非同次 Neumann 境界条件下の熱方程式に対する差分法

MATLAB を使って数値計算

桂田 祐史

2024 年 8 月 18 日書き始め, 2024 年 8 月 24 日公開, 2024 年 9 月 7 日

https://m-katsurada.sakura.ne.jp/labo/text/heat2n-nonhomo.pdf

目 次

1　問題の設定

長方形領域

$$\Omega := (0,W)\times(0,H) = \left\{(x,y)\in\mathbb{R}^2 \;\middle|\; 0<x<W,\ 0<y<H\right\} \tag{1}$$

における熱方程式の初期値境界値問題を差分法で解くプログラムを作ろう。

境界条件は、非同次 Neumann 境界条件とする。これは通常は $\partial\Omega$ で関数 Φ が与えられていて

$$\frac{\partial u}{\partial n}(x,y,t) = \Phi(x,y) \quad ((x,y)\in\partial\Omega,\ t>0))$$

が成り立つということであるが、4 つある角点 $(0,0)$, $(W,0)$, $(0,H)$, (W,H) では、外向き単位法線ベクトル $\boldsymbol{n}$ が存在しないことに注意が必要である。解に緩い滑らかさを仮定すれば、その 4 点では条件を課さなくても、解の一意性や存在が言えそうであるが (変分法的な定式化をするとか、Fourier 級数で考えるとか)、差分法で近似解を求めようという場合は何らかの方程式を与えないと計算ができない (未知数の個数に方程式の個数が足りない状況におちいる)。

ここでは、4 つある角点で $\partial u/\partial x$, $\partial u/\partial y$ が分かっていると仮定する。以下の 4 つの関数 `b_l`, `b_r`, `b_b`, `b_t` が与えられているとする。

(i) 長方形の左辺 (角点含む) で $-\frac{\partial u}{\partial x}$ を与える `b_l`　(`l` は left)

(ii) 長方形の右辺 (角点含む) で $\frac{\partial u}{\partial x}$ を与える `b_r`　(`r` は right)

(iii) 長方形の下辺 (角点含む) で $-\frac{\partial u}{\partial y}$ を与える `b_b`　(`b` は bottom)

(iv) 長方形の右辺 (角点含む) で $\frac{\partial u}{\partial y}$ を与える `b_t`　(`t` は top)

$$\Phi(x,y) = \begin{cases} \texttt{b\_l}(0,y) & (x=0,\ 0<y<H,\ t\geq 0 \text{ のとき}) \\ \texttt{b\_r}(W,y) & (x=W,\ 0<y<H,\ t\geq 0 \text{ のとき}) \\ \texttt{b\_l}(x,0) & (y=0,\ 0<x<W,\ t\geq 0 \text{ のとき}) \\ \texttt{b\_l}(x,H) & (y=H,\ 0<x<W,\ t\geq 0 \text{ のとき}) \end{cases}$$

なお、これらの関数 `b_l`, `b_r`, `b_b`, `b_t` は時刻にも依存するとしても、解くことは難しくない。差分方程式自体はそういう場合に導出することにする。

支配方程式は以下のようになる。

$$u_t(x,y,t) = \triangle u(x,y,t) \qquad ((x,y)\in\Omega,\ t>0), \tag{2a}$$

$$-\frac{\partial u}{\partial x}(0,y,t) = \texttt{b\_l}(0,y,t) \qquad (0\leq y\leq H,\ t>0), \tag{2b}$$

$$\frac{\partial u}{\partial x}(W,y,t) = \texttt{b\_r}(W,y,t) \quad (0\leq y\leq H,\ t>0), \tag{2c}$$

$$-\frac{\partial u}{\partial y}(x,0,t) = \texttt{b\_b}(x,0,t) \qquad (0\leq x\leq W,\ t>0), \tag{2d}$$

$$\frac{\partial u}{\partial y}(x,H,t) = \texttt{b\_t}(x,H,t) \quad (0\leq x\leq W,\ t>0), \tag{2e}$$

$$u(x,y,0) = f(x,y) \qquad ((x,y)\in\overline{\Omega}). \tag{2f}$$

マイナス $-$ がついているのは、`b_*` は、角点を除けば $\frac{\partial u}{\partial n}$ を意味する、という趣旨である。

2　差分方程式

2.1　離散化の設定

変数 (x, y, t) が動く範囲 $\overline{\Omega} \times [0, \infty) = [0, L] \times [0, H] \times [0, \infty)$ を差分格子に切る。
$N_x, N_y \in \mathbb{N}$, $\tau > 0$ に対して、

$$h_x := \frac{W}{N_x}, \quad h_y := \frac{H}{N_y}, \tag{3}$$

$$x_i = ih_x \quad (i = 0, 1, \cdots, N_x), \quad y_j = jh_y \quad (j = 0, 1, \cdots, N_y), \quad t_n = n\tau \quad (n = 0, 1, \cdots) \tag{4}$$

とおき、(x_i, y_j, t_n) を格子点と呼ぶ。

$u^n_{i,j} := u(x_i, y_j, t_n)$ $(0 \le i \le N_x,\ 0 \le j \le N_y,\ n = 0, 1, \cdots)$ の近似値 $U^n_{i,j}$ を求めることを目標にする。$U^n_{i,j}$ は以下に提示する差分方程式の解として定義する。

2.2　θ 法

熱方程式 $\dfrac{\partial u}{\partial t} = \dfrac{\partial^2 u}{\partial x^2} + \dfrac{\partial^2 u}{\partial y^2}$ において、t に関する微分係数を前進差分近似すると、陽解法の方程式

$$\frac{U^{n+1}_{i,j} - U^n_{i,j}}{\tau} = \frac{U^n_{i+1,j} - 2U^n_{i,j} + U^n_{i-1,j}}{h_x^2} + \frac{U^n_{i,j+1} - 2U^n_{i,j} + U^n_{i,j-1}}{h_y^2}$$

が得られる。ここで i, j の範囲はとりあえず $1 \le i \le N_x - 1$, $1 \le j \le N_y - 1$ としておくが、後で仮想格子点を導入する関係で、$0 \le i \le N_x$, $0 \le j \le N_y$ についても考えることになる。

一方、t に関する微分係数を後退差分近似すると、陰解法の方程式

$$\frac{U^{n+1}_{i,j} - U^n_{i,j}}{\tau} = \frac{U^{n+1}_{i+1,j} - 2U^{n+1}_{i,j} + U^{n+1}_{i-1,j}}{h_x^2} + \frac{U^{n+1}_{i,j+1} - 2U^{n+1}_{i,j} + U^{n+1}_{i,j-1}}{h_y^2}$$

が得られる。

θ を $0 \le \theta \le 1$ を満たすパラメーターとして、$(1-\theta) : \theta$ の比で重み付き平均をとった

$$\begin{aligned}\frac{U^{n+1}_{i,j} - U^n_{i,j}}{\tau} = {} & (1-\theta)\left(\frac{U^n_{i+1,j} - 2U^n_{i,j} + U^n_{i-1,j}}{h_x^2} + \frac{U^n_{i,j+1} - 2U^n_{i,j} + U^n_{i,j-1}}{h_y^2}\right) \\ & + \theta\left(\frac{U^{n+1}_{i+1,j} - 2U^{n+1}_{i,j} + U^{n+1}_{i-1,j}}{h_x^2} + \frac{U^{n+1}_{i,j+1} - 2U^{n+1}_{i,j} + U^{n+1}_{i,j-1}}{h_y^2}\right)\end{aligned} \tag{5}$$

を θ 法の差分方程式と呼ぶ。

(5) の両辺に τ をかけて、

$$\lambda_x := \frac{\tau}{h_x^2}, \quad \lambda_y := \frac{\tau}{h_y^2} \tag{6}$$

とおいて代入すると、

$$\begin{aligned}U^{n+1}_{i,j} - U^n_{i,j} = {} & (1-\theta)\left[\lambda_x\left(U^n_{i+1,j} - 2U^n_{i,j} + U^n_{i-1,j}\right) + \lambda_y\left(U^n_{i,j+1} - 2U^n_{i,j} + U^n_{i,j-1}\right)\right] \\ & + \theta\left[\lambda_x\left(U^{n+1}_{i+1,j} - 2U^{n+1}_{i,j} + U^{n+1}_{i-1,j}\right) + \lambda_y\left(U^{n+1}_{i,j+1} - 2U^{n+1}_{i,j} + U^{n+1}_{i,j-1}\right)\right].\end{aligned}$$

移項して ($U^{n+1}_{\circ\triangle}$ は左辺、$U^n_{\circ\triangle}$ は右辺)

$$\begin{aligned}& [1 + 2\theta(\lambda_x + \lambda_y)]\, U^{n+1}_{i,j} - \theta\lambda_x\left(U^{n+1}_{i+1,j} + U^{n+1}_{i-1,j}\right) - \theta\lambda_y\left(U^{n+1}_{i,j+1} + U^{n+1}_{i,j-1}\right) \\ & \quad = [1 - 2(1-\theta)(\lambda_x + \lambda_y)]\, U^n_{i,j} + (1-\theta)\lambda_x\left(U^n_{i+1,j} + U^n_{i-1,j}\right) + (1-\theta)\lambda_y\left(U^n_{i,j+1} + U^n_{i,j-1}\right).\end{aligned}$$

コンパクトに表すため

$$b_x := -\theta\lambda_x, \quad b_y := -\theta\lambda_y, \quad a = 1 + 2\theta\,(\lambda_x + \lambda_y)\,, \tag{7a}$$

$$c_x := (1-\theta)\lambda_x, \quad c_y := (1-\theta)\lambda_y, \quad d = 1 - 2(1-\theta)(\lambda_x + \lambda_y) \tag{7b}$$

とおくと、

$$aU^{n+1}_{i,j} + b_x(U^{n+1}_{i+1,j} + U^{n+1}_{i-1,j}) + b_y(U^{n+1}_{i,j+1} + U^{n+1}_{i,j-1}) = dU^n_{i,j} + c_x(U^n_{i+1,j} + U^n_{i-1,j}) + c_y(U^n_{i,j+1} + U^n_{i,j-1}) \tag{8}$$
$$(0 \le i \le N_x,\ 0 \le j \le N_y,\ n = 0, 1, 2, \cdots).$$

一方、初期条件からは

$$U^0_{i,j} = f(x_i, y_j) \quad (0 \le i \le N_x,\ 0 \le j \le N_y). \tag{9}$$

2.3　仮想格子点の導入による境界条件の中心差分近似

境界条件については、仮想格子点 (x_{-1}, y_j) (x_{N_x+1}, y_j), (x_i, y_{-1}) (x_i, y_{N_y+1}) を導入して、中心差分近似

$$u_x(x_0, y_j) \fallingdotseq \frac{u(x_1, y_j) - u(x_{-1}, y_j)}{2h_x}, \quad u_x(x_{N_x}, y_j) \fallingdotseq \frac{u(x_{N_x+1}, y_j) - u(x_{N_x-1}, y_j)}{2h_x},$$
$$u_y(x_i, y_0) \fallingdotseq \frac{u(x_i, y_1) - u(x_i, y_{-1})}{2h_y}, \quad u_x(x_i, y_{N_y}) \fallingdotseq \frac{u(x_i, y_{N_y+1}) - u(x_i, y_{N_y-1})}{2h_y}$$

を行うと次が得られる。

$$U_{-1,j} = U_{1,j} + 2h_x \mathtt{b\_l}(x_0, y_j) \quad (j = 0, 1, \cdots, N_y), \tag{10a}$$

$$U_{N_x+1,j} = U_{N_x-1,j} + 2h_x \mathtt{b\_r}(x_{N_x}, y_j) \quad (j = 0, 1, \cdots, N_y), \tag{10b}$$

$$U_{i,-1} = U_{i,1} + 2h_y \mathtt{b\_b}(x_i, y_0) \quad (i = 0, 1, \cdots, N_x), \tag{10c}$$

$$U_{i,N_y+1} = U_{i,N_y-1} + 2h_y \mathtt{b\_t}(x_i, y_{N_y}) \quad (i = 0, 1, \cdots, N_x). \tag{10d}$$

2.4　差分方程式 (8) の整理

(10a)–(10d) を用いて、(8) に含まれる仮想格子点上の値 $U^\ell_{-1,j}$, $U^\ell_{N_x+1,j}$, $U^\ell_{i,-1}$, U^ℓ_{i,N_y+1} $(\ell = n, n+1)$ を消去することが出来る。

2.4.1　辺上にない (つまり内部にある) 格子点における差分方程式 (これまでと同じ)

辺上にない格子点における差分方程式は

$$aU^{n+1}_{i,j} + b_x(U^{n+1}_{i+1,j} + U^{n+1}_{i-1,j}) + b_y(U^{n+1}_{i,j+1} + U^{n+1}_{i,j-1}) = dU^n_{i,j} + c_x(U^n_{i+1,j} + U^n_{i-1,j}) + c_y(U^n_{i,j+1} + U^n_{i,j-1}) \tag{11}$$
$$(1 \le i \le N_x - 1,\ 1 \le j \le N_y - 1).$$

左辺を見ると、右辺は機械的に書き下すことができる。

- U^{n+1}_{pq} は U^n_{pq} に、a は d に、b_x は c_x に、 b_y は c_y に、それぞれ置き換えれば良い。

2.4.2　角点における差分方程式

角点 $(0,0)$ における差分方程式は、(8) で $i=j=0$ の場合の式

$$aU_{0,0}^{n+1}+b_x(U_{1,0}^{n+1}+U_{-1,0}^{n+1})+b_y(U_{0,1}^{n+1}+U_{0,-1}^{n+1})=dU_{0,0}^{n}+c_x(U_{1,0}^{n}+U_{-1,0}^{n})+c_y(U_{0,1}^{n}+U_{0,-1}^{n})$$

において

$$\begin{aligned}
U_{1,0}^{n+1}+U_{-1,0}^{n+1}&=2U_{1,0}^{n+1}+2h_x\mathtt{b_l}(x_0,y_0,t_{n+1}),\\
U_{0,1}^{n+1}+U_{0,-1}^{n+1}&=2U_{0,1}^{n+1}+2h_y\mathtt{b_b}(x_0,y_0,t_{n+1}),\\
U_{1,0}^{n}+U_{-1,0}^{n}&=2U_{1,0}^{n+1}+2h_x\mathtt{b_l}(x_0,y_0,t_{n}),\\
U_{0,1}^{n}+U_{0,-1}^{n}&=2U_{0,1}^{n+1}+2h_y\mathtt{b_b}(x_0,y_0,t_{n})
\end{aligned}$$

であるから

$$\begin{aligned}
aU_{0,0}^{n+1}+2b_xU_{1,0}^{n+1}+2b_yU_{0,1}^{n+1}&=dU_{0,0}^{n}+2c_xU_{1,0}^{n}+2c_yU_{0,1}^{n}\\
&\quad+2h_x(c_x\mathtt{b_l}(x_0,y_0,t_n)-b_x\mathtt{b_l}(x_0,y_0,t_{n+1}))\\
&\quad+2h_y(c_y\mathtt{b_b}(x_0,y_0,t_n)-b_y\mathtt{b_b}(x_0,y_0,t_{n+1})).
\end{aligned}\tag{12a}$$

ちなみに、もしも境界値 `b_l()`, `b_b()` が時間に依存しない場合には、(7a), (7b) から得られる

$$c_x-b_x=(1-\theta)\lambda_x-(-\theta\lambda_x)=\lambda_x,\quad c_y-b_y=(1-\theta)\lambda_y-(-\theta\lambda_y)=\lambda_y$$

を用いて、次のように簡単になる。

$$\begin{aligned}
&aU_{0,0}^{n+1}+2b_xU_{1,0}^{n+1}+2b_yU_{0,1}^{n+1}\\
&\quad=dU_{0,0}^{n}+2c_xU_{1,0}^{n}+2c_yU_{0,1}^{n}+2\lambda_xh_x\mathtt{b_l}(x_0,y_0)+2\lambda_yh_y\mathtt{b_b}(x_0,y_0).
\end{aligned}$$

他の 3 つの角点における差分方程式も同様にして、

$$\begin{aligned}
aU_{N_x,0}^{n+1}+2b_xU_{N_x-1,0}^{n+1}+2b_yU_{N_x,1}^{n+1}&=dU_{N_x,0}^{n}+2c_xU_{N_x-1,0}^{n}+2c_yU_{N_x,1}^{n}\\
&\quad+2h_x\left(c_x\mathtt{b_r}(x_{N_x},y_0,t_n)-b_x\mathtt{b_r}(x_{N_x},y_0,t_{n+1})\right)\\
&\quad+2h_y\left(c_y\mathtt{b_b}(x_{N_x},y_0,t_n)-b_y\mathtt{b_b}(x_{N_x},y_0,t_{n+1})\right),
\end{aligned}\tag{12b}$$

$$\begin{aligned}
aU_{0,N_y}^{n+1}+2b_xU_{1,N_y}^{n+1}+2b_yU_{0,N_y-1}^{n+1}&=dU_{0,N_y}^{n}+2c_xU_{1,N_y}^{n}+2c_yU_{0,N_y-1}^{n}\\
&\quad+2h_x\left(c_x\mathtt{b_l}(x_0,y_{N_y},t_n)-b_x\mathtt{b_l}(x_0,y_{N_y},t_{n+1})\right)\\
&\quad+2h_y\left(c_y\mathtt{b_t}(x_0,y_{N_y},t_n)-b_y\mathtt{b_t}(x_0,y_{N_y},t_{n+1})\right).
\end{aligned}\tag{12c}$$

$$\begin{aligned}
aU_{N_x,N_y}^{n+1}+2b_xU_{N_x-1,N_y}^{n+1}+2b_yU_{N_x,N_y-1}^{n+1}&=dU_{N_x,N_y}^{n}+2c_xU_{N_x-1,N_y}^{n}+2c_yU_{N_x,N_y-1}^{n}\\
&\quad+2h_x\left(c_x\mathtt{b_r}(x_{N_x},y_{N_y},t_n)-b_x\mathtt{b_r}(x_{N_x},y_{N_y},t_{n+1})\right)\\
&\quad+2h_y\left(c_y\mathtt{b_t}(x_{N_x},y_{N_y},t_n)-b_y\mathtt{b_t}(x_{N_x},y_{N_y},t_{n+1})\right).
\end{aligned}\tag{12d}$$

2.4.3　角点でない辺上にある格子点における差分方程式

長方形の下の辺上にある角点でない格子点 (x_i,y_0) における差分方程式

$$aU_{i,0}^{n+1}+b_x(U_{i+1,0}^{n+1}+U_{i-1,0}^{n+1})+b_y(U_{i,1}^{n+1}+U_{i,-1}^{n+1})=dU_{i,0}^{n}+c_x(U_{i+1,0}^{n}+U_{i-1,0}^{n})+c_y(U_{i,1}^{n}+U_{i,-1}^{n})$$

において

$$U_{i,1}^{n+1}+U_{i,-1}^{n+1}=2U_{i,1}^{n+1}+2h_y\mathtt{b\_b}(x_i,y_0,t_{n+1}),\quad U_{i,1}^{n}+U_{i,-1}^{n}=2U_{i,1}^{n}+2h_y\mathtt{b\_b}(x_i,y_0,t_{n})$$

であるから

$$aU_{i,0}^{n+1} + b_x(U_{i+1,0}^{n+1} + U_{i-1,0}^{n+1}) + 2b_yU_{i,1}^{n+1} = dU_{i,0}^n + c_x(U_{i+1,0}^n + U_{i-1,0}^n) + 2c_yU_{i,1}^n \\ + 2h_y\left(c_y\texttt{b\_b}(x_i, y_0, t_n) - b_y\texttt{b\_b}(x_i, y_0, t_{n+1})\right) \\ (1 \le i \le N_x - 1). \tag{13a}$$

他の辺上にある格子点 (角点でない) における差分方程式も同様にして導ける。
長方形の上の辺上にある格子点のうち、角点でない点における差分方程式は

$$aU_{i,N_y}^{n+1} + b_x(U_{i+1,N_y}^{n+1} + U_{i-1,N_y}^{n+1}) + 2b_yU_{i,N_y-1}^{n+1} = dU_{i,N_y}^n + c_x(U_{i+1,N_y}^n + U_{i-1,N_y}^n) + 2c_yU_{i,N_y-1}^n \\ + 2h_y\left(c_y\texttt{b\_t}(x_i, y_{N_y}, t_n) - b_y\texttt{b\_t}(x_i, y_{N_y}, t_{n+1})\right) \\ (1 \le i \le N_x - 1). \tag{13b}$$

長方形の左の辺上にある格子点のうち、角点でない点における差分方程式は

$$aU_{0,j}^{n+1} + 2b_xU_{1,j}^{n+1} + b_y(U_{0,j+1}^{n+1} + U_{0,j-1}^{n+1}) = dU_{0,j}^n + 2c_xU_{1,j}^n + c_y(U_{0,j+1}^n + U_{0,j-1}^n) \\ + 2h_x\left(c_x\texttt{b\_l}(x_0, y_j, t_n) - b_x\texttt{b\_l}(x_0, y_j, t_{n+1})\right) \tag{13c}$$

$$(1 \le j \le N_y - 1). \tag{13d}$$

長方形の右の辺上にある格子点のうち、角点でない点における差分方程式は

$$aU_{N_x,j}^{n+1} + 2b_xU_{N_x-1,j}^{n+1} + b_y(U_{N_x,j+1}^{n+1} + U_{N_x,j-1}^{n+1}) = dU_{N_x,j}^n + 2c_xU_{N_x-1,j}^n + c_y(U_{N_x,j+1}^n + U_{N_x,j-1}^n) \\ + 2h_x\left(c_x\texttt{b\_r}(x_{N_x}, y_j, t_n) - b_x\texttt{b\_r}(x_{N_x}, y_j, t_{n+1})\right) \\ (1 \le j \le N_y - 1). \tag{13e}$$

2.4.4 ここまでのまとめ

U_{ij}^{n+1} $(0 \le i \le N_x,\ 0 \le j \le N_y)$ について、$(N_x+1)(N_y+1)$ 個の 1 次方程式からなる連立方程式

(11)	(内部格子点)
(13a), (13b), (13c), (13e)	(辺上にある角点でない格子点)
(12a), (12b), (12c), (12d)	(角点)

が得られた (U_{ij}^n $(0 \le i \le N_x,\ 0 \le j \le N_y)$ を既知として)。

移項のルールは慣れると簡単である。

(a) 辺上にある格子点に対応する方程式で移項が必要である。領域内部にある格子点に対応する方程式では移項は必要ない。

(b) 左右の辺上にある格子点 (x_i, y_y) (つまり $i = 0, N_x$) に対応する方程式では、右辺に

$$2h_x\left(c_xu_x(x_i, y_j, t_n) - b_xu_x(x_i, y_j, t_{n+1})\right)$$

という定数項 ($U_{p,q}^n$ を含まないという意味) が現れる。

(c) 上下の辺上にある格子点 (x_i, y_y) (つまり $j = 0, N_y$) に対応する方程式では

$$2h_y\left(c_yu_y(x_i, y_j, t_n) - b_yu_y(x_i, y_j, t_{n+1})\right)$$

という定数項が現れる。

(d) 角点 (x_i, y_j) (つまり $(i, j) = (0, 0), (N_x, 0), (0, N_y), (N_x, N_y)$) では

$$2h_x\left(c_xu_x(x_i, y_j, t_n) - b_xu_x(x_i, y_j, t_{n+1})\right) + 2h_y\left(c_yu_y(x_i, y_j, t_n) - b_yu_y(x_i, y_j, t_{n+1})\right)$$

となる。

3　対称行列係数の連立一次方程式への帰着

3.1　差分方程式の行列・ベクトル表現

前節で得られた差分方程式を $A\boldsymbol{U} = \boldsymbol{f}$ の形に表そう (A は行列、$\boldsymbol{U}$ と $\boldsymbol{f}$ はベクトル)。

$$U_\ell^n = U_{i,j}^n, \quad \ell = j + (N_x - 1)i \quad (0 \le i \le N_x,\, 0 \le j \le N_y), \tag{14a}$$

$$\boldsymbol{U}^n := \left(U_0^n, \cdots, U_{N-1}^n\right)^\top, \quad N := (N_x + 1)(N_y + 1) \tag{14b}$$

のようにして $\boldsymbol{U}^n$ $(n = 0, 1, \cdots)$ を定義すると、差分方程式は、適当な A, B, $\boldsymbol{c}^n$ を用いて

$$A\boldsymbol{U}^{n+1} = B\boldsymbol{U}^n + \boldsymbol{c}^n \tag{15}$$

の形に表せる。

素朴にやると、係数行列が対称にならないが、対称にするための修正は次項で行う。

((14a), (14b) は、ℓ が変化するとき、列番号 j が速く動き、行番号 i はゆっくり動く、いわゆる row major mode である。MATLAB の meshgrid を使う場合は、row major mode が便利なため、これを採用する。)

$m \in \mathbb{N}$ に対し、I_m, J'_m, K_m を

$$I_m := \begin{pmatrix} 1 & 0 & & & \\ 0 & 1 & 0 & & \\ & \ddots & \ddots & \ddots & \\ & & 0 & 1 & 0 \\ & & & 0 & 1 \end{pmatrix} \in \mathbb{R}^{m\times m}, \quad J'_m := \begin{pmatrix} 0 & 2 & & & \\ 1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 0 & 1 \\ & & & 2 & 0 \end{pmatrix} \in \mathbb{R}^{m\times m}, \quad K_m := -2I_m + J_m \tag{16}$$

で定める。このとき、以下のように A, B, $\boldsymbol{c}^n$ を定めると、前節の差分方程式が (15) の形に表せる。

$$A = I_{N_x+1} \otimes I_{N_y+1} + \theta\lambda_y I_{N_x+1} \otimes K_{N_y+1} + \theta\lambda_x K_{N_x+1} \otimes I_{N_y+1}, \tag{17}$$

$$B = I_{N_x+1} \otimes I_{N_y+1} - (1-\theta)\lambda_y I_{N_x+1} \otimes K_{N_y+1} - (1-\theta)\lambda_x K_{N_x+1} \otimes I_{N_y+1}. \tag{18}$$

(B は、A の定義式の右辺の θ を $-(1-\theta)$ で置き換えたものに等しい。)

$\boldsymbol{c}^n$ は “移項される” 項をまとめたものである。

$$\boldsymbol{c}^n := 2h_x\boldsymbol{c}_x^n + 2h_y\boldsymbol{c}_y^n, \tag{19a}$$

$$\boldsymbol{c}_x^n := \begin{pmatrix} c_x\mathtt{b\_l}(x_0, y_0, t_n) - b_x\mathtt{b\_l}(x_0, y_0, t_{n+1}) \\ c_x\mathtt{b\_l}(x_0, y_1, t_n) - b_x\mathtt{b\_l}(x_0, y_1, t_{n+1}) \\ \cdots \\ c_x\mathtt{b\_l}(x_0, y_{N_y-1}) - b_x\mathtt{b\_l}(x_0, y_{N_y-1}, t_{n+1}) \\ c_x\mathtt{b\_l}(x_0, y_{N_y}) - b_x\mathtt{b\_l}(x_0, y_{N_y}, t_{n+1}) \\ \hline 0 \\ \vdots \\ 0 \\ \hline \vdots \\ \hline \vdots \\ \hline 0 \\ \vdots \\ 0 \\ \hline c_x\mathtt{b\_r}(x_{N_x}, y_0, t_n) - b_x\mathtt{b\_r}(x_{N_x}, y_0, t_{n+1}) \\ c_x\mathtt{b\_r}(x_{N_x}, y_1, t_n) - b_x\mathtt{b\_r}(x_{N_x}, y_1, t_{n+1}) \\ \cdots \\ c_x\mathtt{b\_r}(x_{N_x}, y_{N_y-1}) - b_x\mathtt{b\_r}(x_{N_x}, y_{N_y-1}, t_{n+1}) \\ c_x\mathtt{b\_r}(x_{N_x}, y_{N_y}) - b_x\mathtt{b\_r}(x_{N_x}, y_{N_y}, t_{n+1}) \end{pmatrix}, \tag{19b}$$

$$\boldsymbol{c}_y^n := \begin{pmatrix} c_y\mathtt{b\_b}(x_0, y_0, t_n) - b_y\mathtt{b\_b}(x_0, y_0, t_{n+1}) \\ 0 \\ \vdots \\ 0 \\ c_y\mathtt{b\_t}(x_0, y_{N_y}, t_n) - b_y\mathtt{b\_t}(x_0, y_{N_y}, t_{n+1}) \\ \hline c_y\mathtt{b\_b}(x_1, y_0, t_n) - b_y\mathtt{b\_b}(x_1, y_0, t_{n+1}) \\ 0 \\ \vdots \\ 0 \\ c_y\mathtt{b\_t}(x_1, y_{N_y}, t_n) - b_y\mathtt{b\_t}(x_1, y_{N_y}, t_{n+1}) \\ \hline \vdots \\ \hline \vdots \\ \hline c_y\mathtt{b\_b}(x_{N_x}, y_0, t_n) - b_y\mathtt{b\_b}(x_{N_x}, y_0, t_{n+1}) \\ 0 \\ \vdots \\ 0 \\ c_y\mathtt{b\_t}(x_{N_x}, y_{N_y}, t_n) - b_y\mathtt{b\_t}(x_{N_x}, y_{N_y}, t_{n+1}) \end{pmatrix}. \tag{19c}$$

$\boldsymbol{f}^n := B\boldsymbol{U}^n + \boldsymbol{c}^n$ とおくと、$A\boldsymbol{U}^{n+1} = \boldsymbol{f}^n$. この連立 1 次方程式の係数行列 A は対称でないため、実際の数値計算には適さない。次項では、方程式を変形して、係数行列が対称になるための方法を説明する。

3.2 係数行列の対称化

$\boldsymbol{A}\boldsymbol{U}^{n+1} = \boldsymbol{f}^n$ を、対称行列を係数行列に持つ連立 1 次方程式に変形したい。そのために使える 2 つの方法がある。

3.2.1 1 次元バージョンの説明

まず 1 次元の場合に説明しよう。

方法 1

$$A = \begin{pmatrix} a & 2b & & & \\ b & a & b & & \\ & \ddots & \ddots & \ddots & \\ & & b & a & b \\ & & & 2b & a \end{pmatrix} \in \mathbb{R}^{n\times n}, \quad \boldsymbol{f} = \begin{pmatrix} f_1 \\ f_2 \\ \vdots \\ f_{n-1} \\ f_n \end{pmatrix}, \quad \boldsymbol{x} = \begin{pmatrix} x_1 \\ x_2 \\ \vdots \\ x_{n-1} \\ x_n \end{pmatrix}$$

とするとき、連立一次方程式 $A\boldsymbol{x} = \boldsymbol{f}$ を考える。(Neumann 境界条件を課した 1 次元熱方程式の差分方程式はこの形になる。)

$$S_n := \mathrm{diag}(1/2, 1, \cdots, 1, 1/2) = \begin{pmatrix} \frac{1}{2} & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \frac{1}{2} \end{pmatrix} \in \mathbb{R}^{n\times n}$$

とおき、方程式の両辺に左から S_n をかける。すなわち

$$A' := S_n A, \quad \boldsymbol{f}' = B\boldsymbol{f}$$

とおくと、方程式は $A'\boldsymbol{x} = \boldsymbol{f}'$ に変換されるが、

$$A' = \begin{pmatrix} \frac{a}{2} & b & & & \\ b & a & b & & \\ & \ddots & \ddots & \ddots & \\ & & b & a & b \\ & & & b & \frac{a}{2} \end{pmatrix},$$

であるから、A' は対称行列である。$A'\boldsymbol{x} = \boldsymbol{f}'$ を解いて $\boldsymbol{f}' = (f'_1, \cdots, f'_n)$ が求まれば、

$$\boldsymbol{f} = \begin{pmatrix} 2f'_1 \\ f'_2 \\ \vdots \\ f'_{n-1} \\ 2f'_n \end{pmatrix}$$

により $\boldsymbol{f}$ が求まる。

方法 2　もう一つのやり方は、

$$C := \begin{pmatrix} \sqrt{2} & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \sqrt{2} \end{pmatrix}$$

という行列 C を用意して、$A\boldsymbol{x} = \boldsymbol{f}$ の両辺に左から C^{-1} をかけて、$x = Cy$ と変数変換する。

$$C^{-1}ACy = C^{-1}\boldsymbol{f}.$$

$A' := C^{-1}AC$, $\boldsymbol{f}' := C^{-1}\boldsymbol{f}$ とおくと $A'y = \boldsymbol{f}'$. この A' は実は実対称行列である:

$$A' = A \text{ の第 } 1,n \text{ 行を } 1/\sqrt{2} \text{ 倍して、第 } 1,n \text{ 列を } \sqrt{2} \text{ 倍した行列}$$

$$= \begin{pmatrix} a & \sqrt{2}b & & & & \\ \sqrt{2}b & a & b & & & \\ & b & a & b & & \\ & & \ddots & \ddots & \ddots & \\ & & & b & a & \sqrt{2}b \\ & & & & \sqrt{2}b & a \end{pmatrix}.$$

この方法は、最初の方法より少し面倒であるが、固有値問題 $A\boldsymbol{x} = \lambda\boldsymbol{x}$ を扱うときに役立つ。$x = Cy$ とおくと、$ACy = \lambda Cy$ であるから

$$A'y = \lambda y.$$

A' は A を相似変換したものなので、A' の固有値が A の固有値と一致するのは当然であるが、A' は実対称であるから、固有値 λ がすべて実数であること、相異なる固有値 λ, μ に属する A' の固有ベクトル y, y' は互いに直交することが分かる。

$$y^\top y = y_1y'_1 + y_2y'_2 + \cdots + y_{n-1}y'_{n-1} + y_ny'_n = \frac{1}{2}x_1x'_1 + x_2x'_2 + \cdots + x_{n-1}x'_{n-1} + \frac{1}{2}x_nx'_n$$

であるから、

$$\langle x, x' \rangle := \frac{1}{2}x_1x'_1 + x_2x'_2 + \cdots + x_{n-1}x'_{n-1} + \frac{1}{2}x_nx'_n$$

で定義した内積 $\langle \cdot, \cdot \rangle$ について、A の相異なる固有値に属する固有ベクトルは互いに直交する。

3.2.2　2 次元の場合

第 1 の方法の 2 次元バージョンを採用する。同次 Neumann 境界条件について説明した桂田 [1] では、第 2 の方法を採用したが、Dirichlet 境界条件の場合 ([2]) に近い方が良いと判断した。

次の行列 S を $A\boldsymbol{U}^{n+1} = \boldsymbol{f}^n$ の左からかけた $SA\boldsymbol{U}^{n+1} = S\boldsymbol{f}^n$ の係数行列 SA は実対称行列となる。

$S := S_{N_x+1} \otimes S_{N_y+1}$

$$
= \left(\begin{array}{c|c|c|c|c}
\begin{matrix} \frac{1}{4} & & & & \\ & \frac{1}{2} & & & \\ & & \ddots & & \\ & & & \frac{1}{2} & \\ & & & & \frac{1}{4} \end{matrix} & & & & \\
\hline
& \begin{matrix} \frac{1}{2} & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \frac{1}{2} \end{matrix} & & & \\
\hline
& & \begin{matrix} \ddots & & & & \\ & \ddots & & & \\ & & \ddots & & \\ & & & \ddots & \\ & & & & \ddots \end{matrix} & & \\
\hline
& & & \begin{matrix} \frac{1}{2} & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & \frac{1}{2} \end{matrix} & \\
\hline
& & & & \begin{matrix} \frac{1}{4} & & & & \\ & \frac{1}{2} & & & \\ & & \ddots & & \\ & & & \frac{1}{2} & \\ & & & & \frac{1}{4} \end{matrix}
\end{array}\right).
$$

SA がどうなるか計算 (ほぼ暗算) で分かるので、それで対称であると言っても良いし、次のようにしても分かる。

$$
\begin{aligned}
SA &= (S_{N_x+1} \otimes S_{N_y+1}) \left(I_{N_x+1} \otimes I_{N_y+1} + \theta\lambda_y I_{N_x+1} \otimes K_{N_y+1} + \theta\lambda_x K_{N_x+1} \otimes I_{N_y+1}\right) \\
&= S_{N_x+1} \otimes S_{N_y+1} + \lambda_y S_{N_x+1} \otimes \left(S_{N_y+1} K_{N_y+1}\right) + \theta\lambda_x \left(S_{N_x+1} K_{N_x+1}\right) \otimes S_{N_y+1}
\end{aligned}
$$

(1 次元の場合に確認したように) $S_{N_y+1}K_{N_y+1}$ と $S_{N_x+1}K_{N_x+1}$ はともに対称であるから、SA は対称である (一般に $(A \otimes B)^\top = A^\top \otimes B^\top$ であるから、対称行列のテンソル積は対称である。また対称行列の和は対称行列である。)。

4 MATLAB プログラムによる数値実験

4.1 行列 A, B, S の計算

3.1 の行列 A, B は次のコードで計算できる。

heat2n_mat0.m

```
function [A,B]=heat2n_mat0(Nx,Ny,lambdax,lambday,theta)
  Ix=speye(Nx+1,Nx+1);
  Iy=speye(Ny+1,Ny+1);
  vx=ones(Nx,1);
  Jx=sparse(diag(vx,1)+diag(vx,-1));
  Jx(1,2)=2; Jx(Nx+1,Nx)=2;
  vy=ones(Ny,1);
  Jy=sparse(diag(vy,1)+diag(vy,-1));
  Jy(1,2)=2; Jy(Ny+1,Ny)=2;
  Kx=2*Ix-Jx;
  Ky=2*Iy-Jy;
% column first
  A=kron(Ix,Iy)+theta*lambday*kron(Ix,Ky)+theta*lambdax*kron(Kx,Iy);
  B=kron(Ix,Iy)-(1-theta)*lambday*kron(Ix,Ky)-(1-theta)*lambdax*kron(Kx,Iy);
% row first
%  A=kron(Iy,Ix)+theta*lambdax*kron(Iy,Kx)+theta*lambday*kron(Ky,Ix);
%  B=kron(Iy,Ix)-(1-theta)*lambdax*kron(Iy,Kx)-(1-theta)*lambday*kron(Ky,Ix);
end
```

3.2 の行列 S は次のコードで計算できる。

```
Sx=sparse(diag([1/2;ones(Nx-1,1);1/2]));
Sy=sparse(diag([1/2;ones(Ny-1,1);1/2]));
S=kron(Sx,Sy);
```

4.2 例題とシミュレーション・プログラム

4.2.1 問題の設定

厳密解が分かっている問題を解いて、誤差が十分小さいことを確認しよう。

$$u(x,y,t) = (x-1/2)(y-1/2) + e^{-2\pi^2 t}\cos(\pi z)\cos(\pi y)$$

とおく。(これは調和関数に、同次 Neumann 境界条件の場合の熱方程式の厳密解を足したものである。)
$u_t = \triangle u$ が成り立つ (調和関数は熱方程式を満たす)。
初期値は

$$f(x,y) = u(x,y,0) = (x-1/2)(y-1/2) + \cos(\pi z)\cos(\pi y).$$

境界での法線方向の微分を調べよう。

$$\frac{\partial u}{\partial x}(x,y,t) = y - \frac{1}{2} - e^{-2\pi^2 t}\pi\sin(\pi x)\cos(\pi y), \quad \frac{\partial u}{\partial y}(x,y,t) = x - \frac{1}{2} - e^{-2\pi^2 t}\pi\cos(\pi x)\sin(\pi y)$$

より

$$-\frac{\partial u}{\partial x}(0,y,t) = \frac{1}{2} - y, \quad \frac{\partial u}{\partial x}(1,y,t) = y - \frac{1}{2}, \quad -\frac{\partial u}{\partial y}(x,0,t) = \frac{1}{2} - x, \quad \frac{\partial u}{\partial x}(x,1,t) = x - \frac{1}{2}.$$

ゆえに境界値は次のようにすれば良い。

$$\texttt{b\_l}(0,y,t)=\frac{1}{2}-y,\quad \texttt{b\_r}(1,y,t)=y-\frac{1}{2},\quad \texttt{b\_b}(x,0,t)=\frac{1}{2}-x,\quad \texttt{b\_t}(x,1,t)=x-\frac{1}{2}.$$

角点での値を調べておこう。

$$\begin{aligned}
-\frac{\partial u}{\partial x}(0,0,t)&=\frac{1}{2},\quad -\frac{\partial u}{\partial x}(0,1,t)=-\frac{1}{2},\quad \frac{\partial u}{\partial x}(1,0,t)=-\frac{1}{2},\quad \frac{\partial u}{\partial x}(1,1,t)=\frac{1}{2},\\
-\frac{\partial u}{\partial y}(0,0,t)&=\frac{1}{2},\quad -\frac{\partial u}{\partial y}(1,0,t)=-\frac{1}{2},\quad \frac{\partial u}{\partial x}(0,1,t)=-\frac{1}{2},\quad \frac{\partial u}{\partial x}(1,1,t)=\frac{1}{2}.
\end{aligned}$$

角点では $\frac{\partial u}{\partial n}$ は定義されないが、角点まで込めて連続な拡張を持つことが分かる (左下で $\frac{1}{2}$, 左上で $-\frac{1}{2}$, 右上で $\frac{1}{2}$, 右下で $-\frac{1}{2}$)。

4.2.2 ソース・プログラム heat2n_nh.m

heat2n_nh.m

```
% heat2n_nh.m
% 長方形領域における熱方程式 u_t=△ u（非同次 Neumann 境界条件）を差分法で解く
% 2024/8/24, written by mk.
% 入手 curl -O https://m-katsurada.sakura.ne.jp/program/fdm/heat2n_nh.m
% 解説 https://m-katsurada.sakura.ne.jp/labo/text/heat2n-nonhomo.pdf

function heat2n_nh(Nx, Ny)
  arguments
    Nx = 50;
    Ny = 50;
  end
  function [A,B]=heat2n_mat0(Nx,Ny,lambdax,lambday,theta)
    Ix=speye(Nx+1,Nx+1);
    Iy=speye(Ny+1,Ny+1);
    vx=ones(Nx,1);
    Jx=sparse(diag(vx,1)+diag(vx,-1));
    Jx(1,2)=2; Jx(Nx+1,Nx)=2;
    vy=ones(Ny,1);
    Jy=sparse(diag(vy,1)+diag(vy,-1));
    Jy(1,2)=2; Jy(Ny+1,Ny)=2;
    Kx=2*Ix-Jx;
    Ky=2*Iy-Jy;
  % column first
    A=kron(Ix,Iy)+theta*lambday*kron(Ix,Ky)+theta*lambdax*kron(Kx,Iy);
    B=kron(Ix,Iy)-(1-theta)*lambday*kron(Ix,Ky)-(1-theta)*lambdax*kron(Kx,Iy);
  % row first
  %  A=kron(Iy,Ix)+theta*lambdax*kron(Iy,Kx)+theta*lambday*kron(Ky,Ix);
  %  B=kron(Iy,Ix)-(1-theta)*lambdax*kron(Iy,Kx)-(1-theta)*lambday*kron(Ky,Ix);
  end

  % 長方形領域 (a,b) × (c,d)
  a=0; b=1; c=0; d=1;
  % Neumann 境界値 b() 調和関数を選ぶとこれが平衡解（定常解）
  function val = harmonic(x,y)
    val = (x-0.5).*(y-0.5);
  end
  % 初期値（平衡解を境界値が 0 のもので摂動する）
  function val = iv(x,y)
    val = harmonic(x,y) + cos(pi*x).*cos(pi*y);
  end
  % 厳密解が分かる
  function val = exact(x,y,t)
```

```
  val = harmonic(x,y) + exp(- 2 * pi * pi * t) * cos(pi*x).*cos(pi*y);
end
% 長方形の下の辺での b_b() … -∂ u/∂ y(x,0,t)
function val = b_b(x)
  val = 0.5 - x;
end
% 長方形の上の辺での b_b() … ∂ u/∂ y(x,1,t)
function val = b_t(x)
  val = x - 0.5;
end
% 長方形の左の辺での b_l() … -∂ u/∂ x(0,y,t)
function val = b_l(y)
  val = 0.5 - y;
end
% 長方形の右の辺での b_r() … ∂ u/∂ x(1,y,t)
function val = b_r(y)
  val = y - 0.5;
end
% 誤差を測るためのノルム（最大値ノルム）
function val = mynorm(a)
  [m,n]=size(a);
  val = norm(reshape(a,m*n,1),Inf);
end
% 格子への分割
% 分割数 Nx, Ny は関数引数とした。プログラム先頭の arguments ブロックを見よ。
%Nx=50;
%Ny=50;
hx=(b-a)/Nx;
hy=(d-c)/Ny;
theta=0.5;
tau=0.5/(1/hx^2+1/hy^2); % 陽解法の場合のギリギリの刻み（もっと大きくても OK）
lambdax=tau/hx^2;
lambday=tau/hy^2;
display(sprintf('Nx=%d, Ny=%d, hx=%f, hy=%f, tau=%e, λ x=%f, λ y=%f', ...
  Nx, Ny, hx, hy, tau, lambdax, lambday));

% 差分方程式 A U^{n+1}=B U^n の行列
[A,B]=heat2n_mat0(Nx,Ny,lambdax,lambday,theta);
% 対称にするためのスケーリング用の行列
Sx=sparse(diag([1/2;ones(Nx-1,1);1/2]));
Sy=sparse(diag([1/2;ones(Ny-1,1);1/2]));
S=kron(Sy,Sx);
% A を対称行列に直す
A=S*A;
% 対称かどうかチェックする
er=A-A';
if norm(full(er),inf)>=1
  disp('A が対称でない。')
  pause
end
% LU 分解
[AL,AU,ap]=lu(A,'vector');

% 格子点の座標ベクトル x=(x_1,x_2,...,x_{Nx+1}), y=(y_1,y_2,...,y_{Ny+1})
X=linspace(a,b,Nx+1)';
Y=linspace(c,d,Ny+1)';
% 格子点の x,y 座標の配列 X={X_{ij}}, Y={Y_{ij}}
[x,y]=meshgrid(X,Y);
% 初期値
u= iv(x,y);
% 極限
ulimit = harmonic(x,y);

% 初期値のグラフを描く
```

```
  disp(' 初期値のグラフ')
  meshc(x,y,u); axis([a b c d -1 1]); drawnow; shg;
  disp(' 何かキーを押して下さい')
  pause

  % 境界値 左の辺、右の辺
  blvector=b_l(Y(1:Ny+1));
  brvector=b_r(Y(1:Ny+1));
  lindex=1:Ny+1;
  rindex=Nx*(Ny+1)+1:(Nx+1)*(Ny+1);
  % 境界値 下の辺、上の辺
  bbvector=b_b(X(1:Nx+1));
  btvector=b_t(X(1:Nx+1));
  bindex=1:Ny+1:(Nx+1)*(Ny+1);
  tindex=Ny+1:Ny+1:(Nx+1)*(Ny+1);

  % どこまで計算するか
  Tmax=0.5;
  Nmax = ceil(Tmax/tau);
  % グラフを表示する時間の間隔
  dt=0.005;
  skip=dt/tau;

  U=reshape(u,(Nx+1)*(Ny+1),1);

  disp(' ループに入ります。何かキーを押してください。')
  t=0;
  maxerror=0;
  for n=0:Nmax
    % 連立 1 次方程式の右辺
    U=B*U;
    % 移項
    U(lindex)=U(lindex)+2*hx*lambdax*blvector;
    U(rindex)=U(rindex)+2*hx*lambdax*brvector;
    U(bindex)=U(bindex)+2*hy*lambday*bbvector;
    U(tindex)=U(tindex)+2*hy*lambday*btvector;
    U=S*U;
    % 連立 1 次方程式を解いて差分解を求める
    U=AU\(AL\U(ap,:));
    t=(n+1)*tau;
    % 計算結果の表示
    if mod(n,skip)==0
      u=reshape(U,Ny+1,Nx+1);
      meshc(x,y,u); axis([a b c d -1 1]); drawnow; shg;
      % 誤差の表示
      error = mynorm(u-exact(x,y,t));
      if error > maxerror maxerror = error; end
      display(sprintf('t=%f, ||error||=%e, ||u||=%e, ||u ∞||=%e',...
        t, error, mynorm(u), mynorm(ulimit)));
    end
  end
  display(sprintf('max ||u(・,t)-u(・, ∞)||=%e', mynorm(u-ulimit)));
  display(sprintf('Nx=%d,Ny=%d,max error=%e', Nx, Ny, maxerror));
end
```

4.2.3 入手・コンパイル・実行

まずターミナルで

```
curl -O https://m-katsurada.sakura.ne.jp/program/fdm/heat2n_nh.m
```

これを実行するには、例えば

```
cp -p heat2n_nh.m ~/Documents/MATLAB
```

とコピーして (私はシンボリック・リンクをはることにしている)、MATLAB の中で

```
>> heat2n_nh
```

(分割数はデフォールトの $N_x = 50$, $N_y = 50$ が採用される。)

あるいは、分割数 N_x, N_y を指定して

```
>> heat2n_nh(100,100)
```

とする。

4.2.4 実行結果

$0 \leq t \leq$ Tmax $= 0.5$ における誤差 (max error) は次のようになった。$O(h_x^2 + h_y^2)$ となっていると考えられる。

N_x	N_y	max error
50	50	1.209044e-04
100	100	3.024724e-05
200	200	7.563130e-06

表 1: 分割を細かくしていったとき ($N_x = N_y$ を 50, 100, 200) の誤差の変化

(独白: それにしても、MATLAB の疎行列関係の命令は便利だな。ずっと以前試し始めたとき、Octave とかで使えなかったりして残念に感じたけれど、今はどうなっているのかな。でも今回 `norm()` が使えないのはちょっと驚いた。プログラム中で `full()` を使っているのはそのせい。そのうち不要になったりするのかな。

A Kronecker 積

A.1 定義

$$A \otimes B = \begin{pmatrix} a_{11}B & a_{12}B & \cdots & a_{1n}B \\ a_{21}B & a_{22}B & \cdots & a_{2n}B \\ \vdots & & & \vdots \\ a_{m1}B & a_{m2}B & \cdots & a_{mn}B \end{pmatrix} \quad (A = (a_{ij}) \in \mathbb{R}^{m\times n}, \quad B \in \mathbb{R}^{k\times \ell}).$$

A.2　1 次元同次 Ditichlet

同次 Dirichlet

$$J_m := \begin{pmatrix} 0 & 1 & & & \\ 1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 0 & 1 \\ & & 0 & 1 & 0 \end{pmatrix} \in \mathbb{R}^{m\times m}$$

とおく。

区間を N 等分割するとする。

$$K = -2I_{N-1} + J_{N-1}$$

とおくと $\frac{1}{h^2}K$ が 2 回微分の離散化である。

熱方程式の前進 Euler 法は

$$\frac{1}{\tau}\left(U^{n+1} - U^n\right) = \frac{1}{h^2}KU^n.$$

整理して

$$U^{n+1} - U^n = \lambda KU^n.$$

後退 Euler 法は

$$U^{n+1} - U^n = \lambda KU^{n+1}.$$

θ 法は

$$U^{n+1} - U_n = (1-\theta)\lambda KU^n + \theta\lambda KU^{n+1}$$

移項して

$$(I - \theta\lambda K)U^{n+1} = [I + (1-\theta)\lambda K]U^n.$$

K の代わりに J_{N-1} を使うと見慣れた式が登場する。

$$[(I + 2\theta\lambda)I_{N-1} - \theta\lambda J_{N-1}]U^{n+1} = \{[1 - 2(1-\theta)\lambda]I_{N-1} + (1-\theta)\lambda J_{N-1}\}U^n.$$

1 次元 Dirichlet

```
N=5
myone=ones(N-2,1);
J=diag(myone,1)+diag(myone,-1)
I=eye(N-1,N-1)
K=-2*I+J
theta=0.5
lambda=0.5
A=I-theta*lambda*K
B=I+(1-theta)*lambda*K
```

```
A =

    1.5000   -0.2500         0         0
   -0.2500    1.5000   -0.2500         0
         0   -0.2500    1.5000   -0.2500
         0         0   -0.2500    1.5000


B =

    0.5000    0.2500         0         0
    0.2500    0.5000    0.2500         0
         0    0.2500    0.5000    0.2500
         0         0    0.2500    0.5000
```

A.3 1 次元同次 Neumann

実は同次 Neumann は、J の代わりに $J' := J'_{N+1}$ を使えば良い。
ただし J'_m は

$$J' = \begin{pmatrix} 0 & 2 & & & \\ 1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & 1 & 0 & 1 \\ & & 0 & 2 & 0 \end{pmatrix} \in \mathbb{R}^{m \times m}$$

で定義される。

1 次元 Neumann

```
N=5
myone=ones(N-2,1);
Jp=diag(myone,1)+diag(myone,-1)
Jp(1,2)=2
Jp(N-1,N-2)=2
I=eye(N-1,N-1)
K=-2*I+Jp
theta=0.5
lambda=0.5
A=I-theta*lambda*K
B=I+(1-theta)*lambda*K
```

```
A =

    1.5000   -0.5000         0         0
   -0.2500    1.5000   -0.2500         0
         0   -0.2500    1.5000   -0.2500
         0         0   -0.5000    1.5000


B =

    0.5000    0.5000         0         0
    0.2500    0.5000    0.2500         0
         0    0.2500    0.5000    0.2500
         0         0    0.5000    0.5000
```

A.4 2 次元 Dirichlet

2 次元の Dirichlet の場合。
$(0, W) \times (0, H)$ を N_x 等分, N_y 等分する。

$$h_x := \frac{W}{N_x}, \quad h_y := \frac{H}{N_y}, \quad \lambda_x := \frac{tau}{h_x^2}, \quad \lambda_y := \frac{tau}{h_y^2}$$

とおく。

$$K_x := -2I_{N_x+1} + J'_{N_x+1}, \quad K_y := -2I_{N_y+1} + J'_{N_y+1},$$

とおくと

$$\frac{1}{h_x^2} I_{N_y+1} \otimes K_x + \frac{1}{h_y^2} J_y \otimes I_{N_x+1}$$

は Laplacian の近似となる。
もう面倒なので、I_{N_x+1}, I_{N_y+1}, J_{N_x+1}, J_{N_y+1} をそれぞれ I_x, I_y, J_x, J_y と書くね。
前進 Euler 法は

$$U^{n+1} - U^n = (\lambda_x I_y \otimes K_x + \lambda_y K_y \otimes I_x) U^n.$$

後退 Euler 法は

$$U^{n+1} - U^n = (\lambda_x I_y \otimes K_x + \lambda_y K_y \otimes I_x) U^{n+1}.$$

θ 法は

$$U^{n+1} - U^n = (1-\theta)(\lambda_x I_y \otimes K_x + \lambda_y K_y \otimes I_x) U^n + \theta(\lambda_x I_y \otimes K_x + \lambda_y K_y \otimes I_x) U^{n+1}.$$

整理すると

$$\{[1 - \theta(\lambda_x + \lambda_y)]I - \theta(\lambda_x I_y \otimes J_x + \lambda_y J_y \otimes I_x)\} \boldsymbol{U}^{n+1} = \{[1 + (1-\theta)(\lambda_x + \lambda_y)]I + (1-\theta)(\lambda_x I_y \otimes J_x + \lambda_y J_y \otimes I_x)\} \boldsymbol{U}^n.$$

$$A = \{[1 - \theta(\lambda_x + \lambda_y)]I - \theta(\lambda_x I_y \otimes J_x + \lambda_y J_y \otimes I_x)\},$$
$$B = \{[1 + (1-\theta)(\lambda_x + \lambda_y)]I + (1-\theta)(\lambda_x I_y \otimes J_x + \lambda_y J_y \otimes I_x)\}$$

とおくと

$$A\boldsymbol{U}^{n+1} = B\boldsymbol{U}^n.$$

以上は同次境界条件の場合で、非同次境界条件の場合は、同じ行列を用いて

$$A\boldsymbol{U}^{n+1} = B\boldsymbol{U}^n + \boldsymbol{c}^n$$

という形の差分方程式が得られる。詳しい議論は桂田 [2] を見てもらうとして、結果だけ引用しておくと

$$U_\ell^n = U_{i,j}^n \quad \ell = j + (N_y - 1)(i-1) \quad (1 \le i \le N_x - 1,\ 1 \le j \le N_y - 1) \tag{20}$$

と番号付けたとき

$$A\boldsymbol{U}^{n+1} = B\boldsymbol{U}^n + \lambda_x \begin{pmatrix} b(x_0,y_1) \\ b(x_0,y_2) \\ \vdots \\ b(x_0,y_{N_y-2}) \\ b(x_0,y_{N_y-1}) \\ \hline 0 \\ 0 \\ \vdots \\ 0 \\ 0 \\ \hline \vdots \\ \hline \vdots \\ \hline b(x_{N_x},y_1) \\ b(x_{N_x},y_2) \\ \vdots \\ b(x_{N_x},y_{N_y-2}) \\ b(x_{N_x},y_{N_y-1}) \end{pmatrix} + \lambda_y \begin{pmatrix} b(x_1,y_0) \\ 0 \\ \vdots \\ 0 \\ b(x_1,y_{N_y}) \\ \hline b(x_2,y_0) \\ 0 \\ \vdots \\ 0 \\ b(x_2,y_{N_y}) \\ \hline \vdots \\ \hline \vdots \\ \hline b(x_{N_x-1},y_0) \\ 0 \\ \vdots \\ 0 \\ b(x_{N_x-1},y_{N_y}) \end{pmatrix} \begin{matrix} \leftarrow i=1, j=1 \\ \leftarrow i=1, j=2 \\ \vdots \\ \leftarrow i=1, j=N_y-2 \\ \leftarrow i=1, j=N_y-1 \\ \leftarrow i=2, j=1 \\ \leftarrow i=2, j=2 \\ \vdots \\ \leftarrow i=2, j=N_y-2 \\ \leftarrow i=2, j=N_y-1 \\ \vdots \\ \vdots \\ \leftarrow i=N_x-1, j=1 \\ \leftarrow i=N_x-1, j=2 \\ \vdots \\ \leftarrow i=N_x-1, j=N_y-2 \\ \leftarrow i=N_x-1, j=N_y-1 \end{matrix} \tag{21}$$

2 次元 Dirichlet

```
Nx=4; Ny=4;
myonex=ones(Nx-2,1);
Jx=diag(myonex,1)+diag(myonex,-1)
Ix=eye(Nx-1,Nx-1)
Kx=-2*Ix+Jx
myoney=ones(Ny-2,1);
Jy=diag(myoney,1)+diag(myoney,-1)
Iy=eye(Ny-1,Ny-1)
Ky=-2*Iy+Jy
theta=0.5
lambdax=0.25
lambday=0.25
lambda=lambdax+lambday
I=kron(Iy,Ix);
A=(1-theta*lambda)*I-theta*(lambdax*kron(Iy,Jx)+lambday*kron(Jy,Ix))
B=(1+(1-theta)*lambda)*I+(1-theta)*(lambdax*kron(Iy,Jx)+lambday*kron(Jy,Ix))
```

```
A =

    0.7500   -0.1250         0   -0.1250         0         0         0         0         0
   -0.1250    0.7500   -0.1250         0   -0.1250         0         0         0         0
         0   -0.1250    0.7500         0         0   -0.1250         0         0         0
   -0.1250         0         0    0.7500   -0.1250         0   -0.1250         0         0
         0   -0.1250         0   -0.1250    0.7500   -0.1250         0   -0.1250         0
         0         0   -0.1250         0   -0.1250    0.7500         0         0   -0.1250
         0         0         0   -0.1250         0         0    0.7500   -0.1250         0
         0         0         0         0   -0.1250         0   -0.1250    0.7500   -0.1250
         0         0         0         0         0   -0.1250         0   -0.1250    0.7500


B =

    1.2500    0.1250         0    0.1250         0         0         0         0         0
    0.1250    1.2500    0.1250         0    0.1250         0         0         0         0
         0    0.1250    1.2500         0         0    0.1250         0         0         0
    0.1250         0         0    1.2500    0.1250         0    0.1250         0         0
         0    0.1250         0    0.1250    1.2500    0.1250         0    0.1250         0
         0         0    0.1250         0    0.1250    1.2500         0         0    0.1250
         0         0         0    0.1250         0         0    1.2500    0.1250         0
         0         0         0         0    0.1250         0    0.1250    1.2500    0.1250
         0         0         0         0         0    0.1250         0    0.1250    1.2500
```

A.5 2 次元 Neumann

2 次元の Neumann の場合。

$I'_x = I_{N_x+1}$, $I'_y = I_{N_y+1}$, I_{N_y+1}, $J'_x = J'_{N_x+1}$, $J'_y = J'_{N_y+1}$ を使えば良いはず。

2 次元 Neumann

```
Nx=4; Ny=4;
myonex=ones(Nx,1);
Jxp=diag(myonex,1)+diag(myonex,-1);
Jxp(1,2)=2; Jxp(Nx+1,Nx)=2;
Ixp=eye(Nx+1,Nx+1);
Kxp=-2*Ixp+Jxp
myoney=ones(Ny,1);
Jyp=diag(myoney,1)+diag(myoney,-1)
Jyp(1,2)=2; Jyp(Nx+1,Nx)=2;
Iyp=eye(Ny+1,Ny+1);
Kyp=-2*Iyp+Jyp
theta=0.5
lambdax=0.25
lambday=0.25
lambda=lambdax+lambday
I=kron(Iyp,Ixp);
A=(1-theta*lambda)*I-theta*(lambdax*kron(Iyp,Jxp)+lambday*kron(Jyp,Ixp))
B=(1+(1-theta)*lambda)*I+(1-theta)*(lambdax*kron(Iyp,Jxp)+lambday*kron(Jyp,Ixp))
```

```
A =

    0.7500   -0.2500         0         0         0   -0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
   -0.1250    0.7500   -0.1250         0         0         0   -0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0   -0.1250    0.7500   -0.1250         0         0         0   -0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0   -0.2500    0.7500         0         0         0         0   -0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
   -0.1250         0         0         0         0    0.7500   -0.2500         0         0         0   -0.1250         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0   -0.1250         0         0         0   -0.2500    0.7500         0         0         0         0   -0.1250         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0   -0.1250         0         0         0         0    0.7500   -0.2500         0         0         0   -0.1250         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.2500    0.7500         0         0         0         0   -0.1250         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0   -0.1250         0         0         0         0    0.7500   -0.2500         0         0         0   -0.1250         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.1250    0.7500   -0.1250         0         0         0   -0.1250         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.1250         0         0         0   -0.2500    0.7500         0         0         0         0   -0.1250
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.2500         0         0         0    0.7500   -0.2500         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.2500         0         0         0    0.7500   -0.2500         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.2500         0         0         0    0.7500   -0.2500         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.2500         0         0         0    0.7500   -0.2500         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.2500         0         0         0    0.7500   -0.2500
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0   -0.2500         0         0         0    0.7500

B =

    1.2500    0.2500         0         0         0    0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
    0.1250    1.2500    0.1250         0         0         0    0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0    0.1250    1.2500    0.1250         0         0         0    0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0    0.1250    1.2500    0.1250         0         0         0    0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0    0.2500    1.2500         0         0         0         0    0.2500         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0
    0.1250         0         0         0         0    1.2500    0.2500         0         0         0    0.1250         0         0         0         0         0         0         0         0         0         0         0         0         0         0
         0    0.1250         0         0         0    0.1250    1.2500    0.1250         0         0         0    0.1250         0         0         0         0         0         0         0         0         0         0         0         0         0
         0         0    0.1250         0         0         0    0.1250    1.2500    0.1250         0         0         0    0.1250         0         0         0         0         0         0         0         0         0         0         0         0
         0         0         0    0.1250         0         0         0    0.1250    1.2500    0.1250         0         0         0    0.1250         0         0         0         0         0         0         0         0         0         0         0
         0         0         0         0    0.1250         0         0         0    0.2500    1.2500         0         0         0         0    0.1250         0         0         0         0         0         0         0         0         0         0
         0         0         0         0         0    0.1250         0         0         0         0    1.2500    0.2500         0         0         0    0.1250         0         0         0         0         0         0         0         0         0
         0         0         0         0         0         0    0.1250         0         0         0    0.1250    1.2500    0.1250         0         0         0    0.1250         0         0         0         0         0         0         0         0
         0         0         0         0         0         0         0    0.1250         0         0         0    0.1250    1.2500    0.1250         0         0         0    0.1250         0         0         0         0         0         0         0
         0         0         0         0         0         0         0         0    0.1250         0         0         0    0.1250    1.2500    0.1250         0         0         0    0.1250         0         0         0         0         0         0
         0         0         0         0         0         0         0         0         0    0.1250         0         0         0    0.2500    1.2500         0         0         0         0    0.1250         0         0         0         0         0
         0         0         0         0         0         0         0         0         0         0    0.1250         0         0         0    1.2500    0.2500         0         0         0         0    0.1250         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0    0.1250         0         0         0    1.2500    0.2500         0         0         0         0    0.1250         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0    0.1250         0         0         0    1.2500    0.2500         0         0         0         0    0.1250         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0    0.1250         0         0         0    1.2500    0.2500         0         0         0         0    0.1250         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.1250         0         0         0    1.2500    0.2500         0         0         0         0    0.1250
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.2500         0         0         0    1.2500    0.2500         0         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.2500         0         0         0    1.2500    0.2500         0         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.2500         0         0         0    1.2500    0.2500         0         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.2500         0         0         0    1.2500    0.2500         0
         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0         0    0.2500         0         0         0    1.2500    0.2500

>>
```

参考文献

[1] 桂田祐史：同次 Neumann 境界条件下の熱方程式に対する差分法 — MATLAB を使って数値計算 —, https://m-katsurada.sakura.ne.jp/labo/text/heat2n.pdf (2015 年〜).

[2] 桂田祐史：2 次元熱方程式の非同次 Dirichlet 境界値問題を解く差分法プログラムを作る, https://m-katsurada.sakura.ne.jp/labo/text/heat2d-nonhomo.pdf (2024 年〜).