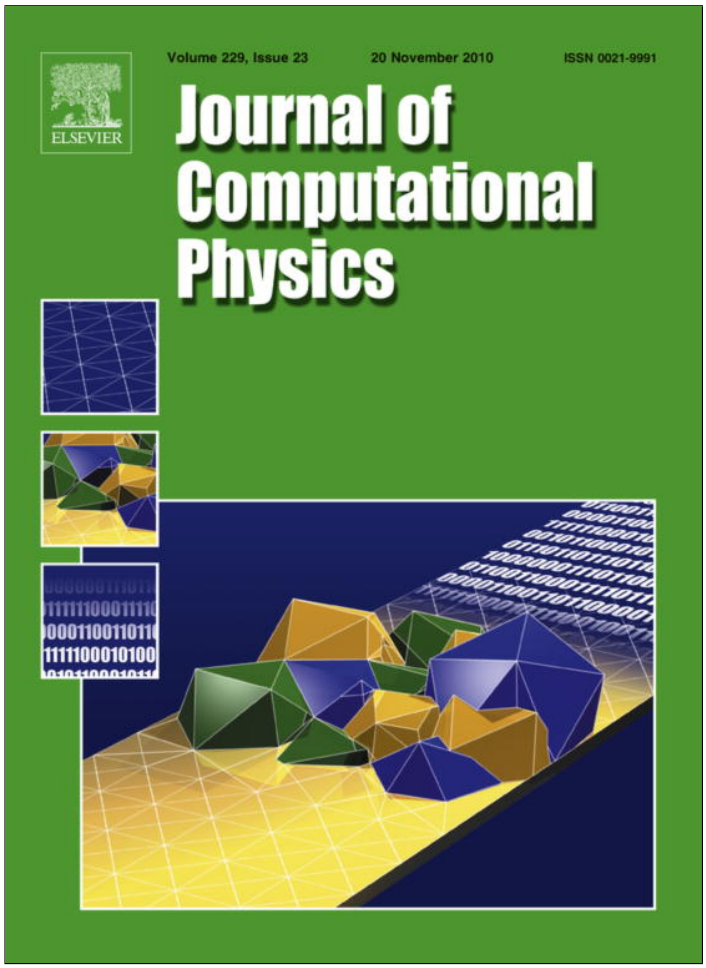


Journal of Computational Physics 229 (2010) 8935–8951

Contents lists available at ScienceDirect

Journal of Computational Physics

journal homepage: www.elsevier.com/locate/jcp

# A cell functional minimization scheme for parabolic problem ☆

Li Yin, Jiming Wu *, Yanzhong Yao

*Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, P.O. Box 8009, Beijing 100088, PR China*

ARTICLE INFO

*Article history:*
Received 22 December 2009
Received in revised form 12 July 2010
Accepted 16 August 2010
Available online 31 August 2010

*Keywords:*
Cell functional minimization
Finite volume scheme
Parabolic problem
Domain decomposition method
Distorted mesh

ABSTRACT

We are interested in a robust and accurate finite volume scheme for 2-D parabolic problems derived from the cell functional minimization approach. The scheme has a local stencil, is locally conservative, treats discontinuity rigorously and leads to a symmetric positive definite linear system. Since the scheme has both cell centered unknowns and cell edge unknowns, the computational cost is an issue and a parallel algorithm is then suggested based on nonoverlapping domain decomposition approach. The interface condition is of the Dirichlet–Robin type and has a parameter $\lambda$. By choosing this parameter properly, the convergence of the iteration process could be sped up. Numerical results for linear and nonlinear problems demonstrate the good performance of the cell functional minimization scheme and its parallel version on distorted meshes.



## 1. Introduction

Accurate and efficient discretization methods for parabolic or diffusion problems on distorted meshes are very important for many applications, such as reservoir simulation, Lagrangian hydrodynamics and magnetohydrodynamics. Numerous efforts have been devoted to this subject and lots of schemes have been suggested, such as the local support operator scheme (LSOM) [28] and its modern version namely the mimetic finite difference scheme (MFD) [20,24,5,9,10,25], the multi-point flux approximation scheme (MPFA) [1,18,2,21,15,19], the nine-point scheme [22,34,32,4,35] and some others [16,7]. A desirable scheme for diffusion or parabolic problems on distorted meshes is usually required to have as many as possible the good numerical properties that range from the classical stability and accuracy to some other ones, including local stencil, local conservation, positivity preserving or monotonicity, simplicity, robustness, cost-efficiency, symmetry and positive definiteness of the resulting linear system, cell centered type, etc. To our knowledge, there exists no scheme satisfying all the above properties. Usually, a scheme possesses some properties at the cost of losing other ones.

In constructing an efficient scheme for parabolic or diffusion problems, one usually faces some commonly known difficulties, including the distortion of the meshes, the discontinuity (sometimes anisotropy) of the diffusion coefficient and so on. As physical modeling improves, the need for more sophisticated numerical methods is increasing, which may cause some more difficulties, such as a great number of mesh cells, complex geometries and complicated material distribution. Compared with the pure cell centered schemes, the schemes that employ both cell centered unknowns and cell edge unknowns have two times of degrees of freedom more than that of the former on a structured quadrilateral mesh in two dimensions. This problem becomes even serious when the schemes are extended to three dimensions. In this case, the cell centered schemes seem competitive, particularly for parabolic problems. Therefore, it is interesting to study the problem of

☆ This work is supported by the National Natural Science Foundation of China (No.10701014, 10871030, 10871029) and a grant from the Laboratory of Computational Physics.

* Corresponding author.
*E-mail addresses:* yinli_kitty122@sina.com (L. Yin), wu_jiming@iapcm.ac.cn (J. Wu), yao_yanzhong@iapcm.ac.cn (Y. Yao).

0021-9991/$ - see front matter © 2010 Elsevier Inc. All rights reserved.
doi:10.1016/j.jcp.2010.08.018

removing the cell edge unknowns from the schemes that possess both cell centered unknowns and cell edge unknowns, which is not easy and may cause some other problems, say, the loss of accuracy or symmetry on certain types of meshes (cf. [8,25]). In the case where complex geometries and complicated material distribution present, it is natural to divide the computational domain into nonoverlapping blocks according to the geometry, material discontinuity and the feature of the solution. The nonoverlapping subdomains are then partitioned into multi-block matching or non-matching meshes, which leads to the so-called nonoverlapping domain decomposition algorithms. The interface algorithm is the key factor of this approach.

This paper aims to derive a robust and efficient 2-D finite volume scheme for problems with complex geometries and complicated material distribution. This scheme can be viewed as an updated version of the scheme originally suggested in [23] by minimizing certain functional defined on the whole computational domain. Due to the constraints of the functional and the choice of the unknowns, the scheme in [23] can only solve problems with Neumann or flux boundary conditions. Besides, it usually loses accuracy on distorted meshes. By contrast, the scheme in this paper is obtained by minimizing certain functional defined on a single cell and, by employing both cell centered unknowns and cell edge unknowns. The idea of introducing cell edge unknowns is certainly not new and is motivated by other peoples' works, such as those for MFD and MPFA. Besides, by a similar technique suggested in [10], the cell functional minimization approach can also generate a family of schemes, whose resulting linear systems are symmetric positive definite and as a result the saddle-point problem suffered by MFD scheme in [10] can be avoided. In general the present scheme has a local stencil, is locally conservative, leads to a symmetric positive definite linear system, treats discontinuity rigorously and moreover, offers approximately a second order accuracy on highly distorted meshes.

The involving of the cell edge unknowns seems an obvious disadvantage of the scheme derived from the cell functional minimization procedure. One approach to solve this problem is to remove the cell edge unknowns from the scheme. As mentioned above, this approach is not so easy and may cause some new problems. Here we must point out once again that we are going to develop an efficient scheme for problems with complex geometries and complicated material distribution, where certain kind of domain decomposition and parallel algorithm are necessary. In this case, we find that the cell edge unknowns become quite useful since they facilitate certain nonoverlapping domain decomposition approach, which in turn leads to an efficient parallel algorithm for our problem. The problem of computational cost caused by the introduction of cell edge unknowns now becomes less serious.

The interface conditions in our nonoverlapping domain decomposition algorithm is of Dirichlet–Robin type with a parameter $\lambda$. The idea of using interface conditions of Robin type in domain decomposition methods was originally suggested in [26], which facilitates the use of nonoverlapping subdomains and has the possibility for a speed up in the convergence by properly selecting the parameter $\lambda$. For this issue, we mention the works in [3,12,31,27] and the works cited therein. For the ease of presentation and clearness of the main idea, we just consider the case where matching multi-block structured quadrilateral meshes are involved. The work for non-matching multi-block meshes is ongoing at the present. In the matching case, the cell edge unknowns in the cell functional minimization scheme make it convenient to handle all kinds of interface conditions. In the non-matching case, some mortar technique similar to that in [6] is a possible choice.

The rest of this paper is organized as follows. In the next section, we describe the formulation of problem. In Section 3, we derive a finite volume scheme based on minimizing a cell functional and suggest also its parallel algorithm based on nonoverlapping domain decomposition. Section 4 presents some numerical experiments for the finite volume scheme and its parallel version. Finally, we give some concluding remarks in the last section.

## 2. Formulation of the problem

The parabolic problem that we seek to solve an be expressed in the general form

$$\frac{\partial u}{\partial t} - \operatorname{div}(\kappa(u)\nabla u) = S(x,y,t), \quad \text{in } \Omega, \tag{2.1}$$

with initial condition

$$u(x,y,0) = \psi(x,y), \tag{2.2}$$

where $u = u(x,y,t)$ denotes the scalar unknown function or the intensity, $\Omega$ is a bounded domain in $\mathbb{R}^2$, $\kappa(u)$ is a positive scalar diffusion coefficient dependent on $u$ and uniformly bounded above and below in $\Omega$, and $S$ denotes a source or driving function. The boundary condition for Eq. (2.1) could be in the following uniform form,

$$\alpha\kappa(u)\frac{\partial u}{\partial \vec{n}} + \beta u = g, \quad \text{on} \quad \partial\Omega, \tag{2.3}$$

where $\vec{n}$ denotes the unit outward normal along the domain boundary $\partial\Omega$, and $\alpha$, $\beta$ are two parameters. If $\alpha = 0$ and $\beta \neq 0$, (2.3) leads to the Dirichlet boundary condition, and if $\alpha \neq 0$ and $\beta = 0$, (2.3) gives the Neumann or flux boundary condition. Furthermore, if $\alpha = 1$ and $\beta > 0$, one obtains the boundary condition of the Robin type.

In our method, it is useful to introduce a vector function, known as the flux and defined by

$$\vec{F} = -\kappa(u)\nabla u. \tag{2.4}$$

Flux $\vec{F}$ has certain physical meanings, for example, it stands for the Darcy velocity in reservoir simulation. By this intermediate variable, (2.1) can be rewritten as the following systems,

$$\frac{\partial u}{\partial t} + \operatorname{div}\vec{F} = S, \tag{2.5}$$

$$\vec{F} = -\kappa(u)\nabla u. \tag{2.6}$$

Integrating Eq. (2.5) over a mesh cell $e$, we obtain

$$\int_e \frac{\partial u}{\partial t} de + \oint_{\partial e} \vec{F}\cdot\vec{n}\, ds = \int_e S\, de, \tag{2.7}$$

where $\vec{n}$ is the unit vector normal to the cell boundary $\partial e$. In physical terms, (2.7) represents the energy or particle conservation over the cell. The main part of a finite volume discretization is now down to find certain approximation of the contour integration in (2.7).

## 3. A finite volume scheme based on minimizing a cell functional

There are several numerical methods arising from the solution of the first order system (2.5) and (2.6) through functional approach. In the mixed finite element method [29], the solution for the homogeneous Dirichlet linear steady state problem associated with (2.5) and (2.6) is the unique saddle point of the Lagrangian $\mathscr{L}(\cdot,\cdot)$ defined on $H(\mathrm{div};\Omega)\times L^2(\Omega)$ by

$$\mathscr{L}(\vec{G}, v) = \frac{1}{2}\int_\Omega \frac{|\vec{G}|^2}{\kappa_c} d\Omega + \int_\Omega (S - \operatorname{div}\vec{G})\ v\ d\Omega,$$

where $\kappa_c$ denotes the constant diffusion coefficient, i.e., the solution $(\vec{F}_c, u) \in H(\mathrm{div};\Omega)\times L^2(\Omega)$ with $\vec{F}_c = -\kappa_c\nabla u$ satisfies

$$\mathscr{L}(\vec{F}_c, v) \leqslant \mathscr{L}(\vec{F}_c, u) \leqslant \mathscr{L}(\vec{G}, u), \quad \text{for all} \quad (\vec{G}, v) \in H(\mathrm{div};\Omega)\times L^2(\Omega). \tag{3.1}$$

Here we are more interested in a finite volume scheme introduced in [23] where the flux variable is obtained by minimizing the following energy functional of a vector field $\vec{G}$ on the whole domain $\Omega$,

$$\widehat{W}(\vec{G}) := \int_\Omega \frac{|\vec{G}|^2}{\kappa(u)}\ d\Omega - 2\int_\Omega u\operatorname{div}\vec{G}\ d\Omega. \tag{3.2}$$

For given $u$ and constant diffusion coefficient $\kappa_c$, minimization of functional (3.2) is equivalent to the solution of the right inequality in (3.1). Although the scheme avoids the errors coming from vertex values and edge diffusion coefficients calculated through interpolation procedure, it loses accuracy on distorted meshes and can only solve problems with Neumman or flux boundary conditions [14,13]. In order to extend the scheme in [23] and overcome its shortcomings, in this section, the relation between flux $\vec{F}$ and density $u$ will be rebuilt by minimizing a cell functional, afterwards a finite volume scheme is obtained by using this new relation. The resulting scheme can deal with all kinds of boundary conditions in the uniform form (2.3) and has almost second order convergence rate on highly distorted meshes.

### 3.1. Cell functional minimization algorithm

We begin the discussion by introducing a cell functional of a vector function $\vec{G}$, given by

$$W(\vec{G}) := \int_e \frac{|\vec{G}|^2}{\kappa(u)}\ de - 2\int_e u\operatorname{div}\vec{G}\ de + 2\oint_{\partial e} u\vec{G}\cdot\vec{n}\ ds. \tag{3.3}$$

Obviously, (3.3) can be viewed as a modified version of (3.2), by replacing the computational domain $\Omega$ with the cell $e$ and by adding a contour integration term. It is straightforward to obtain the theorem below by making use of the Green's formula.

**Theorem 3.1.** *The flux $\vec{F} = -\kappa(u)\nabla u$ minimizes the cell functional (3.3), i.e.,*

$$W(\vec{G}) = W(\vec{F}) + \int_e \frac{|\vec{G}-\vec{F}|^2}{\kappa(u)}\ de \geqslant W(\vec{F}). \tag{3.4}$$

In the following, we discretize the cell functional (3.3) in a special manner. First, we introduce discrete unknowns. The intensity and flux unknowns, with respect to a single quadrilateral cell, are shown in Fig. 1, where $u_e$ and $u_k (1 \leqslant k \leqslant 4)$ denote the intensity unknowns defined at the cell center and the edge midpoints, respectively, while $f_k$ denote the normal components of the edge flux, defined by

$$f_k = \int_{s_k} \vec{F}\cdot\vec{n}\ ds \approx \left(\vec{F}\cdot\vec{n}_k\right) m(s_k), \quad 1 \leqslant k \leqslant 4,$$

where $\vec{n}_k$ denotes unit vector normal to the edge $s_k$ of mesh cell $e$, $m(s_k)$ is the measure of $s_k$.

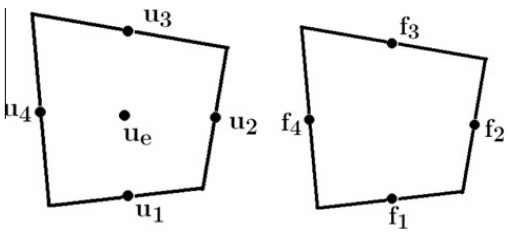



**Fig. 1.** Locations of unknowns. (a) The intensity unknowns are located at cell center and edge midpoints. (b) The edge fluxes are located at edge midpoints.

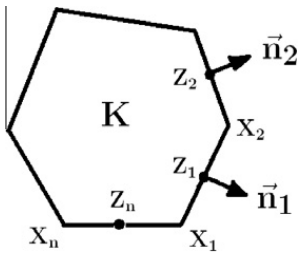



**Fig. 2.** A polygonal domain $K$ in 2-space.

Next, we approximate the divergence integral

$$\int_e u \operatorname{div} \vec{F}\ de \approx u_e \oint_{\partial e} \vec{F} \cdot \vec{n}\ ds = u_e \sum_{k=1}^{4} f_k, \tag{3.5}$$

and discretize the contour integral

$$\oint_{\partial e} u \vec{F} \cdot \vec{n}\ ds \approx \sum_{k=1}^{4} u_k f_k. \tag{3.6}$$

Finally, we approximate the first term in the functional (3.3). The key point is to approximate $|\vec{F}|^2$. For this, we introduce the following result.

**Theorem 3.2.** *For a polygonal domain K in the 2-dimensional space as show in Fig. 2, and a constant vector $\vec{v} = (v_1, v_2)$, we have*

$$m(K)\vec{v} = \sum_{i=1}^{n} m(\partial K_i)(\vec{v} \cdot \vec{n}_i) \vec{Z_* Z_i}, \tag{3.7}$$

*where m(K) and m(∂K_i) denote respectively the measures of polygonal domain K and edge ∂K_i, $\vec{n}_i$ denotes the outward-directed unit normal associated with edge ∂K_i. Here Z^* is an arbitrary point, Z_i is the midpoint of edge ∂K_i, and the vector $\vec{Z_* Z_i} = Z_i - Z_*$.*

**Proof.** It suffices to prove that the identity (3.7) holds for the first component $v_1$. Let $x^*$ and $x_i$ denote respectively the first component coordinates of points $Z^*$ and $Z_i$. Then,

$$\begin{aligned} m(K)v_1 &= \int_K \operatorname{div}((x - x_*)\vec{v}) dK = \oint_{\partial K} (x - x_*)\vec{v} \cdot \vec{n}\ ds = \sum_{i=1}^{n} (\vec{v} \cdot \vec{n}_i) \int_{\partial K_i} (x - x_*) ds = \sum_{i=1}^{n} (\vec{v} \cdot \vec{n}_i) \int_{\partial K_i} (x - x_i + x_i - x_*) ds \\ &= \sum_{i=1}^{n} (\vec{v} \cdot \vec{n}_i) \int_{\partial K_i} (x_i - x_*) ds = \sum_{i=1}^{n} m(\partial K_i)(\vec{v} \cdot \vec{n}_i)(x_i - x_*), \end{aligned}$$

The proof is completed. □

By using Theorem 3.2, we are ready to approximate $|\vec{F}|^2$. As shown in Fig. 3, $\vec{F}_k$ denotes the average value of $\vec{F}$ on the triangle domain $X_{k-1}X_kX_{k+1}$. Hereafter, for simplicity of exposition, we use $k$ as a periodic subscript with 4 period. For example,

$$X_k = X_{(k \bmod 4)},$$

where $(k \bmod 4)$ represents the remainder of $k$ modulo 4, i.e., $X_0 = X_4$, $X_5 = X_1$, etc. Let $\vec{l}_k = X_k \vec{X}_{k+1}$, $l_k = |\vec{l}_k|$, $k = 1, \ldots, 4$, and $Z_k$ is the midpoint of $X_kX_{k+1}$.

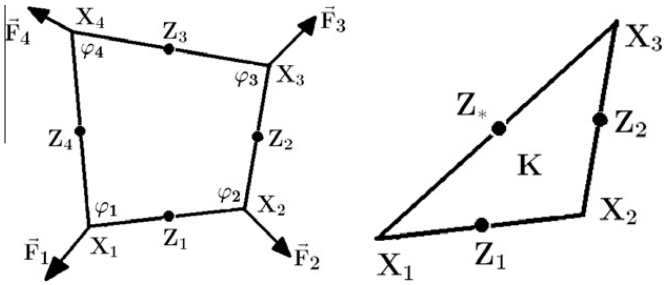



**Fig. 3.** Approximation for $|\vec{F}|^2$ from the trigonal standpoint.

Assume that $\vec{F}_2$ is constant on triangular domain $X_1X_2X_3$. Using Theorem 3.2 on $X_1X_2X_3$ and choosing $Z^*$ to be the midpoint of $X_1X_3$, we get

$$S_2\vec{F}_2 = l_1(\vec{F}_2\cdot\vec{n}_1)\vec{Z_*Z_1} + l_2(\vec{F}_2\cdot\vec{n}_2)\vec{Z_*Z_2} = \frac{1}{2}(-f_1\vec{l}_2 + f_2\vec{l}_1),$$

where $S_2$ is the area of triangle $\triangle X_1X_2X_3$. Thus,

$$|\vec{F}_2|^2 = \frac{1}{4S_2^2}\left[f_1^2l_2^2 + f_2^2l_1^2 - 2f_1f_2(\vec{l}_1\cdot\vec{l}_2)\right].$$

$|\vec{F}_1|^2, |\vec{F}_3|^2, |\vec{F}_4|^2$ are obtained analogously. Now, let

$$\frac{1}{m(e)}\int_e |\vec{F}|^2 de \approx \omega_1|\vec{F}_1|^2 + \omega_2|\vec{F}_2|^2 + \omega_3|\vec{F}_3|^2 + \omega_4|\vec{F}_4|^2, \tag{3.8}$$

where $\omega_k (k = 1,2,3,4)$ are certain weights that will be determined later. Then, (3.8) leads to,

$$\begin{aligned}\int_e \frac{|\vec{F}|^2}{\kappa(u)}\,de &\approx \frac{m(e)}{\kappa(u_e)}\sum_{k=1}^4 \omega_k|\vec{F}_k|^2 = \frac{m(e)}{\kappa(u_e)}\sum_{k=1}^4 \omega_k \frac{f_{k-1}^2l_k^2 + f_k^2l_{k-1}^2 - 2f_{k-1}f_k\left(\vec{l}_{k-1}\cdot\vec{l}_k\right)}{4S_k^2}\\ &= \frac{m(e)}{\kappa(u_e)}\sum_{k=1}^4 \omega_k \frac{f_{k-1}^2l_k^2 + f_k^2l_{k-1}^2 + 2f_{k-1}f_kl_{k-1}l_k\cos\varphi_k}{l_{k-1}^2l_k^2\sin^2\varphi_k},\end{aligned} \tag{3.9}$$

where $m(e)$ denotes the cell area and $\varphi_k$ the angle $\angle X_{k-1}X_kX_{k+1}$, see Fig. 3.

At last, putting (3.5), (3.6) and (3.9) together and through some straightforward algebra, we reach

$$W\left(\vec{F}\right) \approx \mathbf{F}^T\mathbf{A}\mathbf{F} - 2\mathbf{F}^T\mathbf{U}, \tag{3.10}$$

where

$$\mathbf{F} = (f_1, f_2, f_3, f_4)^T, \quad \mathbf{U} = (u_e - u_1, u_e - u_2, u_e - u_3, u_e - u_4)^T,$$

and $\mathbf{A}$ is a symmetric matrix, given by

$$\mathbf{A} = \frac{m(e)}{\kappa(u_e)}\mathbf{B}, \tag{3.11}$$

here the symmetric matrix $\mathbf{B} = (b_{ij})_{4\times4}$ is given by

$$b_{kk} = \frac{\omega_k}{l_k^2\sin^2\varphi_k} + \frac{\omega_{k+1}}{l_k^2\sin^2\varphi_{k+1}}, \quad b_{k,k+1} = \frac{\omega_{k+1}\cos\varphi_{k+1}}{l_kl_{k+1}\sin^2\varphi_{k+1}}, \quad b_{k,k+2} = 0, \quad \text{for} \quad k = 1,2,3,4.$$

For matrix $\mathbf{A}$, we have the following result.

**Theorem 3.3.** *Suppose that $\sin\varphi_k \neq 0$, and $\omega_k > 0$ $(k = 1,2,3,4)$, then matrix $\mathbf{A}$ defined in (3.11) is symmetric positive definite.*

**Proof.** By (3.11), we only need to prove that $\mathbf{B}$ is positive definite. For any nonzero vector $\mathbf{R} = (r_1, r_2, r_3, r_4)^T$, we have, from $\sin\varphi_k \neq 0$ and $\omega_k > 0$,

$$\mathbf{R}^T\mathbf{B}\mathbf{R} = \sum_{k=1}^4 \frac{\omega_k}{\sin^2\varphi_k}\left(\frac{r_{k-1}}{l_{k-1}}, \frac{r_k}{l_k}\right)\begin{pmatrix}1 & \cos\varphi_k\\ \cos\varphi_k & 1\end{pmatrix}\begin{pmatrix}r_{k-1}/l_{k-1}\\ r_k/l_k\end{pmatrix} > 0.$$

The last inequality holds, because

$$\begin{pmatrix} 1 & \cos\varphi_k \\ \cos\varphi_k & 1 \end{pmatrix}$$

is positive definite. Thus, matrix **B** is positive definite, which complete the proof. □

Since **A** is symmetric positive definite, the discrete cell functional (3.10) achieves its minimization if and only if

$$\mathbf{AF} = \mathbf{U}. \tag{3.12}$$

### 3.2. Finite volume scheme based on the cell functional minimization

As can be seen from the discussion in the previous subsection, there are nine unknowns on a single cell, and by (3.12), only five of them are independent. Thus, in selecting the independent unknowns for a cell, we have mainly two choices, i.e.,

- case (i): $(u_e, f_1, f_2, f_3, f_4)$;
- case (ii): $(u_e, u_1, u_2, u_3, u_4)$.

For case (i) the equation corresponding to the cell centered unknown $u_e$ can be obtained from (2.7) by the standard finite volume spatial discretization,

$$\frac{\partial u_e^{(i,j)}}{\partial t} + \frac{1}{m(e_{i,j})}\left(f_1^{(i,j)} + f_2^{(i,j)} + f_3^{(i,j)} + f_4^{(i,j)}\right) = S_e^{(i,j)}, \quad 1 \leqslant i \leqslant I, \quad 1 \leqslant j \leqslant J, \tag{3.13}$$

where $(i,j)$ denotes the index of a structured cell in the 2-D computational domain, $S_e^{(i,j)} = S(x_e^{(i,j)}, y_e^{(i,j)}, t)$ and $(x_e^{(i,j)}, y_e^{(i,j)})$ denote the coordinates of cell center $(i,j,e)$ as shown in Fig. 4(a). The equation corresponding to the edge unknowns $f_k$ can be obtained by using the continuity of the intensity across the cell edges. Taking edge $(i,j,2)$ or $(i+1,j,4)$ in Fig. 4(b) for example, we have $u_2^{(i,j)} = u_4^{(i+1,j)}$ so that

$$\sum_{k=1}^{4} a_{2k}^{(i,j)} f_k^{(i,j)} - u_e^{(i,j)} = \sum_{k=1}^{4} a_{4k}^{(i+1,j)} f_k^{(i+1,j)} - u_e^{(i+1,j)},$$

where $a_{lk}^{(i,j)}$ are the entries of cell matrix **A**. In this approach, the solution of the local linear system (3.12) is unnecessary. However, when the above scheme is applied to the second-order elliptic problems, the resulting algebraic system is of saddle-point type.

For case (ii), we first solve the local linear system (3.12) to get the explicit expressions of the flux variables. Then, by still taking the edge $(i,j,2)$ or $(i+1,j,4)$ in Fig. 4(b) for exposition and using the flux continuity condition $f_2^{(i,j)} = -f_4^{(i+1,j)}$, we obtain the equation corresponding to the cell edge intensity unknown $u_2^{(i,j)}$,

$$-\sum_{k=1}^{4} \hat{a}_{2k}^{(i,j)}\left(u_e^{(i,j)} - u_k^{(i,j)}\right) - \sum_{k=1}^{4} \hat{a}_{4k}^{(i+1,j)}\left(u_e^{(i+1,j)} - u_k^{(i+1,j)}\right) = 0,$$

where $\hat{a}_{lk}^{(i,j)}$ denote the entries of $\mathbf{A}^{-1}$. The equation for the cell centered unknown is obtained by substituting the flux expressions into (3.13),

$$\frac{\partial u_e^{(i,j)}}{\partial t} + \frac{1}{m(e_{i,j})}\sum_{k=1}^{4}\left(\sum_{l=1}^{4} \hat{a}_{lk}^{(i,j)}\right)\left(u_e^{(i,j)} - u_k^{(i,j)}\right) = S_e^{(i,j)}.$$

If all the cell matrices are symmetric positive definite, by almost the same procedure in [28], the resulting algebraic system can be a symmetric positive definite one.

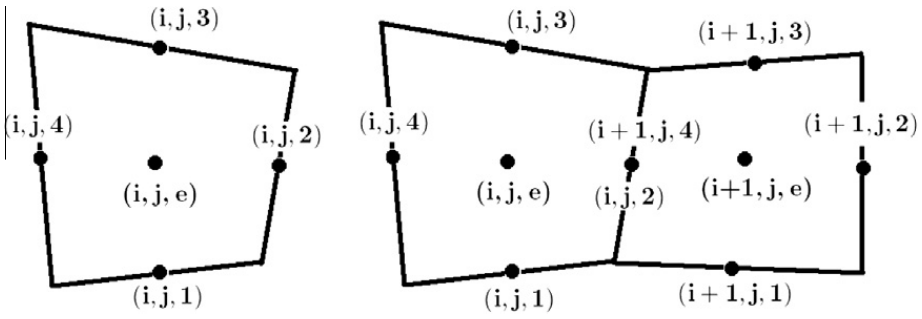



**Fig. 4.** (a) The stencil for the cell-center (i,j,e) of cell (i,j); (b) An example stencil for the inner edge-center (i,j,2) or (i + 1,j,4).

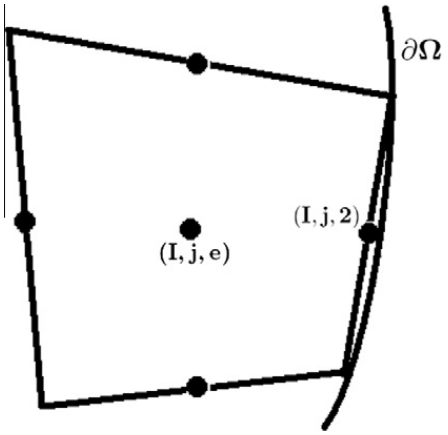



**Fig. 5.** An example stencil for the right edge midpoint (I,j,2) of cell (I,j) on the boundary $\partial\Omega$.

Next, we consider the situation of cell edges on the domain boundary. For instance, let us consider the right edge of the cell $(I,j)$ on the domain boundary $\partial\Omega$, see Fig. 5. From the boundary condition (2.3) and flux (2.4), we have

$$-\int_{s_2^{(I,j)}} \alpha \vec{F}\cdot\vec{n}\, ds + \int_{s_2^{(I,j)}} \beta u\, ds = \int_{s_2^{(I,j)}} g\, ds. \tag{3.14}$$

We approximate above boundary condition,

$$-\alpha f_2^{(I,j)} + \beta u_2^{(I,j)} m\left(s_2^{(I,j)}\right) = g_2^{(I,j)} m\left(s_2^{(I,j)}\right), \tag{3.15}$$

where $g_2^{(I,j)} = g(x_2^{(I,j)}, y_2^{(I,j)}, t)$. If the edge intensity variables are treated as intermediate ones, the discrete boundary condition (3.15) becomes

$$-\alpha f_2^{(I,j)} - \beta\left(\sum_{k=1}^{4} a_{2k}^{(I,j)} f_k^{(I,j)} - u_e^{(I,j)}\right) m\left(s_2^{(I,j)}\right) = g_2^{(I,j)} m\left(s_2^{(I,j)}\right).$$

On the other hand, if we choose flux variables to be intermediate ones, (3.15) leads to

$$-\alpha \sum_{k=1}^{4} \hat{a}_{2k}^{(I,j)}\left(u_e^{(I,j)} - u_k^{(I,j)}\right) + \beta u_2^{(I,j)} m\left(s_2^{(I,j)}\right) = g_2^{(I,j)} m\left(s_2^{(I,j)}\right).$$

In conclusion, our cell functional minimization algorithm yields a 5-point stencil for the cell-center equations, a 9-point stencil for the inner edge-center equations, and a 5-point stencil for the boundary edge-center equations, see Figs. 4 and 5.

### 3.3. The cell matrix **A**

From the above subsection we have seen that the relation (3.12), derived from the cell functional minimization approach, plays an important part in the construction of our finite volume scheme. In this subsection, we shall discuss the problem of constructing the cell matrix **A** in (3.12). The first type of cell matrix is given by (3.11) with the weights specified as

$$\omega_k = \frac{1}{4}, \quad k = 1,2,3,4. \tag{3.16}$$

Our numerical experience indicates that this type of cell matrix usually leads to poor accuracy on distorted meshes. Here we are more interested in searching some new ones. The main idea is to require that (3.12) satisfies the linearity preserving criterion [35], which means that it holds exactly for the linear case where the solution $u$ is a linear function and the diffusion coefficient is a constant on each cell, i.e.,

$$u = a_e x + b_e y + c_e, \quad \kappa(u) = \kappa_e, \quad \text{on } e, \tag{3.17}$$

where $a_e$, $b_e$, $c_e$, $\kappa_e$ are constants.

For the above purpose, we need some more notations and preliminaries. We still employ Fig. 3 for exposition and associate it with some new notations. Denote by $\vec{n}_k$ the outward unit vector normal to the tangential vector $\vec{l}_k = \vec{X_k X_{k+1}}$. Here we point out once again that $k$ is a periodic subscript with 4 period. Denote by $Z_e$ the cell center at which the cell centered unknown $u_e$ is defined. Introduce further the $4\times 2$ matrices $\mathscr{F} = (f_{kj})$ and $\mathscr{U} = (u_{kj})$ by

$$f_{kj} = -\kappa_e l_k(\vec{e}_j\cdot\vec{n}_k) \quad \text{and} \quad u_{kj} = -\vec{e}_j\cdot\vec{Z_e Z_k},$$

where $\vec{e}_1 = (1,0)^T,\ \vec{e}_2 = (0,1)^T,\ l_k = |X_k X_{k+1}|,\ k = 1,2,3,4$ and $j = 1, 2$. By Theorem 3.2,

$$m(e)\vec{e}_j = \sum_{k=1}^{4} l_k(\vec{e}_j \cdot \vec{n}_k)\vec{Z_e Z_k}\,. \tag{3.18}$$

Multiplying the above identity with $\kappa_e \vec{e}_i (i = 1,2)$ yields

$$\mathscr{U}^T \mathscr{F} = \kappa_e m(e)\mathscr{I}, \tag{3.19}$$

where $\mathscr{I}$ is a $2 \times 2$ identity matrix.

Now, by requiring that (3.12) is exact for the linear case (3.17), we reach

$$\mathbf{A}\mathscr{F} = \mathscr{U}. \tag{3.20}$$

Recall that the cell edge unknowns $u_k (k = 1,2,3,4)$ are defined at the midpoints of the cell edges, while the cell centered unknown $u_e$ is defined at the cell center $Z_e$ whose definition has not been given yet. Usually coordinates of the cell center $\overline{Z}_e(\bar{x}_e, \bar{y}_e)$ are obtained by arithmetic averaging of those of the cell nodes, i.e.,

$$\bar{x}_e = \frac{1}{4}\sum_{i=1}^{4} x_i, \quad \bar{y}_e = \frac{1}{4}\sum_{i=1}^{4} y_i. \tag{3.21}$$

However, in our method, the cell center $Z_e(x_e, y_e)$ can be any fixed point in the cell. Denote $\vec{t}_e = \vec{\overline{Z}_e Z_e}$. Then, on the one hand, we have by straightforward calculation,

$$u_{k,j} = \vec{e}_j \cdot \vec{t}_e + \vec{e}_j \cdot \vec{Z_k \overline{Z}_e} = \vec{e}_j \cdot \vec{t}_e + \frac{1}{4}\vec{e}_j \cdot \left(\vec{l}_{k+1} - \vec{l}_{k-1}\right),$$

On the other hand, by the definition of matrix $\mathbf{B}$ in (3.11), we find that

$$\mathbf{B} = \sum_{k=1}^{4} \frac{\omega_k}{l_{k-1} l_k \sin^2 \varphi_k} \mathbf{T}_k \mathbf{B}_k \mathbf{T}_k^T, \tag{3.22}$$

where

$$\mathbf{T}_1 = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \end{pmatrix}^T, \quad \mathbf{T}_2 = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{pmatrix}^T,$$
$$\mathbf{T}_3 = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}^T, \quad \mathbf{T}_4 = \begin{pmatrix} 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}^T$$

and

$$\mathbf{B}_k = \begin{pmatrix} \frac{l_k}{l_{k-1}} & \cos\varphi_k \\ \cos\varphi_k & \frac{l_{k-1}}{l_k} \end{pmatrix}.$$

Still by straightforward calculation,

$$\mathbf{B}_k \mathbf{T}_k^T \mathscr{F} = -\kappa_e \sin\varphi_k \begin{pmatrix} -\vec{l}_k \cdot \vec{e}_1 & -\vec{l}_k \cdot \vec{e}_2 \\ \vec{l}_{k-1} \cdot \vec{e}_1 & \vec{l}_{k-1} \cdot \vec{e}_2 \end{pmatrix}, \tag{3.23}$$

where we have used the identities

$$\begin{aligned} &l_k(\vec{n}_{k-1} + \vec{n}_k \cos\varphi_k) = -\sin\varphi_k \vec{l}_k, \\ &l_{k-1}(\vec{n}_k + \vec{n}_{k-1}\cos\varphi_k) = \sin\varphi_k \vec{l}_{k-1}, \quad k = 1,2,3,4. \end{aligned} \tag{3.24}$$

Putting (3.11), (3.22) and (3.23) together gives

$$\mathbf{A}\mathscr{F} = -m(e)\sum_{k=1}^{4} \frac{\omega_k}{l_{k-1} l_k \sin\varphi_k} \mathbf{T}_k \begin{pmatrix} -\vec{l}_k \cdot \vec{e}_1 & -\vec{l}_k \cdot \vec{e}_2 \\ \vec{l}_{k-1} \cdot \vec{e}_1 & \vec{l}_{k-1} \cdot \vec{e}_2 \end{pmatrix} = (\alpha_{kj})_{4\times 2}, \tag{3.25}$$

where

$$\alpha_{k,j} = \left(\tilde{\omega}_{k+1}\vec{l}_{k+1} - \tilde{\omega}_k \vec{l}_{k-1}\right) \cdot \vec{e}_j, \quad \tilde{\omega}_k = \frac{m(e)\omega_k}{l_{k-1} l_k \sin\varphi_k}.$$

Substituting (3.25) and (3.21) into (3.20), we finally arrive at

$$(4\tilde{\omega}_{k+1} - 1)\vec{l}_{k+1} + (1 - 4\tilde{\omega}_k)\vec{l}_{k-1} = 4\vec{t}_e, \quad k = 1,2,3,4. \tag{3.26}$$

By these identities, we have the result below.

**Theorem 3.4.** *If cell e is not a parallelogram and the cell center coincides with $\overline{Z}_e$, then (3.12), with cell matrix* **A** *given by (3.11), holds exactly for the linear case (3.17) if and only if*

$$\omega_k = \frac{l_{k-1} l_k \sin \varphi_k}{4m(e)}, \quad k = 1,2,3,4. \tag{3.27}$$

**Proof.** Obviously, (3.27) is a sufficient condition. The nontrivial part is to prove that (3.27) is also a necessary one. Since cell $e$ is not a parallelogram, we assume that, without any harm to our argument, $\vec{l}_1$ is not collinear with $\vec{l}_3$. By using (3.26) and noting $\vec{t}_e = 0$, we have

$$\begin{cases} (4\tilde{\omega}_3 - 1)\vec{l}_3 + (1 - 4\tilde{\omega}_2)\vec{l}_1 = 0, \\ (4\tilde{\omega}_1 - 1)\vec{l}_1 + (1 - 4\tilde{\omega}_4)\vec{l}_3 = 0, \end{cases} \tag{3.28}$$

which implies $\tilde{\omega}_k = 1/4 (k = 1,2,3,4)$ and in turn (3.27). □

Although we have provided a systematic way to construct finite volume schemes for parabolic Eq. (2.1), the above result indicates that there is no much flexibility, at least for the structured quadrilateral meshes. The limitation is caused by the fact that the cell matrix **A** should be given by (3.11). If we drop this requirement and relay only on (3.20) to construct a symmetric positive definite cell matrix, things will become different. Motivated by [10,11], we give a constructional algorithm by just choosing

$$\mathbf{A} = \frac{1}{\kappa_e m(e)} \mathscr{U}\mathscr{U}^T + \mathscr{C}\mathscr{D}\mathscr{C}^T, \tag{3.29}$$

where $\mathscr{D}$ is an arbitrary $2 \times 2$ symmetric positive definite matrix, and $\mathscr{C}$ is a $4 \times 2$ matrix whose columns span the null space of the matrix $\mathscr{F}^T$ so that $\mathscr{F}^T\mathscr{C} = 0$. By (3.19) we see that the rank of $\mathscr{F}$ is two and $\mathscr{C}$ is well defined. Due to (3.19), the matrix **A** given by (3.29) satisfies (3.20) and is symmetric and positive semidefinite. The positive definiteness of this new cell matrix follows from its nonsingularity, which can be obtained by using the same procedure in the proof of Theorem 3.1 in [10].

As in [10], (3.29) allows us to construct a family of finite volume schemes and moreover, enable us to extend the present results to the unstructured polygonal meshes. More interesting is that the undetermined parameters in (3.29) include not only the entries of the symmetric positive definite matrix $\mathscr{D}$ but also the coordinates of the cell center $Z_e$. Since only the mass center is allowed for the cell center and only case (i) can be employed for the unknowns in [10], the present approach seems a little more flexible. Moreover, since we have an equivalent choice of unknowns, i.e., case (ii), and the resulting linear system is symmetric positive definite, the saddle-point problem suffered by MFD scheme in [10] can be avoided in our method. During the revision process of this paper, we were informed by one of the reviewer that the MFD scheme in [10] has been extended in [17] where the cell center is allowed to be any point in the cell in order to establish certain relation with the other two finite volume schemes. In addition, in order to be consistent with the framework of the original MFD method, a weight function $w_E$ has to be introduced, although the existence of this weight function has been proved. Since in our derivation there is no requirement for the location of the cell center, it seems natural in the present setting to choose any fixed point to be the cell center.

At last, we point out that the finite volume scheme based on cell functional minimization approach can now be viewed as a family of schemes, some of which have close relations with some schemes studied before, including the famous LSOM and MFD schemes. We list several special cases below:

- *Special case* 1. If we choose case (i) for unknowns and (3.11) with weights given by (3.16) for cell matrix, the resulting scheme is identical to the one in [23,14].
- *Special case* 2. If we choose case (i) for unknowns and (3.11) with weights given by (3.27) for cell matrix, the resulting scheme coincides with the scheme in [33].
- *Special case* 3. If we choose case (ii) for unknowns and use (3.11) and (3.27) to define the cell matrix, the resulting scheme is identical to the LSOM scheme in [28].
- *Special case* 4. If we choose case (i) for unknowns and use (3.29) and mass center to define the cell matrix, the resulting scheme coincides with the MFD scheme in [10] confined to the structured quadrilateral meshes.

### 3.4. A parallel algorithm based on nonoverlapping domain decomposition

Since the scheme discussed above employs both cell centered and cell edge unknowns, it has much more degrees of freedom than the pure cell centered schemes, which not only needs more memory space but also influences the computational speed in the large scale practical simulations. In order to conquer this weakness, we suggest a parallel algorithm based on nonoverlapping domain decomposition approach. We find that the cell edge unknowns in the scheme facilitate the construction of all kinds of interface conditions.

For the sake of simplicity, we only consider the case where the domain $\Omega$ is decomposed into two nonoverlapping subdomains $\Omega_1$ and $\Omega_2$, and denote $\Gamma_{12} = \overline{\Omega}_1 \cap \overline{\Omega}_2$. A parallel algorithm with Dirichlet–Robin interface conditions for (2.1) and (2.3) is suggested as follows

$$\begin{aligned}
&\frac{u_1^{n+1,s} - u_1^n}{\tau} - \operatorname{div}\left(\kappa(u_1^{n+1,s})\nabla u_1^{n+1,s}\right) = S^{n+1}, \quad \text{in} \quad \Omega_1,\\
&\alpha\kappa\left(u_1^{n+1,s}\right)\frac{\partial}{\partial \vec{n}_1}u_1^{n+1,s} + \beta u_1^{n+1,s} = g^{n+1}, \quad \text{on} \quad \partial\Omega \cap \partial\Omega_1,\\
&u_1^{n+1,s} = u_2^{n+1,s-1}, \quad \text{on} \quad \Gamma_{12},\\
&u_1^0 = \psi
\end{aligned}$$

and

$$\begin{aligned}
&\frac{u_2^{n+1,s} - u_2^n}{\tau} - \operatorname{div}\left(\kappa\left(u_2^{n+1,s}\right)\nabla u_2^{n+1,s}\right) = S^{n+1}, \quad \text{in} \quad \Omega_2,\\
&\alpha\kappa\left(u_2^{n+1,s}\right)\frac{\partial}{\partial \vec{n}_2}u_2^{n+1,s} + \beta u_2^{n+1,s} = g^{n+1}, \quad \text{on} \quad \partial\Omega \cap \partial\Omega_2,\\
&\lambda u_2^{n+1,s} + \kappa\left(u_2^{n+1,s}\right)\frac{\partial}{\partial \vec{n}_2}u_2^{n+1,s} = \lambda u_1^{n+1,s-1} - \kappa\left(u_1^{n+1,s-1}\right)\frac{\partial}{\partial \vec{n}_1}u_1^{n+1,s-1}, \quad \text{on} \quad \Gamma_{12},\\
&u_2^0 = \psi,
\end{aligned}$$

where $s$ denotes the iterative number, $u_i^{n+1,0} = u_i^n$, $u_i^n$ is the approximate solution in $\Omega_i$ at time $t = n\tau$, and $\tau$ is the time step.

The implementation of the above algorithm is standard and is then omitted. Here we point out that other types of interface conditions can also be considered in this parallel algorithm. Moreover, if the problem is solved on the whole domain and a big global matrix is involved, the above domain decomposition algorithm can serve as an efficient preconditioner for certain iteration method of Krylov type.

## 4. Numerical experiments

In this section, we provide numerical results for three examples on the distorted structured quadrilateral meshes. One example is a linear problem with a continuous diffusion coefficient. The following example is a linear problem with a discontinuous diffusion coefficient, and the last example is about a nonlinear problem. We will investigate the performance of our finite volume scheme and its parallel version on some typical distorted quadrilateral meshes, including the sine mesh, the Shestakov mesh, the Kershaw mesh and the random mesh. Descriptions of these distorted meshes can be found in [28,35]. Random meshes used in experiments are obtained from their orthogonal counterparts with 20 percent random disturbance of the interior mesh vertices.

We shall examine the following four types of errors on the global computational domain $\Omega = [0,1] \times [0,1]$,

$$\begin{aligned}
&e_c = \max_{i,j}\left|u(x_{i,j,e}, y_{i,j,e,T}) - u_{i,j,e}^n\right|,\\
&e_i = \max_{i,j,k}\left|u(x_{i,j,k}, y_{i,j,k}, T) - u_{i,j,k}^n\right|, \quad k = 2,4,\\
&e_j = \max_{i,j,k}\left|u(x_{i,j,k}, y_{i,j,k}, T) - u_{i,j,k}^n\right|, \quad k = 1,3,\\
&e_{L_2} = \left(\sum_{i,j}\left|u(x_{i,j,e}, y_{i,j,e}, T) - u_{i,j,e}^n\right|^2 m(e_{i,j})\right)^{1/2},
\end{aligned}$$

where the notations $(i,j,k)$, $k = e, 1, \ldots, 4$ for the cell $e_{i,j}$ are shown in Fig. 4. We use the Picard iteration to linearize the nonlinear systems in the nonlinear case, and use GMRES method [30] to solve the linear systems in all experiments. Throughout, we shall choose case (ii) for unknowns and use (3.11) and (3.27) to define the cell matrix.

**Example 5.1.** Consider the following linear problem with a continuous diffusion coefficient,

$$\begin{aligned}
&\frac{\partial u}{\partial t} - \operatorname{div}(\nabla u) = S(x,y,t), \quad (x,y,t) \in \Omega \times (0,T],\\
&u(x,y,0) = 2 + \sin(\pi x)\sin(\pi y), \quad (x,y) \in \Omega,
\end{aligned}$$

with boundary conditions specified by

$$\begin{aligned}
&u(0,y,t) = 2\exp(-2\pi^2 t),\\
&\frac{\partial}{\partial \vec{n}}u(x,0,t) = \frac{\partial}{\partial \vec{n}}u(x,1,t) = -\pi\exp(-2\pi^2 t)\sin(\pi x),\\
&\left(\frac{\partial}{\partial \vec{n}} + 2\right)u(1,y,t) = \exp(-2\pi^2 t)(-\pi\sin(\pi y) + 4),
\end{aligned}$$

and the source term $S(x,y,t) = -4\pi^2 \exp(-2\pi^2 t)$. The exact solution of above linear problem is $u = \exp(-2\pi^2 t)(2 + \sin(\pi x)\sin(\pi y))$.

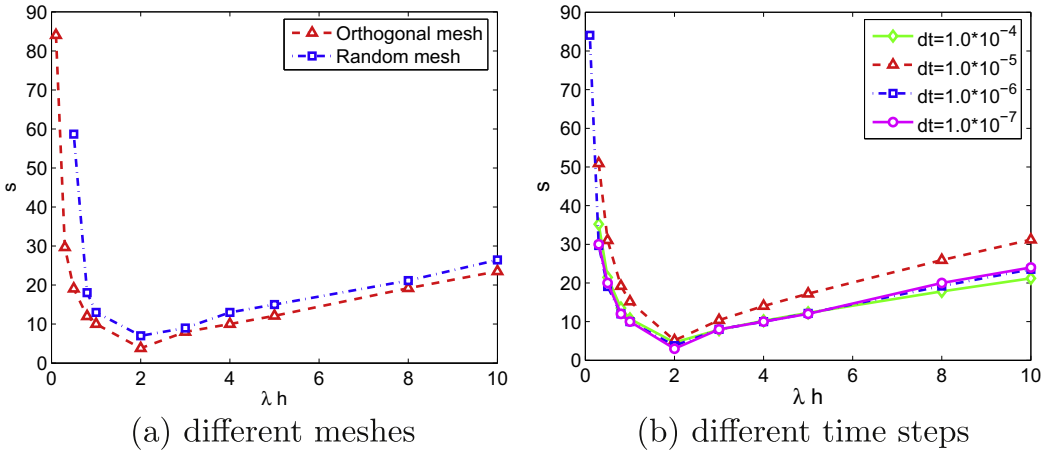
First, we examine the accuracy of our finite volume scheme, in which a sequence of random quadrilateral meshes are employed. Here the time step $\tau = 1.0e{-}6$, and the final time $T = 0.01$. The results are presented in Table 1 where the convergence rate is also given in the last column, which is obtained by a least squares fit. We can see that the discrete scheme based on minimizing the cell functional has a nearly second order accuracy.

Next, we test the effect of parameter $\lambda$ in the Dirichlet–Robin interface conditions of the parallel algorithm. Let $s$ denote the average iterative number per CPU per time step in the whole computing process, and *erb* is the maximum error bound in iterative convergence controlling. Both the x and y directions have $m$ processors, i.e., total CPUs = $m*m$ processors for the whole domain. The relations between $s$ and parameter $\lambda h$ are presented in Fig. 6, where $h$ denotes the average cell size of the mesh. In those two tests, we choose CPUs = $2 * 2$. One is tested on $10 * 10$ orthogonal and random meshes with $\tau = 1.0e{-}6$ and $T = 0.01$. The other is employed on $10 * 10$ orthogonal meshes with different time steps and the total number of time steps is fixed on $10^4$. Fig. 6(a) shows that iterative number on the random mesh is a little bigger than that on the uniform mesh, and the optimal value of $\lambda$ seems the same on both meshes, which implies that the optimal value of $\lambda$ is insensitive to the mesh distortion. Fig. 6(b) shows that the time step does not affect the optimal value of $\lambda$. The optimal value $O(h^{-1/2})$ of $\lambda$ for a Robin–Robin non-overlapping domain decomposition for Poisson equation is derived by using Poincare–Steklov operators in [27]. Making use of techniques in [27], we also get the optimal value $O(h^{-1})$ of $\lambda$ for a Dirichlet–Robin non-overlapping domain decomposition method. Here we omit intricate analysis, but our numerical experiments confirm it.

Now fixing the number of processors and $\lambda$(here $\lambda \approx 2/h$), numerical results on a sequence of random meshes are presented in Table 2. Table 3 shows results of the parallel algorithm with a fixed number of processors and a set of varying $\lambda$

**Table 1**
The accuracy of the scheme for Example 5.1 on a sequence of random meshes.

| Mesh | $10 * 10$ | $20 * 20$ | $40 * 40$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| $e_c$ | 4.6395E−3 | 1.9091E−3 | 4.7524E−4 | 1.9004E−4 | 1.85 |
| $e_i$ | 1.2481E−2 | 3.5619E−3 | 1.0846E−3 | 4.0806E−4 | 1.98 |
| $e_j$ | 1.0888E−2 | 3.4173E−3 | 1.0218E−3 | 4.5532E−4 | 1.80 |
| $e_{L_2}$ | 1.4046E−3 | 4.0909E−4 | 1.0054E−4 | 4.1752E−5 | 1.99 |




(a) different meshes (b) different time steps

**Fig. 6.** Iterative number $s$ versus parameter $\lambda h$.

**Table 2**
The accuracy of the parallel algorithm with CPUs = $2 * 2$.

| Mesh | $10 * 10$ | $20 * 20$ | $40 * 40$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| $\lambda$ | 20.0 | 40.0 | 80.0 | 120.0 | |
| $s$ | 6.9996 | 8.6729 | 6.9994 | 8.3318 | |
| $e_c$ | 4.6399E−3 | 1.9093E−3 | 4.7702E−4 | 1.9021E−4 | 1.85 |
| $e_i$ | 1.2480E−2 | 3.5614E−3 | 1.0831E−3 | 4.0791E−4 | 1.98 |
| $e_j$ | 1.0887E−2 | 3.4172E−3 | 1.0203E−3 | 4.5463E−4 | 1.80 |
| $e_{L_2}$ | 1.4047E−3 | 4.0909E−4 | 1.0078E−4 | 4.1882E−5 | 1.99 |

**Table 3**
Results of the parallel algorithm with varying $\lambda$,*erb* and CPUs = $2 * 2$.

| $\lambda$ | 10.0 | 20.0 | 50.0 | 100.0 |
|---|---|---|---|---|
| *erb* | 1.0E−8 | 1.0E−8 | 1.0E−8 | 1.0E−8 |
| *s* | 8.2247 | 6.9996 | 9.0007 | 14.1232 |
| $e_{L_2}$ | 1.4046E−3 | 1.4047E−3 | 1.4049E−3 | 1.4050E−3 |
| *erb* | 1.0E−10 | 1.0E−10 | 1.0E−10 | 1.0E−10 |
| *s* | 30.8528 | 11.9994 | 31.9983 | 64.1955 |
| $e_{L_2}$ | 1.4046E−3 | 1.4046E−3 | 1.4046E−3 | 1.4046E−3 |
| *erb* | 1.0E−12 | 1.0E−12 | 1.0E−12 | 1.0E−12 |
| *s* | 31.8526 | 12.9992 | 32.9982 | 65.1954 |
| $e_{L_2}$ | 1.4046E−3 | 1.4046E−3 | 1.4046E−3 | 1.4046E−3 |

**Table 4**
Results of the parallel algorithm with varying CPUs, *erb*, and $\lambda = 120$.

| CPUs | $2 * 2$ | $4 * 4$ | $10 * 10$ |
|---|---|---|---|
| *erb* | 1.0E−8 | 1.0E−8 | 1.0E−8 |
| *s* | 8.3318 | 8.9992 | 11.9989 |
| $e_{L_2}$ | 4.1882E−5 | 4.1892E−5 | 4.1753E−5 |
| *erb* | 1.0E−10 | 1.0E−10 | 1.0E−10 |
| *s* | 13.9991 | 15.9534 | 18.1870 |
| $e_{L_2}$ | 4.1752E−5 | 4.1752E−5 | 4.1752E−5 |
| *erb* | 1.0E−12 | 1.0E−12 | 1.0E−12 |
| *s* | 14.9990 | 18.4752 | 21.2068 |
| $e_{L_2}$ | 4.1752E−5 | 4.1752E−5 | 4.1752E−5 |

**Table 5**
Results of the parallel algorithm for the fixed size problem per processor.

| Mesh | $20 * 20$ | $30 * 30$ | $40 * 40$ | $50 * 50$ | $60 * 60$ | $80 * 80$ |
|---|---|---|---|---|---|---|
| CPUs | $2 * 2$ | $3 * 3$ | $4 * 4$ | $5 * 5$ | $6 * 6$ | $8 * 8$ |
| $\lambda$ | 40.0 | 60.0 | 80.0 | 100.0 | 120.0 | 160.0 |
| *s* | 3.9997 | 4.9995 | 4.9995 | 4.9996 | 4.9996 | 4.9996 |
| $e_{L_2}$ | 1.6557E−4 | 7.5416E−5 | 4.3751E−5 | 2.9103E−5 | 2.1183E−5 | 1.3467E−5 |

and *erb* on a $10 * 10$ random mesh. Table 4 presents results with a fixed $\lambda$ and mutative numbers of CPUs and *erb* on a $60 * 60$ random mesh. From Tables 3 and 4, we can see that computational errors do not be affected by the parameter $\lambda$ or the number of subdomains as maximum controlling error *erb* approaches zero. Obviously, it will cost more iterations by using a smaller maximum controlling error. All those Tables 2 and 4 show that the parallel algorithm has almost the same accuracy with the discrete scheme on one CPU by referring the results in Table 1.

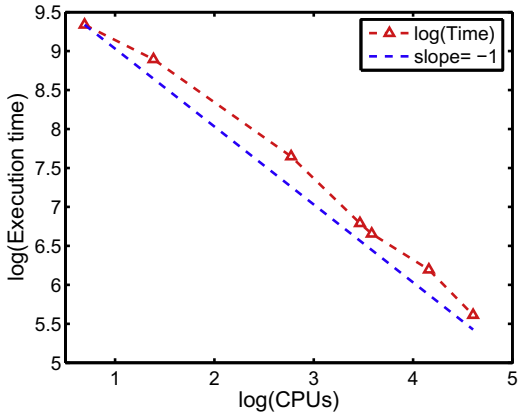



**Fig. 7.** Execution time decreased in inverse proportion to the number of processors.

At last, check the scalability for our unaccelerated parallel algorithm with an optimal value of $\lambda$. Table 5 shows parallel results on a series of orthogonal mesh as problem size and processor number are increased in proportion. It indicates that the iterative number $s$ does not increase much when the mesh is refining and the number of CPUs is increasing. Fig. 7 describes that execution time on a fixed $200 * 200$ orthogonal mesh decreases in inverse proportion to the number of processors.

**Example 5.2.** Consider a linear problem with discontinuous diffusion coefficients,

$$\frac{\partial u}{\partial t} - \operatorname{div}(\kappa(u)\nabla u) = S(x,y,t), \quad (x,y,t) \in \Omega \times (0,T],$$

where

$$\kappa(u) = \begin{cases} 0.4, & (x,y,t) \in \left(0,\frac{2}{3}\right] \times (0,1) \times (0,T], \\ 0.1, & (x,y,t) \in \left(\frac{2}{3},1\right) \times (0,1) \times (0,T] \end{cases}$$

with initial value

$$u(x,y,0) = \begin{cases} 2 + \sin(\pi x)\sin(2\pi y), & (x,y,t) \in \left(0,\frac{2}{3}\right] \times (0,1) \times (0,T], \\ 2 + \sin(4\pi x)\sin(2\pi y), & (x,y,t) \in \left(\frac{2}{3},1\right) \times (0,1) \times (0,T] \end{cases}$$

and boundary conditions

$$u(0,y,t) = 2\exp(-2\pi^2 t),$$
$$\frac{\partial}{\partial \vec{n}} u(x,1,t) = -\frac{\partial}{\partial \vec{n}} u(x,0,t) = \begin{cases} 2\pi\exp(-2\pi^2 t)\sin(\pi x), & x \in \left(0,\frac{2}{3}\right], \\ 2\pi\exp(-2\pi^2 t)\sin(4\pi x), & x \in \left(\frac{2}{3},1\right), \end{cases}$$
$$\left(\frac{\partial}{\partial \vec{n}} + 2\right) u(1,y,t) = 4\pi\exp(-2\pi^2 t)\sin(2\pi y) + 4\exp(-2\pi^2 t).$$

Here the source term $S(x,y,t) = -4\pi^2 \exp(-2\pi^2 t)$. The associated analytic solution is

$$u(x,y,t) = \begin{cases} \exp(-2\pi^2 t)(2 + \sin(\pi x)\sin(2\pi y)), & (x,y,t) \in \left(0,\frac{2}{3}\right] \times (0,1) \times (0,T], \\ \exp(-2\pi^2 t)(2 + \sin(4\pi x)\sin(2\pi y)), & (x,y,t) \in \left(\frac{2}{3},1\right) \times (0,1) \times (0,T]. \end{cases}$$

Table 6 shows the accuracy of the finite volume scheme based on the cell functional minimization algorithm on a sequence of random quadrilateral meshes, in which $\tau = 1.0e{-}6$ and $T = 0.01$.

In our numerical parallel experiments for this example, we will test parallel results in two special cases. One case is that interfaces are not on the material discontinuities, for example CPUs = $2 * 2$. Another case is that interfaces are on the material discontinuities, for example CPUs = $3 * 3$. Tables 7 and 8 present errors of parallel algorithm with optimal values $\lambda$ (here $\lambda \approx 5/(9h)$) on random meshes in those two cases. Comparing Tables 7 and 8 with Table 6, we can see that parallel results for a linear problem with a discontinuous diffusion coefficient also have good accuracy. Obviously, it seems that the accuracy almost keeps the same, the iterative number does not change much, and the optimal value of $\lambda$ nearly does not change whether interfaces are on the material discontinuity or not.

**Table 6**
The accuracy of the scheme for Example 5.2 on random meshes.

| Mesh | $18 * 18$ | $27 * 27$ | $45 * 45$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| $e_c$ | 2.0481E−2 | 1.2423E−2 | 5.4063E−3 | 2.9723E−3 | 1.65 |
| $e_i$ | 5.4576E−2 | 2.3280E−2 | 8.4362E−3 | 5.8177E−3 | 1.79 |
| $e_j$ | 2.8680E−2 | 1.1602E−2 | 5.6470E−3 | 2.8996E−3 | 1.99 |
| $e_{L_2}$ | 4.1674E−3 | 1.9870E−3 | 7.9039E−4 | 4.4058E−4 | 1.89 |

**Table 7**
The accuracy of the parallel algorithm for Example 5.2 with CPUs = $2 * 2$.

| Mesh | $18 * 18$ | $27 * 27$ | $45 * 45$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| $\lambda$ | 10.0 | 15.0 | 25.0 | 33.33 | |
| $s$ | 7.0005 | 8.9999 | 8.9999 | 8.9997 | |
| $e_c$ | 2.0481E−2 | 1.2423E−2 | 5.4064E−3 | 2.9723E−3 | 1.65 |
| $e_i$ | 5.4575E−2 | 2.3280E−2 | 8.4362E−3 | 5.8177E−3 | 1.79 |
| $e_j$ | 2.8680E−2 | 1.1602E−2 | 5.6470E−3 | 2.8996E−3 | 1.99 |
| $e_{L_2}$ | 4.1673E−3 | 1.9870E−3 | 7.9040E−4 | 4.4056E−4 | 1.89 |

**Table 8**
The accuracy of the parallel algorithm for Example 5.2 with CPUs = 3 * 3.

| Mesh | 18 * 18 | 27 * 27 | 45 * 45 | 60 * 60 | Rate |
|---|---|---|---|---|---|
| $\lambda$ | 10.0 | 15.0 | 25.0 | 33.33 | |
| $s$ | 9.0004 | 11.0000 | 10.9660 | 10.9996 | |
| $e_c$ | 2.0480E−2 | 1.2422E−2 | 5.4036E−3 | 2.9729E−3 | 1.65 |
| $e_i$ | 5.4575E−2 | 2.3280E−2 | 8.4372E−3 | 5.8178E−3 | 1.79 |
| $e_j$ | 2.8680E−2 | 1.1602E−2 | 5.6429E−3 | 2.8996E−3 | 1.99 |
| $e_{L_2}$ | 4.1674E−3 | 1.9871E−3 | 7.9045E−4 | 4.4069E−4 | 1.89 |

**Example 5.3.** In this example, we will consider the following nonlinear problem,

$$\frac{\partial u}{\partial t} - \mathrm{div}(u\nabla u) = S(x,y,t), (x,y,t) \in \Omega \times (0,T],$$
$$u(x,y,0) = 2 + \sin(\pi x)\sin(\pi y), (x,y) \in \Omega$$

with Dirichlet boundary condition

$$u(x,y,t) = 2\exp(-2\pi^2 t), \quad (x,y) \in \partial\Omega,$$

and the source term $S(x,y,t) = -2\pi^2\exp(-2\pi^2 t)(2 + \sin(\pi x)\sin(\pi y)) + \pi^2\exp(-4\pi^2 t)[(4 + 2\sin(\pi x)\sin(\pi y))\sin(\pi x)\sin(\pi y) - \cos^2(\pi x)\sin^2(\pi y) - \sin^2(\pi x)\cos^2(\pi y))$. The exact solution is $u = \exp(-2\pi^2 t)(2 + \sin(\pi x)\sin(\pi y))$.

Now, we investigate the performance of the finite volume scheme for Example 5.3 on some typical distorted quadrilateral meshes, including the trapezoidal mesh, the sine mesh, the Shestakov mesh and the Kershaw mesh. Examples of these typical meshes together with the domain decomposition are shown in Fig. 8(a)–(d), respectively. In the parallel experiments for this nonlinear example, the interface conditions are updated after each nonlinear iteration. So in the nonlinear case, the iterative number $s$ combines effects of the nonlinear iteration and the iteration of the domain decomposition algorithm.

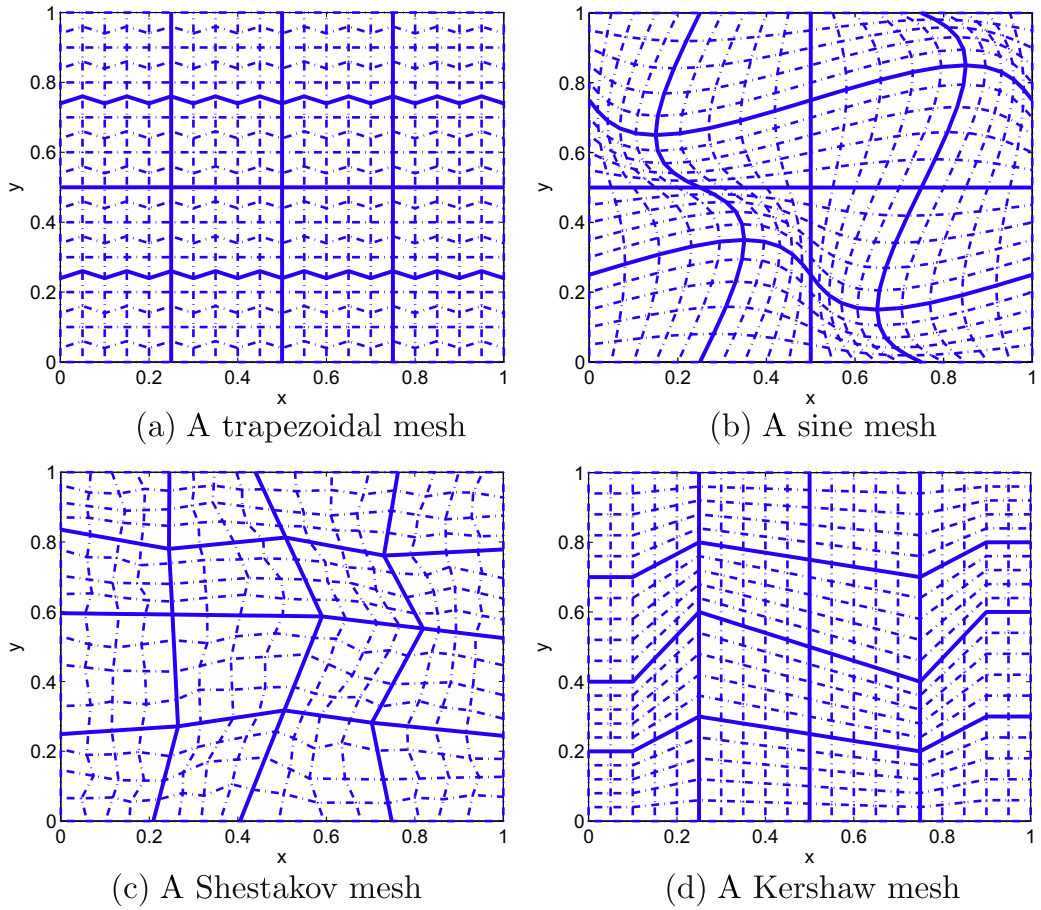



**Fig. 8.** Four typical distorted meshes with 4 * 4 subdomains.

**Table 9**
Numerical results on the trapezoidal mesh.

| Mesh | $10 * 10$ | $20 * 20$ | $40 * 40$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| CPUs | $2 * 2$ | $4 * 4$ | $8 * 8$ | $12 * 12$ | |
| $\lambda$ | 50.0 | 100.0 | 200.0 | 300.0 | |
| $s$ | 7.9746 | 11.5745 | 12.5212 | 13.1619 | |
| $e_c$ | 7.5339E−3 | 2.3076E−3 | 6.2896E−4 | 2.8735E−4 | 1.84 |
| $e_i$ | 1.0102E−2 | 2.7858E−3 | 7.2783E−4 | 3.2804E−4 | 1.92 |
| $e_j$ | 1.0804E−2 | 2.7630E−3 | 7.0011E−4 | 3.1263E−4 | 1.98 |
| $e_{L_2}$ | 3.6554E− | 9.2186E−4 | 2.3079E−4 | 1.0262E−4 | 2.00 |

**Table 10**
Numerical results on the sine mesh.

| Mesh | $10 * 10$ | $20 * 20$ | $40 * 40$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| CPUs | $2 * 2$ | $4 * 4$ | $8 * 8$ | $12 * 12$ | |
| $\lambda$ | 50.0 | 100.0 | 200.0 | 300.0 | |
| $s$ | 10.0578 | 12.0778 | 15.0754 | 16.9986 | |
| $e_c$ | 1.1794E−2 | 3.9348E−3 | 1.1548E−3 | 5.3888E−4 | 1.74 |
| $e_i$ | 2.3899E−2 | 7.6112E− | 2.1642E−3 | 9.9126E−4 | 1.80 |
| $e_j$ | 2.3899E−2 | 7.6111E−3 | 2.1639E−3 | 9.9099E−4 | 1.80 |
| $e_{L_2}$ | 5.0811E−3 | 1.3958E−3 | 3.6016E−4 | 1.6111E−4 | 1.93 |

**Table 11**
Numerical results on the Shestakov mesh.

| Mesh | $8 * 8$ | $16 * 16$ | $32 * 32$ | $64 * 64$ | Rate |
|---|---|---|---|---|---|
| CPUs | $2 * 2$ | $4 * 4$ | $8 * 8$ | $16 * 16$ | |
| $\lambda$ | 40.0 | 80.0 | 160.0 | 320.0 | |
| $s$ | 8.4755 | 11.3688 | 15.3630 | 19.7516 | |
| $e_c$ | 1.9926E−2 | 7.0053E−3 | 2.2056E−3 | 6.0064E−4 | 1.69 |
| $e_i$ | 1.9795E−2 | 6.8708E−3 | 2.0855E−3 | 7.1647E−4 | 1.60 |
| $e_j$ | 1.6536E−2 | 5.5557E−3 | 1.8505E−3 | 6.9889E−4 | 1.52 |
| $e_{L_2}$ | 6.8811E−3 | 1.9148E− | 5.1242E− | 1.3745E−4 | 1.88 |

**Table 12**
Numerical results on the Kershaw mesh.

| Mesh | $10 * 10$ | $20 * 20$ | $40 * 40$ | $60 * 60$ | Rate |
|---|---|---|---|---|---|
| CPUs | $2 * 2$ | $4 * 4$ | $8 * 8$ | $12 * 12$ | |
| $\lambda$ | 50.0 | 100.0 | 200.0 | 300.0 | |
| $s$ | 8.9995 | 12.0632 | 15.8855 | 19.5776 | |
| $e_c$ | 1.0980E−2 | 3.4808E−3 | 1.5726E−3 | 7.2127E−4 | 1.58 |
| $e_i$ | 1.0311E−2 | 2.4488E−3 | 9.4742E−4 | 3.7670E−4 | 1.90 |
| $e_j$ | 2.5099E−2 | 5.4083E−3 | 2.1299E−3 | 9.6244E−4 | 1.84 |
| $e_{L_2}$ | 5.3822E−3 | 1.2649E−3 | 4.0678E−4 | 1.7113E−4 | 1.96 |

Some experiences have been done to investigate the optimal value of $\lambda$ on those four typical distorted meshes. Results are similar with those in Fig. 6 and show that the optimal value of $\lambda$ is independent of mesh types and interface shapes. Numerical results with optimal values $\lambda(\lambda \approx 5/h$ for this nonlinear example) relevant to a sequence of test meshes are presented in Tables 9–12, where $erb = 1.0e{-}8$, $\tau = 1.0e{-}6$, $T = 0.01$. Obviously, our parallel finite volume scheme on those distorted meshes has a good accuracy, and the iterative number $s$ does not acutely increase as meshes refining and the number of processors enhancing.

## 5. Concluding remarks

We have obtained a finite volume scheme through a cell functional minimization approach and derived its parallel algorithm based on nonoverlapping domain decomposition. Unlike the mixed finite element methods or the MFD method, here the edge intensity unknowns are not defined through the hybridization procedure, instead they are introduced simultaneously with the cell centered unknowns and edge flux unknowns. The relations of these unknowns are then established

by minimizing certain cell functional. Moreover, we find that cell edge unknowns play important roles in the construction of the parallel algorithm of the finite volume scheme. They not only prevent the loss of accuracy on the interface due to interpolation but also make it convenient to handle all kinds of interface conditions of subdomains. The Robin condition used here could speed up convergence if $\lambda$ is properly chosen. Our numerical experiments show that the optimal value of $\lambda$ seems insensitive to the mesh distortion, interface shape and location, but relevant to the mesh size. Numerical results demonstrate the good performance of the finite volume scheme and its parallel version. Both of them have almost second order accuracy on many typical distorted quadrilateral meshes. Other future works include investigating the influence of the location of the cell center, comparing multiplicate interface conditions, extending them to the nonconforming interface cases, computing practical multi-material physical problem.

## Acknowledgments

The authors thank our colleagues Z. Dai, Z. Gao, T. Gu and L. Chang for useful discussions and the anonymous referees for valuable suggestions and comments.

## References

[1] I. Aavatsmark, T. Barkve, Ø. Bøe, T. Mannseth, Discretization on non-orthogonal, curvilinear grids for multi-phase flow, in: Proceedings of the Fourth European Conference on the Mathematics of Oil Recovery, Røros, Norway, vol. D, 17, 1994.
[2] I. Aavatsmark, T. Barkve, Ø. Bøe, T. Mannseth, Discretization on non-orthogonal, quadrilateral grids for inhomogeneous, anisotropic media, J. Comput. Phys. 127 (1996) 2–14.
[3] Y. Achdou, C. Japhet, Y. Maday, F. Nataf, A new cement to glue non-conforming grids with Robin interface condition: The finite volume case, Numer. Math. 92 (2002) 593–620.
[4] M.M. Basko, J. Maruhn, An. Tauschwitz, An efficient cell-centered diffusion scheme for quadrilateral grids, J. Comput. Phys. 228 (2009) 2175–2193.
[5] M. Berndt, K. Lipnikov, M. Shashkov, M.F. Wheeler, I. Yotov, Superconvergence of the velocity in mimetic finite difference methods on quadrilaterals, SIAM J. Numer. Anal. 43 (2005) 1728–1749.
[6] M. Berndt, K. Lipnikov, M. Shashkov, M.F. Wheeler, I. Yotov, A mortar mimetic finite difference method on non-matching grids, Numer. Math. 102 (2005) 203–230.
[7] F. Boyer, F. Hubert, Finite volume method for 2D linear and nonlinear elliptic problems with discontinuities, SIAM J. Numer. Anal. 46 (2008) 3032–3070.
[8] J. Breil, P.-H. Maire, A cell-centered diffusion scheme on two-dimensional unstructured meshes, J. Comput. Phys. 224 (2007) 785–823.
[9] F. Brezzi, K. Lipnikov, M. Shashkov, Convergence of the mimetic finite difference method for diffusion problems on polyhedral meshes, SIAM J. Numer. Anal. 43 (2005) 1872–1896.
[10] F. Brezzi, K. Lipnikov, V. Simoncini, A family of mimetic finite difference methods on polygonal and polyhedral meshes, Math. Models Methods. Appl. Sci. 15 (2005) 1533–1551.
[11] F. Brezzi, K. Lipnikov, M. Shashkov, V. Simoncini, A new discretization methodology for diffusion problems on generalized polyhedral meshes, Comput. Methods Appl. Mech. Eng. 196 (2007) 3682–3692.
[12] R. Cautres, R. Herbin, F. Hubert, The Lions domain decomposition algorithm on non-matching cell-centered finite volume meshes, IMA J. Numer. Anal. 24 (2004) 465–490.
[13] G. Chen, D. Li, Z. Wan, Discrete scheme based on variation principle for three dimensional diffusion equation, Chinese J. Comput. Phys. 20 (2003) 291–297.
[14] G. Chen, Y. Zhang, Difference scheme based on variational principle for two dimensional thermal conduction equation, Chinese J. Comput. Phys. 19 (2002) 299–304.
[15] Q.-Y. Chen, J. Wan, Y. Yang, R.T. Mifflin, Enriched multi-point flux approximation for general grids, J. Comput. Phys. 227 (2008) 1701–1721.
[16] K. Domelevo, P. Omnes, A finite volume method for the Laplace equation on almost arbitrary two-dimensional grids, ESAIM: M2AN 39 (2005) 1203–1249.
[17] J. Droniou, R. Eymard, T. Gallouët, R. Herbin, A unified approach to mimetic finite difference, hybrid finite volume and mixed finite volume methods, Math. Models Methods. Appl. Sci. 20 (2010) 265–295.
[18] M.G. Edwards, C.F. Rogers, A flux continuous scheme for the full tensor pressure equation, in: Proceedings of the Fourth European Conference on the Mathematics of Oil Recovery, Røros, Norway, vol. D, 15, 1994.
[19] M.G. Edwards, H. Zheng, Double-families of quasi-positive Darcy-flux approximations with highly anisotropic tensors on structured and unstructured grids, J. Comput. Phys. 229 (2010) 594–625.
[20] J. Hyman, J. Morel, M. Shashkov, S. Steinberg, Mimetic finite difference methods for diffusion equations, Comput. Geosci. 6 (2002) 333–352.
[21] R.A. Klausen, R. Winther, Robust convergence of multi point flux approximation on rough grids, Numer. Math. 104 (2006) 317–337.
[22] D. Li, H. Shui, M. Tang, On the finite difference scheme of two-dimensional parabolic equation in a non-rectangular mesh, J. Numer. Methods Comput. Appl. 1 (1980) 217–224.
[23] D. Li, G. Chen, Introduction to Difference Methods for Parabolic Equation, Science Press, Beijing, 1995. Chinese.
[24] K. Lipnikov, J. Morel, M. Shashkov, Mimetic finite difference methods for diffusion equations on non-orthogonal non-conformal meshes, J. Comput. Phys. 199 (2004) 589–597.
[25] K. Lipnikov, M. Shashkov, I. Yotov, Local flux mimetic finite difference methods, Numer. Math. 112 (2009) 115–152.
[26] P.L. Lions, On the Schwarz alternating method III: a variant for nonoverlapping subdomains, in: Third International Symposium on Domain Decomposition Methods for Partial Differential Equations, SIAM, Philadelphia, 1989, pp. 202–203.
[27] S.H. Lui, A Lions non-overlapping domain decomposition method for domains with an arbitrary interface, IMA J. Numer. Anal. 29 (2009) 332–349.
[28] J.E. Morel, R.M. Roberts, M.J. Shashkov, A local support-operators diffusion discretization scheme for quadrilateral r-z meshes, J. Comput. Phys. 144 (1998) 17–51.
[29] J.E. Roberts, J.-M. Thomas, Mixed and hybrid finite element methods, Handb. Numer. Anal. (1987).
[30] Y. Saad, Iterative Method for Sparse Linear Systems, PWS publishing, New York, 1996.
[31] L. Saas, I. Faille, F. Nataf, F. Willien, Finite volume methods for domain decomposition on nonmatching grids with arbitrary interface conditions, SIAM J. Numer. Anal. 43 (2005) 860–890.
[32] Z. Sheng, G. Yuan, A nine point scheme for the approximation of diffusion operators on distorted quadrilateral meshes, SIAM J. Sci. Comput. 30 (2008) 1341–1361.
[33] J. Wu, Z. Dai, S. Fu, A difference scheme with high resolution based on variation principle for 2D diffusion equation, Internal Report, Institute of Applied Physics and Computational Mathematics, Beijing, China, 2003.

[34] J. Wu, S. Fu, L. Shen, A differences scheme with high resolution for the numerical solution of nonlinear diffusion equation, J. Numer. Methods Comput. Appl. 24 (2003) 116–128. in Chinese.
[35] J. Wu, Z. Dai, Z. Gao, G. Yuan, Linearity preserving nine-point schemes for diffusion equation on distorted quadrilateral meshes, J. Comput. Phys. 229 (2010) 3382–3401.