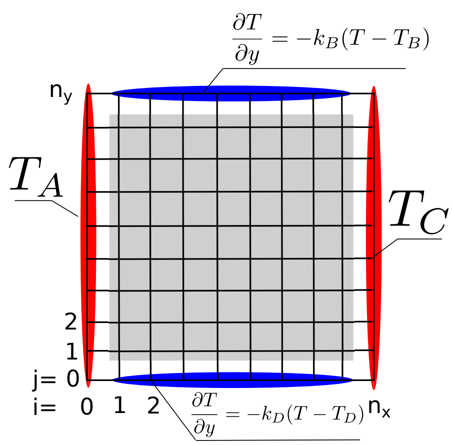
Projekt 9: Dyfuzja ciepła - metoda Cranck-Nicloson.

Tomasz Chwiej

10 stycznia 2019

1 Wstęp



Rysunek 1: Siatka węzłów użyta w obliczeniach z zaznaczonymi warunkami brzegowymi: Dirichleta (czerwony) i von Neumanna (niebieski).

Na zajęciach znajdziemy rozwiązanie zależnego od czasu równania dyfuzji ($T = T(x, y, t)$)

$$\nabla^2 T = \frac{\partial T}{\partial t} \tag{1}$$

Równanie to zdyskretyzujemy na siatce, zapiszemy w postaci macierzowej i znajdziemy jego rozwiązanie w kolejnych chwilach czasowych przy użyciu metody Cranck-Nicolson (do rozwiązania układu równań liniowych w tej metodzie wykorzystamy rozkład LU). Geometria układu wraz z siatką węzłów, w których wyznaczona zostanie temperatura pokazane jest na rys. 1.

1.1 Dyskretyzacja równania + metoda CN

Najperw definiujemy siatkę węzłów, dla położenia (x,y) i czasu (t)

$$\begin{aligned}
\Delta x &= \Delta y = \Delta && (2)\\
x_i &= i \cdot \Delta, \quad i = 0, 1, 2, \ldots, n_x && (3)\\
y_j &= j \cdot \Delta, \quad j = 0, 1, 2, \ldots, n_y && (4)\\
t_n &= n \cdot \Delta t, \quad n = 0, 1, 2, \ldots && (5)\\
T(x, y, t) &\to T(x_i, y_j, t_n) \to T^n_{i,j} && (6)
\end{aligned}$$

(dolny indeks - położenie, górny - czas).
Zapisujemy równanie (1) korzystając ze schematu C-N

$$\frac{1}{2}\left(\nabla^2 T^n + \nabla^2 T^{n+1}\right) = \frac{T^{n+1} - T^n}{\Delta t} \tag{7}$$

zastępujemy pochodne ilorazami różnicowymi $\left[d^2f/dx^2 = (f_{i+1} - 2f_i + f_{i-1})/\Delta^2\right]$ i grupujemy wyrazy względem chwil czasowych

$$\begin{aligned} & \frac{\Delta t}{2\Delta^2}\left(T^{n+1}_{i+1,j} + T^{n+1}_{i-1,j} - 4T^{n+1}_{i,j} + T^{n+1}_{i,j+1} + T^{n+1}_{i,j-1}\right) - T^{n+1}_{i,j} \\ = & -\frac{\Delta t}{2\Delta^2}\left(T^{n}_{i+1,j} + T^{n}_{i-1,j} - 4T^{n}_{i,j} + T^{n}_{i,j+1} + T^{n}_{i,j-1}\right) - T^{n}_{i,j} \end{aligned} \tag{8}$$

Dokonujemy teraz reindeksacji węzłów (parę wskaźników (i,j) zastępujemy jednym l)

$$l = i + j \cdot (n_x + 1), \quad l = 0, 1, 2, \ldots, N \tag{9}$$
$$N = (n_x + 1) \cdot (n_y + 1) \tag{10}$$
$$j = floor\left(\frac{l}{n_x + 1}\right) \tag{11}$$
$$i = l - j \cdot (n_x + 1) \tag{12}$$

Równanie 8 możemy zapisać w postaci macierzowej

$$A \cdot \vec{T}^{n+1} = B \cdot \vec{T}^{n} + \vec{c} \tag{13}$$

[uwaga: pojawienie się wektora $\vec{c}$ nie wynika bezpośrednio z równania (8), ale dodaliśmy je aby zapewnić spełnienie warunków brzegowych von Neumanna, dla WB Dirichleta przyjmujemy $\vec{c} = 0$. (góra/dół na rysunku 1],
oraz z uwzględnieniem tylko niezerowych elementów macierzowych

$$\begin{aligned} & a_{l,l-nx-1}T^{n+1}_{l-nx-1} + a_{l,l-1}T^{n+1}_{l-1} + a_{l,l}T^{n+1}_{l} + a_{l,l+1}T^{n+1}_{l+1} + a_{l,l+n_x+1}T^{n+1}_{l+n_x+1} \\ = & b_{l,l-nx-1}T^{n}_{l-nx-1} + b_{l,l-1}T^{n}_{l-1} + b_{l,l}T^{n}_{l} + b_{l,l+1}T^{n}_{l+1} + b_{l,l+n_x+1}T^{n}_{l+n_x+1} + c_l \end{aligned} \tag{14}$$

1.1.1 Niezerowe elementy macierzowe z uwzględnieniem WB

- Wnętrze obszaru (szary obszar na rysunku)

$$i = 1, 2, \ldots, n_x - 1 \tag{15}$$
$$j = 1, 2, \ldots, n_y - 1 \tag{16}$$
$$a_{l,l-n_x-1} = a_{l,l-1} = a_{l,l+1} = a_{l,l+n_x+1} = \frac{\Delta t}{2\Delta^2} \tag{17}$$
$$a_{l,l} = -\frac{2\Delta t}{\Delta^2} - 1 \tag{18}$$
$$b_{l,l-n_x-1} = b_{l,l-1} = b_{l,l+1} = b_{l,l+n_x+1} = -\frac{\Delta t}{2\Delta^2} \tag{19}$$
$$b_{l,l} = \frac{2\Delta t}{\Delta^2} - 1 \tag{20}$$

- WB Dirichleta (lewy i prawy brzeg)

$$i = 0, n_x \tag{21}$$
$$j = 0, 1, 2, \ldots, n_y \tag{22}$$
$$a_{l,l} = 1 \tag{23}$$
$$b_{l,l} = 1 \tag{24}$$
$$c_l = 0 \tag{25}$$

- WB von Neumanna na górnym brzegu dla chwili $n + 1$

$$\frac{\partial T^{n+1}}{\partial y} = -k_B(T^{n+1} - T_B) \tag{26}$$

po zastąpieniu pochodnych ilorazami jest następujący

$$\frac{T_{i,n_y}^{n+1} - T_{i,n_y-1}^{n+1}}{\Delta} = -k_B(T_{i,n_y}^{n+1} - T_B) \tag{27}$$

a po reindeksacji węzłów i po pogrupowaniu wyrazów

$$a_{l,l-nx-1}T_{l-n_x-1}^{n+1} + a_{l,l}T_l^{n+1} = c_l \tag{28}$$
$$\tag{29}$$

gdzie

$$\begin{aligned}
i &= 1, 2, \ldots, n_x - 1 && (30)\\
j &= n_y && (31)\\
a_{l,l-n_x-1} &= -\frac{1}{k_B\Delta} && (32)\\
a_{l,l} &= 1 + \frac{1}{k_B\Delta} && (33)\\
c_l &= T_B \quad \text{(wektor c)} && (34)\\
b_{l,*} &= 0, \quad \text{(cały wiersz)} && (35)
\end{aligned}$$

- WB von Neumanna na dolnym brzegu dla chwili $n+1$

$$\frac{\partial T^{n+1}}{\partial y} = -k_D(T^{n+1} - T_D) \tag{36}$$

po zastąpieniu pochodnych ilorazami jest następujący

$$\frac{T_{i,0}^{n+1} - T_{i,1}^{n+1}}{\Delta} = -k_D(T_{i,0}^{n+1} - T_B) \tag{37}$$

a po reindeksacji węzłów i po pogrupowaniu wyrazów

$$a_{l,l}T_l^{n+1} + a_{l,l+nx+1}T_{l+n_x+1}^{n+1} = c_l \tag{38}$$
$$\tag{39}$$

gdzie

$$\begin{aligned}
i &= 1, 2, \ldots, n_x - 1 && (40)\\
j &= 0 && (41)\\
a_{l,l} &= 1 + \frac{1}{k_D\Delta} && (42)\\
a_{l,l+n_x+1} &= -\frac{1}{k_D\Delta} && (43)\\
c_l &= T_D \quad \text{(wektor c)} && (44)\\
b_{l,*} &= 0, \quad \text{(cały wiersz)} && (45)
\end{aligned}$$

1.1.2 Warunki Początkowe

WP narzucamy na wektor startowy $\vec{T}^0$

$$\begin{aligned}
T_l^0 &= T_A, \quad i = 0,\ j = 0, 1, 2 \ldots, n_y, \quad \text{(lewy brzeg)} && (46)\\
T_l^0 &= T_C, \quad i = n_x,\ j = 0, 1, 2 \ldots, n_y, \quad \text{(prawy brzeg)} && (47)\\
T_l^0 &= 0, \quad \text{w pozostałym obszarze} && (48)
\end{aligned}$$

2 Algorytm CN

Algorytm CN dla równania dyfuzji w wersji macierzowej

```
inicjalizacja: A, B, $\vec{c}$, $\vec{T} = \vec{T}^0$
oblicz rozkład LU: A
FOR it=0 TO IT_MAX STEP 1 DO
  $\vec{d} = B \cdot \vec{T} + \vec{c}$           //$T = T^n$
  rozwiąż ukł. równ  (LU):  $A \cdot \vec{T} = \vec{d}$ //$T = T^{n+1}$
END DO
```

3 Zadania do wykonania

W obliczeniach do znalezienia rozkładu LU, rozwiązania układu równań liniowych, mnożenia macierz-wektor czy dodawania wektorów należy użyć biblioteki numerycznej np. GSL (lub innej dostępnej na Taurusie).

1. W obliczeniach należy użyć wartości parametrów: $n_x = 40$, $n_y = 40$, $N = (n_x + 1) \cdot (n_y + 1)$, $\Delta = 1$, $\Delta t = 1$, $T_A = 40$, $T_B = 0$, $T_C = 30$, $T_D = 0$, $k_B = 0.1$, $k_D = 0.6$, $IT\_MAX = 2000$.

2. Utworzyć macierze $A[N][N]$, $B[N][N]$ oraz wektor $c[N]$ i wypełnić je zgodnie z wzorami zamieszczonymi w sekcji 1.1.1

3. Utworzyć wektor startowy $T[N]$ i narzucić warunki początkowe (sekcja 1.1.2)

4. Znaleźć rozkład LU macierzy A (GSL)

5. Zaimplementować algorytm CN (sekcja 2)

6. Wykonać IT_MAX kroków.

7. Dla $it = 100, 200, 500, 1000, 2000$ sporządzić mapy rozkładu temperatury w pomieszczeniu $T(x, y)$ (50 pkt.)

8. W stanie ustalonym równanie dyfuzji redukuje się do $\nabla^2 T = 0$ (brak zmian w czasie - znika pochodna czasowa). Dla $it = 100, 200, 500, 1000, 2000$ proszę sporządzić mapy rozkładu $\nabla^2 T(x, y)$ i sprawdzić czy tak jest. (50 pkt.)

4 Przydatne procedury z biblioteki GSL

- rozkład LU

  ```
  int gsl_linalg_LU_decomp(gsl_matrix * A, gsl_permutation * p, int * signum)
  ```

  Po wywołaniu procedury znaleziony rozkład LU jest wpisany do macierzy A

- rozwiązanie układu równań z użyciem LU

  ```
  int gsl_linalg_LU_solve(const gsl_matrix * LU, const gsl_permutation * p,
                          const gsl_vector * b, gsl_vector * x)
  ```

- mnożenie macierz-wektor (BLAS 2)

$y = \alpha\, op(A) \cdot \vec{x} + \beta\, \vec{y}$
$op(A) = A, A^T, A^H$

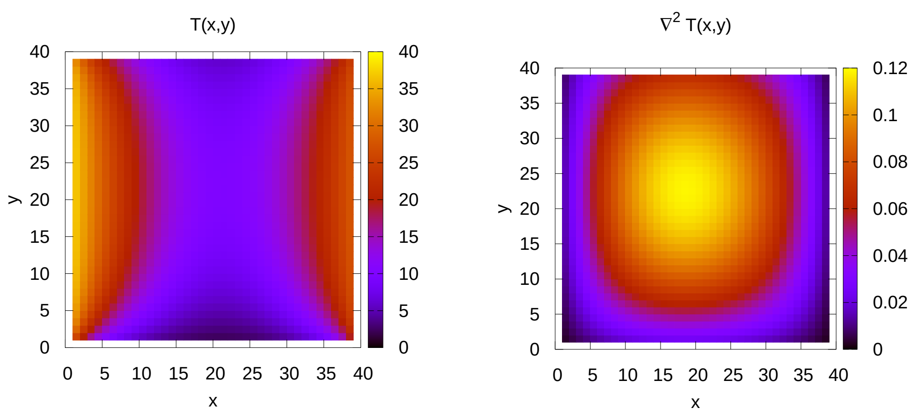
```
int gsl_blas_dgemv(CBLAS_TRANSPOSE_t TransA, double alpha,
                   const gsl_matrix * A, const gsl_vector * x,
                   double beta, gsl_vector * y)

(TransA = CblasNoTrans, CblasTrans, CblasConjTrans)
```

- dodawanie dwóch wektorów (BLAS 1)

  $\vec{y} = \alpha\vec{x} + \vec{y}$

```
int gsl_blas_daxpy(double alpha, const gsl_vector * x, gsl_vector * y)
```

5 Przykładowe wyniki



Rysunek 2: Wyniki dla $it = 100$