一维抛物型方程的差分格式

肖名财　刘礼海

一、引言

在研究热传导过程、气体膨胀过程和电磁场的传播等问题时，常常会遇到抛物型偏微分方程。这类方程的自变量中有一个是实际中的时间变量，用 t 表示。本文研究一维非齐次热传导方程的 Dirichlet 初边值问题：

$$\begin{cases} \dfrac{\partial u}{\partial t} - a\dfrac{\partial^2 u}{\partial x^2} = f(x,t), 0 \le x \le 1, 0 \le t \le T & \text{(1a)} \\ u(x,0) = \varphi(x), 0 \le x \le 1 & \text{(1b)} \\ u(0,t) = \alpha(t), u(1,t) = \beta(t), 0 \le t \le 1 & \text{(1c)} \end{cases}$$

其中 a 为正常数，$f(x,t), \varphi(t), \alpha(t), \beta(t)$ 为已知函数，$\varphi(0) = \alpha(0), \varphi(1) = \beta(0)$

称 (1b) 为初值条件，(1c) 为边值条件。

二、数值格式

首先，将区间 [0,1] 作 m 等分，将区间 [0,T] 作 n 等分，记 $h = \dfrac{1}{m}, \tau = \dfrac{T}{h}$，$x_i = ih, 0 \le i \le m, t_k = k\tau, 0 \le k \le n$

分别称 h 和 τ 为空间步长和时间步长。

称在 t=0,x=0,x=1 处的节点为内节点，其他节点为边界节点，称在直线 $t = t_k$ 上的所有节点 $(x_i, t_k)|0 \le i \le m$ 为第 k 层节点。

2.1 向后 Euler 格式

在节点 (x_i, t_k) 处考虑微分方程 (1a)，有

$$\frac{\partial u}{\partial t}(x_i, t_k) - a\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = f(x_i, t_k), 1 \le i \le m-1, 0 \le k \le n-1 \tag{2}$$

用二阶中心差商代替二阶导数

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = \frac{1}{h^2}[u(x_{i-1,t_k}) - 2u(x_i, t_k) + u(x_{i+1}, t_k)] + O(h^2) \tag{3}$$

用向后差商代替一阶导数

$$\frac{\partial u}{\partial t}(x_i, t_k) = \frac{u(x_i, t_k) - u(x_i, t_{k-1})}{\tau} + O(\tau) \tag{4}$$

将 (3) 和 (4) 带入 (2) 式，得

$$\frac{u(x_i,t_k)-u(x_i,t_{k-1})}{\tau}-a\frac{u(x_{i-1,t_k})-2u(x_i,t_k)+u(x_{i+1},t_k)}{h^2}=f(x_i,t_k)+O(h^2+\tau) \quad (5)$$

截断误差为 $O(h^2+\tau)$.

在 (5) 中略去截断误差, 得差分格式

$$\begin{cases} \dfrac{u(x_i,t_k)-u(x_i,t_{k-1})}{\tau}-a\dfrac{u(x_{i-1,t_k})-2u(x_i,t_k)+u(x_{i+1},t_k)}{h^2}=f(x_i,t_k) & \text{(6a)} \\ u_i^0=\varphi(x_i), 0\le i\le m & \text{(6b)} \\ u_0^k=\alpha(t_k), u_m^k=\beta(t_k), 1\le k\le n & \text{(6c)} \end{cases}$$

称 (6a)-(6c) 为向后 Euler 格式.

记 $\gamma=\dfrac{a\tau}{h^2}$, 称 γ 为步长比，(6a) 整理得

$$-\gamma u_{i-1}^k+(1+2\gamma)u_i^k-\gamma u_{i+1}^k=u_i^{k-1}+\tau f(x_i,t_k), 1\le i\le m-1, 0\le k\le n-1 \quad (7)$$

写成矩阵形式

$$Au=f \quad (8)$$

其中

$$A=\begin{bmatrix} 1+2\gamma & -\gamma & & \\ -\gamma & 1+2\gamma & -\gamma & \\ & \ddots & \ddots & \ddots \\ & & -\gamma & 1+2\gamma \end{bmatrix}$$

$$u=\begin{bmatrix} u_1^k \\ u_2^k \\ \vdots \\ u_{m-2}^k \\ u_{m-1}^k \end{bmatrix}$$

.

$$f=\begin{bmatrix} u_1^{k-1}+\gamma u_0^k \\ u_2^{k-1} \\ \vdots \\ u_{m-2}^{k-1} \\ u_{m-1}^{k-1}+\gamma u_m^k \end{bmatrix}+\begin{bmatrix} \tau f(x_1,t_k) \\ \tau f(x_2,t_k) \\ \vdots \\ \tau f(x_{m-2},t_k) \\ \tau f(x_{m-1},t_k) \end{bmatrix}$$

.

解方程 (8), 可以得到数值解。

2.2 Crank-Nicolson 格式

引入半整数点记 $t_{k+\frac{1}{2}} = \frac{1}{2}(t_k + t_{k+1})$

在点 $(x_i, t_{k+\frac{1}{2}})$ 处考虑微分方程 (1a), 有

$$\frac{\partial u}{\partial t}(x_i, t_{k+\frac{1}{2}}) - a\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = f(x_i, t_{k+\frac{1}{2}}), 1 \le i \le m-1, 0 \le k \le n-1 \tag{9}$$

由 Taylor 展开得

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = \frac{\partial^2 u}{\partial x^2}(x_i, t_{k+\frac{1}{2}}) - \frac{\tau}{2}\frac{\partial^3 u}{\partial x^2 \partial t} + \frac{1}{2}(\frac{\tau}{2})^2\frac{\partial^4 u}{\partial x^2 \partial t^2} + O(\tau^3) \tag{10}$$

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1}) = \frac{\partial^2 u}{\partial x^2}(x_i, t_{k+\frac{1}{2}}) + \frac{\tau}{2}\frac{\partial^3 u}{\partial x^2 \partial t} + \frac{1}{2}(\frac{\tau}{2})^2\frac{\partial^4 u}{\partial x^2 \partial t^2} + O(\tau^3) \tag{11}$$

将 (10) 和 (11) 相加得

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+\frac{1}{2}}) = \frac{1}{2}[\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1}) + \frac{\partial^2 u}{\partial x^2}(x_i, t_k)] + O(\tau^2) \tag{12}$$

将 (12) 带入 (9) 得

$$\frac{\partial u}{\partial t}(x_i, t_{k+\frac{1}{2}}) - \frac{1}{2}a[\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1}) + \frac{\partial^2 u}{\partial x^2}(x_i, t_k)] = f(x_i, t_{k+\frac{1}{2}}) + O(\tau^2) \tag{13}$$

用一阶中心差商近似一阶导数

$$\frac{\partial u}{\partial t}(x_i, t_{k+\frac{1}{2}}) = \frac{u(x_i, t_{k+1}) - u(x_i, t_k))}{\tau} + O(\tau^2) \tag{14}$$

用二阶中心差商近似二阶导数

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = \frac{u(x_{i+1}, t_k) - 2u(x_i, t_k) + u(x_{i-1}, t_k)}{h^2} + O(h^2) \tag{15}$$

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1}) = \frac{u(x_{i+1}, t_{k+1}) - 2u(x_i, t_{k+1}) + u(x_{i-1}, t_{k+1})}{h^2} + O(h^2) \tag{16}$$

将 (14)、(15) 和 (16) 代入 (13)，得

$$\begin{aligned} &\frac{u(x_i, t_{k+1}) - u(x_i, t_k))}{\tau} - \frac{a}{2}[\frac{u(x_{i+1}, t_k) - 2u(x_i, t_k) + u(x_{i-1}, t_k)}{h^2} + \\ &\frac{u(x_{i+1}, t_{k+1}) - 2u(x_i, t_{k+1}) + u(x_{i-1}, t_{k+1})}{h^2}] = f(x_i, t_{k+\frac{1}{2}}) + O(h^2 + \tau^2) \end{aligned} \tag{17}$$

略去截断误差，从而可得差分方程

$$\begin{cases} \dfrac{u_i^{k+1} - u_i^k}{\tau} = \dfrac{a}{2}[\dfrac{u_{i+1}^k - 2u_i^k + u_{i-1}^k}{h^2} + \dfrac{u_{i+1}^{k+1} - 2u_i^{k+1} + u_{i-1}^{k+1}}{h^2}] + f(x_i, t_{k+\frac{1}{2}}) & \text{(18a)} \\ u_i^0 = \varphi(x_i), 0 \le i \le m & \text{(18b)} \\ u_0^k = \alpha(t_k), u_m^k = \beta(t_k), 1 \le k \le n & \text{(18c)} \end{cases}$$

截断误差为 $O(\tau^2+h^2)$.

将 (18a) 写成矩阵形式

$$A_1u^{k+1}=A_2u^k+f \tag{19}$$

其中

$$A_1=\begin{bmatrix}1+\gamma & -\frac{\gamma}{2} & & \\ -\frac{\gamma}{2} & 1+\gamma & -\frac{\gamma}{2} & \\ & \ddots & \ddots & \ddots \\ & & -\frac{\gamma}{2} & 1+\gamma\end{bmatrix}$$

$$A_2=\begin{bmatrix}1-\gamma & \frac{\gamma}{2} & & \\ \frac{\gamma}{2} & 1-\gamma & \frac{\gamma}{2} & \\ & \ddots & \ddots & \ddots \\ & & \frac{\gamma}{2} & 1-\gamma\end{bmatrix}$$

$$u^k=\begin{bmatrix}u_1^k\\ u_2^k\\ \vdots\\ u_{m-2}^k\\ u_{m-1}^k\end{bmatrix}$$

.

$$f=\begin{bmatrix}\frac{\gamma}{2}(u_0^k+u_0^{k+1})+\tau f(x_1,t_{k+\frac{1}{2}})\\ \tau f(x_2,t_{k+\frac{1}{2}})\\ \vdots\\ \tau f(x_{m-2},t_{k+\frac{1}{2}})\\ \frac{\gamma}{2}(u_m^k+u_m^{k+1})+\tau f(x_{m-1},t_{k+\frac{1}{2}})\end{bmatrix}$$

解 (19), 即可得数值解。

2.3 二阶 BDF 差分格式

当 k=1 时，直接对 (1a) 运用向后 Euler 差分格式，有

$$\frac{u_i^1-u_i^0}{\tau}=a\frac{u_{i-1}^1-2u_i^1+u_{i+1}^1}{h^2}+f_i^1, 1\le i\le m-1 \tag{20}$$

截断误差 $R=O(h^2+\tau)$

当 $k\ge 2$ 时, 在点 (x_i,t_k) 处考虑方程 (1a)，有

$$\frac{\partial u}{\partial t}(x_i,t_k)=a\frac{\partial^2 u}{\partial t^2}(x_i,t_k)+f(x_i,t_k) \tag{21}$$

将 $u(x_i,t_{k-1})$,$u(x_i,t_{k-2})$ 在点 (x_i,t_k) 处 Taylor 展开，有

$$u(x_i,t_{k-1})=u(x_i,t_k)-\tau\frac{\partial u}{\partial t}(x_i,t_k)+\frac{1}{2!}\tau^2\frac{\partial^2 u}{\partial t^2}(x_i,t_k)-\frac{1}{3!}\tau^3\frac{\partial^3 u}{\partial t^3}(x_i,t_k)+O(\tau^3) \tag{22}$$

$$u(x_i,t_{k-2})=u(x_i,t_k)-2\tau\frac{\partial u}{\partial t}(x_i,t_k)+\frac{1}{2!}(2\tau)^2\frac{\partial^2 u}{\partial t^2}(x_i,t_k)-\frac{1}{3!}(2\tau)^3\frac{\partial^3 u}{\partial t^3}(x_i,t_k)+O(\tau^3) \tag{23}$$

(23) $-\,4\times$ (22) 得

$$\frac{u(x_i,t_{k-2})-4u(x_i,t_{k-1})+3u(x_i,t_k)}{2\tau}=\frac{\partial u}{\partial t}(x_i,t_k)+O(\tau^2) \tag{24}$$

用二阶中心差商代替二阶导数

$$\frac{\partial^2 u}{\partial x^2}(x_i,t_k)=\frac{u(x_{i+1},t_k)-2u(x_i,t_k)+u(x_{i-1},t_k)}{h^2}+O(h^2) \tag{25}$$

将 (24) 和 (25) 代入方程 (21), 有

$$\begin{aligned}\frac{u(x_i,t_{k-2})-4u(x_i,t_{k-1})+3u(x_i,t_k)}{2\tau}&=a\frac{u(x_{i+1},t_k)-2u(x_i,t_k)+u(x_{i-1},t_k)}{h^2}\\&+f(x_i,t_k)+O(\tau^2+h^2)\end{aligned} \tag{26}$$

所以差分格式为

$$\begin{cases}\dfrac{3u_i^k-4u_i^{k-1}+u_i^{k-2}}{2\tau}=a\dfrac{u_{i-1}^k-2u_i^k+u_{i+1}^k}{h^2}+f_i^k, k\geq 2, 1\leq i\leq m-1 & \text{(27a)}\\ \dfrac{u_i^1-u_i^0}{\tau}=a\dfrac{u_{i-1}^1-2u_i^1+u_{i+1}^1}{h^2}+f_i^1, k=1, 1\leq i\leq m-1 & \text{(27b)}\\ u_i^0=\varphi(x_i), 1\leq i\leq m, u_0^k=\alpha(t_k), u_m^k=\beta(t_k), 1\leq k\leq n & \text{(27c)}\end{cases}$$

当 $k\geq 2$ 时，整理为

$$\gamma u_{i-1}^k-(2\gamma+\frac{3}{2})u_i^k+\gamma u_{i+1}^k=-2u_i^{k-1}+\frac{1}{2}u_i^{k-2}-\tau f_i^k \tag{28}$$

写成矩阵形式

$$Au^k=-2u^{k-1}+\frac{1}{2}u^{k-2}+f \tag{29}$$

其中

$$A=\begin{bmatrix}-(2\gamma+\frac{3}{2}) & \gamma & & \\ \gamma & -(2\gamma+\frac{3}{2}) & \gamma & \\ & \ddots & \ddots & \ddots \\ & & -(2\gamma+\frac{3}{2}) & \gamma\end{bmatrix}$$

$$u^k = \begin{bmatrix} u_1^k \\ u_2^k \\ \vdots \\ u_{m-2}^k \\ u_{m-1}^k \end{bmatrix}$$

.

$$f = \begin{bmatrix} -\gamma u_0^k - \tau f(x_1, t_k) \\ -\tau f(x_1, t_k) \\ \vdots \\ -\tau f(x_{m-2}, t_k) \\ -\gamma u_m^k - \tau f(x_{m-1}, t_k) \end{bmatrix}$$

解方程 (29), 即可得数值解。

三、 数值例子

数值例子为

$$\begin{cases} \dfrac{\partial u}{\partial t} = \dfrac{\partial^2 u}{\partial x^2} & , \quad 0 \le x \le 1, 0 \le t \le 1 \\ u(x,0) = e^x & , \quad 0 \le x \le 1 \\ u(0,t) = e^t, u(1,t) = e^{1+t}, & , \quad 0 \le t \le 1 \end{cases} \tag{30}$$

该问题的精确解为 $u(x,t) = e^{x+t}$.

定义误差为

$$E_\infty(h,\tau) = \max_{\substack{1 \le i \le m-1 \\ 1 \le k \le n}} |u(x_i, t_k) - u_i^k)|$$

现在分别使用三种数值格式对该例子进行求解，解题程序运行于 Matlab 2018a.

3.1 向后 Euler 格式

当 $\tau = \dfrac{1}{100}, h = \dfrac{1}{10}$ 时,t=1 处的数值解和精确解见图1, 从图像上看很接近。

当 $\tau = \dfrac{1}{100}, h = \dfrac{1}{10}$ 时, 取 x=0.5, 不同 t 处的值见表1, 随着 t 的增大，误差不断累积，越来越大, 到 t=1 处误差变得最大。

取不同 τ 和 h 时，t=1 处的误差见图2, 步长越小，误差也越小。

取不同步长时，误差和误差比见表2，τ 变为原来的 4 倍，h 变为原来的 2 倍，误差会变为原来的 4 倍，符合 $O(\tau + h^2)$ 的截断误差

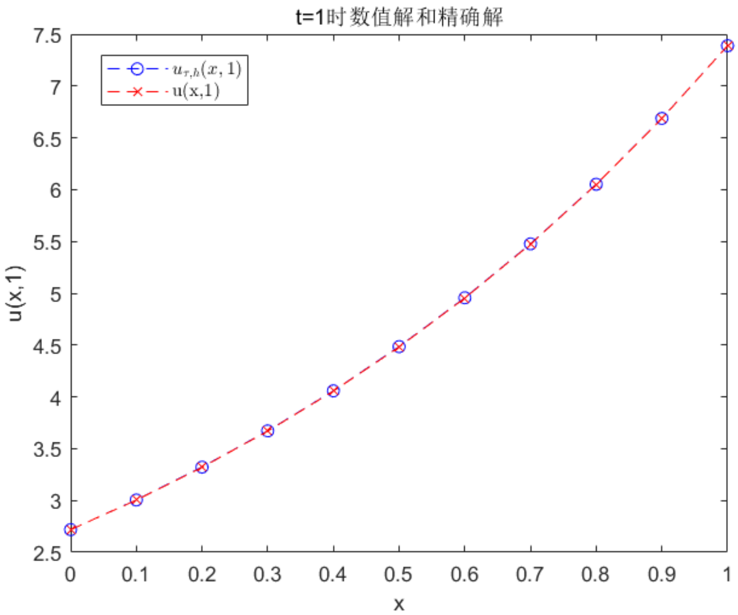

图 **1**　**t=1** 处的数值解和精确解 **(**向后 **Euler** 格式**)**

表 **1**　**x=0.5** 时，不同 **t** 处的数值解、精确解和误差 **(**向后 **Euler** 格式**)**

k	t	数值解	精确解	误差
10	0.1	1.822891	1.822119	7.7174E-04
20	0.2	2.014927	2.013753	1.1743E-03
30	0.3	2.226965	2.225541	1.4242E-03
40	0.4	2.461227	2.459603	1.6237E-03
50	0.5	2.720096	2.718282	1.8140E-03
60	0.6	3.006178	3.004166	2.0124E-03
70	0.7	3.322344	3.320117	2.2271E-03
80	0.8	3.671759	3.669297	2.4625E-03
90	0.9	4.057922	4.055200	2.7219E-03
100	1	4.484697	4.481689	3.0084E-03

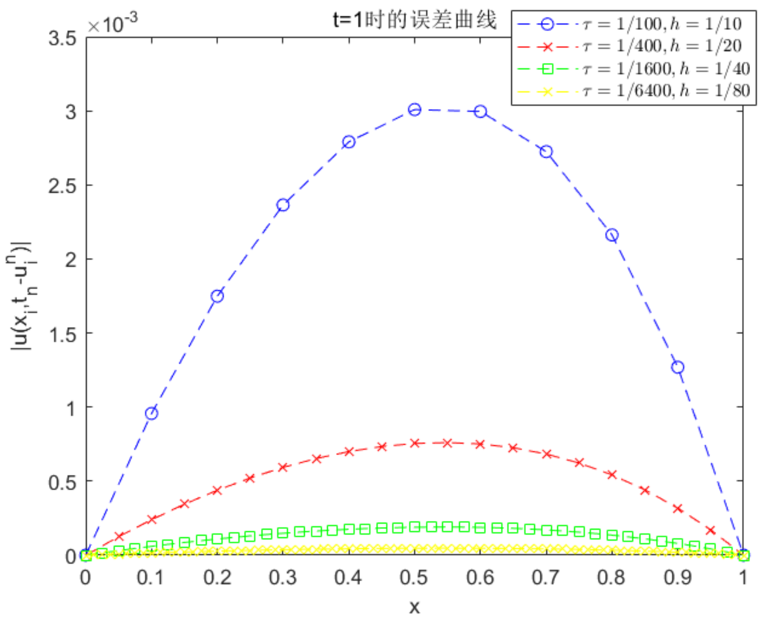

图 2　不同步长下的误差 (向后 **Euler** 格式)

表 2　取不同步长时的误差和误差比 (向后 **Euler** 格式)

h, τ	$E_\infty(h,\tau)$	$E_\infty(2h,4\tau)/E_\infty(h,\tau)$
1/100,1/10	3.0084E-03	*
1/400,1/20	7.6035E-04	3.956615
1/1600,1/40	1.9023E-04	3.997046
1/6400,1/80	4.7566E-05	3.999261

3.2 Crank-Nicolson 格式

当 $\tau=\dfrac{1}{10}, h=\dfrac{1}{10}$ 时,t=1 处的数值解和精确解见图3, 从图像上看很接近。

当 $\tau=\dfrac{1}{10}, h=\dfrac{1}{10}$ 时, 取 x=0.5, 不同 t 处的值见表3, 当层数越深时，误差越大，这是因为, 每一次由 k 层求解 k+1 层时都有误差，随着 t 的增大，误差不断累积，越来越大, 到 t=1 处误差变得最大。

取不同 τ 和 h 时，t=1 处的误差见图4, 步长越小，误差也越小。

使用 Crank-Nicolson 格式求数值解，取不同步长时最大误差和最大误差的比值见表4, τ 变为原来的 2 倍，h 变为原来的 2 倍，误差会变为原来的 4 倍，符合 $O(\tau^2+h^2)$ 的截断误差。

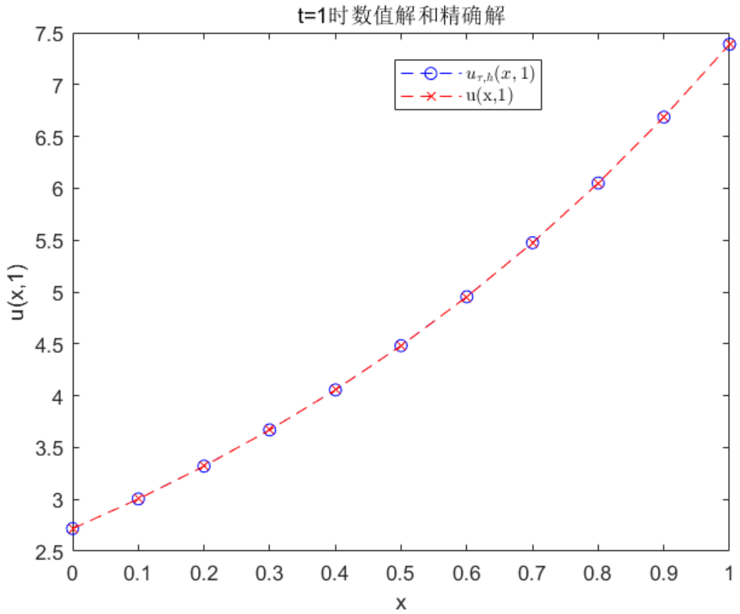

图 **3**　**t=1** 处的数值解和精确解 **(Crank-Nicolson** 格式**)**

表 **3**　**x=0.5** 时，不同 **t** 处的数值解、精确解和误差 **(Crank-Nicolson** 格式**)**

k	t	数值解	精确解	误差
1	0.1	1.822349	1.822119	2.3053E-04
2	0.2	2.014105	2.013753	3.5224E-04
3	0.3	2.225953	2.225541	4.1241E-04
4	0.4	2.460072	2.459603	4.6922E-04
5	0.5	2.718802	2.718282	5.2042E-04
6	0.6	3.004743	3.004166	5.7700E-04
7	0.7	3.320755	3.320117	6.3794E-04
8	0.8	3.670002	3.669297	7.0507E-04
9	0.9	4.055979	4.055200	7.7949E-04
10	1	4.482550	4.481689	8.6123E-04

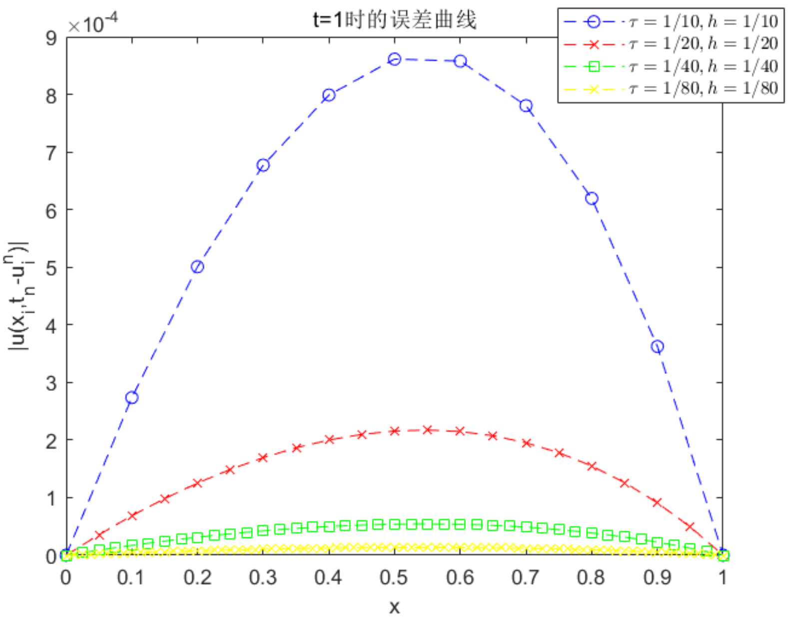

图 4　不同步长下的误差 **(Crank-Nicolson** 格式**)**

表 4　不同步长的最大误差和最大误差的比 **(Crank-Nicolson** 格式**)**

h,τ	$E_\infty(h,\tau)$	$E_\infty(2h,2\tau)/E_\infty(h,\tau)$
1/10,1/10	8.61E-04	*
1/20,1/20	2.17E-04	3.962274
1/40,1/40	5.44E-05	3.998647
1/80,1/80	1.36E-05	3.999657

3.3 二阶 BDF 差分格式

当 $\tau=\dfrac{1}{10},h=\dfrac{1}{10}$ 时,t=1 处的数值解和精确解见图5, 从图像上看很接近。

当 $\tau=\dfrac{1}{10},h=\dfrac{1}{10}$ 时, 取 x=0.5, 不同 t 处的值见表5。

取不同 τ 和 h 时，t=1 处的误差见图6, 步长越小，误差也越小。

取不同步长时，误差和误差比见表6，τ 变为原来的 2 倍，h 变为原来的 2 倍，误差变为原来的 2 到 3 倍，这是因为，虽然 $k\geq 2$ 时，截断误差为 $O(\tau^2+h^2)$，但是 $k=1$ 时，截断误差为 $O(\tau+h^2)$, 所以达不到 4 倍。

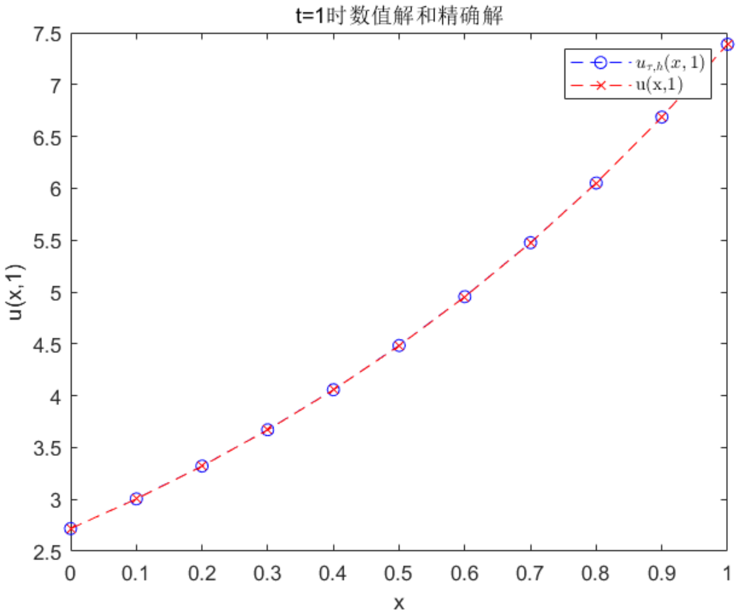

图 **5** **t=1** 处的数值解和精确解 **(**二阶 **BDF** 差分格式**)**

表 **5** **x=0.5** 时，不同 **t** 处的数值解、精确解和误差 **(**二阶 **BDF** 差分格式**)**

k	t	数值解	精确解	误差
1	0.1	1.827620	1.822119	5.50E-03
2	0.2	2.018779	2.013753	5.03E-03
3	0.3	2.228923	2.225541	3.38E-03
4	0.4	2.461791	2.459603	2.19E-03
5	0.5	2.719898	2.718282	1.62E-03
6	0.6	3.005622	3.004166	1.46E-03
7	0.7	3.321620	3.320117	1.50E-03
8	0.8	3.670940	3.669297	1.64E-03
9	0.9	4.057022	4.055200	1.82E-03
10	1	4.483712	4.481689	2.02E-03

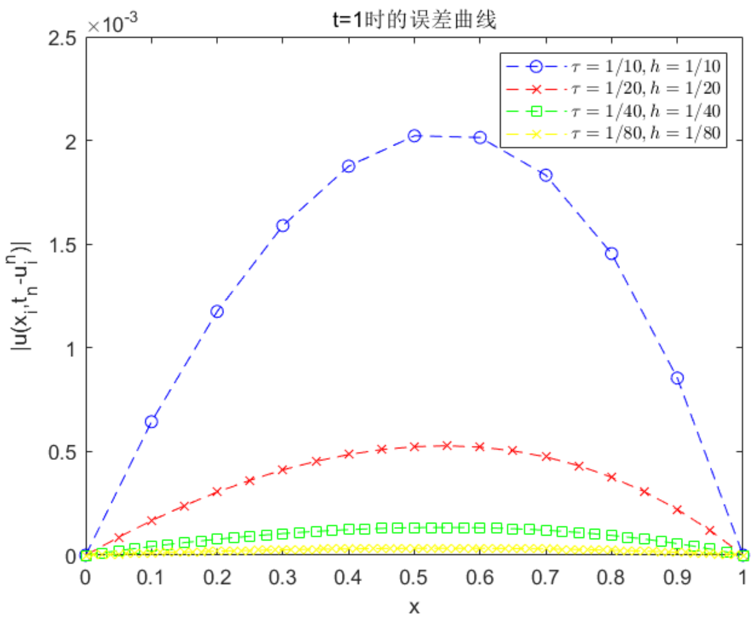

图 **6**　不同步长下的误差 (二阶 **BDF** 差分格式)

表 **6**　取不同步长时的误差和误差比 (二阶 **BDF** 差分格式)

h,τ	$E_\infty(h,\tau)$	$E_\infty(2h,4\tau)/E_\infty(h,\tau)$
1/10,1/10	5.61E-03	*
1/20,1/20	1.90E-03	2.9544
1/40,1/40	6.18E-04	3.0719
1/80,1/80	1.81E-04	3.4202

四、 总结

本文建立了三种一维抛物型方程差分格式：向后 Euler 格式、Crank-Nicolson 格式和二阶 BDF 格式。并运用这三种格式，通过编写 matlab 程序，对具体的算例进行了数值求解。向后 Euler 格式的截断误差为 $O(\tau+h^2)$,Crank-Nicolson 格式的截断误差为 $O(\tau^2+h^2)$, 而二阶 BDF 格式的截断误差则分为两段：当 $k=1$ 时，截断误差为 $O(\tau+h^2)$，当 $k\geq 2$ 时，截断误差为 $O(\tau^2+h^2)$，精度为：Crank-Nicolson 格式 > 二阶 BDF 格式 > 向后 Euler 格式.

参考文献

[1] 李荣华. 偏微分方程数值解法 [M]. 高等教育出版社, 2005.

[2] 孙志忠, 计算数学. 偏微分方程数值解法 [M]. 科学出版社, 2005.

附录 A　程序流程图

程序的流程见图7

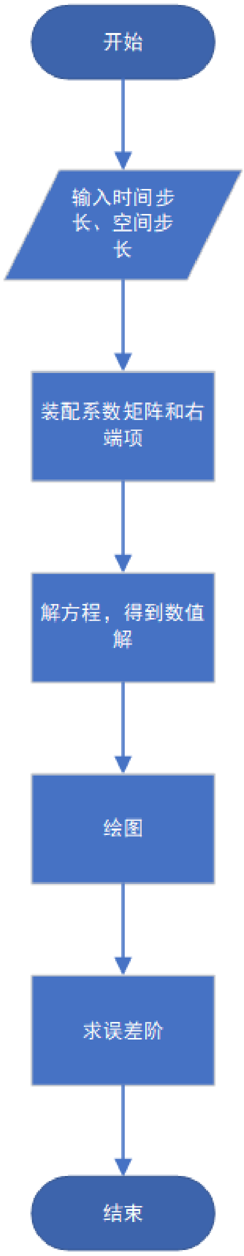

图 **7**　程序流程图

附录 B　向后 Euler 格式代码

fa.m 求精确解

```
function r =fa(t,x)
% 求精确解
```

```
% @t 时间向量
% @x 空间向量
[t,x]=meshgrid(t,x);
t=t';
x=x';
r=exp(x+t);
end
```

fsolve.m 用向后 Euler 格式求数值解

```
function [t,x,u] = fsolve(tau,h)
%  用向后Euler法解抛物方程
%  @t 时间向量
%  @x 空间向量
%  @tau 时间步长
%  @h 空间步长
%  @u 数值解
N=1/tau;%t被分割的区间数
M=1/h;%x被分割的区间数
t=0:tau:1;
x=0:h:1;
u=ones(N+1,M+1);
u(1,:)=exp(x);%初值
%边值
u(:,1)=exp(t);
u(:,M+1)=exp(1+t);

r=tau/h^2;%步长系数
a=ones(M-2,1)*(-r);%下对角线
c=ones(M-2,1)*(-r);%上对角线
b=ones(M-1,1)*(1+2*r);%主队角线
A=diag(b,0)+diag(a,-1)+diag(c,1);%系数矩阵
for k=2:N+1
%右端项
f=u(k-1,2:M);
f(1)=f(1)+r*u(k,1);
f(M-1)=f(M-1)+r*u(k,M+1);
u(k,2:M)=(A\f')';%由k-1层求第k层的值
end
end
```

fig.m 绘制误差图，精确解和数值解对比图

```
clc;clear;
tau=[1/100,1/400,1/1600,1/6400];%时间步长
h=[1/10,1/20,1/40,1/80];%空间步长
K=size(tau,2);
```

```
t_cell=cell(K,1);%时间向量
x_cell=cell(K,1);%空间向量
u_cell=cell(K,1);%数值解
ua_cell=cell(K,1);%精确解
epsilon_cell=cell(K,1);%绝对误差
for n=1:K
[t_cell{n,1},x_cell{n,1},u_cell{n,1}]=fsolve(tau(n),h(n));%求数值解
ua_cell{n,1}=fa(t_cell{n,1},x_cell{n,1});%精确解
epsilon_cell{n,1}=abs(ua_cell{n,1}-u_cell{n,1});%求绝对误差
end
figure(1);
%t=1,tau=1/100,h=1/10时数值解和精确解对比
plot(x_cell{1,1},u_cell{1,1}(1/tau(1)+1,:),'b--o'...
,x_cell{1,1},ua_cell{1,1}(1/tau(1)+1,:),'r--x')
h=legend('$u_{\tau,h}(x,1)$','u(x,1)');
set(h,'Interpreter','latex') %设置legend为latex解释器
title('t=1时数值解和精确解');
xlabel('x');ylabel('u(x,1)');

%绘制t=1时的误差曲线
figure(2)
plot(x_cell{1,1},epsilon_cell{1,1}(1/tau(1)+1,:),'b--o');
hold on;
plot(x_cell{2,1},epsilon_cell{2,1}(1/tau(2)+1,:),'r--x');
hold on;
plot(x_cell{3,1},epsilon_cell{3,1}(1/tau(3)+1,:),'g--s');
hold on;
plot(x_cell{4,1},epsilon_cell{4,1}(1/tau(4)+1,:),'y--x');
hold on;
h=legend('$\tau=1/100,h=1/10$','$\tau=1/400,h=1/20$','$\tau=1/1600,h=1/40$','$\tau=1/6400,h=1/80$');
set(h,'Interpreter','latex') %设置legend为latex解释器
title('t=1时的误差曲线');
xlabel('x');ylabel('|u(x_{i},t_{n}-u_{i}^{n})|');
```

data.m 求特定点的数值解，误差分析。

```
clc;clear;
tao=[1/100,1/400,1/1600,1/6400];%时间步长
h=[1/10,1/20,1/40,1/80];%空间步长
K=size(tao,2);
t_cell=cell(K,1);%时间向量
x_cell=cell(K,1);%空间向量
u_cell=cell(K,1);%数值解
ua_cell=cell(K,1);%精确解
epsilon_cell=cell(K,1);%绝对误差
max_epsilon=ones(K,1);
rate=ones(3,1);%误差比
```

```
for n=1:K
[t_cell{n,1},x_cell{n,1},u_cell{n,1}]=fsolve(tao(n),h(n));%求数值解
ua_cell{n,1}=fa(t_cell{n,1},x_cell{n,1});%精确解
epsilon_cell{n,1}=abs(ua_cell{n,1}-u_cell{n,1});%求绝对误差
max_epsilon(n,1)=max(epsilon_cell{n,1}(:));%求最大误差
end

%E(t,h)/E(4t,2h)
for n=1:K-1
rate(n,1)=max_epsilon(n,1)/max_epsilon(n+1,1);
end

%当x=0.5时，取t=0.1,0.2,...1时，误差的变化,tao=1/100,h=1/10;
u_1=ones(10,1);%特定点数值解
ua_1=ones(10,1);%特定点精确解
[t,x,u]=fsolve(tao(1),h(1));
ua=fa(t,x);
u_1=u(11:10:101,6);
ua_1=ua(11:10:101,6);
epsion_1=abs(u_1-ua_1);%误差
```

附录 C　Crank-Nicolson 格式代码

fa.m 求精确解: 与向后 Euler 格式相同

fsolve.m 用 Crank-Nicolson 格式求数值解

```
function [t,x,u] = fsolve(tau,h)
%   用Crank-Nicolson格式解求数值解
%   @t 时间向量
%   @x 空间向量
%   @tau 时间步长
%   @h 空间步长
%   @u 数值解
N=1/tau;%t被分割的区间数
M=1/h;%x被分割的区间数
t=0:tau:1;
x=0:h:1;
u=ones(N+1,M+1);
u(1,:)=exp(x);%初值
%边值
u(:,1)=exp(t);
u(:,M+1)=exp(1+t);

r=tau/h^2;%步长系数
a1=ones(M-2,1)*(-r/2);%下对角线
```

```
b1=ones(M-1,1)*(1+r);%主队角线
c1=ones(M-2,1)*(-r/2);%上对角线
A1=diag(b1,0)+diag(a1,-1)+diag(c1,1);%系数矩阵

a2=-a1;
b2=ones(M-1,1)*(1-r);
c2=-c1;
A2=diag(b2,0)+diag(a2,-1)+diag(c2,1);%系数矩阵
for k=2:N+1
%右端项
f=A2*u(k-1,2:M)';
f(1)=f(1)+r/2*(u(k,1)+u(k-1,1));
f(M-1)=f(M-1)+r/2*(u(k,M+1)+u(k-1,M+1));
u(k,2:M)=(A1\f)';%由k-1层求第k层的值
end
end
```

fig.m 绘制误差图，精确解和数值解对比图

```
clc;clear;
tau=[1/10,1/20,1/40,1/80];%时间步长
h=[1/10,1/20,1/40,1/80];%空间步长
K=size(tau,2);
t_cell=cell(K,1);%时间向量
x_cell=cell(K,1);%空间向量
u_cell=cell(K,1);%数值解
ua_cell=cell(K,1);%精确解
epsilon_cell=cell(K,1);%绝对误差
for n=1:K
[t_cell{n,1},x_cell{n,1},u_cell{n,1}]=fsolve(tau(n),h(n));%求数值解
ua_cell{n,1}=fa(t_cell{n,1},x_cell{n,1});%精确解
epsilon_cell{n,1}=abs(ua_cell{n,1}-u_cell{n,1});%求绝对误差
end
figure(1);
%t=1,tau=1/100,h=1/10时数值解和精确解对比
plot(x_cell{1,1},u_cell{1,1}(1/tau(1)+1,:),'b--o'...
,x_cell{1,1},ua_cell{1,1}(1/tau(1)+1,:),'r--x')
h=legend('$u_{\tau,h}(x,1)$','u(x,1)');
set(h,'Interpreter','latex') %设置legend为latex解释器
title('t=1时数值解和精确解');
xlabel('x');ylabel('u(x,1)');

%绘制t=1时的误差曲线
figure(2)
plot(x_cell{1,1},epsilon_cell{1,1}(1/tau(1)+1,:),'b--o');
hold on;
plot(x_cell{2,1},epsilon_cell{2,1}(1/tau(2)+1,:),'r--x');
```

```
hold on;
plot(x_cell{3,1},epsilon_cell{3,1}(1/tau(3)+1,:),'g--s');
hold on;
plot(x_cell{4,1},epsilon_cell{4,1}(1/tau(4)+1,:),'y--x');
hold on;
h=legend('$\tau=1/10,h=1/10$','$\tau=1/20,h=1/20$','$\tau=1/40,h=1/40$','$\tau=1/80,h=1/80$');
set(h,'Interpreter','latex') %设置legend为latex解释器
title('t=1时的误差曲线');
xlabel('x');ylabel('|u(x_{i},t_{n}-u_{i}^{n})|');
```

data.m 求特定点的数值解，误差分析。

```
clc;clear;
tau=[1/10,1/20,1/40,1/80];%时间步长
h=[1/10,1/20,1/40,1/80];%空间步长
K=size(tau,2);
t_cell=cell(K,1);%时间向量
x_cell=cell(K,1);%空间向量
u_cell=cell(K,1);%数值解
ua_cell=cell(K,1);%精确解
epsilon_cell=cell(K,1);%绝对误差
max_epsilon=ones(K,1);
rate=ones(3,1);%误差比
for n=1:K
[t_cell{n,1},x_cell{n,1},u_cell{n,1}]=fsolve(tau(n),h(n));%求数值解
ua_cell{n,1}=fa(t_cell{n,1},x_cell{n,1});%精确解
epsilon_cell{n,1}=abs(ua_cell{n,1}-u_cell{n,1});%求绝对误差
max_epsilon(n,1)=max(epsilon_cell{n,1}(:));%求最大误差
end

%E(t,h)/E(2t,2h)
for n=1:K-1
rate(n,1)=max_epsilon(n,1)/max_epsilon(n+1,1);
end

%当x=0.5时，取t=0.1,0.2,...1时，误差的变化(tau=1/10,h=1/10);
u_1=zeros(10,1);%特定点数值解
ua_1=zeros(10,1);%特定点精确解
[t,x,u]=fsolve(tau(1),h(1));
ua=fa(t,x);
u_1=u(2:1:11,1/h(1)/2+1);
ua_1=ua(2:1:11,1/h(1)/2+1);
epsion_1=abs(u_1-ua_1);%误差
```

附录 D　二阶 BDF 格式代码

fa.m 求精确解, 与前面相同

fig.m 绘制误差图，精确解和数值解对比图，与 Crank-Nicolson 格式相同

data.m 求特定点的数值解，误差分析，与 Crank-Nicolson 格式相同

fsolve.m 用 BDF 格式求数值解

```
function [t,x,u] = fsolve(tau,h)
%  用BDF格式解抛物方程
%  @t 时间向量
%  @x 空间向量
%  @tau 时间步长
%  @h 空间步长
%  @u 数值解
N=1/tau;%t被分割的区间数
M=1/h;%x被分割的区间数
t=0:tau:1;
x=0:h:1;
u=ones(N+1,M+1);
u(1,:)=exp(x);%初值
%边值
u(:,1)=exp(t);
u(:,M+1)=exp(1+t);

r=tau/h^2;%步长系数
a=ones(M-2,1)*(-r);%下对角线
c=ones(M-2,1)*(-r);%上对角线
b=ones(M-1,1)*(1+2*r);%主队角线
A1=diag(b,0)+diag(a,-1)+diag(c,1);%向后Euer法系数矩阵

a=ones(M-2,1)*(r);%下对角线
c=ones(M-2,1)*(r);%上对角线
b=ones(M-1,1)*(-(2*r+3/2));%主队角线
A2=diag(b,0)+diag(a,-1)+diag(c,1);%向后Euer法系数矩阵

%由第0层求第1层
f=u(1,2:M);
f(1)=f(1)+r*u(2,1);
f(M-1)=f(M-1)+r*u(2,M+1);
u(2,2:M)=(A1\f')';%由0层求第1层的值

%用Crank-Nicolson格式求出的第1层替代向后Euler法求出的第一层数值解
%    [t2,x2,u2]=fsolve12(tau,h);
%    u(2,2:M)=u2(2,2:M);

%第2层到N层
```

```
for k=3:N+1
%右端项
F=-2*u(k-1,2:M)+1/2*u(k-2,2:M);
F(1)=F(1)-r*u(k,1);
F(M-1)=F(M-1)-r*u(k,M+1);
u(k,2:M)=(A2\F')';%由k-1层求第k层的值
end
end
```