MONTE CARLO METHOD FOR SOLVING A PARABOLIC PROBLEM

by

Yi TIAN and Zai-Zai YAN *

College of Sciences, Inner Mongolia University of Technology, Hohhot, China

Original scientific paper
DOI: 10.2298/TSCI1603933T

In this paper, we present a numerical method based on random sampling for a parabolic problem. This method combines use of the Crank-Nicolson method and Monte Carlo method. In the numerical algorithm, we first discretize governing equations by Crank-Nicolson method, and obtain a large sparse system of linear algebraic equations, then use Monte Carlo method to solve the linear algebraic equations. To illustrate the usefulness of this technique, we apply it to some test problems.

Key words: *Monte Carlo method, Crank-Nicolson method, system of linear algebraic equations*

Introduction

Recently there are many methods to solve heat conduction equations that are used for characterization thermal problems, Tian *et al*. [1] combined the Crank-Nicolson and Monte Carlo methods to solve a class of heat conduction equations. Jia *et al*. [2] used the semi-inverse method to establish a variational principle for an unsteady heat conduction equation. Liu *et al.* [3] adopted He's fractional derivative to study the heat conduction in fractal medium.

In this paper, we will consider a problem of determining an unknown function in the heat conduction equation:

$$u_t \quad a(t)u_{xx} \quad f(x,t), \quad 0 \quad x \quad 1, \quad t \quad 0 \tag{1}$$

with initial condition $u(x, 0) = \varphi(x)$, $0 < x < 1$, and boundary conditions $u(0, t) = a(t)$, $u(1, t) = \beta(t)$, $t > 0$, where $a(t) > 0$ is a known function and $\varphi(x)$, $a(t)$, $\beta(t)$, and $f(x, t)$ are known continuous functions.

The Crank-Nicolson method is employed to discretize the problem domain. Owing to the application of the Crank-Nicolson method, a large sparse system of linear algebraic equations is obtained, then we use Monte Carlo method to solve large systems of linear algebraic equations $AX = b$, where $A \quad R^{n \quad n}$, and $X, b \quad R^n$ [4]. Monte Carlo algorithms have many advantages. For one thing, these algorithms are parallel algorithms, they have high parallel efficiency [5], for another thing, Monte Carlo methods are preferable for solving large sparse systems of linear algebraic equations, such as those arising from approximations of partial differential equation [6-9].

* Corresponding author; e-mail: zz.yan@163.com

Crank-Nicolson method for discretization

The domain $[0, 1] \times [0, T]$ is divided into an $n \times N$ mesh with the spatial-step size $h = 1/n$ in the x direction and the time-step size $\tau = T/N$, respectively. Grid points (x_i, t_k) are defined by $x_i = ih$, $t_k = k\tau$, $(0 \le i \le n, 0 \le k \le N)$. The notation u_i^k is used for the finite difference approximation of $u(ih, k\tau)$.We define the operators:

$$u_i^{k+1/2} = \frac{1}{2}(u_i^k + u_i^{k+1}), \delta_t u_i^{k+1/2} = \frac{1}{\tau}(u_i^{k+1} - u_i^k), \delta_x^2 u_i^k = \frac{1}{h^2}(u_{i+1}^k - 2u_i^k + u_{i-1}^k), \delta_x u_i^k = \frac{1}{h}(u_{i+1/2}^k - u_{i-1/2}^k)$$

Let's consider eq. (1) at point $(x_i, t_{k+1/2})$:

$$\frac{\partial u}{\partial t}(x_i, t_{k+1/2}) = a(t_{k+1/2})\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1/2}) + f(x_i, t_{k+1/2}) \tag{2}$$

by Taylor's formula:

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1}) = \frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1/2}) + \frac{\partial^3 u}{\partial x^2 \partial t}(x_i, t_{k+1/2})(t_{k+1} - t_{k+1/2}) + \frac{1}{2!}\frac{\partial^4 u}{\partial x^2 \partial t^2}(x_i, \xi_{ik})(t_{k+1} - t_{k+1/2})^2, \quad t_{k+1/2} < \xi_{ik} < t_{k+1} \tag{3}$$

and

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = \frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1/2}) + \frac{\partial^3 u}{\partial x^2 \partial t}(x_i, t_{k+1/2})(t_k - t_{k+1/2}) + \frac{1}{2!}\frac{\partial^4 u}{\partial x^2 \partial t^2}(x_i, \xi_{ik})(t_k - t_{k+1/2})^2, \quad t_k < \xi_{ik} < t_{k+1/2} \tag{4}$$

hence

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1/2}) = \frac{1}{2}\left[\frac{\partial^2 u}{\partial x^2}(x_i, t_k) + \frac{\partial^2 u}{\partial x^2}(x_i, t_{k+1})\right] - \frac{\tau^2}{8}\frac{\partial^4 u}{\partial x^2 \partial t^2}(x_i, \zeta_{ik}) \tag{5}$$

where $1 \le i \le n-1$, $k \ge 0$, and $t_k < \zeta_{ik} < t_{k+1}$. Similarly, we have:

$$\frac{\partial u}{\partial x^2}(x_i, t_{k+1/2}) = \delta_t u_i^{k+1/2} - \frac{\tau^2}{24}\frac{\partial^3 u}{\partial t^3}(x_i, \eta_{ik}), \quad t_k < \eta_{ik} < t_{k+1} \tag{6}$$

and

$$\frac{\partial^2 u}{\partial x^2}(x_i, t_k) = \delta_x^2 u_i^k - \frac{h^2}{12}\frac{\partial^4 u}{\partial x^4}(\xi_{ik}, t_k), \quad t_k < \xi_{ik} < t_{k+1} \tag{7}$$

Take eqs. (5)-(7) into (2), we obtain:

$$\delta_i u_i^{k+1/2} - a(t_{k+1/2})\delta_x^2 u_i^{k+1/2} = f(x_i, t_{k+1/2}) + R_{ik} \tag{8}$$

where

$$R_{ik} = \left[\frac{1}{24}\frac{\partial^3 u}{\partial t^3}(x_i, \eta_{ik}) - \frac{a(t_{k+1/2})}{8}\frac{\partial^4 u}{\partial x^2 \partial t^2}(x_i, \zeta_{ik})\right]\tau^2 - \frac{a(t_{k+1/2})}{24}\left[\frac{\partial^4 u}{\partial x^4}(\xi_{ik}, t_k) + \frac{\partial^4 u}{\partial x^4}(\xi_{i,k+1}, t_{k+1})\right]h^2$$

Omit R_{ik} from eq. (8), consider initial condition and boundary conditions at point (x_i, t_k), we have:

$$\delta_i u_o^{k\ \ 1/2} \quad a(t_{k\ \ 1/2})\delta_x^2 u_i^{k\ \ 1/2} \quad f(x_i, t_{k\ \ 1/2}),\quad 1 \quad i \quad n \quad 1, \quad k \quad 0 \tag{9}$$

with $u_i^0 = \varphi(x_i)$, 0 i n, and $u_0^k = \alpha(t_k)$, $u_n^k = \beta(t_k)$, k 0.

According to eq. (9), we have:

$$\frac{a(t_{k\ \ 1/2})r}{2} u_{i\ \ 1}^{k\ \ 1} \quad [1 \quad a(t_{k\ \ 1/2})r]u_i^{k\ \ 1} \quad \frac{a(t_{k\ \ 1/2})r}{2} u_{i\ \ 1}^{k\ \ 1}$$

$$\frac{a(t_{k\ \ 1/2})r}{2} u_{i\ \ 1}^{k} \quad [1 \quad a(t_{k\ \ 1/2})r]u_i^{k} \quad \frac{a(t_{k\ \ 1/2})r}{2} u_{i\ \ 1}^{k} \quad \tau f(x_i, t_{k\ \ 1/2}) \tag{10}$$

where $r = \tau/h^2$, 1 i $n - 1$, k 0.

Equation (10) can be written as the following Matrix form:

$$AU = b \tag{11}$$

where

$$A \quad a(t_{k\ \ 1/2})r \quad \begin{array}{ccccc} 1 & \frac{1}{2} & & & \\ \frac{1}{2} & 1 & \frac{1}{2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{2} & 1 & \frac{1}{2} \\ & & & \frac{1}{2} & 1 \end{array} \qquad \begin{array}{ccccc} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 1 \end{array}$$

$$U^t \quad (u_1^{k\ \ 1} u_2^{k\ \ 1} \cdots u_{n\ \ 2}^{k\ \ 1} u_{n\ \ 1}^{k\ \ 1})$$

$$\overline{b} \quad a(t_{k\ \ 1/2})r \quad \begin{array}{ccccc} 1 & \frac{1}{2} & & & \\ \frac{1}{2} & 1 & \frac{1}{2} & & \\ & \ddots & \ddots & \ddots & \\ & & \frac{1}{2} & 1 & \frac{1}{2} \\ & & & \frac{1}{2} & 1 \end{array} \qquad \begin{array}{ccccc} 1 & & & & \\ & 1 & & & \\ & & \ddots & & \\ & & & 1 & \\ & & & & 1 \end{array}$$

$$b \quad \overline{b} \quad \begin{array}{c} u_1^k \\ u_2^k \\ \vdots \\ u_{n\ \ 2}^k \\ u_{n\ \ 1}^k \end{array} \qquad \begin{array}{c} \frac{a(t_{k\ \ 1/2})}{2}[\alpha(k) \quad \alpha(k \quad 1)] \quad \tau f(x_1, t_{k\ \ 1/2}) \\ \tau f(x_2, t_{k\ \ 1/2}) \\ \vdots \\ \tau f(x_{n\ \ 2}, t_{k\ \ 1/2}) \\ \frac{a(t_{k\ \ 1/2})r}{2}[\beta(k) \quad \beta(k \quad 1)] \quad \tau f(x_{n\ \ 1}, t_{k\ \ 1/2}) \end{array}$$

Discussion of the numerical result

In this section, we perform numerical tests for the algorithm described in [1].

Example 1. Consider the problem $u_t - a(t)u_{xx} = f(x, t)$, $0 < x < 1$, $t > 0$ with $u(x, 0) = (e^{-x} + e^x)e - e^x$, $0 < x < 1$; $u(0, t) = 2e^{e^t} \quad 1$ and $u(1, t) = (e^{-1} + e)e^{e^t} \quad e$, $t > 0$, where $a(t) = e^t$ and $f(x, t) = e^{(x+t)}$. The exact solution is $u(x, t) = (e^{-x} + e^x)e^{e^t} - e^x$.

The results obtained for $u(x, t)$ are presented in tab. 1.

Table 1. Result for *u* with γ = 0.98, *h* = 0.1, *t* = 0.005, *k* = 11, and *N* = 5000

i	$u_{1,k}$		$u_{5,k}$		$u_{9,k}$	
	Numerical	Exact	Numerical	Exact	Numerical	Exact
7	4.556774	4.556724	4.704200	4.703988	5.614308	5.613979
20	4.964895	4.964533	5.161609	5.161554	6.195289	6.195495
50	6.153402	6.153264	6.497873	6.495325	7.891494	7.890568
60	6.647447	6.647192	7.052375	7.049517	8.595858	8.594884
80	7.830298	7.829717	8.376961	8.376323	10.282640	10.281106
100	9.348352	9.347527	10.084721	10.079323	12.444029	12.445429

Example 2. Consider the problem $u_t - a(t)u_{xx} = f(x, t)$, $0 < x < 1$, $t > 0$ with $u(x, 0) = e^x$, $0 < x < 1$; $u(0, t) = 2e^{e^t} \quad 1$ and $u(1, t) = e^{t \quad t^3/3}$, $t > 0$, where $a(t) = t^2 + 1$ and $f(x, t) = xt^2 - x$. The exact solution is $u(x, t) \quad e^{x \quad t \quad t^3/3} \quad xt^3/3 \quad xt$.

The results obtained for $u(x, t)$ are presented in tab. 2.

Table 2. Result for *u* with γ = 0.98, *h* = 0.1, *t* = 0.01, *k* = 15, and *N* = 2000

i	$u_{1,k}$		$u_{5,k}$		$u_{9,k}$	
	Numerical	Exact	Numerical	Exact	Numerical	Exact
10	1.212557	1.211843	1.772773	1.772893	2.628217	2.629488
20	1.333497	1.333730	1.921650	1.920463	2.832696	2.834588
40	1.646830	1.646405	2.329271	2.323305	3.409811	3.407616
60	2.111359	2.111290	2.964007	2.964443	4.342121	4.341071
80	2.853976	2.854390	4.045561	4.037469	5.927285	5.926223
100	4.127121	4.125985	5.936542	5.921368	8.728614	8.730917

Conclusion

In this paper, an alternative method is proposed to solve a parabolic problem. We first discretize governing equations by Crank-Nicolson method, and obtain a large sparse system of linear algebraic equations $Ax = b$, then use Jacobi over-relaxation iterative method and Monte Carlo method to solve the algebraic equations. The numerical results show that the proposed numerical method is accurate to estimate the exact solutions of eq. (1).

Acknowledgment

The work is supported by National Natural Science Foundation of China (Grant No. 11361036), the Joint Specialized Research Fund for the Doctoral Program of Higher Education, Inner Mongolia Education Department (20131514110005).

References

[1] Tian, Y., *et al*., A New Method for Solving a Class of Heat Conduction Equation, *Thermal Science, 19* (2015), 4, pp. 1205-1210

[2] Jia, Z. J., *et al*., Variational Principle for Unsteady Heat Conduction Equation, *Thermal Science, 18* (2014), 3, pp. 1045-1047

[3] Liu, F. J., *et al*., He's Fractional Derivative for Heat Conduction in a Fractal Medium Arising in Silkworm Cocoon Hierarchy, *Thermal Science, 19* (2015), 4, pp. 1155-1159

[4] Dimov, I. T., *et al*., A New Iterative Monte Carlo Approach for Inverse Matrix Problem, *Journal of Computational and Applied Mathematics, 92* (1998), 1, pp. 15-35

[5] Dimov, I. T., *et al*., Monte Carlo Algorithms: Performance Analysis for some Computer Architectures, *Journal of Computational and Applied Mathematics, 48* (1993), 3, pp. 253-277

[6] Farnoosh, R., *et al*., Monte Carlo Simulation Via a Numerical Algorithm for Solving a Nonlinear Inverse Problem, *Commun Nonlinear Sci Numer Simulat, 15* (2010), 9, pp. 2436-2444

[7] Shidfar, A., *et al.*, A Numerical Method for Solving of a Nonlinear Inverse Diffusion Problem, *Computers and Mathematics with Applications, 52* (2006), 6-7, pp. 1021-1030

[8] Shidfar, A., *et al.*, A Numerical Solution Technique for a One-Dimensional Inverse Nonlinear Parabolic Problem, *Applied Mathematica and Computation, 184* (2007), 2, pp. 308-315

[9] Hu, B., *et al*., Crank-Nicolson Finite Difference Scheme for the Rosenau-Burgers Equation, *Applied Mathematica and Computation, 204* (2008), 1, pp. 311-316

Paper submitted: November 9, 2015
Paper revised: February 5, 2016
Paper accepted: February 5, 2016