SCIENCE CHINA
Physics, Mechanics & Astronomy

•Article•

April 2019 Vol. 62 No. 4: 040711
https://doi.org/10.1007/s11433-018-9299-8

The dimension splitting element-free Galerkin method for 3D transient heat conduction problems

ZhiJuan Meng[1,2], Heng Cheng[3], LiDong Ma[4], and YuMin Cheng[1*]

[1] *Shanghai Institute of Applied Mathematics and Mechanics, Shanghai Key Laboratory of Mechanics in Energy Engineering, Shanghai University, Shanghai 200072, China;*
[2] *School of Applied Science, Taiyuan University of Science and Technology, Taiyuan 030024, China;*
[3] *Department of Civil Engineering, Shanghai University, Shanghai 200444, China;*
[4] *School of Materials Science and Engineering, Taiyuan University of Science and Technology, Taiyuan 030024, China*

Received July 19, 2018; accepted September 5, 2018; published online November 19, 2018

By transforming a 3D problem into some related 2D problems, the dimension splitting element-free Galerkin (DSEFG) method is proposed to solve 3D transient heat conduction problems. The improved element-free Galerkin (IEFG) method is used for 2D transient heat conduction problems, and the finite difference method is applied in the splitting direction. The discretized system equation is obtained based on the Galerkin weak form of 2D problem; the essential boundary conditions are imposed with the penalty method; and the finite difference method is employed in the time domain. Four exemplary problems are chosen to verify the efficiency of the DSEFG method. The numerical solutions show that the efficiency and precision of the DSEFG method are greater than ones of the IEFG method for 3D problems.

improved element-free Galerkin (IEFG) method, dimension splitting method, finite difference method, dimension splitting element-free Galerkin (DSEFG) method, transient heat conduction problem

PACS number(s): 02.06.-x, 02.70.-c, 02.03.Jr, 44.10.+i

Citation: Z. J. Meng, H. Cheng, L. D. Ma, and Y. M. Cheng, The dimension splitting element-free Galerkin method for 3D transient heat conduction problems, Sci. China-Phys. Mech. Astron. **62**, 040711 (2019), https://doi.org/10.1007/s11433-018-9299-8

1 Introduction

Heat conduction problems exist widely in many engineering fields. However, only for linear problems with simple boundary conditions and geometrics can analytical solutions be obtained [1,2]. Thus, many researchers have developed effective numerical methods, such as finite element method (FEM) [3], boundary element method (BEM) [4,5], and meshless methods [6-9], to solve these problems.

The FEM and BEM are popular for obtaining numerical solutions of a lot of science and engineering problems [10,11]. However, these methods rely strongly on the mesh of the problem domain, so a remeshing technique must be used to obtain the numerical results with high precision for crack growth and extremely large deformation problems.

In meshless methods, the shape function only relies on the node distribution in the problem domain, meaning that many complicated problems can be solved with great computational precision without the remeshing technique [12-14].

3D heat conduction problems can be effectively solved with the improved element-free Galerkin (IEFG) method [15]. Because the weighted orthogonal functions are used, fewer coefficients are included in IEFG method than in the original EFG method. The computational speed of IEFG

*Corresponding author (email: ymcheng@shu.edu.cn)

method is higher—up to 30% less CPU time; moreover, other engineering and science problems can also be solved using the IEFG method. Zhang et al. [16,17] applied it to solve 3D potential problems and 3D wave equations. Peng et al. [18] applied it to solve 3D viscoelasticity problems. However, the computational efficiency of the IEFG method is still much lower than that of the FEM. Therefore, the IEFG method must be further improved to obtain higher computational efficiency.

Li and Shen [19] first present a dimension splitting method (DSM). The key point of the DSM is to divide a 3D problem into many related 2D ones. Li et al. [20-22] applied the DSM to a 3D linearly elastic shell and to 3D compressible, rotating, and incompressible Navier-Stokes equations. Hansen and Ostermann [23,24] solved quasilinear and evolution parabolic equations with the DSM. In fact, the DSM is a convenient and efficient numerical method for various equations [25-28]. It is obvious that the DSM can improve computational efficiency greatly.

In this paper, by introducing the dimension splitting method, a 3D transient heat conduction problem is transformed into some related 2D problems; then, a DSEFG method for 3D transient heat conduction problems is presented. The IEFG method is used for these 2D problems, and the finite difference method is applied in the splitting direction. The discretized system equation is obtained based on the Galerkin weak form of the 2D transient heat conduction problem, the essential boundary conditions are imposed with a penalty method, and the finite difference method is employed in the time domain. Four exemplary problems are chosen to verify the efficiency of the DSEFG method, and the numerical solutions show that the efficiency and precision of the DSEFG method are greater than those of the IEFG method.

2 Dimension splitting procedure of 3D heat conduction problems

Consider the 3D transient heat equation:

$$\rho c\cdot\frac{\partial T}{\partial t}-k\nabla^2 T-Q(\mathbf{x},t)=0,\quad (\mathbf{x}\in\Omega), \tag{1}$$

with the boundary conditions:

$$T-\overline{T}=0,\quad (\mathbf{x}\in\Gamma_1), \tag{2}$$

$$\mathbf{n}\cdot k\nabla T-\overline{q}=0,\quad (\mathbf{x}\in\Gamma_2), \tag{3}$$

and

$$\mathbf{n}\cdot k\nabla T-h(T_a-T_\infty)=0,\quad (\mathbf{x}\in\Gamma_3), \tag{4}$$

and initial condition:

$$T|_{t=0}=T_0, \tag{5}$$

where T is temperature; ρ is density; t is time; c is heat capacity; k is thermal conductivity; $Q(\mathbf{x}, t)$ is the ratio of heat per unit volume per unit time; Γ is the boundary of the problem domain Ω, given by $\Gamma=\Gamma_1\cup\Gamma_2\cup\Gamma_3$, where Γ_1, Γ_2, and Γ_3 are the boundary with known temperature $\overline{T}$, flux $\overline{q}$, and heat transfer coefficient h, respectively; $T_a=T_a(\mathbf{x}, t)$ is medium temperature; T_∞ is the temperature of the surrounding fluid; T_0 is initial temperature, and $\mathbf{n}$ is the unit vector outward normal to Γ.

Assume that domain Ω is divided into L planes along direction x_3. The distance between adjacent planes is Δx_3. Then, L+1 2D sub-domains $\Omega^{(k)}$, (k=0,1, …, L) are obtained as:

$$\Omega=\bigcup_{k=0}^{L-1}\Omega^{(k)}\times[x_3^{(k)},x_3^{(k+1)}]\cup\Omega^{(L)}, \tag{6}$$

where

$$a=x_3^{(0)}<x_3^{(1)}\cdots<x_3^{(L)}=c,\ \ x_3\in[a,c], \tag{7}$$

$$\Delta x_3=x_3^{(k+1)}-x_3^{(k)}=(c-a)/L. \tag{8}$$

For a fixed $x_3^{(k)}$ and time t, a 3D transient heat conduction problem is transformed into several related 2D problems, i.e.,

$$k\left(\frac{\partial^2 T}{\partial x_1^2}+\frac{\partial^2 T}{\partial x_2^2}\right)=\rho c\cdot\frac{\partial T}{\partial t}-k\frac{\partial^2 T}{\partial x_3^2}-Q(\mathbf{x},t),\quad \mathbf{x}=(x_1,x_2,x_3^{(k)}),\ \ (x_1,x_2)\in\Omega^{(k)}, \tag{9}$$

and the corresponding boundary conditions are

$$T=\overline{T}(\mathbf{x},t),\ \mathbf{x}=(x_1,x_2,x_3^{(k)}),\ \ (x_1,x_2)\in\Gamma_1^{(k)}, \tag{10}$$

$$\mathbf{n}\cdot k\nabla T=\overline{q}(\mathbf{x},t),\ \mathbf{x}=(x_1,x_2,x_3^{(k)}),\ \ (x_1,x_2)\in\Gamma_2^{(k)}, \tag{11}$$

$$\mathbf{n}\cdot k\nabla T=h(T_a(\mathbf{x},t)-T_\infty(\mathbf{x},t)),\quad \mathbf{x}=(x_1,x_2,x_3^{(k)}),\ \ (x_1,x_2)\in\Gamma_3^{(k)}. \tag{12}$$

For a fixed time t, $T=T(x_1,x_2,x_3^{(k)},t)$ is the temperature in the sub-domain $\Omega^{(k)}$ with boundary $\Gamma^{(k)}$, and $\Gamma^{(k)}=\Gamma_1^{(k)}\cup\Gamma_2^{(k)}\cup\Gamma_3^{(k)}$.

Eqs. (9)-(12) can be solved with the IEFG method in the sub-domain $\Omega^{(k)}$. The finite difference method is applied in the splitting direction x_3 and in the time domain. Then, the numerical solutions of eqs. (1)-(5) can be successfully obtained.

Eqs. (9)-(11) are equivalent to the following functional:

$$\begin{aligned}\Pi=&\frac{1}{2}\int_{\Omega^{(k)}}\nabla^{\mathrm{T}}T\left[k\left(\left(\frac{\partial T}{\partial x_1}\right)^2+\left(\frac{\partial T}{\partial x_2}\right)^2\right)\right]\mathrm{d}\Omega^{(k)}\\&+\int_{\Omega^{(k)}}\left[T\left(\rho c\frac{\partial T}{\partial t}-k\frac{\partial^2 T}{\partial x_3^2}-Q\right)\right]\mathrm{d}\Omega^{(k)}\\&-\int_{\Gamma_2^{(k)}}T\overline{q}\,\mathrm{d}\Gamma^{(k)}-\int_{\Gamma_3^{(k)}}h\left(\frac{T^2}{2}-TT_\infty\right)\mathrm{d}\Gamma^{(k)}.\end{aligned} \tag{13}$$

Selecting the penalty method to apply essential boundary conditions, we obtain

$$\begin{aligned}\Pi^* = &\frac{1}{2}\int_{\Omega^{(k)}} \nabla^{\mathrm{T}} T\left[k\left(\left(\frac{\partial T}{\partial x_1}\right)^2+\left(\frac{\partial T}{\partial x_2}\right)^2\right)\right]\mathrm{d}\Omega^{(k)}\\ &+\int_{\Omega^{(k)}}\left[T\left(\rho c\frac{\partial T}{\partial t}-k\frac{\partial^2 T}{\partial x_3^2}-Q\right)\right]\mathrm{d}\Omega^{(k)}\\ &-\int_{\Gamma_2^{(k)}} T\overline{q}\,\mathrm{d}\Gamma^{(k)}-\int_{\Gamma_3^{(k)}} h\left(\frac{T^2}{2}-TT_\infty\right)\mathrm{d}\Gamma^{(k)}\\ &+\frac{1}{2}\int_{\Gamma_1^{(k)}}(T-\overline{T})\,\alpha\,(T-\overline{T})\,\mathrm{d}\Gamma^{(k)},\end{aligned} \tag{14}$$

where α is the penalty factor.

The variation of eq. (14) is

$$\begin{aligned}\delta\Pi^* = &\int_{\Omega^{(k)}} \nabla^{\mathrm{T}} T\left[k\delta\left(\left(\frac{\partial T}{\partial x_1}\right)^2+\left(\frac{\partial T}{\partial x_2}\right)^2\right)\right]\mathrm{d}\Omega^{(k)}\\ &+\int_{\Omega^{(k)}}\left[\delta T\left(\rho c\frac{\partial T}{\partial t}-k\frac{\partial^2 T}{\partial x_3^2}-Q\right)\right]\mathrm{d}\Omega^{(k)}\\ &-\int_{\Gamma_2^{(k)}}\delta T\overline{q}\,\mathrm{d}\Gamma^{(k)}-\int_{\Gamma_3^{(k)}}\delta T\cdot h\,(T_a-T_\infty)\mathrm{d}\Gamma^{(k)}\\ &+\int_{\Gamma_1^{(k)}}\delta T\cdot\alpha(T-\overline{T})\,\mathrm{d}\Gamma^{(k)}\\ &=0.\end{aligned} \tag{15}$$

3 The DSEFG method for 3D transient heat conduction problems

3.1 IMLS approximation

In 2D sub-domain $\Omega^{(k)}$, M nodes $\mathbf{x}_I^{(k)}$, I=1,2,…,M, are distributed. The temperature, $T(\mathbf{x}, t)$, at point $\mathbf{x}_I^{(k)}$ at time t is represented as:

$$T_I(x_3^{(k)},t)=T(\mathbf{x}_I^{(k)},x_3^{(k)},t), \tag{16}$$

where $T(\mathbf{x}^{(k)},x_3^{(k)},t)$ at any point $\mathbf{x}^{(k)}$=(x_1, x_2) is related to the nodes $\mathbf{x}_I^{(k)}$, (I=1,2,…,n), whose domains of influence cover the point $\mathbf{x}^{(k)}$.

In this paper, the IMLS approximation is selected to obtain shape functions. In the 2D sub-domain $\Omega^{(k)}$, $T^h(\mathbf{x}^{(k)},x_3^{(k)},t)$ is the approximation of $T(\mathbf{x}, t)$ at point $\mathbf{x}^{(k)}=(x_1,x_2)$, and the trial function is

$$\begin{aligned}T^h(\mathbf{x}^{(k)},x_3^{(k)},t)&=\sum_{i=1}^{m}p_i(\mathbf{x}^{(k)})a_i(\mathbf{x}^{(k)})\\ &=\mathbf{P}^T(\mathbf{x}^{(k)})\mathbf{a}(\mathbf{x}^{(k)}),\\ &\mathbf{x}^{(k)}\in\Omega^{(k)},\end{aligned} \tag{17}$$

where $p_i(\mathbf{x}^{(k)})$ is a basis function, m is the number of basis functions, $\mathbf{P}^{\mathrm{T}}(\mathbf{x}^{(k)})$ is the vector of basis functions, and $\mathbf{a}(\mathbf{x}^{(k)})$ is the vector of the coefficient $a_i(x^{(k)})$.

For the 2D sub-domain $\Omega^{(k)}$, the basis function can be chosen as:

Linear basis

$$\mathbf{P}^{\mathrm{T}}=(1,x_1,x_2),\quad (m=3), \tag{18}$$

or

Quadratic basis

$$\mathbf{P}^{\mathrm{T}}=(1,x_1,x_2,x_1^2,x_1x_2,x_2^2),\quad (m=6). \tag{19}$$

A weighted least-squares method is applied to obtain the coefficients $a_i(\mathbf{x}^{(k)})$ in eq. (17). The difference between the functions $T(\mathbf{x}^{(k)},x_3^{(k)},t)$ and $T^h(\mathbf{x}^{(k)},x_3^{(k)},t)$ must be minimized.

Define

$$\begin{aligned}J&=\sum_{I=1}^{n}w(\mathbf{x}^{(k)}-\mathbf{x}_I^{(k)})\left[T^h(\mathbf{x}_I^{(k)},x_3^{(k)},t)-T_I\right]^2\\ &=\sum_{I=1}^{n}w(\mathbf{x}^{(k)}-\mathbf{x}_I^{(k)})\left[\sum_{i=1}^{m}p_i(\mathbf{x}^{(k)})\,a_i(\mathbf{x}^{(k)})-T_I\right]^2,\end{aligned} \tag{20}$$

where $\mathbf{x}_I^{(k)}$, (I=1,2,…,n), are nodes that influence domains covering the point $\mathbf{x}^{(k)}=(x_1,x_2)$, $w\left(\mathbf{x}^{(k)}-\mathbf{x}_I^{(k)}\right)$ is the weight function, and $T_I=T(\mathbf{x}_I^{(k)},x_3^{(k)},t)$.

The matrix form of eq. (20) is

$$J=(\mathbf{Pa}-\mathbf{T})^{\mathrm{T}}\mathbf{W}(\mathbf{Pa}-\mathbf{T}), \tag{21}$$

where

$$\mathbf{T}^{\mathrm{T}}=(T_1,T_2,\ldots,T_n), \tag{22}$$

$$\mathbf{P}=\begin{bmatrix}p_1(\mathbf{x}_1^{(k)}) & p_2(\mathbf{x}_1^{(k)}) & \cdots & p_m(\mathbf{x}_1^{(k)})\\ p_1(\mathbf{x}_2^{(k)}) & p_2(\mathbf{x}_2^{(k)}) & \cdots & p_m(\mathbf{x}_2^{(k)})\\ \vdots & \vdots & \ddots & \vdots\\ p_1(\mathbf{x}_n^{(k)}) & p_2(\mathbf{x}_n^{(k)}) & \cdots & p_m(\mathbf{x}_n^{(k)})\end{bmatrix}, \tag{23}$$

and

$$\mathbf{W}=\begin{bmatrix}w(\mathbf{x}^{(k)}-\mathbf{x}_1^{(k)}) & 0 & \cdots & 0\\ 0 & w(\mathbf{x}^{(k)}-\mathbf{x}_2^{(k)}) & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & w(\mathbf{x}^{(k)}-\mathbf{x}_n^{(k)})\end{bmatrix}. \tag{24}$$

To obtain $\mathbf{a}(\mathbf{x}^{(k)})$, the minimization condition requires that

$$\frac{\partial J}{\partial\mathbf{a}}=\overline{\mathbf{A}}(\mathbf{x}^{(k)})\mathbf{a}(\mathbf{x}^{(k)})-\overline{\mathbf{B}}(\mathbf{x}^{(k)})\mathbf{T}=0, \tag{25}$$

where

$$\overline{\mathbf{A}}(\mathbf{x}^{(k)})=\mathbf{P}^{\mathrm{T}}\mathbf{WP}, \tag{26}$$

$$\overline{\mathbf{B}}(\mathbf{x}^{(k)})=\mathbf{P}^{\mathrm{T}}\mathbf{W}. \tag{27}$$

From eq. (25), we have

$$\mathbf{a}(\mathbf{x}^{(k)})=\overline{\mathbf{A}}^{-1}(\mathbf{x}^{(k)})\overline{\mathbf{B}}(\mathbf{x}^{(k)})\mathbf{T}. \tag{28}$$

Thus, the local approximation is

$$T^h(\mathbf{x}^{(k)},x_3^{(k)},t)=\sum_{I=1}^{n}\Phi_I(\mathbf{x}^{(k)})T_I=\mathbf{\Phi}(\mathbf{x}^{(k)})\mathbf{T}, \tag{29}$$

where the shape function is

$$\begin{aligned}\mathbf{\Phi}(\mathbf{x}^{(k)})&=(\Phi_1(\mathbf{x}^{(k)}),\Phi_2(\mathbf{x}^{(k)}),...,\Phi_n(\mathbf{x}^{(k)}))\\&=\mathbf{P}^T(\mathbf{x}^{(k)})\overline{\mathbf{A}}^{-1}(\mathbf{x}^{(k)})\overline{\mathbf{B}}(\mathbf{x}^{(k)}).\end{aligned} \tag{30}$$

For $\forall f(\mathbf{x}),g(\mathbf{x})\in \mathrm{span}(p_1,p_2,\cdots,p_m)$, let us define

$$(f,g)=\sum_{I=1}^{n}w(\mathbf{x}-\mathbf{x}_I)f(\mathbf{x}_I)g(\mathbf{x}_I). \tag{31}$$

In span(p_1, p_2, …, p_m), if the set of functions $p_1(\mathbf{x}),p_2(\mathbf{x}),\dots,p_m(\mathbf{x})$ and the weight function $\{w_i\}$ satisfy

$$\begin{aligned}(p_k,p_j)&=\sum_{i=1}^{n}w_ip_k(x_i)p_j(x_i)\\&=\begin{cases}0, & k\neq j,\\ A_k, & k=j,\end{cases}\\&\quad(k,j=1,2,\dots,m).\end{aligned} \tag{32}$$

This is called a weighted orthogonal function set about points $\{x_i\}$.

From eq. (31), eq. (25) is written in detail as:

$$\begin{bmatrix}(p_1,p_1) & (p_1,p_2) & \cdots & (p_1,p_m)\\ (p_2,p_1) & (p_2,p_2) & \cdots & (p_2,p_m)\\ \vdots & \vdots & \ddots & \vdots\\ (p_m,p_1) & (p_m,p_2) & \cdots & (p_m,p_m)\end{bmatrix}\begin{bmatrix}a_1(\mathbf{x}^{(k)})\\ a_2(\mathbf{x}^{(k)})\\ \vdots\\ a_m(\mathbf{x}^{(k)})\end{bmatrix}$$

$$=\begin{bmatrix}(p_1,T_I)\\ (p_2,T_I)\\ \vdots\\ (p_m,T_I)\end{bmatrix}. \tag{33}$$

If the function set $\left\{p_i(\mathbf{x}^{(k)})\right\}$, $(i=1,2,\dots,m)$, is weighted orthogonal about points $\{x_i\}$, i.e.,

$$(p_i,p_j)=0,\quad i\neq j, \tag{34}$$

then eq. (33) becomes

$$\begin{bmatrix}(p_1,p_1) & 0 & \cdots & 0\\ 0 & (p_2,p_2) & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & (p_m,p_m)\end{bmatrix}\begin{bmatrix}a_1(\mathbf{x}^{(k)})\\ a_2(\mathbf{x}^{(k)})\\ \vdots\\ a_m(\mathbf{x}^{(k)})\end{bmatrix}$$

$$=\begin{bmatrix}(p_1,T_I)\\ (p_2,T_I)\\ \vdots\\ (p_m,T_I)\end{bmatrix}. \tag{35}$$

Then we can directly obtain

$$a_i(\mathbf{x}^{(k)})=\frac{(p_i,T_I)}{(p_i,p_i)},\ i=1,2,\dots,m. \tag{36}$$

The corresponding matrix form of eq. (36) is

$$\mathbf{a}(\mathbf{x}^{(k)})=\mathbf{A}^*(\mathbf{x}^{(k)})\overline{\mathbf{B}}(\mathbf{x}^{(k)})\mathbf{T}, \tag{37}$$

where

$$\mathbf{A}^*(\mathbf{x}^{(k)})=\begin{bmatrix}\frac{1}{(p_1,p_1)} & 0 & \cdots & 0\\ 0 & \frac{1}{(p_2,p_2)} & \cdots & 0\\ \vdots & \vdots & \ddots & \vdots\\ 0 & 0 & \cdots & \frac{1}{(p_m,p_m)}\end{bmatrix}. \tag{38}$$

Substituting eq. (36) into eq. (17) yields

$$\begin{aligned}T^h(\mathbf{x}^{(k)},x_3^{(k)},t)&=\sum_{I=1}^{n}\Phi_I^*(\mathbf{x}^{(k)})T_I(\mathbf{x}^{(k)},x_3^{(k)},t)\\&=\mathbf{\Phi}^*(\mathbf{x}^{(k)})\mathbf{T},\end{aligned} \tag{39}$$

where the shape function is

$$\begin{aligned}\mathbf{\Phi}^*(\mathbf{x}^{(k)})&=(\Phi_1^*(\mathbf{x}^{(k)}),\Phi_2^*(\mathbf{x}^{(k)}),\dots,\Phi_n^*(\mathbf{x}^{(k)}))\\&=\mathbf{P}^{\mathrm{T}}(\mathbf{x}^{(k)})\mathbf{A}^*(\mathbf{x}^{(k)})\overline{\mathbf{B}}(\mathbf{x}^{(k)}).\end{aligned} \tag{40}$$

This is IMLS approximation for 2D problems. In this method, the coefficient $a_i(\mathbf{x}^{(k)})$ can be obtained simply and directly, and singular or ill-conditioned equations can be avoided. Then, the computational efficiency and solution precision of the MLS approximation are improved.

3.2 DSEFG method for 3D heat conduction problems

From eq. (39), we obtain

$$\begin{aligned}\frac{\partial T(\mathbf{x}^{(k)},x_3^{(k)},t)}{\partial t}&=\frac{\partial}{\partial t}\sum_{I=1}^{n}\Phi_I^*(\mathbf{x}^{(k)})T_I(\mathbf{x}^{(k)},x_3^{(k)},t)\\&=\sum_{I=1}^{n}\Phi_I^*(\mathbf{x}^{(k)})\frac{\partial T_I(\mathbf{x}^{(k)},x_3^{(k)},t)}{\partial t}\\&=\sum_{I=1}^{n}\Phi_I^*(\mathbf{x}^{(k)})\dot{T}_I(\mathbf{x}^{(k)},x_3^{(k)},t)\\&=\mathbf{\Phi}^*(\mathbf{x}^{(k)})\dot{\mathbf{T}},\end{aligned} \tag{41}$$

$$\begin{aligned}\frac{\partial^2 T(\mathbf{x}^{(k)},x_3^{(k)},t)}{\partial x_3^2}&=\frac{\partial}{\partial x_3^2}\sum_{I=1}^{n}\Phi_I^*T_I(\mathbf{x}^{(k)},x_3^{(k)},t)\\&=\sum_{I=1}^{n}\Phi_I^*\frac{\partial^2 T_I(\mathbf{x}^{(k)},x_3^{(k)},t)}{\partial x_3^2}\\&=\mathbf{\Phi}^*(\mathbf{x}^{(k)})\mathbf{T}'',\end{aligned} \tag{42}$$

$$\begin{aligned}\mathbf{L}T(\mathbf{x}^{(k)},x_3^{(k)},t)&=\sum_{I=1}^{n}\begin{bmatrix}\frac{\partial}{\partial x_1}\\ \frac{\partial}{\partial x_2}\end{bmatrix}\Phi_I^*(\mathbf{x}^{(k)})T_I(\mathbf{x}^{(k)},x_3^{(k)},t)\\&=\sum_{I=1}^{n}\mathbf{B}_I(\mathbf{x}^{(k)})T_I(\mathbf{x}^{(k)},x_3^{(k)},t)\\&=\mathbf{B}(\mathbf{x}^{(k)})\mathbf{T},\end{aligned} \tag{43}$$

where

$$\dot{\mathbf{T}} = \left(\frac{\partial T_1(\mathbf{x}^{(k)}, x_3^{(k)}, t)}{\partial t}, \frac{\partial T_2(\mathbf{x}^{(k)}, x_3^{(k)}, t)}{\partial t}, \cdots, \frac{\partial T_n(\mathbf{x}^{(k)}, x_3^{(k)}, t)}{\partial t} \right)^{\mathrm{T}}, \tag{44}$$

$$\mathbf{T}'' = \left(\frac{\partial^2 T_1(\mathbf{x}^{(k)}, x_3^{(k)}, t)}{\partial x_3^2}, \frac{\partial^2 T_2(\mathbf{x}^{(k)}, x_3^{(k)}, t)}{\partial x_3^2}, \cdots, \frac{\partial^2 T_n(\mathbf{x}^{(k)}, x_3^{(k)}, t)}{\partial x_3^2} \right)^{\mathrm{T}}, \tag{45}$$

$$\mathbf{B}(\mathbf{x}^{(k)}) = (\mathbf{B}_1(\mathbf{x}^{(k)}), \mathbf{B}_2(\mathbf{x}^{(k)}), \cdots, \mathbf{B}_n(\mathbf{x}^{(k)})), \tag{46}$$

$$\mathbf{B}_I(\mathbf{x}^{(k)}) = \begin{bmatrix} \Phi_{I,1}^*(\mathbf{x}^{(k)}) \\ \Phi_{I,2}^*(\mathbf{x}^{(k)}) \end{bmatrix}. \tag{47}$$

Substituting eqs. (39) and (41)-(45) into eq. (15), we have

$$\begin{aligned} \delta \Pi^* = & \int_{\Omega^{(k)}} \nabla^{\mathrm{T}} (\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot (k\delta \nabla (\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathbf{T}))\mathrm{d}\Omega^{(k)} \\ & + \int_{\Omega^{(k)}} [\delta(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T})\ (\rho c\ \boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\dot{\mathbf{T}} \\ & - k\ \boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathbf{T}'' - Q)]\,\mathrm{d}\Omega^{(k)} \\ & - \int_{\Gamma_2^{(k)}} \delta(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathbf{T})\ \cdot \bar{q}\ \mathrm{d}\Gamma^{(k)} \\ & - \int_{\Gamma_3^{(k)}} \delta(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot h\ (T_\mathrm{a} - T_\infty)\,\mathrm{d}\Gamma^{(k)} \\ & + \int_{\Gamma_1^{(k)}} \delta\ (\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot \alpha\,(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\ \mathbf{T} - \overline{T})\,\mathrm{d}\Gamma^{(k)} \\ & = 0. \end{aligned} \tag{48}$$

The integration terms of eq. (48) are rendered as follows:

$$\begin{aligned} & \int_{\Omega^{(k)}} \nabla^{\mathrm{T}} (\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot (k\delta \nabla (\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}))\,\mathrm{d}\Omega^{(k)} \\ & = k[\int_{\Omega^{(k)}} \delta(\mathbf{B}(\mathbf{x}^{(k)})\,\mathbf{T})^{\mathrm{T}} \mathbf{B}(\mathbf{x}^{(k)})\,\mathrm{d}\Omega^{(k)}] \cdot \mathbf{T} \\ & = \delta\,\mathbf{T}^{\mathrm{T}} \cdot [k \int_{\Omega^{(k)}} \mathbf{B}^{\mathrm{T}}(\mathbf{x}^{(k)})\mathbf{B}(\mathbf{x}^{(k)})\,\mathrm{d}\Omega^{(k)}] \cdot \mathbf{T} \\ & = \delta\,\mathbf{T}^{\mathrm{T}} \cdot \mathbf{K} \cdot \mathbf{T}, \end{aligned} \tag{49}$$

$$\begin{aligned} & \int_{\Omega^{(k)}} [\delta\,(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T})(\rho c\ \boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\dot{\mathbf{T}} \\ & \quad - k\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathbf{T}'' - Q)]\mathrm{d}\Omega^{(k)} \\ & = \delta\mathbf{T}^{\mathrm{T}} \left[\int_{\Omega^{(k)}} \rho c\ \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)}) \cdot \boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathrm{d}\Omega^{(k)} \cdot \dot{\mathbf{T}} \right. \\ & \quad - k \int_{\Omega^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)}) \cdot \boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathrm{d}\Omega^{(k)} \cdot \mathbf{T}'' \\ & \quad \left. - \int_{\Omega^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)}) Q\,\mathrm{d}\Omega^{(k)} \right] \\ & = \delta\mathbf{T}^{\mathrm{T}}[\mathbf{A}\dot{\mathbf{T}} - \mathbf{C}\mathbf{T}'' - \mathbf{f}^{(1)}], \end{aligned} \tag{50}$$

$$\begin{aligned} & \int_{\Gamma_2^{(k)}} \delta\,(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot \bar{q}\,\mathrm{d}\Gamma^{(k)} \\ & = \delta\mathbf{T}^{\mathrm{T}} \int_{\Gamma_2^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)}) \cdot \bar{q}\,\mathrm{d}\Gamma^{(k)} \\ & = \delta\mathbf{T}^{\mathrm{T}} \cdot \mathbf{f}^{(2)}, \end{aligned} \tag{51}$$

$$\begin{aligned} & \int_{\Gamma_3^{(k)}} \delta\,(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot h\ (T_\mathrm{a} - T_\infty)\,\mathrm{d}\Gamma^{(k)} \\ & = \delta\mathbf{T}^{\mathrm{T}} \cdot h \int_{\Gamma_3^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})T_\mathrm{a} - T_\infty)\,\mathrm{d}\Gamma^{(k)} \\ & = \delta\mathbf{T}^{\mathrm{T}}(\mathbf{H} \cdot \mathbf{T} - \mathbf{f}^{(3)}), \end{aligned} \tag{52}$$

$$\begin{aligned} & \int_{\Gamma_1^{(k)}} \delta\ (\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\,\mathbf{T}) \cdot \alpha(\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\ \mathbf{T} - \overline{T})\,\mathrm{d}\Gamma^{(k)} \\ & = \delta\mathbf{T}^{\mathrm{T}} \left[\int_{\Gamma_1^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,\alpha\,\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathrm{d}\Gamma^{(k)} \cdot \mathbf{T} \right. \\ & \quad \left. - \int_{\Gamma_1^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,\alpha \cdot \overline{T}\mathrm{d}\Gamma^{(k)} \right] \\ & = \delta\mathbf{T}^{\mathrm{T}}[\mathbf{K}^{\alpha} \cdot \mathbf{T} - \mathbf{f}^{\alpha}], \end{aligned} \tag{53}$$

where

$$\mathbf{K} = k \int_{\Omega^{(k)}} \mathbf{B}^{\mathrm{T}}(\mathbf{x}^{(k)})\mathbf{B}(\mathbf{x}^{(k)})\,\mathrm{d}\Omega^{(k)}, \tag{54}$$

$$\mathbf{A} = \rho c \int_{\Omega^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\ \mathrm{d}\Omega^{(k)}, \tag{55}$$

$$\mathbf{C} = k \int_{\Omega^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathrm{d}\Omega^{(k)}, \tag{56}$$

$$\mathbf{f}^{(1)} = \int_{\Omega^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,Q\mathrm{d}\Omega^{(k)}, \tag{57}$$

$$\mathbf{f}^{(2)} = \int_{\Gamma_2^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)}) \cdot \bar{q}\,\mathrm{d}\Gamma^{(k)}, \tag{58}$$

$$\mathbf{H} = \int_{\Gamma_3^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)}) \cdot h \cdot \boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathrm{d}\Gamma^{(k)}, \tag{59}$$

$$\mathbf{f}^{(3)} = \int_{\Gamma_3^{(k)}} \boldsymbol{\Phi}^*(\mathbf{x}^{(k)}) \cdot h \cdot T_\infty\,\mathrm{d}\Gamma^{(k)}, \tag{60}$$

$$\mathbf{K}^{\alpha} = \int_{\Gamma_1^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,\alpha\,\boldsymbol{\Phi}^*(\mathbf{x}^{(k)})\mathrm{d}\Gamma^{(k)}, \tag{61}$$

and

$$\mathbf{f}^{\alpha} = \int_{\Gamma_1^{(k)}} \boldsymbol{\Phi}^{*\mathrm{T}}(\mathbf{x}^{(k)})\,\alpha\,\overline{T}\mathrm{d}\Gamma^{(k)}. \tag{62}$$

Then, we obtain

$$\begin{aligned} & \delta\mathbf{T}^{\mathrm{T}}[-\mathbf{C}\mathbf{T}'' + \mathbf{A}\dot{\mathbf{T}} + (\mathbf{K} + \mathbf{H} + \mathbf{K}^{\alpha})\mathbf{T} \\ & \quad - (\mathbf{f}^{(1)} + \mathbf{f}^{(2)} + \mathbf{f}^{(3)} + \mathbf{f}^{\alpha})] = 0. \end{aligned} \tag{63}$$

As $\delta\mathbf{T}^{\mathrm{T}}$ is arbitrary, we obtain

$$-\mathbf{C}\mathbf{T}'' + \mathbf{A}\dot{\mathbf{T}} + \overline{\mathbf{K}}\mathbf{T} = \mathbf{F}, \tag{64}$$

where

$$\overline{\mathbf{K}} = \mathbf{K} + \mathbf{H} + \mathbf{K}^{\alpha}, \tag{65}$$

$$\mathbf{F} = \mathbf{f}^{(1)} + \mathbf{f}^{(2)} + \mathbf{f}^{(3)} + \mathbf{f}^{\alpha}. \tag{66}$$

To solve eq. (64), the problem domain Ω is uniformly divided into L planes by inserting $L-1$ points $x_3^{(1)}, x_3^{(2)}, \cdots, x_3^{(L-1)}$ along the direction x_3. The distance between the adjacent planes is Δx_3. Then, we calculate the temperature at time t of the plane when

$x_3 = x_3^{(1)}, x_3^{(2)}, \ldots, x_3^{(L-1)}$.

Suppose $\mathbf{T}(x_1, x_2, x_3^{(1)}, t)$, $\mathbf{T}(x_1, x_2, x_3^{(2)}, t)$, …, $\mathbf{T}(x_1, x_2, x_3^{(L-1)}, t)$ is the approximate value of the temperature in the plane $x_3 = x_3^{(1)}, x_3^{(2)}, \ldots, x_3^{(L-1)}$, respectively. Let

$$\mathbf{T}(x_1, x_2, x_3^{(0)}, t) = \mathbf{T}_{(t)}^{(0)} = \mathbf{T}(a, t), \tag{67}$$

$$\mathbf{T}(x_1, x_2, x_3^{(1)}, t) = \mathbf{T}_{(t)}^{(1)}, \tag{68}$$

$$\mathbf{T}(x_1, x_2, x_3^{(2)}, t) = \mathbf{T}_{(t)}^{(2)}, \tag{69}$$

…

$$\mathbf{T}(x_1, x_2, x_3^{(L-1)}, t) = \mathbf{T}_{(t)}^{(L-1)}, \tag{70}$$

$$\mathbf{T}(x_1, x_2, x_3^{(L)}, t) = \mathbf{T}_{(t)}^{(L)} = \mathbf{T}(c, t). \tag{71}$$

Using the finite difference method, we obtain

$$\mathbf{T}''^{(k)} \approx \frac{\mathbf{T}_{(t)}^{(k-1)} - 2\mathbf{T}_{(t)}^{(k)} + \mathbf{T}_{(t)}^{(k+1)}}{(\Delta x_3)^2}, \quad (k = 1, 2, \ldots, L-1), \tag{72}$$

$$\dot{\mathbf{T}} \approx \frac{\mathbf{T}_{(t+\Delta t)}^{(k)} - \mathbf{T}_{(t)}^{(k)}}{\Delta t}, \quad (k = 1, 2, \ldots, L-1). \tag{73}$$

Then, from eq. (64) we have

$$-\mathbf{C}\frac{\mathbf{T}_{(t+\Delta t)}^{(0)} - 2\mathbf{T}_{(t+\Delta t)}^{(1)} + \mathbf{T}_{(t+\Delta t)}^{(2)}}{(\Delta x_3)^2} + \mathbf{A}\frac{\mathbf{T}_{(t+\Delta t)}^{(1)} - \mathbf{T}_{(t)}^{(1)}}{\Delta t} + \overline{\mathbf{K}}\mathbf{T}_{(t+\Delta t)}^{(1)} = \mathbf{F}_{(t+\Delta t)}^{(1)}, \tag{74}$$

$$-\mathbf{C}\frac{\mathbf{T}_{(t+\Delta t)}^{(1)} - 2\mathbf{T}_{(t+\Delta t)}^{(2)} + \mathbf{T}_{(t+\Delta t)}^{(3)}}{(\Delta x_3)^2} + \mathbf{A}\frac{\mathbf{T}_{(t+\Delta t)}^{(2)} - \mathbf{T}_{(t)}^{(2)}}{\Delta t} + \overline{\mathbf{K}}\mathbf{T}_{(t+\Delta t)}^{(2)} = \mathbf{F}_{(t+\Delta t)}^{(2)}, \tag{75}$$

$$-\mathbf{C}\frac{\mathbf{T}_{(t+\Delta t)}^{(2)} - 2\mathbf{T}_{(t+\Delta t)}^{(3)} + \mathbf{T}_{(t+\Delta t)}^{(4)}}{(\Delta x_3)^2} + \mathbf{A}\frac{\mathbf{T}_{(t+\Delta t)}^{(3)} - \mathbf{T}_{(t)}^{(3)}}{\Delta t} + \overline{\mathbf{K}}\mathbf{T}_{(t+\Delta t)}^{(3)} = \mathbf{F}_{(t+\Delta t)}^{(3)}, \tag{76}$$

…

$$-\mathbf{C}\frac{\mathbf{T}_{(t+\Delta t)}^{(L-3)} - 2\mathbf{T}_{(t+\Delta t)}^{(L-2)} + \mathbf{T}_{(t+\Delta t)}^{(L-1)}}{(\Delta x_3)^2} + \mathbf{A}\frac{\mathbf{T}_{(t+\Delta t)}^{(L-2)} - \mathbf{T}_{(t)}^{(L-2)}}{\Delta t} + \overline{\mathbf{K}}\mathbf{T}_{(t+\Delta t)}^{(L-2)} = \mathbf{F}_{(t+\Delta t)}^{(L-2)}, \tag{77}$$

$$-\mathbf{C}\frac{\mathbf{T}_{(t+\Delta t)}^{(L-2)} - 2\mathbf{T}_{(t+\Delta t)}^{(L-1)} + \mathbf{T}_{(t+\Delta t)}^{(L)}}{(\Delta x_3)^2} + \mathbf{A}\frac{\mathbf{T}_{(t+\Delta t)}^{(L-1)} - \mathbf{T}_{(t)}^{(L-1)}}{\Delta t} + \overline{\mathbf{K}}\mathbf{T}_{(t+\Delta t)}^{(L-1)} = \mathbf{F}_{(t+\Delta t)}^{(L-1)}, \tag{78}$$

The corresponding matrix form is

$$\begin{bmatrix} \mathbf{H} & \mathbf{D} & & & & \\ \mathbf{D} & \mathbf{H} & \mathbf{D} & & & \\ & \mathbf{D} & \mathbf{H} & \mathbf{D} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \mathbf{D} & \mathbf{H} & \mathbf{D} \\ & & & & \mathbf{D} & \mathbf{H} \end{bmatrix} \begin{bmatrix} \mathbf{T}_{(t+\Delta t)}^{(1)} \\ \mathbf{T}_{(t+\Delta t)}^{(2)} \\ \mathbf{T}_{(t+\Delta t)}^{(3)} \\ \vdots \\ \mathbf{T}_{(t+\Delta t)}^{(L-2)} \\ \mathbf{T}_{(t+\Delta t)}^{(L-1)} \end{bmatrix}$$

$$= \begin{bmatrix} -\mathbf{D}\mathbf{T}_{(t+\Delta t)}^{(0)} + \mathbf{G}\mathbf{T}_{(t)}^{(1)} + \mathbf{F}_{(t+\Delta t)}^{(1)} \\ \mathbf{G}\mathbf{T}_{(t)}^{(2)} + \mathbf{F}_{(t+\Delta t)}^{(2)} \\ \mathbf{G}\mathbf{T}_{(t)}^{(3)} + \mathbf{F}_{(t+\Delta t)}^{(3)} \\ \vdots \\ \mathbf{G}\mathbf{T}_{(t)}^{(L-2)} + \mathbf{F}_{(t+\Delta t)}^{(L-2)} \\ -\mathbf{D}\mathbf{T}_{(t+\Delta t)}^{(L)} + \mathbf{G}\mathbf{T}_{(t)}^{(L-1)} + \mathbf{F}_{(t+\Delta t)}^{(L-1)} \end{bmatrix}, \tag{79}$$

where

$$\mathbf{D} = \frac{-\mathbf{C}}{(\Delta x_3)^2}, \tag{80}$$

$$\mathbf{H} = \frac{2\mathbf{C}}{(\Delta x_3)^2} + \frac{\mathbf{A}}{\Delta t} + \overline{\mathbf{K}}, \tag{81}$$

and

$$\mathbf{G} = \frac{\mathbf{A}}{\Delta t}. \tag{82}$$

In eq. (79), the initial values $\mathbf{T}_{(t+\Delta t)}^{(0)}$ and $\mathbf{T}_{(t+\Delta t)}^{(L)}$ can be obtained from eqs. (67) and (71), respectively. Eqs. (67) and (71) are the boundary conditions in the splitting direction of the original problem.

Let

$$\mathbf{E} = \begin{bmatrix} \mathbf{H} & \mathbf{D} & & & & \\ \mathbf{D} & \mathbf{H} & \mathbf{D} & & & \\ & \mathbf{D} & \mathbf{H} & \mathbf{D} & & \\ & & \ddots & \ddots & \ddots & \\ & & & \mathbf{D} & \mathbf{H} & \mathbf{D} \\ & & & & \mathbf{D} & \mathbf{H} \end{bmatrix}, \tag{83}$$

$$\mathbf{Z} = \left[(\mathbf{T}_{(t+\Delta t)}^{(1)})^{\mathrm{T}}, (\mathbf{T}_{(t+\Delta t)}^{(2)})^{\mathrm{T}}, (\mathbf{T}_{(t+\Delta t)}^{(3)})^{\mathrm{T}}, \cdots, (\mathbf{T}_{(t+\Delta t)}^{(L-2)})^{\mathrm{T}}, (\mathbf{T}_{(t+\Delta t)}^{(L-1)})^{\mathrm{T}}\right]^{\mathrm{T}}, \tag{84}$$

and

$$\mathbf{R} = \begin{bmatrix} -\mathbf{D}\mathbf{T}_{(t+\Delta t)}^{(0)} + \mathbf{G}\mathbf{T}_{(t)}^{(1)} + \mathbf{F}_{(t+\Delta t)}^{(1)} \\ \mathbf{G}\mathbf{T}_{(t)}^{(2)} + \mathbf{F}_{(t+\Delta t)}^{(2)} \\ \mathbf{G}\mathbf{T}_{(t)}^{(3)} + \mathbf{F}_{(t+\Delta t)}^{(3)} \\ \vdots \\ \mathbf{G}\mathbf{T}_{(t)}^{(L-2)} + \mathbf{F}_{(t+\Delta t)}^{(L-2)} \\ -\mathbf{D}\mathbf{T}_{(t+\Delta t)}^{(L)} + \mathbf{G}\mathbf{T}_{(t)}^{(L-1)} + \mathbf{F}_{(t+\Delta t)}^{(L-1)} \end{bmatrix}. \tag{85}$$

Eq. (79) can be written as:

$$\mathbf{EZ} = \mathbf{R}. \tag{86}$$

The solutions of eq. (86) are the temperatures of the nodes on each plane $\Omega^{(k)}$, $(k = 1,2,...,L-1)$ at time $t_0+\Delta t$. Furthermore, the temperature at any time t can be solved. Then, applying linear interpolation, we can obtain the temperature $T(x_1, x_2, x_3, t)$ of any nodes in the problem domain Ω as:

$$\mathbf{T}(x_1,x_2,x_3,t) = \frac{x_3 - x_3^{(k)}}{\Delta x_3}\mathbf{T}(x_1,x_2,x_3^{(k)},t) + \frac{x_3^{(k+1)} - x_3}{\Delta x_3}\mathbf{T}(x_1,x_2,x_3^{(k+1)},t). \tag{87}$$

4 Numerical examples

Four exemplary problems are chosen to show the advantage of the DSEFG method by analyzing numerical results under different scale parameters, node distributions, and time step lengths.

In this section, a regular node distribution and the background mesh for numerical integrations are selected to obtain the final discretized system of equations; the 4×4 Gaussian quadrature is selected to compute integrals on each cell of the background mesh, the cubic spline function is selected as the weight function, and the basis function is linear.

4.1 3D problem with heat loss

Consider the equation:

$$T_{,t} = T_{,11} + T_{,22} + T_{,33} - 2T, \mathbf{x} \in \Omega,\ t \in [0, t_0], \tag{88}$$

with boundary conditions:

$$T|_{x_1=0} = T|_{x_1=\pi} = T|_{x_2=0} = T|_{x_2=\pi} = T|_{x_3=0} = T|_{x_3=\pi} = 0, \tag{89}$$

and initial condition:

$$T(\mathbf{x}, 0) = \sin x_1 \sin x_2 \sin x_3, \tag{90}$$

where Ω=[0, π]×[0, π]×[0, π].

The analytical solution is

$$T(\mathbf{x}, t) = \mathrm{e}^{-5t}\sin x_1 \sin x_2 \sin x_3. \tag{91}$$

To discuss the convergence of the DSEFG method for 3D transient heat conduction problems, the influences of scale parameter, node distribution, and time step in the numerical computation are presented.

For this example, 13 planes are uniformly inserted into domain Ω along the splitting direction x_3, and 15×15 nodes are uniformly distributed on each plane with α=1.0×10^5.

The numerical results of the DSEFG method under different values of $d_{\max}$ are shown in Figure 1. We can see that when the value of $d_{\max}$ is 1.05-1.3, the numerical solution of the DSEFG method has great computational precision.

Table 1 shows the relative error norms and CPU time of the DSEFG and IEFG methods under different values of $d_{\max}$. The computational precision and speed of the DSEFG method are greater than those of the IEFG method for the same $d_{\max}$. In this numerical example, we let $d_{\max}$ =1.1.

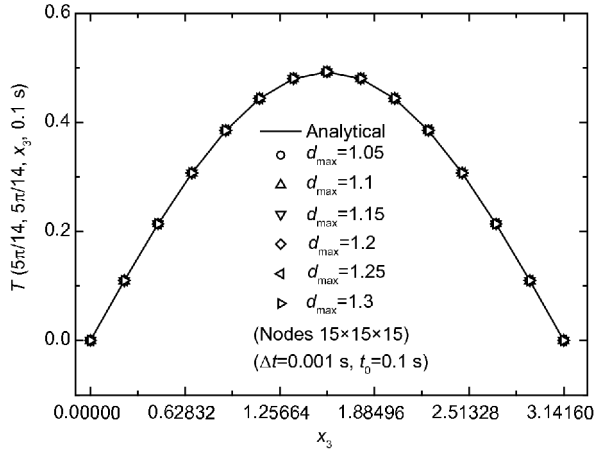

Figure 1 The results obtained by the DSEFG method with different $d_{\max}$.

Table 1 The relative error norm and CPU time of the DSEFGand IEFG methods under different $d_{\max}$

$d_{\max}$	Relative error norm		CPU time (s)	
	DSEFG	IEFG	DSEFG	IEFG
1.05	0.0013	0.0014	15.01	94.24
1.1	0.0012	0.0016	14.89	93.91
1.15	0.0012	0.0020	15.00	140.21
1.2	0.0013	0.0028	15.00	137.86
1.25	0.0015	0.0035	15.04	138.74
1.3	0.0017	0.0044	14.98	139.13

Figure 2 shows the numerical solutions of the DSEFG method under different node distributions. It can be seen that the numerical results obtained with the DSEFG method are close to the analytical ones under different node distribu-

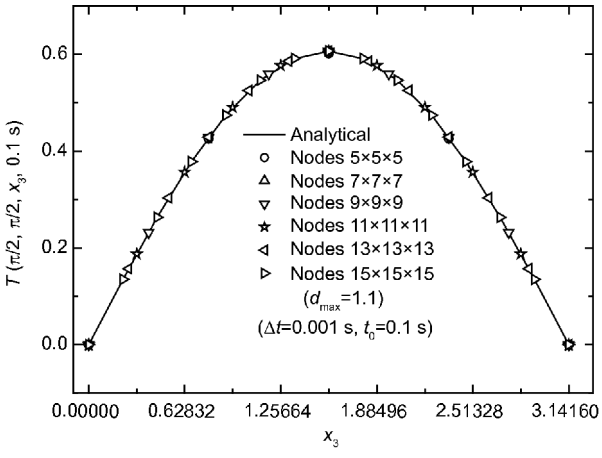

Figure 2 The results obtained with the DSEFG method with different node distributions.

tions.

Table 2 shows the relative error norms and CPU times of the DSEFG and IEFG methods with different node distributions. As in the first example, the computational precision and speed of the DSEFG method are greater than those of the IEFG method with the same node distribution. Based on the relative error norm of the DSEFG method, the method is convergent.

Table 3 shows the relative error norms and CPU time of the DSEFG and IEFG methods with different time step lengths. From Table 3, we can obtain the conclusion that the time step length Δt has more important effects upon the numerical results of the DSEFG method. The error of the DSEFG method decreases as the time step length Δt decreases, which means that the DSEFG method is convergent about the time step length Δt. Because the IEFG method considers 3D equations completely, the numerical results show that the method is affected a little by the time step length; thus, a large time step length is selected for the method.

From the example, we can see that the computational speed can be greatly improved through the DSEFG method; and, under the same node distribution, the computational precision of the DSEFG method is greater than that of the IEFG method.

The numerical solutions along the axis x_3 with the DSEFG and IEFG methods are shown in Figure 3. It can be concluded that both methods are effective, and that the DSEFG method is faster and more accurate than the IEFG method.

For this example, the CPU time of the IEFG method does not change much with the time step length Δt, because most of the CPU time is spent computing shape functions at nodes in the first time step, and the time step length Δt affects the CPU time a little.

Table 2 The relative error norm and CPU time of the DSEFG and IEFG methods under different node distributions

Number of nodes	Relative error norm		CPU time (s)	
	DSEFG	IEFG	DSEFG	IEFG
5×5×5	0.0076	0.0180	3.29	17.19
7×7×7	0.0025	0.0079	5.40	27.36
9×9×9	7.9153×10^{-4}	0.0044	8.01	38.83
11×11×11	4.6640×10^{-5}	0.0028	10.49	52.62
13×13×13	3.5768×10^{-5}	0.0020	12.90	69.20
15×15×15	2.9648×10^{-5}	0.0014	14.89	93.91

Table 3 The relative error norm and CPU time of the DSEFG and IEFG methods under different time step Δt for t_0=1 s

Δt (s)	Relative error norm		CPU time (s)	
	DSEFG	IEFG	DSEFG	IEFG
0.001	0.0012	0.0016	14.89	93.91
0.002	0.0023	0.0016	10.38	86.44
0.004	0.0046	0.0018	7.95	84.69
0.005	0.0058	0.0014	7.54	84.01
0.01	0.0118	0.0015	6.53	83.35

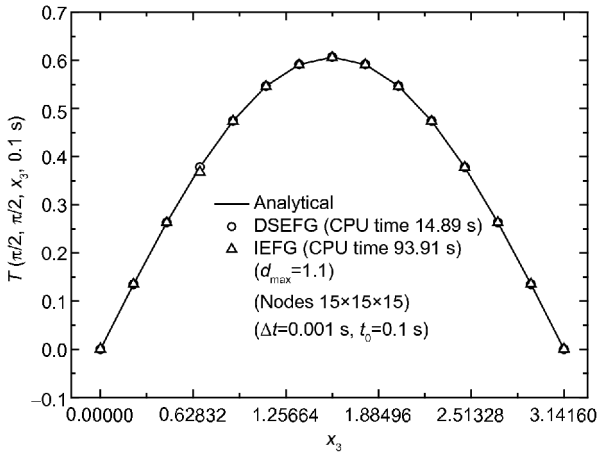

Figure 3 The results obtained by the DSEFG and IEFG methods along direction x_3.

4.2 A 3D problem with natural boundary conditions

Study the equation:

$$T_{,t} = T_{,11} + T_{,22} + T_{,33}, \mathbf{x} \in \Omega, t \in [0, t_0], \tag{92}$$

with natural boundary conditions:

$$\begin{aligned} T_{,1}\big|_{x_1=0} &= T_{,1}\big|_{x_1=\pi} = T_{,2}\big|_{x_2=0} \\ &= T_{,2}\big|_{x_2=\pi} = T_{,3}\big|_{x_3=0} = T_{,3}\big|_{x_3=\pi} = 0, \end{aligned} \tag{93}$$

and initial condition:

$$\begin{aligned} T(\mathbf{x}, 0) = {} & 1 + 2\cos x_1 \cos x_2 \cos x_3 \\ & + 3\cos(2x_1)\cos(3x_2)\cos(4x_3), \end{aligned} \tag{94}$$

where Ω=[0, π]×[0, π]×[0, π].

The analytical solution is

$$\begin{aligned} T(\mathbf{x}, t) = {} & 1 + 2\mathrm{e}^{-3t}\cos x_1 \cos x_2 \cos x_3 \\ & + 3\mathrm{e}^{-29t}\cos(2x_1)\cos(3x_2)\cos(4x_3). \end{aligned} \tag{95}$$

When the DSEFG method is used to solve this problem, we choose x_3 as the splitting direction, and the distance between adjacent planes is Δx_3. The natural boundary conditions are $T_{,3}\big|_{x_3=0} = T_{,3}\big|_{x_3=\pi} = 0$, such that the values at the nodes on the plane x_3=0 are approximately equal to the values at nodes on the plane x_3=Δx_3 when Δx_3 is smaller. This means that the number of planes along x_3 should be large. In this paper, we choose the number of planes to be 50, with α=1.0×10^5, $d_{\max}$=1.4, Δt=0.01 s, and t_0=0.3 s. The node number is 11×11×51. Then, the error is 0.0025 and the CPU time is 11.95 s.

When using the IEFG method to solve this problem, the

node distribution is 11×11×11 and α=1.0×10^5, $d_{\max}$=1.4, Δt=0.01 s, and t_0=0.3 s; then the error is 0.0031 and the CPU time is 19.37 s. If the node distribution of the IEFG method is also 11×11×51, the CPU time will be 177.03 s.

The relative error norms and CPU times of the DSEFG method, IEFG method, and FEM at different times t are shown in Table 4, with 10×10×10 cubic elements with 8 nodes being used in FEM. FEM is shown to be faster than the DSEFG method, which has greater computational accuracy than the IEFG method and FEM.

When the problem domain is divided into 20×20×20 cubic elements with 8 nodes, the relative error norm of FEM is 0.0066 when t_0=0.3 s, but the CPU time is 76.92 s. Compared with the DSEFG method with error 0.0025 and CPU time 11.95 s, FEM spends much more CPU time than the DSEFG method, and its computational accuracy is lower.

Figures 4-6 show the values of T obtained by both the DSEFG and IEFG methods along the directions x_1, x_2, and x_3 at different times t. Both of the two methods are shown to be effective, and the DSEFG method is both faster and more accurate than the IEFG method.

Figure 7 shows the numerical results of heat flux along direction x_1. The heat flux Q_1 along x_1 is defined as $Q_1 = -k\dfrac{\partial T}{\partial x_1}$, where k is the thermal conductivity and the negative sign indicates that the heat flux moves from the higher temperature region to the lower one.

Table 4 The relative error norm and CPU time of the DSEFG method, IEFG method, and FEM

Time	Relative error norm			CPU time (s)		
	DSEFG	IEFG	FEM	DSEFG	IEFG	FEM
t_0=0.3 s	0.0025	0.0031	0.0159	11.95	19.37	11.08
t_0=0.5 s	0.0014	0.0017	0.0175	14.68	20.40	11.76
t_0=0.7 s	8.0774×10^{-4}	8.7544×10^{-4}	0.0169	16.97	21.47	13.16
t_0=0.9 s	4.6178×10^{-4}	4.4814×10^{-4}	0.0143	19.44	22.52	14.8

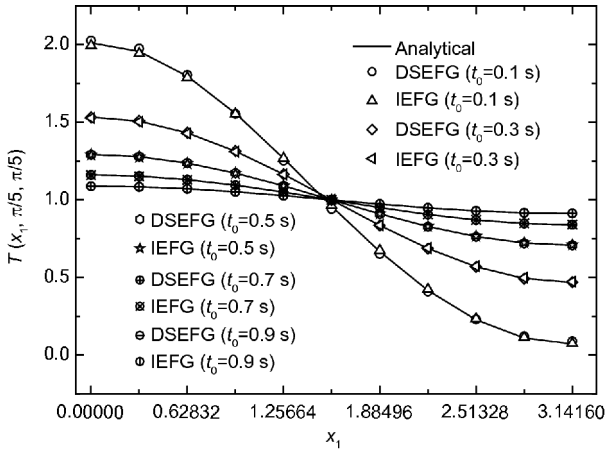

Figure 4 Temperature results at (x_1, π/5, π/5).

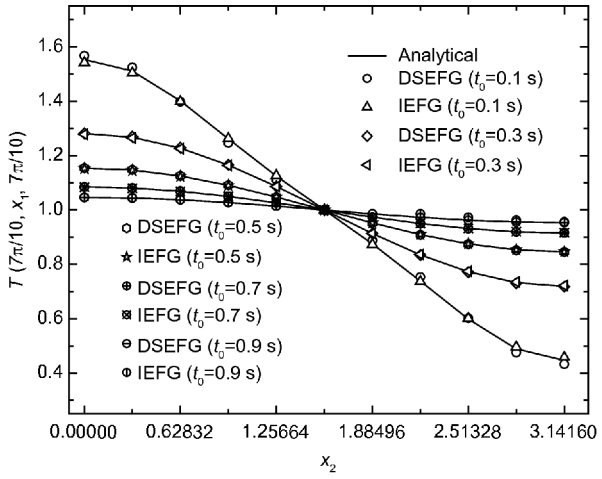

Figure 5 Temperature results at (7π/10, x_2, 7π/10).

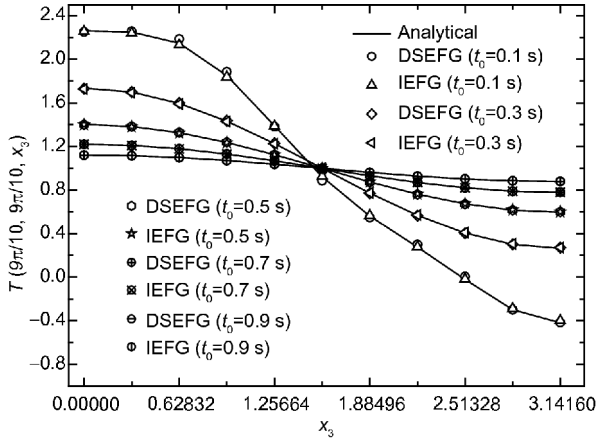

Figure 6 Temperature results at (9π/10, 9π/10, x_3).

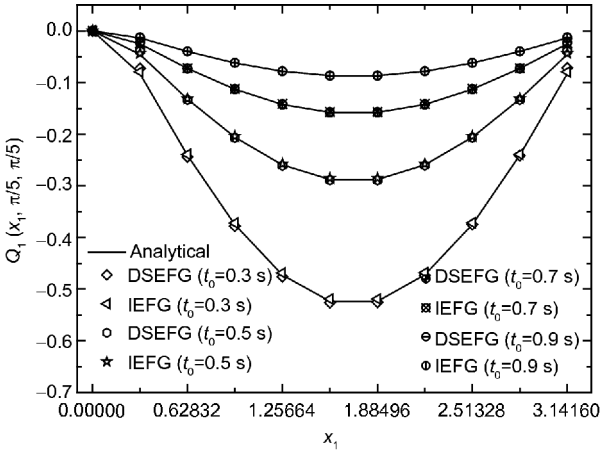

Figure 7 Heat flux results at (x_1, π/5, π/5).

4.3 3D problem with essential boundary conditions

Let us consider the equation

$$T_{,t} = (T_{,11} + T_{,22} + T_{,33}) + \sin x_3,\ \mathbf{x} \in \Omega,\ t \in [0, t_0], \qquad (96)$$

with essential boundary conditions:

$$T|_{x_1=0} = \sin x_3 + \mathrm{e}^{-2t}\sin x_2, \tag{97}$$

$$T|_{x_1=\pi} = \sin x_3 - \mathrm{e}^{-2t}\sin x_2, \tag{98}$$

$$T|_{x_2=0} = \sin x_3 + \mathrm{e}^{-2t}\sin x_1, \tag{99}$$

$$T|_{x_2=\pi} = \sin x_3 - \mathrm{e}^{-2t}\sin x_1, \tag{100}$$

$$T|_{x_3=0} = T_{x_3=\pi} = \mathrm{e}^{-2t}\sin(x_1 + x_2), \tag{101}$$

and initial condition:

$$T(\mathbf{x}, 0) = \sin(x_1 + x_2) + \sin x_3, \tag{102}$$

where Ω=[0, π]×[0, π]×[0, π].

The analytical solution is

$$T(\mathbf{x}, t) = \mathrm{e}^{-2t}\sin(x_1 + x_2) + \sin x_3. \tag{103}$$

For this problem, the parameters selected for the DSEFG and IEFG methods are α=1.0×10^3, $d_{\max}$=1.2, Δt=0.01 s. The node number is 9×9×9. If we choose x_1 or x_2 as the splitting direction, the relative error norm is 0.0055. If we choose x_3 as the splitting direction, this norm is 0.0053. Then, the splitting direction affects the performance of the DSEFG method little. In general, we choose the splitting direction to simplify application of the boundary conditions; here, x_3 is chosen for that purpose.

Table 5 shows the relative error norms and CPU times of the DSEFG and IEFG methods under different values of $d_{\max}$. Table 6 shows the same quantities with different node distributions.

The relative error norm and CPU time of the DSEFG and IEFG methods at different times t are shown in Table 7. It is apparent that the computational accuracy and speed of the DSEFG method make it advantageous over the IEFG method.

Figures 8 and 9 show the heat conduction solutions along directions x_2 and x_3. Both of the two methods are effective, with the DSEFG method being faster and more accurate. Figure 10 shows the numerical results of the heat flux along x_1.

4.4 Heat conduction problem with Dirichlet boundary conditions on a half-torus cylinder

As a fourth example, we study the heat conduction problem:

$$T_{,t} = (T_{,11} + T_{,22} + T_{,33}) + T,$$
$$(r \in [1,2],\ \theta \in [0,\pi],\ x_3 \in [0,1]), \tag{104}$$

with Dirichlet boundary conditions:

$$T(1, \theta, x_3, t) = (\sin\theta + x_3)\cdot \mathrm{e}^t, \tag{105}$$

$$T(2, \theta, x_3, t) = x_3 \cdot \mathrm{e}^t, \tag{106}$$

$$T(r, 0, x_3, t) = x_3 \cdot \mathrm{e}^t, \tag{107}$$

$$T(r, \pi, x_3, t) = x_3 \cdot \mathrm{e}^t, \tag{108}$$

Table 5 The relative error norm and CPU time of the DSEFG and IEFG methods under different $d_{\max}$

$d_{\max}$	Relative error norm		CPU time (s)	
	DSEFG	IEFG	DSEFG	IEFG
1.1	0.0053	0.0111	2.06	26.91
1.2	0.0053	0.0098	2.03	27.36
1.3	0.0062	0.0082	2.01	26.69
1.4	0.0070	0.0070	2.12	26.54
1.5	0.0079	0.0071	2.09	27.26

Table 6 The relative error norm and CPU time of the DSEFG and IEFG methods under different node distributions

Number of nodes	Relative error norm		CPU time (s)	
	DSEFG	IEFG	DSEFG	IEFG
5×5×5	0.0218	0.0490	0.41	4.12
7×7×7	0.0092	0.0190	1.01	11.83
9×9×9	0.0053	0.0098	2.02	27.36
11×11×11	0.0040	0.0060	3.61	52.25
13×13×13	0.0035	0.0040	5.88	97.55

Table 7 The relative error norm and CPU time of the DSEFG and the IEFG methods

Time	Relative error norm		CPU time (s)	
	DSEFG	IEFG	DSEFG	IEFG
t_0=0.1 s	0.0053	0.0098	2.50	27.36
t_0=0.2 s	0.0053	0.0097	2.90	40.99
t_0=0.3 s	0.0051	0.0096	3.79	55.66

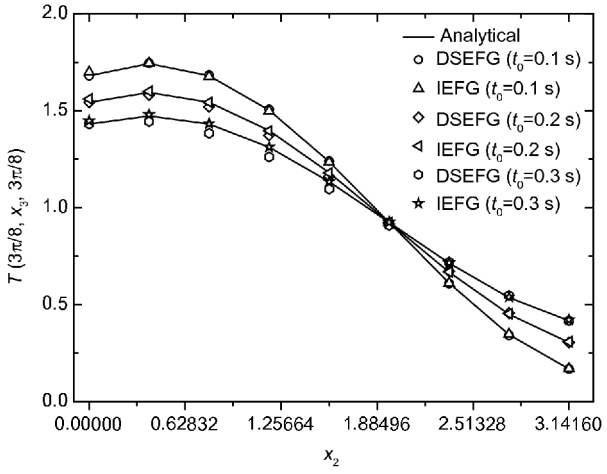

Figure 8 Temperature results at (3π/8, x_2, 3π/8).

$$T(r, \theta, 0, t) = \frac{4}{3}\left(\frac{1}{r} - \frac{r}{4}\right)\sin\theta \cdot \mathrm{e}^t, \tag{109}$$

and

$$T(r, \theta, 1, t) = \left(\frac{4}{3}\left(\frac{1}{r} - \frac{r}{4}\right)\sin\theta + 1\right)\cdot \mathrm{e}^t, \tag{110}$$

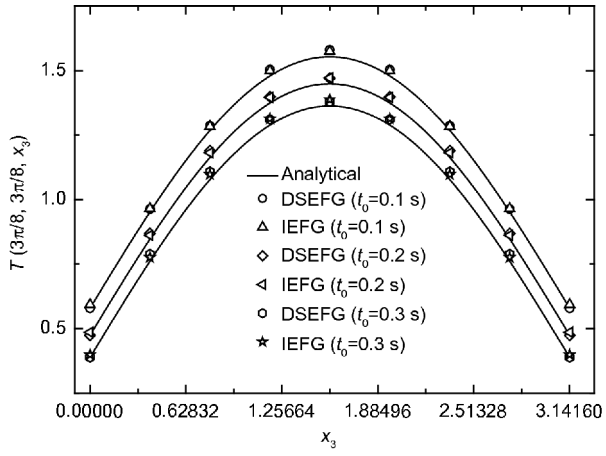

Figure 9 Temperature results at (3π/8, 3π/8, x_3).

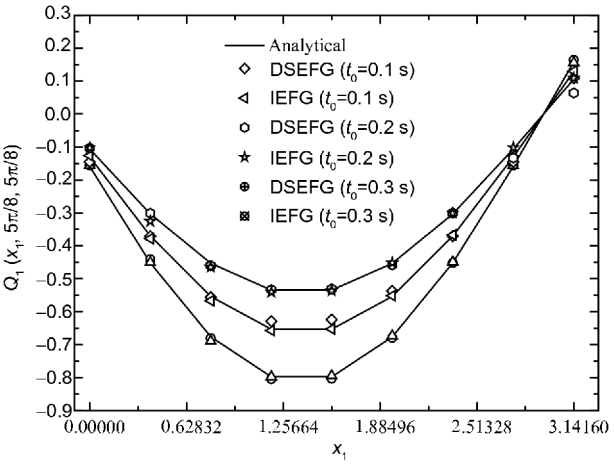

Figure 10 Temperature results at (x_1, 5π/8, 5π/8).

and initial condition:

$$T(r,\theta,x_3,0)=\frac{4}{3}\left(\frac{1}{r}-\frac{r}{4}\right)\sin\theta+x_3. \tag{111}$$

The analytical solution of this problem is

$$T(r,\theta,x_3,t)=\left(\frac{4}{3}\left(\frac{1}{r}-\frac{r}{4}\right)\sin\theta+x_3\right)\cdot \mathrm{e}^t. \tag{112}$$

For this example, the parameters selected for the IEFG method are α=1.0×10^4, $d_{\max}$=1.2, Δt=0.001 s; the number of time steps is 10, and 9×31×21 nodes are distributed. We select the cubic spline function as the weight function. Then, the relative error norm is 0.0024 and the CPU time is 554.05 s.

The same parameters are selected for the DSEFG method. When x_3 is chosen as the splitting direction, the problem domain is divided into 20 equal parts and 9×31 nodes are distributed non-uniformly on the two-dimensional sub-domain of the half-torus, as shown in Figure 11. The relative error norm is 0.0016 and the CPU time is 11.92 s. When r is chosen as the splitting direction under the same node distribution, the relative error norm is 0.0052 and CPU time is 18.38 s. The DSEFG method also can be applied to the problem of the domain shape varying in the splitting direction.

From the above results, the accuracy and efficiency are much better for splitting direction x_3 than for splitting direction r. This is because, when x_3 is chosen, eqs. (105)-(108) (which have relatively simple forms) are the boundary conditions of the corresponding two-dimensional problems; and when r is chosen, eqs. (107)-(110) (which are more complicated) are the boundary conditions. In other words, the splitting direction affects the accuracy and efficiency of the DSEFG method. The greater accuracy and efficiency allowed by splitting along x_3 can also be shown through numerical computation by MATLAB.

Choosing x_3 as the splitting direction, Figures 12-14 show the results obtained by the IEFG and DSEFG methods along the different directions. We can see that both methods offer great computational precision, with the DSEFG method having greater computational speed than the IEFG method.

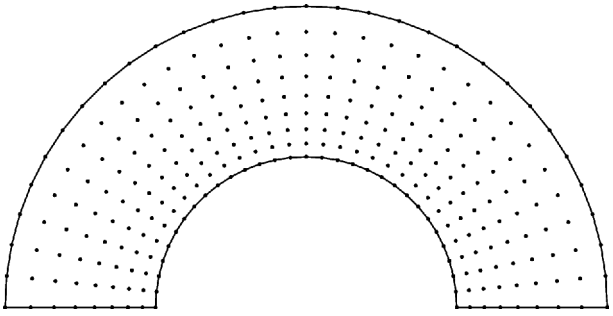

Figure 11 Node distribution in a two-dimensional sub-domain of a half-torus.

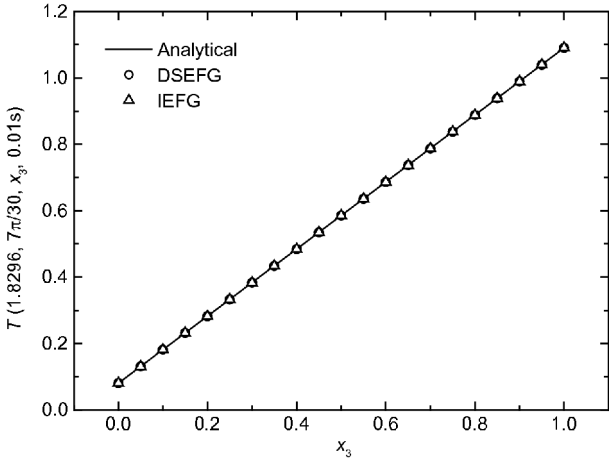

Figure 12 Temperature of analytical, DSEFG, and IEFG results along direction x_3.

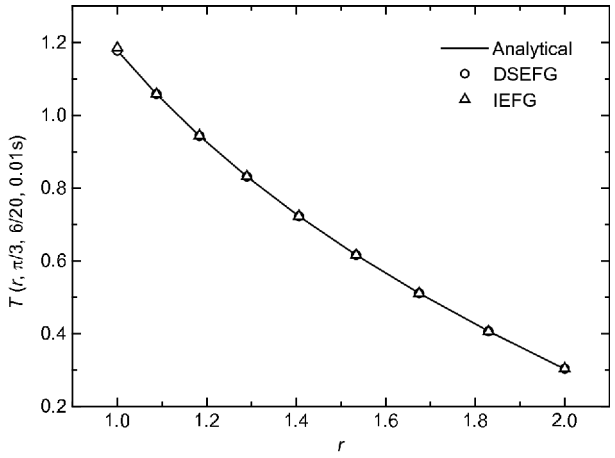

Figure 13 Temperature of analytical, DSEFG, and IEFG results along direction r.

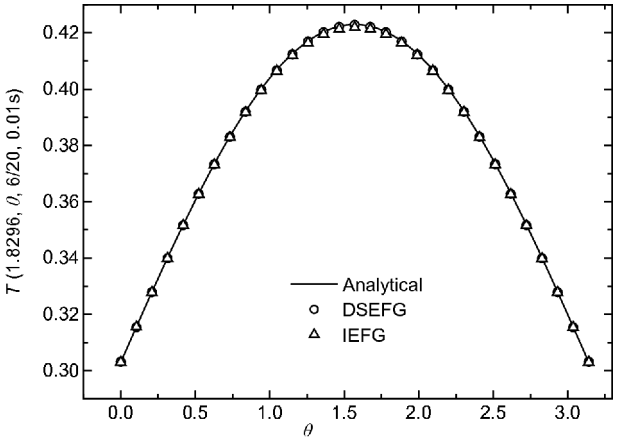

Figure 14 Temperature of analytical, DSEFG, and IEFG results along direction θ.

5 Conclusions

The DSEFG method for 3D transient heat conduction problems was introduced in this paper.

By introducing the dimension splitting method, a 3D heat conduction problem was transformed into some related 2D problems, which were solved by the IEFG method. The discretized system equation was obtained with the Galerkin weak form, the essential boundary conditions were imposed by the penalty method, and the finite difference method was employed in the time domain and the splitting direction. Then, the formulae of the DSEFG method for 3D transient heat conduction problems were obtained.

Four numerical examples show that the DSEFG method is efficient and that the final numerical solutions are affected by the scaling parameter, node distribution, and time step length. The numerical results show that the computational efficiency and computational precision of the DSEFG method are greater than those of the IEFG method.

This work was supported by the National Natural Science Foundation of China (Grant Nos. 11571223, and 51404160) and the Science and Technology Innovation Foundation of Higher Education of Shanxi Province (Grant No. 2016163).

1 M. N. Ozisik, *Boundary Values Problems of Heat Conduction* (International Textbook Company, Scranton, 1968).
2 Y. C. Hu, W. T. Bi, S. Y. Li, and Z. S. She, Sci. China-Phys. Mech. Astron. **60**, 124711 (2017).
3 R. W. Lewis, P. Nithiarasu, and K. N. Seetharamu, *Fundamentals of the Finite Element Method for Heat and Fluid Flow* (Wiley, Chichester, 2004).
4 Y. Gu, W. Chen, and X. Q. He, Int. J. Heat Mass Transfer **55**, 4837 (2012).
5 B. Yu, H. L. Zhou, H. L. Chen, and Y. Tong, Int. J. Heat Mass Transfer **91**, 110 (2015).
6 R. J. Cheng, and K. M. Liew, Eng. Anal. Bound. Elem. **36**, 1322 (2012).
7 B. Dai, B. Zheng, Q. Liang, and L. Wang, Appl. Math. Comput. **219**, 10044 (2013).
8 J. M. Wu, and W. Q. Tao, Int. J. Heat Mass Transfer **51**, 1179 (2008).
9 Q. Li, S. Chen, and X. Luo, Appl. Math. Comput. **300**, 103 (2017).
10 K. J. Bathe, and M. R. Khoshgoftaar, Nucl. Eng. Des. **51**, 389 (1979).
11 Y. M. Cheng, X. Ji, and P. E. He, Acta Mech. Sin. **36**, 43 (2004).
12 S. Li, and W. K. Liu, Appl. Mech. Rev. **55**, 1 (2002).
13 T. Belytschko, Y. Krongauz, D. Organ, M. Fleming, and P. Krysl, Comput. Methods Appl. Mech. Eng. **139**, 3 (1996).
14 J. Dolbow, and T. Belytschko, Arch Computat Methods Eng. **5**, 207 (1998).
15 Z. Zhang, J. F. Wang, Y. M. Cheng, and K. M. Liew, Sci. China-Phys. Mech. Astron. **56**, 1568 (2013).
16 Z. Zhang, P. Zhao, and K. M. Liew, Comput. Mech. **44**, 273 (2009).
17 Z. Zhang, D. M. Li, Y. M. Cheng, and K. M. Liew, Acta Mech. Sin. **28**, 808 (2012).
18 M. J. Peng, R. X. Li, and Y. M. Cheng, Eng. Anal. Bound. Elem. **40**, 104 (2014).
19 K. T. Li, and X. Shen, Int. J. Comput. Math. **84**, 807 (2007).
20 K. T. Li, A. X. Huang and W. Zhang, Commun. Numer. Meth. En. **18**, 1 (2002).
21 K. T. Li, J. Yu, F. Shi and A. X. Huang, Acta Math. App. Sin. **28**, 417 (2012).
22 H. Chen, K. Li, and S. Wang, Int. J. Numer. Meth. Fluids **73**, 409 (2013).
23 E. Hansen, and A. Ostermann, IMA J. Numer. Anal. **30**, 857 (2010).
24 E. Hansen, and A. Ostermann, Numer. Math. **108**, 557 (2008).
25 Y. Hou, and H. Wei, Int. J. Comput. Math. **89**, 112 (2012).
26 E. J. W. T. Maten, Computing, **37**, 335 (1986).
27 M. D. Bragin, and B. V. Rogov, Dokl. Math. **94**, 382 (2016).
28 R. M. D'Souza, N. H. Margolus, and M. A. Smith, J. Stat. Phys. **107**, 401 (2002).