Advances in Difference Equations
a SpringerOpen Journal

RESEARCH Open Access

A new parallel algorithm for solving parabolic equations

Guanyu Xue[1] and Hui Feng[1*]

*Correspondence:
hfeng.math@whu.edu.cn
[1]School of Mathematics and Statistics, Wuhan University, Wuhan, China

Abstract

In this paper, a new parallel algorithm for solving parabolic equations is proposed. The new algorithm includes two domain decomposition methods, each method is applied to compute the values at $(n + 1)$st time level by use of known numerical solutions at nth time level, respectively. Then the average of two above values is chosen to be the numerical solutions at $(n + 1)$st time level. The new algorithm obtains satisfactory accuracy while maintaining parallelism and unconditional stability. This algorithm can be extended to solve two-dimensional parabolic equations by alternating direction implicit (ADI) technique. Both error analysis and numerical experiments illustrate the accuracy and efficiency of the new algorithm.

Keywords: Parabolic equations; Finite difference; Unconditional stability; Parallelism

1 Introduction

With the development of large-scale scientific and engineering computations, the parallel difference method for parabolic equations has been studied rapidly. In 1983, Evans and Abdullah [1] proposed the Group Explicit (GE) scheme for solving the parabolic equations by Saul'yev asymmetric schemes [2]. Evans [3] first constructed the Alternating Group Explicit (AGE) scheme for the diffusion equation two years later. The Alternating Segment Explicit–Implicit (ASE-I) scheme and the Alternating Segment Crank–Nicolson (ASC-N) scheme were designed in [4–6]. Afterwards, the alternating segment algorithms (AGE scheme, ASE-I scheme, and ASC-N scheme) above became very effective methods for some parabolic equations, such as heat equation [7], convection–diffusion equation [8–10], dispersive equation [11–16], forth-order parabolic equation [17–19]. Meanwhile, domain decomposition methods (DDMs) for the partial differential equations have been studied extensively [20–29]. The concept called "intrinsic parallelism" was presented in [30–32]. In 1999, the alternating difference schemes were presented, the unconditional stability analysis was given in [33]. The unconditionally stable domain decomposition method was obtained by the alternating technique in [34–36]. In fact, an alternating segment algorithm is also a form of the domain decomposition method, which is not only suitable for parallel computation but also unconditionally stable. However, the accuracy of alternating segment algorithm is unsatisfactory.

Inspired by the alternating segment algorithm, we present a new parallel algorithm for parabolic equations in this paper. The new parallel algorithm consists of two DDMs. Each one is applied to compute the values at $(n + 1)$st time level by use of known numerical solutions at nth time level, respectively. Then the average of two above values is chosen to be the numerical solutions at $(n+1)$st time level. The new algorithm can be stated as follows:

1. DDM I is applied to compute the values at $(n+1)$st time level noted as V^{n+1} by use of known value U^n at nth time level.
2. DDM II is also applied to compute the values at $(n+1)$st time level noted as W^{n+1} by use of known value U^n at nth time level.
3. In order to improve accuracy, let $U^{n+1} = (V^{n+1} + W^{n+1})/2$ be the numerical solutions at $(n+1)$st time level.

This paper is organized as follows: In Sect. 2, we introduce a Crank–Nicolson scheme and four corresponding Saul'yev asymmetric difference schemes to construct the parallel algorithm for parabolic equations. For simplicity of presentation, we focus on a model problem, namely one-dimensional parabolic equations. The new parallel algorithm and detailed presentations are given. The accuracy of the new algorithm is given in Sect. 3. The existence and uniqueness of solution by the new algorithm are discussed in Sect. 4, while the stability of the new algorithm is given in Sect. 5. In Sect. 6, we extend the new parallel algorithm to solve two-dimensional parabolic equations by ADI technique. Finally, we give some numerical experiments, which illustrate the accuracy and efficiency of the new algorithm proposed in this paper.

2 Algorithm presentation

Considering the model problem of one-dimensional parabolic equations

$$\frac{\partial u}{\partial t} - a\frac{\partial^2 u}{\partial x^2} = f(x,t), \quad x \in (0,l), t \in (0,T], \tag{1}$$

with the initial and boundary conditions

$$u(x,0) = u_0(x), \quad x \in [0,l], \tag{2}$$

$$u(0,t) = g_0(t), \qquad u(l,t) = g_l(t), \quad t \in (0,T], \tag{3}$$

where $a > 0$ is a constant.

Let h and τ be the spatial and temporal step sizes, respectively. Denote $x_j = jh$, $j = 0,1,\ldots,m$, $t_n = n\tau$, $n = 0,1,\ldots,N$. Let u_j^n be the approximate solution at (x_j,t_n). $u(x,t)$ represents the exact solution of (1).

The Crank–Nicolson scheme and Saul'yev asymmetric difference schemes will be used in our new algorithm. The Crank–Nicolson scheme can be written as

$$-\frac{ar}{2}u_{j-1}^{n+1} + (1+ar)u_j^{n+1} - \frac{ar}{2}u_{j+1}^{n+1} = \frac{ar}{2}u_{j-1}^n + (1-ar)u_j^n + \frac{ar}{2}u_{j+1}^n + \tau f_j^n, \tag{4}$$

where $r = \tau/h^2$.

Corresponding Saul'yev asymmetric difference schemes have four forms as follows:

$$-\frac{ar}{2}u_{j-1}^{n+1} + \left(1+\frac{3ar}{2}\right)u_j^{n+1} - aru_{j+1}^{n+1} = \frac{ar}{2}u_{j-1}^n + \left(1-\frac{ar}{2}\right)u_j^n + \tau f_j^n, \tag{5}$$

$$-aru_{j-1}^{n+1} + \left(1+\frac{3ar}{2}\right)u_j^{n+1} - \frac{ar}{2}u_{j+1}^{n+1} = \left(1-\frac{ar}{2}\right)u_j^n + \frac{ar}{2}u_{j+1}^n + \tau f_j^n, \tag{6}$$

$$\left(1+\frac{ar}{2}\right)u_j^{n+1}-\frac{ar}{2}u_{j+1}^{n+1}=aru_{j-1}^n+\left(1-\frac{3ar}{2}\right)u_j^n+\frac{ar}{2}u_{j+1}^n+\tau f_j^n, \tag{7}$$

$$-\frac{ar}{2}u_{j-1}^{n+1}+\left(1+\frac{ar}{2}\right)u_j^{n+1}=\frac{ar}{2}u_{j-1}^n+\left(1-\frac{3ar}{2}\right)u_j^n+aru_{j+1}^n+\tau f_j^n. \tag{8}$$

Assume $m-1=6K$, where K is a positive integer. We consider two domain decomposition methods, DDM I and DDM II, at $(n+1)$st time level.

DDM I:

For the values u_1^{n+1}, u_2^{n+1}, u_3^{n+1} by using the formulas as follows:

$$\begin{cases}-\frac{ar}{2}u_0^{n+1}+(1+ar)u_1^{n+1}-\frac{ar}{2}u_2^{n+1}=\frac{ar}{2}u_0^n+(1-ar)u_1^n+\frac{ar}{2}u_2^n+\tau f_1^n,\\ -\frac{ar}{2}u_1^{n+1}+(1+ar)u_2^{n+1}-\frac{ar}{2}u_3^{n+1}=\frac{ar}{2}u_1^n+(1-ar)u_2^n+\frac{ar}{2}u_3^n+\tau f_2^n,\\ -\frac{ar}{2}u_2^{n+1}+(1+\frac{ar}{2})u_3^{n+1}=\frac{ar}{2}u_2^n+(1-\frac{3ar}{2})u_3^n+aru_4^n+\tau f_3^n.\end{cases} \tag{9}$$

Finding the values $[u_{6k-2}^{n+1},u_{6k-1}^{n+1},u_{6k}^{n+1},u_{6k+1}^{n+1},u_{6k+2}^{n+1},u_{6k+3}^{n+1}]$ by using the following formulas $(k=1,2,\ldots,K-1)$:

$$\begin{cases}(1+\frac{ar}{2})u_{6k-2}^{n+1}-\frac{ar}{2}u_{6k-1}^{n+1}=aru_{6k-3}^n+(1-\frac{3ar}{2})u_{6k-2}^n+\frac{ar}{2}u_{6k-1}^n+\tau f_{6k-2}^n,\\ -\frac{ar}{2}u_{6k-2}^{n+1}+(1+ar)u_{6k-1}^{n+1}-\frac{ar}{2}u_{6k}^{n+1}=\frac{ar}{2}u_{6k-2}^n+(1-ar)u_{6k-1}^n+\frac{ar}{2}u_{6k}^n+\tau f_{6k-1}^n,\\ -\frac{ar}{2}u_{6k-1}^{n+1}+(1+\frac{3ar}{2})u_{6k}^{n+1}-aru_{6k+1}^{n+1}=\frac{ar}{2}u_{6k-1}^n+(1-\frac{ar}{2})u_{6k}^n+\tau f_{6k}^n,\\ -aru_{6k}^{n+1}+(1+\frac{3ar}{2})u_{6k+1}^{n+1}-\frac{ar}{2}u_{6k+2}^{n+1}=(1-\frac{ar}{2})u_{6k+1}^n+\frac{ar}{2}u_{6k+2}^n+\tau f_{6k+1}^n,\\ -\frac{ar}{2}u_{6k+1}^{n+1}+(1+ar)u_{6k+2}^{n+1}-\frac{ar}{2}u_{6k+3}^{n+1}=\frac{ar}{2}u_{6k+1}^n+(1-ar)u_{6k+2}^n+\frac{ar}{2}u_{6k+3}^n+\tau f_{6k+2}^n,\\ -\frac{ar}{2}u_{6k+2}^{n+1}+(1+\frac{ar}{2})u_{6k+3}^{n+1}=\frac{ar}{2}u_{6k+2}^n+(1-\frac{3ar}{2})u_{6k+3}^n+aru_{6k+4}^n+\tau f_{6k+3}^n.\end{cases} \tag{10}$$

Obviously, each subdomain contains six nodes which can be computed by (10) independently.

For the values u_{m-3}^{n+1}, u_{m-2}^{n+1}, u_{m-1}^{n+1} by using the formulas as follows:

$$\begin{cases}(1+\frac{ar}{2})u_{m-3}^{n+1}-\frac{ar}{2}u_{m-2}^{n+1}=aru_{m-4}^n+(1-\frac{3ar}{2})u_{m-3}^n+\frac{ar}{2}u_{m-2}^n+\tau f_{m-3}^n,\\ -\frac{ar}{2}u_{m-3}^{n+1}+(1+ar)u_{m-2}^{n+1}-\frac{ar}{2}u_{m-1}^{n+1}=\frac{ar}{2}u_{m-3}^n+(1-ar)u_{m-2}^n+\frac{ar}{2}u_{m-1}^n+\tau f_{m-2}^n,\\ -\frac{ar}{2}u_{m-2}^{n+1}+(1+ar)u_{m-1}^{n+1}-\frac{ar}{2}u_m^{n+1}=\frac{ar}{2}u_{m-2}^n+(1-ar)u_{m-1}^n+\frac{ar}{2}u_m^n+\tau f_{m-1}^n.\end{cases} \tag{11}$$

Let $\gamma=ar=a\tau/h^2$, the DDM I can be written as the matrix form:

$$(I+\gamma G_1)U^{n+1}=(I-\gamma G_2)U^n+F_1^n, \tag{12}$$

where $U^n=(u_1^n,u_2^n,\ldots,u_{m-1}^n)^T$, $F_1^n=(\frac{\gamma}{2}(u_0^n+u_0^{n+1})+\tau f_1^n,\tau f_2^n,\ldots,\tau f_{m-2}^n,\frac{\gamma}{2}(u_m^n+u_m^{n+1})+\tau f_{m-1}^n)^T$.

The matrices G_1 and G_2 are block diagonal matrices as follows:

$$G_1=\begin{bmatrix}P_1&&&&\\&Q&&&\\&&\ddots&&\\&&&Q&\\&&&&P_2\end{bmatrix},\qquad G_2=\begin{bmatrix}Q_1&&&&\\&Q&&&\\&&\ddots&&\\&&&Q&\\&&&&Q_2\end{bmatrix},$$

where

$$P_1 = \begin{bmatrix} 1 & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & \frac{1}{2} \end{bmatrix}, \qquad P_2 = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & 0 \\ -\frac{1}{2} & 1 & -\frac{1}{2} \\ 0 & -\frac{1}{2} & 1 \end{bmatrix},$$

$$Q = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & & & & \\ -\frac{1}{2} & 1 & -\frac{1}{2} & & & \\ & -\frac{1}{2} & \frac{3}{2} & -1 & & \\ & & -1 & \frac{3}{2} & -\frac{1}{2} & \\ & & & -\frac{1}{2} & 1 & -\frac{1}{2} \\ & & & & -\frac{1}{2} & \frac{1}{2} \end{bmatrix},$$

$$Q_1 = \begin{bmatrix} 1 & -\frac{1}{2} & & & & \\ -\frac{1}{2} & 1 & -\frac{1}{2} & & & \\ & -\frac{1}{2} & \frac{3}{2} & -1 & & \\ & & -1 & \frac{3}{2} & -\frac{1}{2} & \\ & & & -\frac{1}{2} & 1 & -\frac{1}{2} \\ & & & & -\frac{1}{2} & \frac{1}{2} \end{bmatrix}, \qquad Q_2 = \begin{bmatrix} \frac{1}{2} & -\frac{1}{2} & & & & \\ -\frac{1}{2} & 1 & -\frac{1}{2} & & & \\ & -\frac{1}{2} & \frac{3}{2} & -1 & & \\ & & -1 & \frac{3}{2} & -\frac{1}{2} & \\ & & & -\frac{1}{2} & 1 & -\frac{1}{2} \\ & & & & -\frac{1}{2} & 1 \end{bmatrix}.$$

Each block matrix systems (i.e., each subdomain) can be solved independently. It is evident that DDM I (12) has intrinsic parallelism.

DDM II:

For the values $u_1^{n+1}, u_2^{n+1}, \dots, u_6^{n+1}$ by using the following formulas:

$$\begin{cases} -\frac{ar}{2}u_0^{n+1} + (1+ar)u_1^{n+1} - \frac{ar}{2}u_2^{n+1} = \frac{ar}{2}u_0^n + (1-ar)u_1^n + \frac{ar}{2}u_2^n + \tau f_1^n, \\ -\frac{ar}{2}u_1^{n+1} + (1+ar)u_2^{n+1} - \frac{ar}{2}u_3^{n+1} = \frac{ar}{2}u_1^n + (1-ar)u_2^n + \frac{ar}{2}u_3^n + \tau f_2^n, \\ -\frac{ar}{2}u_2^{n+1} + (1+\frac{3ar}{2})u_3^{n+1} - aru_4^{n+1} = \frac{ar}{2}u_2^n + (1-\frac{ar}{2})u_3^n + \tau f_3^n, \\ -aru_3^{n+1} + (1+\frac{3ar}{2})u_4^{n+1} - \frac{ar}{2}u_5^{n+1} = (1-\frac{ar}{2})u_4^n + \frac{ar}{2}u_5^n + \tau f_4^n, \\ -\frac{ar}{2}u_4^{n+1} + (1+ar)u_5^{n+1} - \frac{ar}{2}u_6^{n+1} = \frac{ar}{2}u_4^n + (1-ar)u_5^n + \frac{ar}{2}u_6^n + \tau f_5^n, \\ -\frac{ar}{2}u_5^{n+1} + (1+\frac{ar}{2})u_6^{n+1} = \frac{ar}{2}u_5^n + (1-\frac{3ar}{2})u_6^n + aru_7^n + \tau f_6^n. \end{cases} \tag{13}$$

Finding the values $[u_{6k+1}^{n+1}, u_{6k+2}^{n+1}, u_{6k+3}^{n+1}, u_{6k+4}^{n+1}, u_{6k+5}^{n+1}, u_{6k+6}^{n+1}]$ by using the formulas as follows ($k = 1, 2, \dots, K-2$):

$$\begin{cases} (1+\frac{ar}{2})u_{6k+1}^{n+1} - \frac{ar}{2}u_{6k+2}^{n+1} = aru_{6k}^n + (1-\frac{3ar}{2})u_{6k+1}^n + \frac{ar}{2}u_{6k+2}^n + \tau f_{6k+1}^n, \\ -\frac{ar}{2}u_{6k+1}^{n+1} + (1+ar)u_{6k+2}^{n+1} - \frac{ar}{2}u_{6k+3}^{n+1} = \frac{ar}{2}u_{6k+1}^n + (1-ar)u_{6k+2}^n + \frac{ar}{2}u_{6k+3}^n + \tau f_{6k+2}^n, \\ -\frac{ar}{2}u_{6k+2}^{n+1} + (1+\frac{3ar}{2})u_{6k+3}^{n+1} - aru_{6k+4}^{n+1} = \frac{ar}{2}u_{6k+2}^n + (1-\frac{ar}{2})u_{6k+3}^n + \tau f_{6k+3}^n, \\ -aru_{6k+3}^{n+1} + (1+\frac{3ar}{2})u_{6k+4}^{n+1} - \frac{ar}{2}u_{6k+5}^{n+1} = (1-\frac{ar}{2})u_{6k+4}^n + \frac{ar}{2}u_{6k+5}^n + \tau f_{6k+4}^n, \\ -\frac{ar}{2}u_{6k+4}^{n+1} + (1+ar)u_{6k+5}^{n+1} - \frac{ar}{2}u_{6k+6}^{n+1} = \frac{ar}{2}u_{6k+4}^n + (1-ar)u_{6k+5}^n + \frac{ar}{2}u_{6k+6}^n + \tau f_{6k+5}^n, \\ -\frac{ar}{2}u_{6k+5}^{n+1} + (1+\frac{ar}{2})u_{6k+6}^{n+1} = \frac{ar}{2}u_{6k+5}^n + (1-\frac{3ar}{2})u_{6k+6}^n + aru_{6k+7}^n + \tau f_{6k+6}^n. \end{cases} \tag{14}$$

Obviously, each subdomain contains six nodes which can be computed by (14) independently.

Finding the values $u_{m-6}^{n+1}, u_{m-5}^{n+1}, \dots, u_{m-1}^{n+1}$ by using the formulas as follows:

$$\begin{cases} (1+\frac{ar}{2})u_{m-6}^{n+1} - \frac{ar}{2}u_{m-5}^{n+1} = aru_{m-7}^{n} + (1-\frac{3ar}{2})u_{m-6}^{n} + \frac{ar}{2}u_{m-5}^{n} + \tau f_{m-6}^{n}, \\ -\frac{ar}{2}u_{m-6}^{n+1} + (1+ar)u_{m-5}^{n+1} - \frac{ar}{2}u_{m-4}^{n+1} = \frac{ar}{2}u_{m-6}^{n} + (1-ar)u_{m-5}^{n} + \frac{ar}{2}u_{m-4}^{n} + \tau f_{m-5}^{n}, \\ -\frac{ar}{2}u_{m-5}^{n+1} + (1+\frac{3ar}{2})u_{m-4}^{n+1} - aru_{m-3}^{n+1} = \frac{ar}{2}u_{m-5}^{n} + (1-\frac{ar}{2})u_{m-4}^{n} + \tau f_{m-4}^{n}, \\ -aru_{m-4}^{n+1} + (1+\frac{3ar}{2})u_{m-3}^{n+1} - \frac{ar}{2}u_{m-2}^{n+1} = (1-\frac{ar}{2})u_{m-3}^{n} + \frac{ar}{2}u_{m-2}^{n} + \tau f_{m-3}^{n}, \\ -\frac{ar}{2}u_{m-3}^{n+1} + (1+ar)u_{m-2}^{n+1} - \frac{ar}{2}u_{m-1}^{n+1} = \frac{ar}{2}u_{m-3}^{n} + (1-ar)u_{m-2}^{n} + \frac{ar}{2}u_{m-1}^{n} + \tau f_{m-2}^{n}, \\ -\frac{ar}{2}u_{m-2}^{n+1} + (1+ar)u_{m-1}^{n+1} - \frac{ar}{2}u_{m}^{n+1} = \frac{ar}{2}u_{m-2}^{n} + (1-ar)u_{m-1}^{n} + \frac{ar}{2}u_{m}^{n} + \tau f_{m-1}^{n}. \end{cases} \tag{15}$$

The DDM II can be written as the matrix form:

$$(I+\gamma G_2)U^{n+1} = (I-\gamma G_1)U^n + F_1^n, \tag{16}$$

where $U^n = (u_1^n, u_2^n, \dots, u_{m-1}^n)^T$, $F_1^n = (\frac{\gamma}{2}(u_0^n + u_0^{n+1}) + \tau f_1^n, \tau f_2^n, \dots, \tau f_{m-2}^n, \frac{\gamma}{2}(u_m^n + u_m^{n+1}) + \tau f_{m-1}^n)^T$.

The matrices G_1 and G_2 are block diagonal matrices as follows:

$$G_1 = \begin{bmatrix} P_1 & & & & \\ & Q & & & \\ & & \ddots & & \\ & & & Q & \\ & & & & P_2 \end{bmatrix}, \qquad G_2 = \begin{bmatrix} Q_1 & & & & \\ & Q & & & \\ & & \ddots & & \\ & & & Q & \\ & & & & Q_2 \end{bmatrix}.$$

Each block matrix system (i.e., each subdomain) can be solved independently. It is evident that DDM II (16) has intrinsic parallelism.

Schemes (9)–(11) and schemes (13)–(15) construct two domain decomposition methods (12) and (16), respectively. The corresponding algorithm can be described as follows in Algorithm 1.

The matrix form of Algorithm 1 can be written as follows:

$$\begin{cases} (I+\gamma G_1)V^{n+1} = (I-\gamma G_2)U^n + F_1^n, \\ (I+\gamma G_2)W^{n+1} = (I-\gamma G_1)U^n + F_1^n, \qquad n = 0,1,2,\dots, \\ U^{n+1} = \frac{1}{2}(V^{n+1} + W^{n+1}), \end{cases} \tag{17}$$

where $U^n = (u_1^n, u_2^n, \dots, u_{m-1}^n)^T$, $V^n = (v_1^n, v_2^n, \dots, v_{m-1}^n)^T$, $W^n = (w_1^n, w_2^n, \dots, w_{m-1}^n)^T$.

Algorithm 1 The new parallel algorithm for one-dimensional parabolic equations

```
Require: Initialization $U^0(x_j) \leftarrow u_0(x_j)$.
  for $n = 0,1,\dots,N$ do
    for $j = 0,1,\dots,m$ do
      Solve the values $V_j^{n+1}$ by using DDM I (12).
      Solve the values $W_j^{n+1}$ by using DDM II (16).
      The average of two values will be calculated, i.e., $U_j^{n+1} = \frac{1}{2}(V_j^{n+1} + W_j^{n+1})$.
    end for
  end for
Ensure: Output $U^N(x_j)$.
```

3 The accuracy of Algorithm 1

In this section, we illustrate the accuracy of Algorithm 1. From the Taylor expansion at (x_j, t_{n+1}), we have the following truncation errors for the Crank–Nicolson scheme (4) and Saul'yev asymmetric schemes (5)–(8), respectively.

$$\begin{aligned} T_4 &= \left(\frac{\partial u}{\partial t}\right)_j^{n+1} - a\left(\frac{\partial^2 u}{\partial t^2}\right)_j^{n+1} - \frac{\tau}{2}\left[\left(\frac{\partial^2 u}{\partial t^2}\right)_j^{n+1} - a\left(\frac{\partial^3 u}{\partial x^2 \partial t}\right)_j^{n+1}\right] + O(\tau^2 + h^2) \\ &= O(\tau^2 + h^2). \end{aligned} \tag{18}$$

$$\begin{aligned} T_5 = {}& -a\left[\frac{\tau}{h}\left(\frac{\partial^2 u}{\partial x \partial t}\right)_j^{n+1} + \frac{\tau}{2}\left(\frac{\partial^3 u}{\partial x^2 \partial t}\right)_j^{n+1} + \frac{\tau h}{6}\left(\frac{\partial^4 u}{\partial x^3 \partial t}\right)_j^{n+1} + \frac{\tau^3}{6h}\left(\frac{\partial^4 u}{\partial x \partial t^3}\right)_j^{n+1}\right] \\ & + O(\tau^2 + h^2). \end{aligned} \tag{19}$$

$$\begin{aligned} T_6 = {}& -a\left[-\frac{\tau}{h}\left(\frac{\partial^2 u}{\partial x \partial t}\right)_j^{n+1} + \frac{\tau}{2}\left(\frac{\partial^3 u}{\partial x^2 \partial t}\right)_j^{n+1} - \frac{\tau h}{6}\left(\frac{\partial^4 u}{\partial x^3 \partial t}\right)_j^{n+1} - \frac{\tau^3}{6h}\left(\frac{\partial^4 u}{\partial x \partial t^3}\right)_j^{n+1}\right] \\ & + O(\tau^2 + h^2). \end{aligned} \tag{20}$$

$$\begin{aligned} T_7 = {}& -a\left[\frac{\tau}{h}\left(\frac{\partial^2 u}{\partial x \partial t}\right)_j^{n+1} - \frac{\tau}{2}\left(\frac{\partial^3 u}{\partial x^2 \partial t}\right)_j^{n+1} + \frac{\tau h}{6}\left(\frac{\partial^4 u}{\partial x^3 \partial t}\right)_j^{n+1} + \frac{\tau^3}{6h}\left(\frac{\partial^4 u}{\partial x \partial t^3}\right)_j^{n+1}\right] \\ & + O(\tau^2 + h^2). \end{aligned} \tag{21}$$

$$\begin{aligned} T_8 = {}& -a\left[-\frac{\tau}{h}\left(\frac{\partial^2 u}{\partial x \partial t}\right)_j^{n+1} - \frac{\tau}{2}\left(\frac{\partial^3 u}{\partial x^2 \partial t}\right)_j^{n+1} - \frac{\tau h}{6}\left(\frac{\partial^4 u}{\partial x^3 \partial t}\right)_j^{n+1} - \frac{\tau^3}{6h}\left(\frac{\partial^4 u}{\partial x \partial t^3}\right)_j^{n+1}\right] \\ & + O(\tau^2 + h^2). \end{aligned} \tag{22}$$

It is obvious that the truncation error of the Crank–Nicolson scheme (4) is $O(\tau^2 + h^2)$. Compared with the truncation error T_5 of scheme (5) and the truncation error T_8 of scheme (8), the signs of leading terms of T_5 and T_8 are opposite. Similarly, compared with the truncation error T_6 of scheme (6) and the truncation error T_7 of scheme (7), the signs of leading terms of T_6 and T_7 are opposite.

It is easy to see that:

1. When DDM I use (5) to compute solution at (x_j, t_{n+1}), DDM II will use (8).
2. When DDM I use (6) to compute solution at (x_j, t_{n+1}), DDM II will use (7).
3. When DDM I use (7) to compute solution at (x_j, t_{n+1}), DDM II will use (6).
4. When DDM I use (8) to compute solution at (x_j, t_{n+1}), DDM II will use (5).

Therefore the leading terms of truncation error can be eliminated, we can get the following theorem.

Theorem 1 *The truncation error of Algorithm* 1 *is approximately* $O(\tau^2 + h^2)$.

Proof Since $(\frac{1}{2}) \times [(5) + (8)]$ is obtained as follows:

$$-\frac{ar}{2}u_{j-1}^{n+1} + (1 + ar)u_j^{n+1} - \frac{ar}{2}u_{j+1}^{n+1} = \frac{ar}{2}u_{j-1}^n + (1 - ar)u_j^n + \frac{ar}{2}u_{j+1}^n + \tau f_j^n. \tag{23}$$

Similarly, $(\frac{1}{2}) \times [(6) + (7)]$ is

$$-\frac{ar}{2}u_{j-1}^{n+1} + (1 + ar)u_j^{n+1} - \frac{ar}{2}u_{j+1}^{n+1} = \frac{ar}{2}u_{j-1}^n + (1 - ar)u_j^n + \frac{ar}{2}u_{j+1}^n + \tau f_j^n. \tag{24}$$

From algorithm (17) and the above-mentioned results, the truncation error of Algorithm 1 is approximately equal to the truncation error of the Crank–Nicolson scheme T_4. □

4 Existence and uniqueness

In order to discuss the existence and uniqueness of the solution by Algorithm 1, the following lemmas of Kellogg [37] are required.

Lemma 1 *If* $\theta > 0$ *and* $C + C^T$ *is nonnegative definite, then* $(\theta I + C)^{-1}$ *exists and*

$$\left\|(\theta I + C)^{-1}\right\|_2 \le \theta^{-1}. \tag{25}$$

Lemma 2 *Under the conditions of Lemma* 1, *there is*

$$\left\|(I + \theta C)^{-1}\right\| \le 1. \tag{26}$$

Theorem 2 *The solution of Algorithm* 1 *exists and is unique.*

Proof Assuming the solution U^n at nth time level is known, the solution U^{n+1} at at $(n+1)$st time level is solved by (17).

For DDM I

$$(I + \gamma G_1)V^{n+1} = (I - \gamma G_2)U^n + F_1^n,$$

$(I + \gamma G_1)^{-1}$ exists by Lemma 2. It is proved that DDM I has a unique solution V^{n+1}.

In the same way, for DDM II

$$(I + \gamma G_2)W^{n+1} = (I - \gamma G_1)U^n + F_1^n,$$

it is also proved that DDM II has a unique solution W^{n+1}. Then $U^{n+1} = \frac{1}{2}(V^{n+1} + W^{n+1})$ exists and is unique. □

5 Unconditional stability

In this section, we discuss unconditional stability of Algorithm 1.

Theorem 3 *Algorithm* 1 *is unconditionally stable.*

Proof Algorithm (17) can be rewritten as

$$U^{n+1} = TU^n, \tag{27}$$

where T is the growth matrix,

$$T = \frac{1}{2}\left[(I + \gamma G_1)^{-1}(I - \gamma G_2) + (I + rG_2)^{-1}(I - \gamma G_1)\right].$$

For $G_1 + G_1^T$, $G_2 + G_2^T$ are nonnegative definite matrices, by Kellogg's Lemma 1, we have

$$\left\|(I + \gamma G_i)^{-1}\right\|_2 \le 1, \quad i = 1, 2,$$

then

$$\begin{aligned}\|T\|_2 &\le \tfrac{1}{2}\left(\left\|(I+\gamma G_1)^{-1}\right\|_2\left\|(I-\gamma G_2)\right\|_2+\left\|(I+\gamma G_2)^{-1}\right\|_2\left\|(I-\gamma G_1)\right\|_2\right)\\ &\le \tfrac{1}{2}\left(\left\|(I-\gamma G_1)\right\|_2+\left\|(I-\gamma G_2)\right\|_2\right).\end{aligned}$$

It is obvious that $(I-\gamma G_1)$ and $(I-\gamma G_2)$ are not only normal matrices, but also strictly diagonally dominant matrices. By properties on the strictly diagonally dominant matrix and the normal matrix, we obtain

$$\rho(T) \le \|T\|_2 \le \frac{1}{2}\left[\rho(I-\gamma G_1)+\rho(I-\gamma G_2)\right] < 1, \tag{28}$$

where $\rho(T)$, $\rho(I-\gamma G_1)$ and $\rho(I-\gamma G_2)$ are the spectral radii of the matrices T, $(I-\gamma G_1)$ and $(I-\gamma G_2)$.

Therefore, Algorithm 1 given by (17) is unconditionally stable. □

6 Extension to two-dimensional parabolic equations

In this section, we extend **Algorithm** 1 to solve two-dimensional parabolic equations

$$\frac{\partial u}{\partial t}-\frac{\partial}{\partial x}\left(a\frac{\partial u}{\partial x}\right)-\frac{\partial}{\partial y}\left(b\frac{\partial u}{\partial y}\right)=f(x,y,t), \quad (x,y)\in\Omega, t\in(0,T], \tag{29}$$

$$u(x,y,0)=u_0(x,y), \quad (x,y)\in\Omega, \tag{30}$$

$$u(x,y,t)=0, \quad (x,y)\in\partial\Omega, t\in(0,T], \tag{31}$$

where the domain $\Omega\in(0,L_x)\times(0,L_y)$; $a>0$ and $b>0$ are diffusion coefficients.

Let $u^n_{i,j}$ be the approximate solution at (x_i,y_j,t_n), $u(x,y,t)$ represents the exact solution of (27). With the same time and space discretization of algorithm (17), we obtain its extended algorithm by alternating direction implicit (ADI) technique [38] for Eqs. (27)–(29).

x-direction:

Let $r_1=a\tau/(2h^2)$, $r_2=b\tau/(2h^2)$, the matrix form of the new parallel algorithm in x-direction can be written as follows:

$$\begin{cases}(I+r_1G_1)V_1^{n+\frac{1}{2}}=(I-r_1G_2)U^n+b_1^n,\\ (I+r_1G_2)W_1^{n+\frac{1}{2}}=(I-r_1G_1)U^n+b_1^n,\\ U^{n+\frac{1}{2}}=\frac{1}{2}(V_1^{n+\frac{1}{2}}+W_1^{n+\frac{1}{2}}),\end{cases} \tag{32}$$

where

$$b_1^n=\begin{bmatrix}[(r_2u^n_{0,j}+r_1u^{n+\frac{1}{2}}_{0,j})/2]+r_2(u^n_{1,j-1}-2u^n_{1,j}+u^n_{1,j+1})+\tau f^n_{1,j}/2\\ r_2(u^n_{2,j-1}-2u^n_{2,j}+u^n_{2,j+1})+\tau f^n_{2,j}/2\\ \vdots\\ r_2(u^n_{m-2,j-1}-2u^n_{m-2,j}+u^n_{m-2,j+1})+\tau f^n_{m-2,j}/2\\ [(r_2u^n_{m,j}+r_1u^{n+\frac{1}{2}}_{m,j})/2]+r_2(u^n_{m-1,j-1}-2u^n_{m-1,j}+u^n_{m-1,j+1})+\tau f^n_{m-1,j}/2\end{bmatrix},$$

$$\begin{cases} U^{n+\frac{1}{2}} = (u_{1,j}^{n+\frac{1}{2}}, u_{2,j}^{n+\frac{1}{2}}, \dots, u_{m-1,j}^{n+\frac{1}{2}})^T, \\ V_1^{n+\frac{1}{2}} = (v_{1,j}^{n+\frac{1}{2}}, v_{2,j}^{n+\frac{1}{2}}, \dots, v_{m-1,j}^{n+\frac{1}{2}})^T, \qquad j = 1,2,\dots,m-1. \\ W_1^{n+\frac{1}{2}} = (w_{1,j}^{n+\frac{1}{2}}, w_{2,j}^{n+\frac{1}{2}}, \dots, w_{m-1,j}^{n+\frac{1}{2}})^T, \end{cases}$$

y-direction:

The matrix form of the new parallel algorithm in y-direction can be written as follows:

$$\begin{cases} (I + r_2G_1)V_2^{n+1} = (I - r_2G_2)U^{n+\frac{1}{2}} + b_2^{n+\frac{1}{2}}, \\ (I + r_2G_2)W_2^{n+1} = (I - r_2G_1)U^{n+\frac{1}{2}} + b_2^{n+\frac{1}{2}}, \\ U^{n+1} = \frac{1}{2}(V_2^{n+1} + W_2^{n+1}), \end{cases} \tag{33}$$

where

$$b_2^{n+\frac{1}{2}} = \begin{bmatrix} [(r_2u_{i,0}^{n+\frac{1}{2}} + r_1u_{i,0}^{n+1})/2] + r_1(u_{i-1,1}^{n+1} - 2u_{i,1}^{n+1} + u_{i+1,1}^{n+1}) + \tau f_{i,1}^{n+\frac{1}{2}}/2 \\ r_1(u_{i-1,2}^{n+1} - 2u_{i,2}^{n+1} + u_{i+1,2}^{n+1}) + \tau f_{i,2}^{n+\frac{1}{2}}/2 \\ \vdots \\ r_1(u_{i-1,m-2}^{n+1} - 2u_{i,m-2}^{n+1} + u_{i+1,m-2}^{n+1}) + \tau f_{i,m-2}^{n+\frac{1}{2}}/2 \\ [(r_2u_{i,m}^{n+\frac{1}{2}} + r_1u_{i,m}^{n+1})/2] + r_1(u_{i-1,m-1}^{n+1} - 2u_{i,m-1}^{n+1} + u_{i+1,m-1}^{n+1}) + \tau f_{i,m-1}^{n+\frac{1}{2}}/2 \end{bmatrix},$$

$$\begin{cases} U^{n+1} = (u_{i,1}^{n+1}, u_{i,2}^{n+1}, \dots, u_{i,m-1}^{n+1})^T, \\ V_2^{n+1} = (v_{i,1}^{n+1}, v_{i,2}^{n+1}, \dots, v_{i,m-1}^{n+1})^T, \qquad i = 1,2,\dots,m-1. \\ W_2^{n+1} = (w_{i,1}^{n+1}, w_{i,2}^{n+1}, \dots, w_{i,m-1}^{n+1})^T, \end{cases}$$

The corresponding algorithm can be described as follows in Algorithm 2.

Similar to Algorithm 1, it is obvious that Algorithm 2 has unconditional stability and parallelism.

Remark 1 In Algorithm 2, the domain is divided into many subdomains by using two DDMs. In each time interval, we first solve the values along x-direction by (32) at half-time step and then solve the values along y-direction by (33) at next half-time step. Schemes (32) and (33) lead to block diagonal algebraic systems that can be solved independently. So Algorithm 2 not only suits for parallel computation, but also improves the accuracy.

Algorithm 2 The new parallel algorithm for two-dimensional parabolic equations

```
Require: Initialization $U^0(x_i, y_j) \leftarrow u_0(x_i, y_j)$.
  for $n = 0, 1, \dots, N$ do
    for $i = 0, 1, \dots, m$ do
      for $j = 0, 1, \dots, m$ do
        Solve the values $U_{i,j}^{n+\frac{1}{2}}$ by using scheme (32).
        Solve the values $U_{i,j}^{n+1}$ again by using scheme (33).
      end for
    end for
  end for
Ensure: Output $U^N(x_i, y_j)$.
```

Keeping the advantage of ADI technique, Algorithm 2 reduces computational complexities. Though it is developed for two-dimensional problems, Algorithm 2 can be easily extended to solve high-dimensional parabolic equations.

7 Numerical experiments

To illustrate the accuracy and stability of the new parallel Algorithm 1 and Algorithm 2 for parabolic equations, we present two numerical experiments to verify the accuracy, convergence order in space, stability, and parallel efficiency. In addition, we will compare the accuracy of the new algorithm with the existing method.

Example 1

$$\begin{cases} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} = (1+\pi^2)e^t \sin(\pi x), & x \in (0,1), t \in (0,1], \\ u(x,0) = \sin(\pi x), & x \in [0,1], \\ u(0,t) = u(1,t) = 0, & t \in (0,1]. \end{cases} \tag{34}$$

The exact solution of Example 1 is

$$u(x,t) = e^t \sin(\pi x). \tag{35}$$

Firstly, we examine the convergence rate of Algorithm 1. We divide the mesh point into many segments such as $K = 3$, $K = 4$, $K = 5$, and $K = 6$. Let U_j^n be the exact solution of Example 1, we calculate errors $L^\infty = \|U - u\|$ in maximum-norm taking $\tau = 0.001$. The rate of convergence in space is as follows:

$$\text{Rate} \approx \frac{\log(L_{h1}^\infty / L_{h2}^\infty)}{\log(h1/h2)}.$$

Clearly the errors appear to be of order $O(h^2)$ in Table 1.

Next, we present the error results of Algorithm 1 in terms of the absolute errors and the relative errors, where the absolute error (A. E.) is defined by

$$e_j^n = \left| u_j^n - u(x_j, t_n) \right|,$$

and the relative error (R. E.) is defined by

$$E_j^n = \frac{e_j^n}{|u(x_j, t_n)|} \times 100\%.$$

Tables 2 and 3 display the absolute errors and the relative errors obtained by the presented Algorithm 1 for $h = 1/19$ (i.e., $m-1 = 18 = 6K, K = 3$), $h = 1/25$ (i.e., $m-1 = 24 = 6K$,

Table 1 Convergence rate of Algorithm 1 for h at $t = 0.4$

h	L^∞ error	Rate
1/19 ($m-1 = 18 = 6K, K = 3$)	4.998746e–4	–
1/25 ($m-1 = 24 = 6K, K = 4$)	3.085232e–4	2.0167
1/31 ($m-1 = 30 = 6K, K = 5$)	2.002844e–4	1.9955
1/37 ($m-1 = 36 = 6K, K = 6$)	1.412289e–4	1.9903

Table 2 The absolute errors and relative errors of numerical solutions to Example 1 for $h = 1/19$ (i.e., $m - 1 = 18 = 6K$, $K = 3$)

x_j	Algorithm 1 ($t = 0.2$)		Algorithm 1 ($t = 0.4$)		Algorithm 1 ($t = 0.8$)	
	A. E.	R. E.	A. E.	R. E.	A. E.	R. E.
0.11	0.3731e–3	1.7559e–2	0.5021e–3	1.9312e–2	0.7577e–3	1.9534e–2
0.21	0.6184e–3	1.5420e–2	0.8428e–3	1.7176e–2	1.2739e–3	1.7399e–2
0.31	0.8750e–3	1.5999e–2	1.1881e–3	1.7754e–2	1.7950e–3	1.7976e–2
0.42	0.9753e–3	1.5409e–2	1.3294e–3	1.7165e–2	2.0093e–3	1.7388e–2
0.52	1.0066e–3	1.5469e–2	1.3715e–3	1.7225e–2	2.0729e–3	1.7447e–2
0.63	0.9135e–3	1.5280e–2	1.2463e–3	1.7037e–2	1.8839e–3	1.7259e–2
0.73	0.7582e–3	1.5778e–2	1.0310e–3	1.7534e–2	1.5579e–3	1.7757e–2
0.84	0.5469e–3	1.7563e–2	0.7361e–3	1.9316e–2	1.1109e–3	1.9538e–2
0.95	0.1891e–3	1.7556e–2	0.2545e–3	1.9309e–2	0.3841e–3	1.9532e–2

Table 3 The absolute errors and relative errors of numerical solutions to Example 1 for $h = 1/25$ (i.e., $m - 1 = 24 = 6K$, $K = 4$)

x_j	Algorithm 1 ($t = 0.2$)		Algorithm 1 ($t = 0.4$)		Algorithm 1 ($t = 0.8$)	
	A. E.	R. E.	A. E.	R. E.	A. E.	R. E.
0.04	0.5579e–4	9.0292e–3	0.7499e–4	9.9277e–3	1.1317e–4	1.0041e–2
0.12	1.6391e–4	9.0313e–3	2.2031e–4	9.9298e–3	3.3247e–4	1.0043e–2
0.24	2.7890e–4	8.2703e–3	3.7803e–4	9.1694e–3	5.7102e–4	0.9283e–2
0.36	3.5674e–4	8.0054e–3	4.8512e–4	8.9048e–3	7.3305e–4	0.9018e–2
0.48	3.8737e–4	7.8818e–3	5.2760e–4	8.7813e–3	7.9739e–4	0.8895e–2
0.60	3.6642e–4	7.8242e–3	4.9945e–4	8.7237e–3	7.5491e–4	0.8837e–2
0.72	2.9685e–4	7.8239e–3	4.0462e–4	8.7235e–3	6.1158e–4	0.8837e–2
0.84	1.8851e–4	7.9457e–3	2.5655e–4	8.8452e–3	3.8769e–4	0.8959e–2
0.96	0.5579e–4	9.0292e–3	0.7499e–4	9.9277e–3	1.1317e–4	1.0041e–2

Table 4 Comparisons by the maximum errors to Example 1 for $h = 1/19$ (i.e., $m - 1 = 18 = 6K$, $K = 3$)

Methods	r	$t = 0.2$	$t = 0.4$	$t = 0.6$	$t = 0.8$
Algorithm 1	0.36	3.5539e–4	4.8387e–4	5.9795e–4	7.3131e–4
ASC-N scheme [6]		4.7468e–4	7.1743e–4	1.0705e–3	1.5969e–3
Algorithm 1	0.72	5.7866e–4	7.8813e–4	9.7397e–4	1.1912e–3
ASC-N scheme [6]		7.3439e–4	1.1100e–3	1.6563e–3	2.4709e–3
Algorithm 1	1.08	7.9673e–4	1.0815e–3	1.3381e–3	1.6382e–3
ASC-N scheme [6]		1.1764e–3	1.7767e–3	2.1187e–3	3.9468e–3
Algorithm 1	1.45	1.0066e–3	1.3715e–3	1.6949e–3	2.0729e–3
ASC-N scheme [6]		1.0999e–3	1.6591e–3	2.4754e–3	3.6929e–3
Algorithm 1	1.80	1.2153e–3	1.6560e–3	2.0465e–3	2.5029e–3
ASC-N scheme [6]		1.2843e–3	1.9356e–3	2.8880e–3	4.3084e–3

$K = 4$) at $t = 0.2$, $t = 0.4$, and $t = 0.8$, when taking $r = 1.5$ ($r = \tau/h^2$). From Tables 2 and 3, it is obvious that our algorithm has high accuracy.

Now, we compare Algorithm 1 with the ASC-N scheme in [6] by the maximum errors for $h = 1/19$ (i.e., $m - 1 = 18 = 6K$, $K = 3$), $h = 1/25$ (i.e., $m - 1 = 24 = 6K$, $K = 4$), and $h = 1/31$ (i.e., $m - 1 = 30 = 6K$, $K = 5$) at different time $t = 0.2$, $t = 0.4$, $t = 0.6$, $t = 0.7$, and $t = 0.8$. With the increase in computation time, the errors of the ASC-N scheme in [6] increase more than those of Algorithm 1 for different r ($r = \tau/h^2$) in Tables 4 and 5. We can see that Algorithm 1 has higher accuracy. We consider an example for $h = 1/121$ ($m - 1 = 120 = 6K$, $K = 20$) with large grid ratio $r = 15$ ($r = \tau/h^2$). Table 6 shows that Algorithm 1 has better accuracy than two others. It is indicated that Algorithm 1 is stable, which is consistent with the theoretical results obtained in Sect. 5.

Table 5 Comparisons by the maximum errors to Example 1 for $h = 1/25$ (i.e., $m - 1 = 24 = 6K$, $K = 4$)

Methods	r	$t = 0.2$	$t = 0.4$	$t = 0.6$	$t = 0.8$
Algorithm 1	0.38	1.5929e–4	2.1698e–4	2.6806e–4	3.2777e–4
ASC-N scheme [6]		2.4785e–4	3.7370e–4	4.7785e–4	8.3215e–4
Algorithm 1	0.63	2.2660e–4	3.0852e–4	3.8124e–4	4.6627e–4
ASC-N scheme [6]		2.8997e–4	4.3826e–4	6.5392e–4	9.7553e–4
Algorithm 1	0.94	3.0760e–4	4.1963e–4	5.1826e–4	6.3353e–4
ASC-N scheme [6]		4.5949e–4	6.9190e–4	8.3264e–4	1.5416e–3
Algorithm 1	1.35	3.8737e–4	5.2761e–4	6.5199e–4	7.9740e–4
ASC-N scheme [6]		4.2810e–4	6.4710e–4	9.6552e–4	1.4404e–3
Algorithm 1	1.76	4.6541e–4	6.3397e–4	7.8343e–4	9.5815e–4
ASC-N scheme [6]		4.6674e–4	7.0553e–4	1.0527e–3	1.5704e–3

Table 6 Maximum error comparison for $r = 15$, $h = 1/121$ ($m - 1 = 120 = 6K$, $K = 20$)

	Algorithm 1	ASC-N method [6]	AGE method [3]
$t = 0.2$	1.3994e–4	1.4759e–4	1.2717e–3
$t = 0.4$	3.2238e–4	2.8034e–4	3.2242e–3
$t = 0.6$	5.0639e–4	6.5004e–4	6.2125e–3
$t = 0.8$	6.9237e–4	8.3824e–4	9.2284e–3

Example 2

$$\begin{cases} \frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 u}{\partial y^2} = f(x,y,t), & (x,y) \in \Omega, t \in (0,1], \\ u(x,y,0) = \sin(\pi x)\sin(\pi y), & (x,y) \in \Omega, \\ u(0,y,t) = u(1,y,t) = 0, & y \in [0,1], t \in (0,1], \\ u(x,0,t) = u(x,1,t) = 0, & x \in [0,1], t \in (0,1], \end{cases} \tag{36}$$

where the domain is $\Omega = (0,1) \times (0,1)$ and the right-hand side function is

$$f(x,y,t) = (1 + 2\pi^2)e^t \sin(\pi x)\sin(\pi y). \tag{37}$$

The corresponding exact solution is

$$u(x,y,t) = e^t \sin(\pi x)\sin(\pi y). \tag{38}$$

We take $h = 1/19$ (i.e., $m - 1 = 18 = 6K$, $K = 3$), $h = 1/25$ (i.e., $m - 1 = 24 = 6K$, $K = 4$), $h = 1/31$ (i.e., $m - 1 = 30 = 6K$, $K = 5$), and $h = 1/37$ (i.e., $m - 1 = 36 = 6K$, $K = 6$), respectively. Table 7 displays the maximum errors of Algorithm 2 for $\tau = 0.0004$ at different time. It is obvious that the accuracy of Algorithm 2 is good. The comparison among Algorithm 2, classical C-N scheme, and ASC-N scheme is shown CPU calculation time for $r = 1.2$ at $t = 0.5$ in Table 8. Obviously, Algorithm 2 not only has high accuracy, but also has good parallel efficiency. Take all the programs of Algorithm 2 running on the uniform mesh, three different grids are given in Fig. 2, Fig. 3, and Fig. 4 that are defined by $h = 1/19$, $h = 1/25$, and $h = 1/37$. Figure 1 shows the exact solutions at $t = 0.4$. Comparison of the exact solutions and the numerical solutions are shown in Figs. 1–4.

Table 7 The maximum errors of Algorithm 2 to Example 2 for $\tau = 0.0004$

	$h = 1/19$	$h = 1/25$	$h = 1/31$	$h = 1/37$
$t = 0.2$	6.0275e–4	1.5451e–4	8.0677e–5	4.9096e–5
$t = 0.4$	4.4976e–4	1.9272e–4	1.0052e–4	6.1127e–5
$t = 0.5$	5.1593e–4	2.1305e–4	1.1113e–4	6.7574e–5
$t = 0.6$	5.7284e–4	2.3546e–4	1.2282e–4	7.4684e–5
$t = 0.8$	7.0014e–4	2.8760e–4	1.5001e–4	9.1219e–5

Table 8 Comparison of three schemes calculation time for $r = 1.2$ at $t = 0.5$

h	Algorithm 2	C-N scheme	ASC-N scheme
1/37 ($m - 1 = 36 = 6K, K = 6$)	16.4121 s	20.2625 s	16.7089 s
1/49 ($m - 1 = 48 = 6K, K = 8$)	41.4226 s	49.9863 s	41.0513 s
1/61 ($m - 1 = 60 = 6K, K = 10$)	148.2623 s	175.2632 s	146.0665 s
1/91 ($m - 1 = 90 = 6K, K = 15$)	981.5628 s	1167.2047 s	966.9273 s

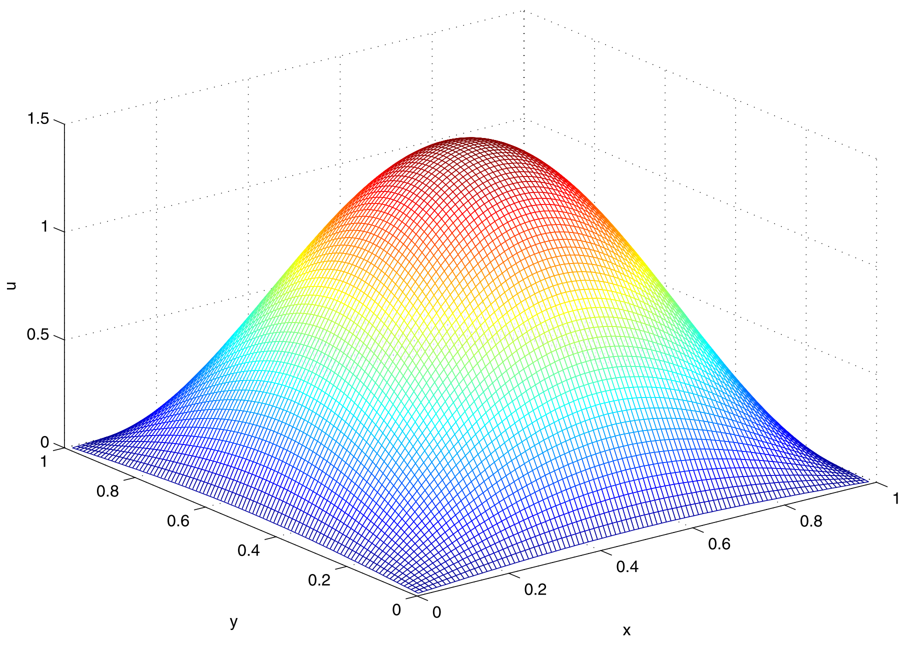

Figure 1 The exact solutions at $t = 0.4$

Based on the experiments above, Algorithm 1 and Algorithm 2 presented in this paper are suitable and efficient for solving parabolic equations.

8 Conclusion

We have proposed and analyzed a new parallel algorithm for parabolic equations. This algorithm consists of two DDMs, each one is used to solve the values in the same time level, respectively. Then the average of two values is calculated. Stability and error analysis show that the new algorithm is unconditionally stable and has the truncation error of order two in both space and time. The new algorithm allows possibly efficient and accurate computation on massive parallel computers in general. Then we extend the new algorithm to two-dimensional parabolic equations by the ADI technique, which means that high-

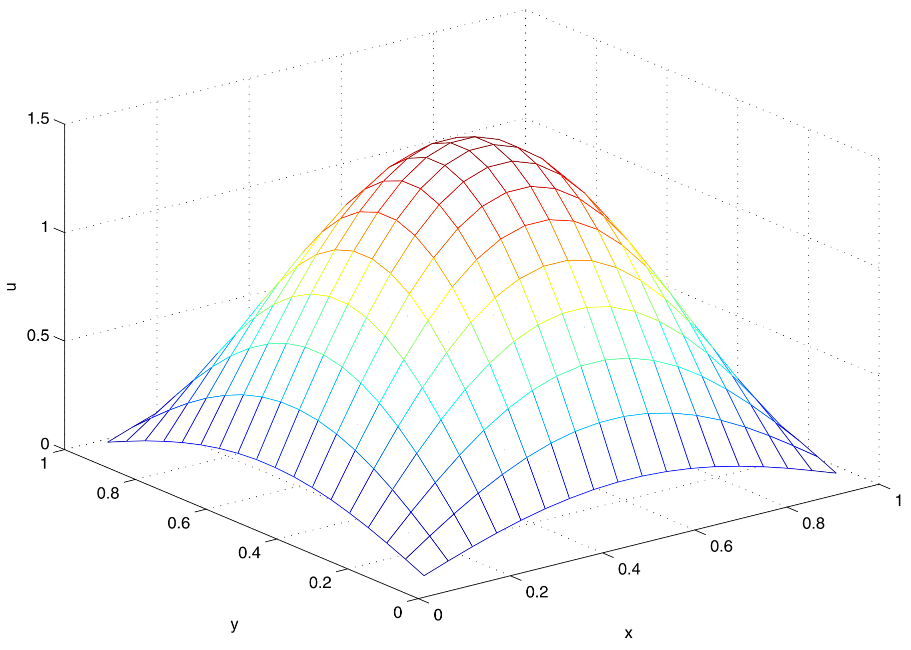

Figure 2 The numerical solutions for $r = 1.0$, $h = 1/19$ at $t = 0.4$

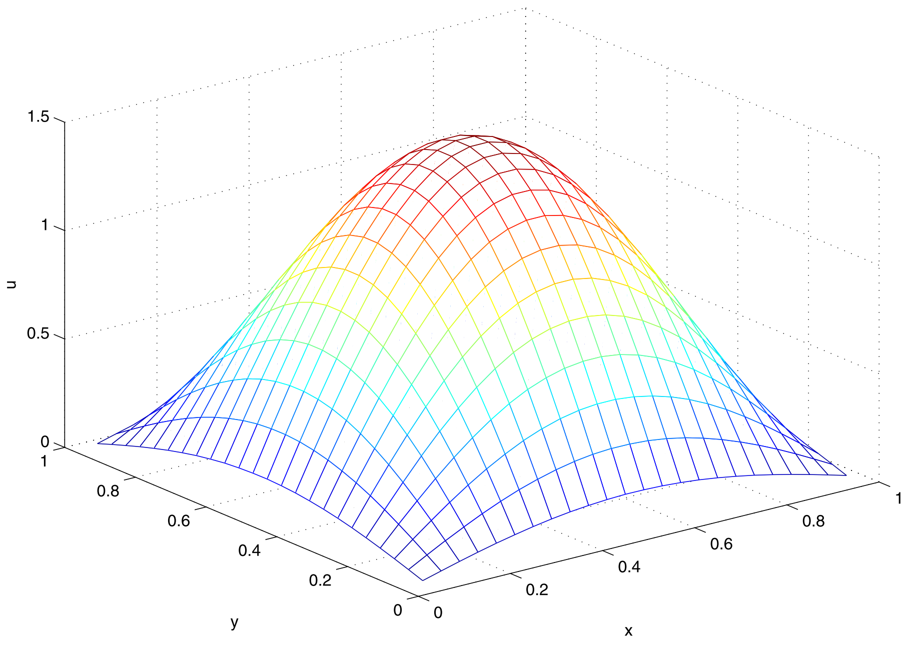

Figure 3 The numerical solutions for $r = 1.0$, $h = 1/25$ at $t = 0.4$

dimensional parabolic equations can be solved by the proposed algorithm in this paper. Numerical experiments illustrate the good performance of the new algorithm.

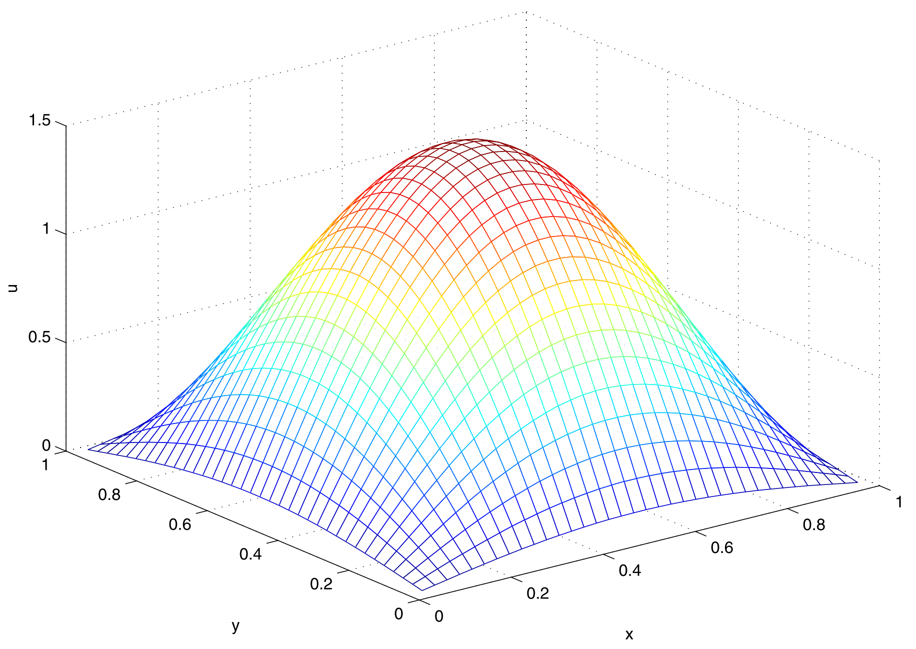

Figure 4 The numerical solutions for $r = 1.0$, $h = 1/37$ at $t = 0.4$

Acknowledgements
The authors would like to thank the anonymous referees for helpful comments and suggestions. This research is supported by the National Natural Science Foundation of China (91130022, 10971159) and the Doctoral Fund of Ministry of Education of China (20130141110026).

Competing interests
The authors declare that they have no competing interests.

Authors' contributions
All authors contributed equally and significantly in writing this article. All authors read and approved the final manuscript.

Publisher's Note

Received: 14 January 2018 Accepted: 25 April 2018 Published online: 09 May 2018

References
1. Evans, D.J., Abdullah, A.R.B.: Group explicit method for parabolic equations. Int. J. Comput. Math. **14**, 73–105 (1983)
2. Saul'yev, V.K.: Integration of Equations of Parabolic Type by Method of Nets, New York (1964)
3. Evans, D.J.: Alternating group explicit method for the diffusion equation. Appl. Math. Model. **9**, 201–206 (1985)
4. Zhang, B.: Alternating segment explicit–implicit method for the diffusion equation. Chin. J. Numer. Methods Comput. Appl. **41**, 245–251 (1991)
5. Chen, J., Zhang, B.: A class of alternating block Crank–Nicolson method. Int. J. Comput. Math. **45**, 89–112 (1991)
6. Zhang, B., Li, W.: On alternating segment Crank–Nicolson scheme. Parallel Comput. **20**, 897–902 (1994)
7. Cao, J.Y., Zhang, D.K.: A new parallel algorithm for the parabolic equation. Guizhou Sci. **25**(2), 27–33 (2007)
8. Evans, D.J., Abdullah, A.R.B.: A new explicit method for the diffusion–convection equation. Comput. Math. Appl. **11**, 145–154 (1985)
9. Lu, J., Zhang, B., Xu, T.: Alternating segment explicit–implicit method for the convection–diffusion equation. Chin. J. Numer. Methods Comput. Appl. **3**, 161–167 (1998)
10. Wang, W.: A class of alternating segment Crank–Nicolson methods for solving convection–diffusion equations. Computing **73**, 41–55 (2004)
11. Zhu, S., Yuan, G., Shen, L.: Alternating group explicit method for the dispersive equation. Int. J. Comput. Math. **75**, 97–105 (2000)
12. Zhu, S., Zhao, J.: The alternating segment explicit–implicit scheme for the dispersive equation. Appl. Math. Lett. **14**, 657–662 (2001)
13. Wang, W., Fu, S.: An unconditionally stable alternating segment difference scheme of eight points for the dispersive equation. Int. J. Numer. Methods Eng. **67**, 435–447 (2006)
14. Zhang, Q., Wang, W.: A four-order alternating segment Crank–Nicolson scheme for the dispersive equation. Comput. Math. Appl. **57**, 283–289 (2009)

15. Wang, W.Q., Zhang, Q.: A highly accurate alternating 6-point group method for the dispersive equation. Int. J. Comput. Math. **87**(7), 1512–1521 (2010)
16. Guo, G., Lü, S., Liu, B.: Unconditional stability of alternating difference schemes with variable time step lengths for dispersive equation. Appl. Math. Comput. **262**, 249–259 (2015)
17. Guo, G., Liu, B.: Unconditional stability of alternating difference schemes with intrinsic parallelism for the fourth-order parabolic equation. Appl. Math. Comput. **219**, 7319–7328 (2013)
18. Guo, G., Zhai, Y., Liu, B.: The alternating segment explicit–implicit scheme for the fourth-order parabolic equation. J. Inf. Comput. Sci. **10**, 2981–2991 (2013)
19. Guo, G., Lü, S.: Unconditional stability of alternating difference schemes with intrinsic parallelism for two-dimensional fourth-order diffusion equation. Comput. Math. Appl. **71**, 1944–1959 (2016)
20. Du, Q., Mu, M., Wu, Z.: Efficient parallel algorithms for parabolic problems. SIAM J. Numer. Anal. **39**(5), 1469–1487 (2001)
21. Zhuang, Y., Sun, X.: Stabilized explicit–implicit domain decomposition methods for the numerical solution of parabolic equations. SIAM J. Sci. Comput. **24**, 335–358 (2002)
22. Shi, H., Liao, H.: Unconditional stability of corrected explicit–implicit domain decomposition algorithms for parallel approximation of heat equations. SIAM J. Numer. Anal. **44**, 1584–1611 (2006)
23. Boglaev, I.: Domain decomposition for a parabolic convection–diffusion problem. Numer. Methods Partial Differ. Equ. **22**, 1361–1378 (2006)
24. Dryja, M., Tu, X.: A domain decomposition discretization of parabolic problems. Numer. Math. **107**, 625–640 (2007)
25. Li, C., Yuan, Y.: A modified upwind difference domain decomposition method for convection–diffusion equations. Appl. Numer. Math. **59**, 1584–1598 (2009)
26. Liang, D., Du, C.: The efficient S-DDM scheme and its analysis for solving parabolic equations. J. Comput. Phys. **272**, 46–69 (2014)
27. Arrarás, A., Gaspar, F.J., Portero, L., Rodrigo, C.: Domain decomposition multigrid methods for nonlinear reaction–diffusion problems. Commun. Nonlinear Sci. Numer. Simul. **20**, 699–710 (2015)
28. Kumar, S., Kumar, M.: An analysis of overlapping domain decomposition methods for singularly perturbed reaction–diffusion problems. J. Comput. Appl. Math. **281**, 250–262 (2015)
29. Zhou, Z., Liang, D.: A time second-order mass-conserved implicit–explicit domain decomposition scheme for solving the diffusion equations. Adv. Appl. Math. Mech. **9**(4), 795–817 (2017)
30. Zhou, Y.: Finite difference method with intrinsic parallelism for quasilinear parabolic systems. Beijing Math. **2**, 1–19 (1996)
31. Zhou, Y.: General finite difference schemes with intrinsic parallelism for nonlinear parabolic system. Beijing Math. **2**, 20–36 (1996)
32. Zhou, Y., Shen, L., Yuan, G.: Some practical difference schemes with intrinsic parallelism for nonlinear parabolic systems. Chin. J. Numer. Methods Comput. Appl. **19**, 46–57 (1997)
33. Yuan, G., Shen, L., Zhou, Y.: Unconditional stability of alternating difference schemes with intrinsic parallelism for two-dimensional parabolic systems. Numer. Methods Partial Differ. Equ. **15**, 625–636 (1999)
34. Sheng, Z., Yuan, G., Hang, X.: Unconditional stability of parallel difference schemes with second order accuracy for parabolic equation. Appl. Math. Comput. **184**(2), 1015–1031 (2007)
35. Yuan, G., Sheng, Z., Hang, X.: The unconditional stability of parallel difference schemes with scheme order convergence for nonlinear parabolic system. Numer. Methods Partial Differ. Equ. **20**(1), 45–64 (2007)
36. Zhuang, Y.: An alternating explicit–implicit domain decomposition method for the parallel solution of parabolic equations. J. Comput. Appl. Math. **206**, 549–566 (2007)
37. Kellogg, R.B.: An alternating direction method for operator equations. J. Soc. Ind. Appl. Math. **12**, 848–854 (1964)
38. Zeng, F., Zhang, Z., Karniadakis, G.E.: Fast difference schemes for solving high-dimensional time-fractional subdiffusion equations. J. Comput. Phys. **307**, 15–33 (2016)