Наочна демонстрація рівняння теплопровідності

Кирило Стрельбицький

Метою роботи є побудова програми для створення анімованої теплової карти, що буде наглядно демонструвати розповсюдження тепла, за допомогою побудови графіку засобами мови Python.

Математичне обгрунтування

Класичне представлення рівняння теплопровідності в просторі з довільною системою координат $r = (r_1,\ ...,\ r_n)$ виглядає так:

$$\frac{\partial u}{\partial t} - a^2 \nabla^2 u = f(r, t) \tag{1}$$

В рівнянні (1) маємо:

- a – константне значення, коєфіцієнт термопровідності;
- ∇^2 – позначення оператора Лапласа;
- $f(r, t)$ – функція теплових джерел;
- $u = u(r, t)$ – шукана функція, що задає температуру в точці з координатами r в момент часу t.

Ми розглядатимемо закрити систему, в якій немає втрат тепла, тож $f(r, t) = 0$.
Для знаходження рішення в площині декартової системи координат, перепишемо рівняння:

$$\frac{\partial u}{\partial t} - a^2 \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right) = 0 \tag{2}$$

Отже, для побудови тепловою карти в певний момент часу t нам необхідно знати значення функції $u(x,\ y,\ t)$. Для знаходження числового значення скористаємось розкладенням в ряд Тейлора.

Розкладемо в ряд Тейлора:

$$f(x + \Delta x) = f(x) + (\Delta x) f'(x) + \frac{(\Delta x)^2}{2!} f''(x) + \frac{(\Delta x)^3}{3!} f'''(x) + ... \tag{3}$$

$$f(x - \Delta x) = f(x) - (\Delta x) f'(x) + \frac{(\Delta x)^2}{2!} f''(x) - \frac{(\Delta x)^3}{3!} f'''(x) + ... \tag{4}$$

Додамо (3) та (4):

$$f(x+\Delta x)+f(x-\Delta x)\approx 2f(x)+\frac{2(\Delta x)^2}{2!}f''(x) \tag{5}$$

Отже, з (5) маємо, що:

$$f''(x)\approx \frac{f(x+\Delta x)-2f(x)+f(x-\Delta x)}{(\Delta x)^2} \tag{6}$$

Наближене значення диференціала знайдемо з означення похідної:

$$f'(x)=\frac{f(x+\Delta x)-f(x)}{\Delta x} \tag{7}$$

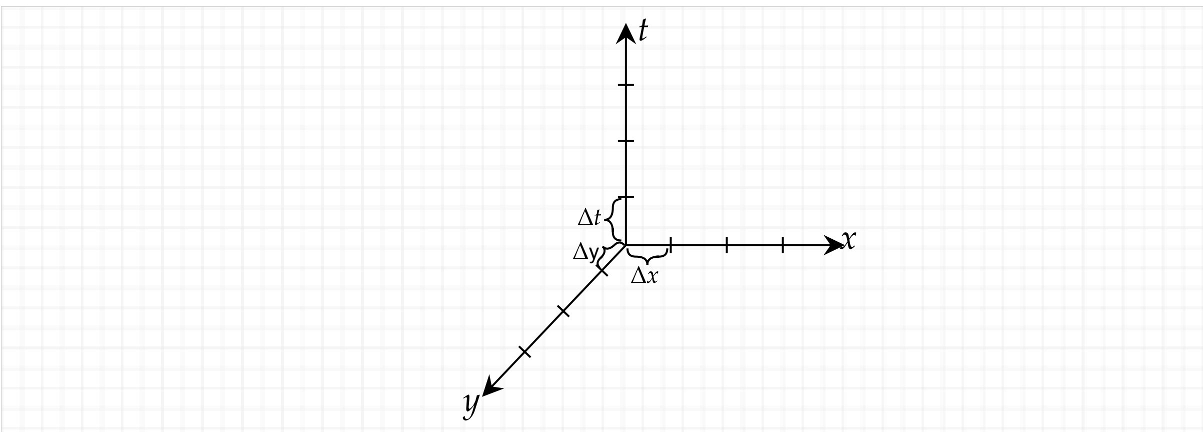
Застосуємо (6) та (7) до (2):

$$\frac{u(x,y,t+\Delta t)-u(x,y,t)}{\Delta t}\approx$$
$$\approx a^2\left(\frac{u(x+\Delta x,y,t)+u(x-\Delta x,y,t)+u(x,y+\Delta y,t)+u(x,y-\Delta y,t)-4u(x,y,t)}{(\Delta x)^2}\right)$$

Нехай $k=\dfrac{a^2\Delta t}{(\Delta x)^2}$, тоді:

$$u(x,y,t+\Delta t)=u(x,y,t)+$$
$$+k(u(x+\Delta x,y,t)+u(x-\Delta x,y,t)+u(x,y+\Delta y,t)+u(x,y-\Delta y,t)-4u(x,y,t))$$

Розміри Δx та Δy визначаються за розмірами сітки площини, а значення Δt візьмемо за $\dfrac{\Delta x^2}{4a}$.

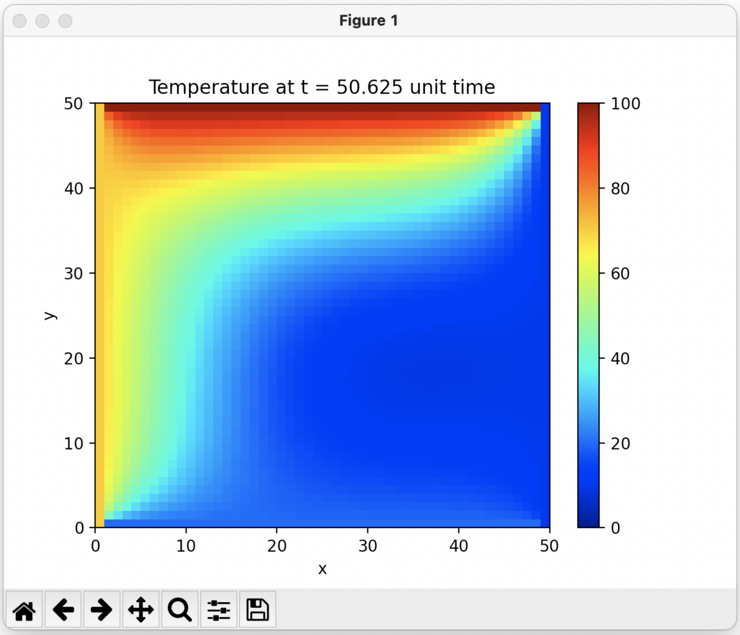


На цьому, математична частина добігає кінця.

Програмна складова

Програмна складова була реалізовано з використанням мови *Python* та модулів *matplotlib* та *numpy.* Посилання на GitHub з вихідним кодом в кінці роботи.

Для запуску модуля необхідно викликати $\$\ python3\ main.py$. Після цього користувач бачить анімований графік температури поверхні, що змінюється з плином часу.



Зображення 1: Вікно з тепловою картою

Реалізація класу **HeatMap** дозволяє встановлювати такі початкові параметри, як Δx та a. Також можно встановлювати:

- кількість кадрів в симуляції;
- швидкість зміни кадрів симуляції;
- початкові значення на карті;
- температуру бортів карти.

```
hmp = HeatMap(
    plate_length=50,
    max_iter_time=500,
    alpha=2,
    delta_x=1)

hmp.create_field()
hmp.set_borders(0, (100, 70, 20, 10))
```

Зображення 2: Виклик функцій з класу **HeatMap**

Обрахунок головної формули та побудова температурної карти відбувається в функції **calculate**, яка ітерує по координаті x та y, а також перебирає змінну часу t.

```
def calculate(self):
    """Making main calculation using heat equation"""

    for k in range(0, self.max_iter_time-1, 1):
        for i in range(1, self.plate_length-1, self.delta_x):
            for j in range(1, self.plate_length-1, self.delta_x):
                self.field[k + 1, i, j] = self.gamma * (self.field[k][i+1][j] +
                                                        self.field[k][i-1][j] +
                                                        self.field[k][i][j+1] +
                                                        self.field[k][i][j-1] -
                                                        4*self.field[k][i][j])
                self.field[k + 1, i, j] += self.field[k][i][j]
```

Зображення 3: Функція **calculate**

Висновки

Мета роботи була досягнута. Побудовано модуль, що дозволяє створювати анімації для начоної демонстрації роботи рівняння теплопровідності. Було використано розкладання в ряд Тейлора та проведена робота з дифіренціалами другого порядку.

Посилання на GitHub: https://github.com/Kirillstrelbitskiy/heatmap-ucu