x_0)(x_2 - x_1)} = \frac{x(x - h_1)}{2h_1(2h_1 + h_1)} = \frac{x^2 - h_1x}{2h_1^2}$$

Derivata polinomului $p(x)$ este:

$$p'(x) = u(x_0, y_j)L_{2,0}'(x) + u(x_1, y_j)L_{2,1}'(x) + u(x_2, y_j)L_{2,2}'(x)$$

unde:

$$L_{2,0}'(x) = \frac{2x - 3h_1}{2h_1^2}$$

$$L_{2,1}'(x) = -\frac{2x - 2h_1}{h_1^2}$$

$$L_{2,2}'(x) = \frac{2x - h_1}{2h_1^2}$$

În punctul $x = 0$, polinoamele Lagrange devin:

$$L_{2,0}'(0) = -\frac{3}{2h_1}$$

$$L_{2,1}'(0) = \frac{2}{h_1}$$

$$L_{2,2}(0) = -\frac{1}{2h_1}$$

Pentru $x = 0$, polinomul specific devine:

$$\begin{aligned} p(0) &= \frac{-3u(x_0, y_j) + 4u(x_1, y_j) - u(x_2, y_j)}{2h_1} \Rightarrow \\ \Rightarrow \frac{\partial u}{\partial x} &= \frac{-3u(x_0, y_j) + 4u(x_1, y_j) - u(x_2, y_j)}{2h_1} + \mathcal{O}(h^2) \\ i. e. \quad -\frac{s}{k(x, y)} &= \frac{-3u(x_0, y_j) + 4u(x_1, y_j) - u(x_2, y_j)}{2h_1} + \mathcal{O}(h^2) \end{aligned}$$

2.13 Construirea sistemului de ecuații pentru un exemplu arbitrar

Ne propunem în continuare a construi sistemul de ecuații pentru un caz particular cu mult mai mic decât ceea ce ne propunem. Astfel, implementăm grafic naiv un domeniu Ω deja discretizat pentru cazul $n = 6 ; m = 3$.

Dreptunghiul va arăta astfel:

y_3	N_{21}	N_{22}	N_{23}	N_{24}	N_{25}	N_{26}	N_{27}
y_2	N_{14}	N_{15}	N_{16}	N_{17}	N_{18}	N_{19}	N_{20}
y_1	N_7	N_8	N_9	N_{10}	N_{11}	N_{12}	N_{13}
y_0	N_0	N_1	N_2	N_3	N_4	N_5	N_6
	x_0	x_1	x_2	x_3	x_4	x_5	x_6

Figură 3- Partiționarea unui dreptunghi arbitrar

Nodurile evidențiate cu portocaliu sunt date Dirichlet, cele roșii reies din prelucrarea ecuațiilor cu date Neumann, iar cele interioare din prelucrarea cu diferențe finite a ecuației principale (*). Aduse împreună, aceste seturi de ecuații ar trebui să arate astfel, formând un sistem:

<i>Coordonate noduri</i>	<i>Nod</i>	<i>Ecuația specifică</i>
(x_0, y_0)	N_0	$u(x_0, y_0)$
(x_1, y_0)	N_1	$u(x_1, y_0)$
(x_2, y_0)	N_2	$u(x_2, y_0)$
(x_3, y_0)	N_3	$u(x_3, y_0)$
(x_4, y_0)	N_4	$u(x_4, y_0)$
(x_5, y_0)	N_5	$u(x_5, y_0)$
(x_6, y_0)	N_6	$u(x_6, y_0)$
(x_0, y_1)	N_7	$k(x_0, y_1) \frac{-3u(x_0, y_1) + 4u(x_1, y_1) - u(x_2, y_1)}{2h_1} + O(h^2)$
(x_1, y_1)	N_8	$k(x_1, y_1) \left[-\frac{1}{h_1^2} u(x_0, y_1) + \frac{2}{h_1^2} u(x_1, y_1) - \frac{1}{h_1^2} u(x_2, y_1) - \frac{1}{h_2^2} u(x_1, y_0) + \right. \\ \left. + \frac{2}{h_2^2} u(x_1, y_1) - \frac{1}{h_2^2} u(x_1, y_2) \right] + O(h^2)$
(x_2, y_1)	N_9	$k(x_2, y_1) \left[-\frac{1}{h_1^2} u(x_1, y_1) + \frac{2}{h_1^2} u(x_2, y_1) - \frac{1}{h_1^2} u(x_3, y_1) - \frac{1}{h_2^2} u(x_2, y_0) + \right. \\ \left. + \frac{2}{h_2^2} u(x_2, y_1) - \frac{1}{h_2^2} u(x_2, y_2) \right] + O(h^2)$
(x_3, y_1)	N_{10}	$k(x_3, y_1) \left[-\frac{1}{h_1^2} u(x_2, y_1) + \frac{2}{h_1^2} u(x_3, y_1) - \frac{1}{h_1^2} u(x_4, y_1) - \frac{1}{h_2^2} u(x_3, y_0) + \right. \\ \left. + \frac{2}{h_2^2} u(x_3, y_1) - \frac{1}{h_2^2} u(x_3, y_2) \right] + O(h^2)$
(x_4, y_1)	N_{11}	$k(x_4, y_1) \left[-\frac{1}{h_1^2} u(x_3, y_1) + \frac{2}{h_1^2} u(x_4, y_1) - \frac{1}{h_1^2} u(x_5, y_1) - \frac{1}{h_2^2} u(x_4, y_0) + \right. \\ \left. + \frac{2}{h_2^2} u(x_4, y_1) - \frac{1}{h_2^2} u(x_4, y_2) \right] + O(h^2)$
(x_5, y_1)	N_{12}	$k(x_5, y_1) \left[-\frac{1}{h_1^2} u(x_4, y_1) + \frac{2}{h_1^2} u(x_5, y_1) - \frac{1}{h_1^2} u(x_6, y_1) - \frac{1}{h_2^2} u(x_5, y_0) + \right. \\ \left. + \frac{2}{h_2^2} u(x_5, y_1) - \frac{1}{h_2^2} u(x_5, y_2) \right] + O(h^2)$
(x_6, y_1)	N_{13}	$u(x_6, y_1)$
(x_0, y_2)	N_{14}	$k(x_0, y_2) \frac{-3u(x_0, y_2) + 4u(x_1, y_2) - u(x_2, y_2)}{2h_1} + O(h^2)$
(x_1, y_2)	N_{15}	$k(x_1, y_2) \left[-\frac{1}{h_1^2} u(x_0, y_2) + \frac{2}{h_1^2} u(x_1, y_2) - \frac{1}{h_1^2} u(x_2, y_2) - \frac{1}{h_2^2} u(x_1, y_1) + \right. \\ \left. + \frac{2}{h_2^2} u(x_1, y_2) - \frac{1}{h_2^2} u(x_1, y_3) \right] + O(h^2)$

(x_2, y_2)	N_{16}	$k(x_2, y_2) \left[-\frac{1}{h_1^2} u(x_1, y_2) + \frac{2}{h_1^2} u(x_2, y_2) - \frac{1}{h_1^2} u(x_3, y_2) - \frac{1}{h_2^2} u(x_2, y_1) + \right.$ $\left. + \frac{2}{h_2^2} u(x_2, y_2) - \frac{1}{h_2^2} u(x_2, y_3) \right] + O(h^2)$
(x_3, y_2)	N_{17}	$k(x_3, y_2) \left[-\frac{1}{h_1^2} u(x_2, y_2) + \frac{2}{h_1^2} u(x_3, y_2) - \frac{1}{h_1^2} u(x_4, y_2) - \frac{1}{h_2^2} u(x_3, y_1) + \right.$ $\left. + \frac{2}{h_2^2} u(x_3, y_2) - \frac{1}{h_2^2} u(x_3, y_3) \right] + O(h^2)$
(x_4, y_2)	N_{18}	$k(x_4, y_2) \left[-\frac{1}{h_1^2} u(x_3, y_2) + \frac{2}{h_1^2} u(x_4, y_2) - \frac{1}{h_1^2} u(x_5, y_2) - \frac{1}{h_2^2} u(x_4, y_1) + \right.$ $\left. + \frac{2}{h_2^2} u(x_4, y_2) - \frac{1}{h_2^2} u(x_4, y_3) \right] + O(h^2)$
(x_5, y_2)	N_{19}	$k(x_5, y_2) \left[-\frac{1}{h_1^2} u(x_4, y_2) + \frac{2}{h_1^2} u(x_5, y_2) - \frac{1}{h_1^2} u(x_6, y_2) - \frac{1}{h_2^2} u(x_5, y_1) + \right.$ $\left. + \frac{2}{h_2^2} u(x_5, y_2) - \frac{1}{h_2^2} u(x_5, y_3) \right] + O(h^2)$
(x_6, y_2)	N_{20}	$u(x_6, y_2)$
(x_0, y_3)	N_{21}	$u(x_0, y_3)$
(x_1, y_3)	N_{22}	$u(x_1, y_3)$
(x_2, y_3)	N_{23}	$u(x_2, y_3)$
(x_3, y_3)	N_{24}	$u(x_3, y_3)$
(x_4, y_3)	N_{25}	$u(x_4, y_3)$
(x_5, y_3)	N_{26}	$u(x_5, y_3)$
(x_6, y_3)	N_{27}	$u(x_6, y_3)$

2.14 Rezolvarea efectivă a sistemului

Ne propunem a rezolva sistemul:

$$A * U = F, \text{ unde}$$

- $U \in \mathbb{R}^{(n+1) \times (m+1)}$ - conține aproximările temperaturii în nodurile considerate
- $F = F_{i+j(n+1)}_{0 \leq i \leq n \text{ si } 0 \leq j \leq m} \in \mathbb{R}^{(n+1) \times (m+1)}$ definită astfel încât:

$$F_{i+j(n+1)} = \begin{cases} f(x_i, y_j) & \text{in cazul general} \\ gD(x_i, y_j) & \text{daca pentru indici se aplica data Dirichlet} \\ gN(x_i, y_j) & \text{daca pentru indici se aplica data Neumann} \end{cases}$$

- $A \in \mathbb{R}^{(n+1)^2 \times (m+1)^2}$ matricea determinata de indicii functiei U de la punctul 2.13

Pentru exemplul particular dat, o explicitare a matricei A pe blocuri ar fi destul de complicat de realizat în cadrul documentației, având în vedere variația conductivității precum și posibilitatea de discretizare în mai multe noduri ($h_1 \neq h_2$). Totuși, procedeul este simplu: se iau indicii din fața funcțiilor u din cadrul ecuațiilor specifice și se construiește o matrice asociată ce va ajuta în rezolvarea problemei în mod computațional mai târziu.

3. Implementarea practică

3.1 Discretizarea domeniului

Începem rezolvarea problemei prin discretizarea celor două noduri pe axele O_x respectiv O_y prin folosirea a $n + 1$ noduri pentru abscisă și un număr de $m + 1$ pentru ordonată. Discretizarea se face folosind formula:

$$h = \frac{\text{lungimea efectivă a intervalului}}{\text{numarul ales de noduri} - 1}$$

Pentru cazul nostru particular, alegem nodurile astfel încât pe axa O_x sunt de două ori mai multe decât pe O_y . Atunci:

$$h_1 = \frac{2}{n+1} \text{ si } h_2 = \frac{1}{m+1}; \text{ dar } m+1 = 2 * (n+1) \Rightarrow \\ \Rightarrow h_1 = h_2 = h.$$

```
#Exemplu unde m+1 = nr de noduri pe Oy, n+1 nr de noduri pe Ox, L = valoarea extrema a intervalului pe (1) Ox si (2) Oy
def discretizareDomeniu(m):
    n = 2*m
    L1, L2 = 2, 1
    h1, h2 = L1/n, L2/m
    print("Pasul dintre cele {0} noduri de pe Ox, respectiv {1} noduri de pe Oy este: {2}, respectiv {3}".format(n+1, m+1, h1, h2))
    print("Vom avea {0} noduri".format((n+1)*(m+1)))
    if(h1 == h2):
        h = h1 #daca sunt egale, "unim" definitia
    return n, h

m0 = 3
n0, h0 = discretizareDomeniu(m0)
```

Pasul dintre cele 7 noduri de pe Ox, respectiv 4 noduri de pe Oy este: 0.3333333333333333, respectiv 0.3333333333333333
Vom avea 28 noduri

Figură 4 – Discretizarea domeniului în Python

3.2 Funcția pentru data Dirichlet

Ne reamintim faptul că datele Dirichlet se aplică pe o porțiune arbitrar aleasă de pe frontiera domeniului Ω ; un drum Γ_D . Astfel, este creată o funcție notată g_D , cu rol în stabilirea unor valori ale căldurii știute în totalitatea nodurilor din componența acestor partiții:

```
def gD(x, y):
    if 0 - eps <= y <= 0 + eps:
        return v2
    if 1 - eps <= y <= 1 + eps:
        return v3
    if 2 - eps <= x <= 2 + eps:
        return v1

v1, v2, v3 = 0, 0, 0
eps = 1e-10
print("Datele Dirichlet: ", gD(2,0.5), gD(1,0), gD(1,1), "(°C)")

Datele Dirichlet:  0 0 0 (°C)
```

Figură 5 – Stabilirea funcției g_D în Python

3.3 Funcția pentru data Neumann

Datele Neumann se aplică pe porțiunea rămasă de frontieră, un drum $\Gamma_G = \partial\Omega \setminus \Gamma_D$. Este creată în acest caz o funcție g_N , cu rol în definirea fluxului de căldură care circulă prin porțiunea definită:

```
def gN(x, y):
    if 0 - eps <= x <= 0 + eps and 0 < y < 1:
        return s
    return 0

s = 0
print("Datele Neumann: ", gN(0, 0.5), "(W/m^2)")

Datele Neumann:  0 (W/m^2)
```

Figură 6 – Stabilirea funcției g_N în Python

3.4 Funcția pentru conductivitatea materialului

În funcțiile de caracteristicile fizice ale domeniului luat în discuție, acesta poate fi compus din materiale diferite care prezintă o conductivitate electrică diferită. Astfel, am construit o funcție k , care primește valorile numerice ale conductivității în cele patru zone de interes. Zonele au fost stabilite prin “tăierea” cu două axe de simetrie, una latitudinală și alta longitudinală a domeniului Ω :


```
import numpy as np

def k(x, y):
    if 0 <= x <= 1 and 0 <= y <= 0.5:
        return t1
    if 1 < x <= 2 and 0 <= y <= 0.5:
        return t2
    if 1 < x <= 2 and 0.5 < y <= 1:
        return t3
    if 0 <= x <= 1 and 0.5 < y <= 1:
        return t4
    return 0

t1, t2, t3, t4 = 16, 17, 15, 14
print("Valorile conductivitatii termice: ", k(0,0), k(2,0), k(2,1), k(0,1), " (W/m*K)")
```

Figură 7 – Stabilirea funcției k în Python

Avem nevoie de o funcție suplimentară care răspunde la întrebarea:

Ce conductivitate corespunde unui nod interior domeniului Ω ?

```
def determinareK(n, m, h):
    nrEcuatii = (n+1)*(m+1)
    K = np.zeros((nrEcuatii, 1))
    for j in range(m+1):
        for i in range(n+1):
            #x va fi 0 + i * h, iar y va fi 0 + j * h
            x = 0 + i * h
            y = 0 + j * h
            t = i + j * (n+1) #obtin indexul corespunzator coloanei i de pe linia j
            if i in range(1, n) and j in range(1, m): #daca sunt interne
                K[t] = k(x,y)
            elif i==0 and j in range(1, m): #daca se aplica date Neumann
                K[t] = k(x,y)
            else:
                K[t] = 1 #nu ma intereseaza valorile k de la frontiera
    return K

K = determinareK(n0, m0, h0)

print(K)
print("Obtinem {0} rezultate".format(K.shape[0]))
```

Figură 8 – Stabilirea conductivității unui nod oarecare în Python

3.5 Componenta F a sistemului de rezolvat

Sursa internă de căldură a materialului trebuie stabilită corect pentru a putea crea vectorul – coloană F de dimensiune $(n + 1) * (m + 1)$. Indicele lui F corespunde nodului cu același indice, definit astfel:

$$F_{i+j(n+1)} = \begin{cases} f(x_i, y_j) & \text{in cazul general} \\ gD(x_i, y_j) & \text{daca pentru indici se aplica data Dirichlet} \\ gN(x_i, y_j) & \text{daca pentru indici se aplica data Neumann} \end{cases}$$

Funcția f este sinusoidală, definită arbitrar $f(x, y) = \sin(2xy)$. În Python, sursa de căldură este stabilită astfel:

```

import numpy as np

def f(x,y):
    return np.sin(2*x*y)

def determinareF(n, m, h):
    nrEcuatii = (n+1)*(m+1)
    F = np.zeros(nrEcuatii, 1)
    for j in range(m+1):
        for i in range(n+1):
            #x va fi 0 + i * h, iar y va fi 0 + j * h
            x = 0 + i * h
            y = 0 + j * h
            k = i + j * (n+1) #obtin indexul corespunzator coloanei i de pe linia j
            if i in range(1, n) and j in range(1, m): #daca sunt interne
                F[k] = f(x,y)
            elif i==0 and j in range(1, m): #daca se aplica date Neumann
                F[k] = gN(x,y)
            else:
                F[k] = gD(x,y) #daca se aplica date Dirichlet
    return F

F = determinareF(n0, m0, h0)

print(F)
print("Obtinem {} rezultate".format(F.shape[0]))

```

Figură 9 – Stabilirea conductivității unui nod oarecare în Python

3.6 Componenta A a sistemului de rezolvat

Scopul lucrării este determinarea componentei U din sistemul de ecuații de rezolvat. Matricea A conține indicii celor $(n + 1)(m + 1)$ ecuații de la punctul 2.13 și poate fi scrisă pe blocuri, după cum urmează:

$$A = \begin{bmatrix} I_{m+1} & 0 & 0 & 0 & \dots & 0 \\ A_{1,1} & A_{1,1} & A_{2,1} & 0 & \dots & 0 \\ 0 & A_{1,2} & A_{2,2} & A_{3,2} & \dots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 & I_n \end{bmatrix} \in M_{(n+1)(m+1), (n+1)(m+1)}(R)$$

Figură 10 – Forma matricei A

Unde:

$$A_{ik} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ 0 & \frac{-1}{R^2} & 0 & \dots & 0 \\ 0 & 0 & \frac{-1}{R^2} & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \dots & \frac{-1}{R^2} & 0 \\ 0 & 0 & \dots & 0 & 0 \end{bmatrix} \in M_{m+1, m+1}(\mathbb{R})$$

$\forall k = \overline{1, m-1}$

Figură 11 – Forma blocului de forma A_{ik}

Și:

$$A_{k,k} = \begin{bmatrix} \frac{-3}{2R} & \frac{2}{R} & \frac{-1}{2R} & 0 & \vdots \\ \frac{-1}{R^2} & \frac{4}{R^2} & \frac{-1}{R^2} & 0 & \vdots \\ 0 & \frac{-1}{R^2} & \frac{4}{R^2} & \frac{-1}{R^2} & \vdots \\ \vdots & \vdots & \vdots & \vdots & \ddots \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix} \in M_{m+1, m+1}(\mathbb{R})$$

$\forall k = \overline{1, m-1}$

Figură 12 – Forma blocului de forma A_{kk}

3.7 Rezolvarea sistemului

Pentru eficiență, în rezolvarea sistemului $A * U = F$, vom folosi factorizarea LU și metoda Gauss cu pivotare parțială definite în librăria *numpy*. Apar și erori de reprezentare

foarte apropiate de 0 (deși existente) pe care le vom memora și urmărim a le evidenția în continuare:

```
import numpy as np

m1 = 4
n1,h1 = discretizareDomeniu(m1)

F = determinareF(n1,m1,h1)
K = determinareK(n1,m1,h1)
A = determinareA(n1,m1,h1)

try:
    U = np.linalg.solve(A,F)
except:
    U = np.linalg.lstsq(A, F, rcond=None)[0]

Faprox = A@U

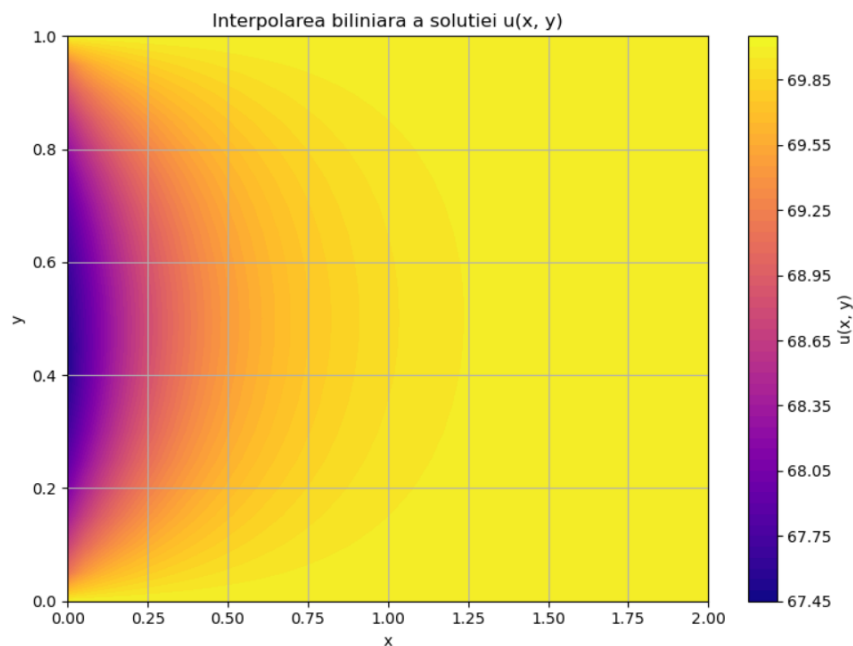
print("Vectorii-coloana Faprox si F, unul langa celalalt: \n",np.c_[Faprox, F])
```

Figură 13 – Rezolvarea sistemului $A * U = F$ în Python

Observație: În acest scop, vom da un exemplu ușor de urmărit, care să evidențieze toate aceste erori, unde $m_1=4$, iar restul componentelor sunt calculate folosind funcțiile anterioare.

3.8 Vizualizarea grafică în cadrul unui heatmap

Folosind algoritmul de interpolare cu spline liniar realizăm astfel o interpolare biliniară folosindu-ne de întregul domeniu nou-descoperit, U . Evaluăm comportamentul funcției într-un subdomeniu ales, cu date Dirichlet și Neumann alese arbitrar. Astfel, funcția spline se evaluează prima dată evaluată în y_0 , apoi în y_1 și în final în punctele obținute anterior. Obținem următorul heatmap:

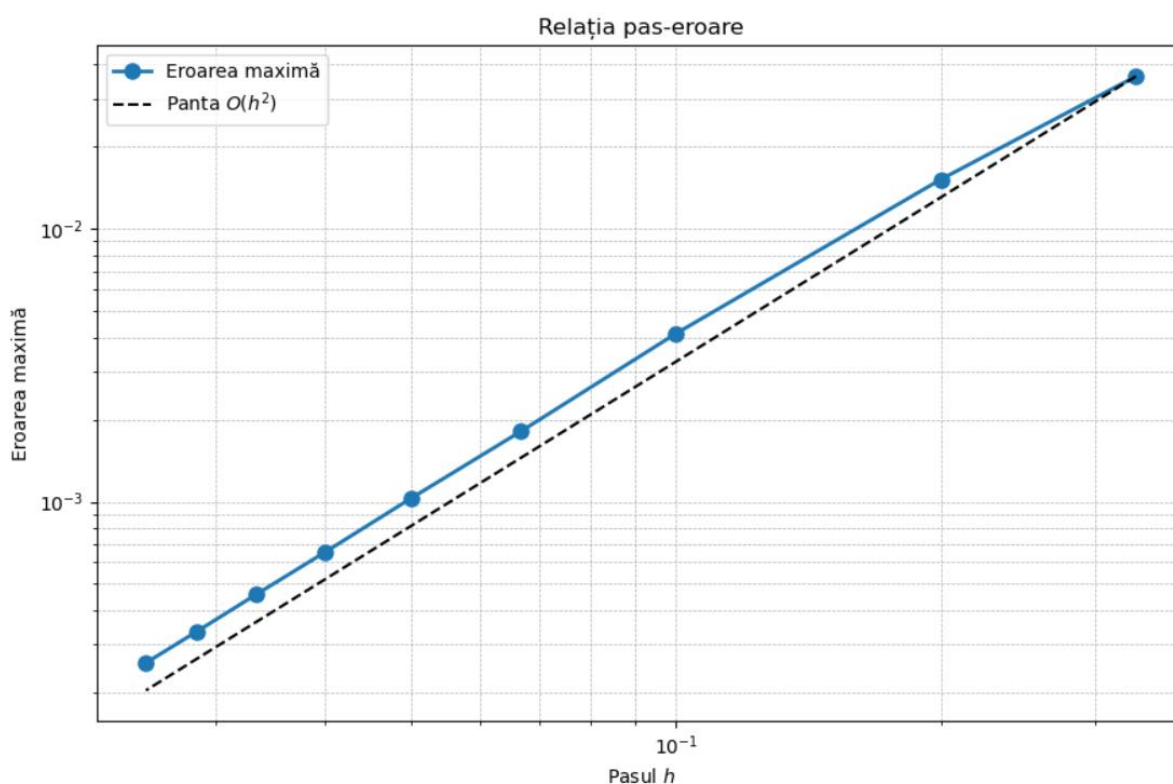


Figură 14 – Heatmap-ul Python

Observație: Modul exact în care este realizat heatmap-ul este documentat în proiectul efectiv.

3.9 Evaluarea erorii

Modificăm funcțiile și condițiile astfel încât eroarea să poată fi determinată (a se vedea în proiect). Cu interpolare, folosim funcțiile anterioare pentru mai multe valori ale lui m , dar calculăm și valoarea exactă a lui U . Se poate observa că, pe măsură ce crește numărul de noduri, pasul h scade odată cu eroarea maximă.



Figură 15 – Estimarea erorii în Python

4 Concluzii

Realizarea acestui proiect ne-a ajutat să ne îmbunătățim atât cunoștințele cât și abilitățile în domeniu dar și în lucrul cu aplicația. Am reușit să realizăm ce înseamnă efectiv cercetarea unui lucru practic, atât în manieră matematică cât și informatică.

Am întâmpinat dificultăți atât tehnice cât și teoretice de-a lungul proiectului. Ne-am îmbunătățit răbdarea și capacitatea de a căuta soluții pragmatice la probleme practice.

A trebuit să ne asigurăm de faptul că detaliile tehnice sunt respectate cu exactitate, fără a produce însă daune proiectării computaționale concentrându-ne, nu în ultimul rând, la aspectul modular al lucrării în sine.