East Asian Journal on Applied Mathematics
doi: 10.4208/eajam.2023-185.301023

Vol. **15**, No. 1, pp. 113-137
February 2025

A Selectively Relaxed Splitting Preconditioning Strategy for the Flux-Limited Multi-Group Radiation Diffusion Equations in Three Dimensions

Xiaoqiang Yue[1], Chenxi Zhang[1], Chunyan Chen[1], Xiaowen Xu[2,*] and Shi Shu[1]

[1]*National Center for Applied Mathematics in Hunan, Key Laboratory of Intelligent Computing & Information Processing of Ministry of Education, Hunan Key Laboratory for Computation and Simulation in Science and Engineering, Xiangtan University, Xiangtan 411105, China.*
[2]*Laboratory of Computational Physics, Institute of Applied Physics and Computational Mathematics, Beijing 100094, China.*

Received 31 July 2023; Accepted (in revised version) 30 October 2023.

Abstract. This article is concerned with a matrix splitting preconditioning technique with two selective relaxations and algebraic multigrid subsolves for $(G+2)\times(G+2)$ block-structured sparse linear systems derived from the three-dimensional flux-limited multi-group radiation diffusion equations, where G is the number of photon energy groups. We introduce an easy-to-implement algebraic selection strategy for the sole contributing parameter, report a spectral analysis and investigate the degree of the minimal polynomial of its left and right preconditioned matrices, and discuss its sequential practical implementation together with the two-level parallelization. Experiments are run with the representative real-world unstructured capsule implosion test cases and it is found that the numerical robustness, computational efficiency and parallel scalability of the proposed preconditioner evaluated on the Tianhe-2A supercomputer with up to 2,816 processor cores are superior to some existing popular monolithic and block preconditioning approaches.

AMS subject classifications: 65F10, 65N55, 65Y05, 65Z05

Key words: Radiation diffusion equation, matrix splitting preconditioner, selective relaxation, algebraic multigrid, parallel computing.

1. Introduction

The numerical simulation of the thermal radiation transport [39] requires solving complicated partial differential equations (PDEs) of parabolic type with highly nonlinear coefficients in a background medium with numerous materials. It is worth noting that the

*Corresponding author. *Email addresses:* yuexq@xtu.edu.cn (X.Q. Yue), xwxu@iapcm.ac.cn (X.W. Xu), shushi@xtu.edu.cn (S. Shu)

thermal radiation transport process occurs in various branches of physics, such as the optical remote sensing, the massive star formation and the inertial confinement fusion experiments. During the past couple of decades, a large amount of research efforts have been devoted to the development of mathematically less complicated and computationally cheaper yet numerically more accurate approximations. Among these approximations, the simplest and the most extensively used one is the flux-limited multi-group radiation diffusion (MGD) equations [21], where the frequency-dependent radiation energy densities are categorized into multiple photon energy (or frequency) groups and, more importantly, are assumed to be equably distributed over the respective frequency ranges.

Traditionally, the adaptive backward Eulerian time-integration is utilized to eliminate the need of severe time-step constraints for the numerical stability. For the resulting nonlinear reaction-diffusion systems, we take advantage of the method of frozen coefficients [30] as the iterative linearization technique, followed by a cell-centered finite volume discretization scheme [11,44] ensuring a local conservation property, however, requiring the solution of a great deal of sparse, ill-conditioned, unsymmetric but positive definite linear systems with the number of degrees of freedom ranging from 10^7 to 10^{11} because of the presence of hydrodynamic instabilities as well as the wave-like propagation characteristics and multiple spatio-temporal scales in exact and approximate solutions, which is computationally expensive, generally accounting for more than eighty percent of the total simulation time. Therefore, this motivates research efforts aimed at the development of robust, accurate and reliable numerical solution algorithms in an efficient and scalable manner.

Despite the reliability and accessibility of sparse direct solvers such as MUMPS [2], PARDISO [42], PaStiX [23], STRUMPACK [19], SuperLU [34], SuperMF [48] and UMFPACK [13] for small systems of linear equations, the memory requirements and difficulties in developing effective massively parallel implementations restrict their scope of practical applications. While on the contrary, the sparse iterative solvers become increasingly attractive to meet the sustaining demand for higher-spatial resolutions, due to their smaller memory utilizations, far easier to implement on parallel computers and higher degrees of parallelism. One of the most powerful sparse iterative solvers is on the strength of specific projections/orthogonalizations onto Krylov subspaces — e.g. Bi-CGSTAB [46], CG [25], GMRES [40] and MINRES [37] as the prominent representatives, whose numerical performance, such as their convergence behaviors, needs to be further boosted via favorable preconditioners. They consistently transform the original system of linear equations into a mathematically equivalent linear system, however, with some more advantageous properties — e.g. smaller (spectral) condition numbers and a more clustered eigenvalue distribution. It is worth highlighting that the two fundamental peculiarities of modern preconditioning approaches are the numerical robustness (with reference to the geometric, physical and discrete parameters and the number of processors) and the implementation scalability — i.e. the setup phase and every iteration step should be scalable in a parallel environment [12]. The numerical robustness is, without doubt, a preemptive requirement to attain a scalable implementation.

Multitudinous scholars and experts at home and abroad have come up with a wide range of resultful preconditioning schemes in an efficient and scalable manner. They can

be principally categorized into three different styles: the monolithic, block (also known as physical-variable-based) as well as symmetric and asymmetric combined preconditioning algorithms accompanied by some sort of adaptive strategy — cf. Refs. [3, 6, 9, 20, 26, 27, 36, 43, 49–51, 53–58]. Another critical criterion of classifying these preconditioners is that how much application information is used. From this point of view, they are supposed to descend into three different categories: the sparse purely-algebraic (general-purpose), semi-algebraic (block-structure specific) and purely-analytic (application specific) preconditioning techniques. Although great progress has been made in this area of research, with the continuous refinement on physical modelings — i.e. much more complicated application characteristics, and the further increase of grid resolutions — i.e. much more significant computational requirements, new challenges and higher demands are put forward for the numerical solution algorithms, primarily on the advanced preconditioning approaches, in the acquisition of well-timed and reliable modeling predictions. In the present article, we propose, analyze, implement and examine a novel matrix splitting block preconditioner of the semi-algebraic category with two selective relaxations (including three positive parameters, however, two of them have no effect on the preconditioning behavior) and algebraic multigrid subsolves for better numerical and parallel scalabilities. It should be emphasized that the proposed preconditioning strategy does not require any other information except the flux-limited MGD coefficient matrix and its inherent physics-informed sparse block structure.

The remaining sections of this study are structured as follows. After introducing in Section 2 the model problem as well as its linearized and discretized formulations, in Section 3 we present the principal contribution, including the construction of the proposed preconditioning algorithm induced by a two-step alternating direction implicit iteration scheme, the algebraic quasi-optimal choice strategy of the involved parameter together with the spectral distribution and the degree of the minimal polynomial of its preconditioned matrix. In Section 4, we provide the preconditioner's sequential implementation procedure and its two-level parallelization. Several sequential and parallel computational results of realistic unstructured benchmark problems are also reported and analyzed in this section to inspect the numerical robustness, computational efficiency as well as parallel strong and weak scaling properties of our preconditioning algorithm. Section 5 concludes the manuscript and suggests a couple of future directions.

2. Problem Configuration, Linearization and Discretization

In the present work we are interested in the numerical solution of the flux-limited nonlinear time-dependent MGD equations, on a spherically symmetrical bounded geometry $\Omega \subset \mathbb{R}^3$, given by

$$\frac{\partial E_g}{\partial t} = \nabla \cdot \big(D_g(E_g)\nabla E_g\big) + c\big(\sigma_{Bg}B_g(T_E) - \sigma_{Pg}E_g\big) + S_g, \quad g = 1, \ldots, G, \tag{2.1a}$$

$$\rho c_I \frac{\partial T_I}{\partial t} = \nabla \cdot \big(D_I(T_I)\nabla T_I\big) - w_{IE}(T_I - T_E), \tag{2.1b}$$

$$\rho c_E \frac{\partial T_E}{\partial t} = \nabla \cdot \big(D_E(T_E)\nabla T_E\big) - c\sum_{g=1}^{G}\big(\sigma_{Bg}B_g(T_E) - \sigma_{Pg}E_g\big) + w_{IE}(T_I - T_E) \tag{2.1c}$$

supplied with appropriate initial and boundary conditions, in search of the radiation energy density functions $E_1, \cdots, E_G : \overline{\Omega} \times \mathscr{I} \to \mathbb{R}$, the ion temperature function $T_I : \overline{\Omega} \times \mathscr{I} \to \mathbb{R}$ and the electron temperature function $T_E : \overline{\Omega} \times \mathscr{I} \to \mathbb{R}$ for some given bounded open temporal interval $\mathscr{I} \subset \mathbb{R}^+$, the density of medium ρ updated in hydrodynamics process, specific heat capacities c_I and c_E, nonlinear thermal-conductivity coefficients $D_I(T_I)$ and $D_E(T_E)$ together with the nonlinear radiation diffusion coefficient $D_g(E_g)$, scattering and absorption coefficients σ_{Bg} and σ_{Pg}, source item S_g and electron scattering energy density $B_g(T_E)$ for the photon frequency group index $g = 1, \ldots, G$.

Discretizing the nonlinear coupled PDE system (2.1) by the adaptive backward Eulerian scheme results in the chain of semi-discrete nonlinear systems of the form

$$\begin{aligned}
&-\nabla \cdot \big(D_g(E_g)\nabla E_g\big) + \left(\frac{1}{\Delta t_{k+1}} + c\sigma_{Pg}\right)E_g - c\sigma_{Bg}B_g(T_E)\\
&= S_g + \frac{1}{\Delta t_{k+1}}E_g^{(k)}, \quad g = 1, \ldots, G,\\
&-\nabla \cdot \big(D_I(T_I)\nabla T_I\big) + \left(\frac{\rho c_I}{\Delta t_{k+1}} + w_{IE}\right)T_I - w_{IE}T_E = \frac{\rho c_I}{\Delta t_{k+1}}T_I^{(k)},\\
&-\nabla \cdot \big(D_E(T_E)\nabla T_E\big) + \left(\frac{\rho c_E}{\Delta t_{k+1}} + w_{IE}\right)T_E + c\sum_{g=1}^{G}\sigma_{Bg}B_g(T_E)\\
&\quad - c\sum_{g=1}^{G}\sigma_{Pg}E_g - w_{IE}T_I = \frac{\rho c_E}{\Delta t_{k+1}}T_E^{(k)}
\end{aligned} \tag{2.2}$$

at the $(k+1)$-th time level, where Δt_{k+1} is the immediate temporal mesh size and terms with superscript (k) symbolize the discrete-in-time counterparts at the k-th time level. Subsequently, we linearize the nonlinear system (2.2) iteratively by means of the method of frozen coefficients [30], thus obtaining a suite of coupled systems of linear reaction-diffusion equations

$$\begin{aligned}
&-\nabla \cdot \big(D_g^{(\delta)}\nabla E_g\big) + \left(\frac{1}{\Delta t_{k+1}} + c\sigma_{Pg}^{(\delta)}\right)E_g - c\sigma_{Bg}^{(\delta)}\left(\frac{\partial B_g}{\partial T_E}\right)^{(\delta)}T_E\\
&= S_g^{(\delta)} + \frac{1}{\Delta t_{k+1}}E_g^{(k)} + c\sigma_{Bg}^{(\delta)}\left[B_g^{(\delta)} - \left(\frac{\partial B_g}{\partial T_E}\right)^{(\delta)}T_E^{(\delta)}\right], \quad g = 1, \ldots, G,
\end{aligned} \tag{2.3a}$$

$$-\nabla \cdot \big(D_I^{(\delta)}\nabla T_I\big) + \left(\frac{\rho c_I^{(\delta)}}{\Delta t_{k+1}} + w_{IE}^{(\delta)}\right)T_I - w_{IE}^{(\delta)}T_E = \frac{\rho c_I^{(\delta)}}{\Delta t_{k+1}}T_I^{(k)}, \tag{2.3b}$$

$$-\nabla \cdot \big(D_E^{(\delta)}\nabla T_E\big) + \left[\frac{\rho c_E^{(\delta)}}{\Delta t_{k+1}} + w_{IE}^{(\delta)} + c\sum_{g=1}^{G}\sigma_{Bg}^{(\delta)}\left(\frac{\partial B_g}{\partial T_E}\right)^{(\delta)}\right]T_E - c\sum_{g=1}^{G}\sigma_{Pg}^{(\delta)}E_g - w_{IE}^{(\delta)}T_I$$

$$= \frac{\rho c_E^{(\delta)}}{\Delta t_{k+1}} T_E^{(k)} - c \sum_{g=1}^{G} \sigma_{Bg}^{(\delta)} \left[B_g^{(\delta)} - \left(\frac{\partial B_g}{\partial T_E} \right)^{(\delta)} T_E^{(\delta)} \right] \tag{2.3c}$$

at the $(\delta+1)$-th nonlinear iterative step, where the initial guess spatial function at t_{k+1}, i.e. the specific case with $\delta = 0$, is just a relevant approximate spatial function at t_k. In the end we employ a cell-centered locally conservative finite volume scheme for the spatial discretization of (2.3), yielding the undermentioned fully-discrete coupled system of linear algebraic equations in an arrowhead form

$$\mathbf{Au} \equiv \begin{bmatrix} A_R & O & D_{RE} \\ O & A_I & D_{IE} \\ D_{ER} & D_{EI} & A_E \end{bmatrix} \begin{bmatrix} e_R \\ t_I \\ t_E \end{bmatrix} = \begin{bmatrix} f_R \\ f_I \\ f_E \end{bmatrix} \equiv \mathbf{f}, \tag{2.4}$$

where the sub-matrices and sub-vectors intimately related to the discrete radiation variables are written as single block entities — i.e.

$$A_R = \begin{bmatrix} A_1 & & \\ & \ddots & \\ & & A_G \end{bmatrix} \in \mathbb{R}^{Gn \times Gn}, \quad D_{RE} = \begin{bmatrix} D_{1E} \\ \vdots \\ D_{GE} \end{bmatrix} \in \mathbb{R}^{Gn \times n},$$

$$e_R = \begin{bmatrix} e_1 \\ \vdots \\ e_G \end{bmatrix} \in \mathbb{R}^{Gn}, \qquad f_R = \begin{bmatrix} f_1 \\ \vdots \\ f_G \end{bmatrix} \in \mathbb{R}^{Gn}$$

and $D_{ER} = (D_{E1}, \cdots, D_{EG}) \in \mathbb{R}^{n \times Gn}$ while A_I, D_{IE}, D_{EI} and A_E are real $n \times n$ matrices. Here n is the number of mesh cells and n-dimensional sub-vectors e_g, $g = 1, \ldots, G$, t_I and t_E serve the purpose of approximating the g-th radiation energy density, ion and electron temperatures on each spatial cell. It is important to mention that the implicit solution process of the linear system (2.4) will be taken place within each nonlinear iterative step at each time level. Several important observations on the coefficient matrix **A** defined by (2.4) that should be made are

- the nonzero structure of each diagonal sub-block is, without exception, shaped as that of a discrete linear scalar reaction-diffusion operator, however, with the involved coefficients in pretty different orders of magnitude and potentially strong discontinuities, which indicates that sub-blocks $A_1, \cdots, A_G, A_I$, and A_E are sparse, symmetric positive definite and multiscale;

- each off-diagonal sub-block $D_{gg'}, g \neq g'$ is diagonal with non-positive entries at possibly remarkably different scales and compliant with the undermentioned conditions

  $$D_{EI} = D_{IE} \quad \text{while} \quad D_{Eg} \neq D_{gE} \quad \text{for} \quad g = 1, \ldots, G,$$

  which means that **A** must be sparse, nonsymmetric positive definite, multiscale and multi-physics coupled.

It follows from the above analysis that the algebraic characteristics of the coefficient matrix **A** are much worse than those of its diagonal sub-blocks, which is precisely the primary reason why multifarious block preconditioning studies have been reported previously — i.e. instead of directly solving (2.4), but making use of the flexible restarted GMRES solver, denoted by FGMRES(m), to solve a mathematically equivalent preconditioned system

$$\mathbf{P}^{-1}\mathbf{A}\mathbf{u} = \mathbf{P}^{-1}\mathbf{f} \quad \text{or} \quad (\mathbf{A}\mathbf{P}^{-1})\mathbf{x} = \mathbf{f} \quad \text{with} \quad \mathbf{u} = \mathbf{P}^{-1}\mathbf{x},$$

where m is the number of Krylov directions to orthogonalize against and **P** is a block preconditioner chosen appropriately.

3. A Selectively Relaxed Splitting Preconditioner

3.1. Related works

Let us firstly make reference to related works. The relaxed splitting preconditioning scheme was developed originally for the saddle point problems from the incompressible Navier-Stokes equations in [45], which is on the strength of the relaxed type preconditioner investigated in [7] together with Hermitian and skew-Hermitian splitting idea originally proposed in [4]. Yet another relaxed splitting preconditioner was introduced in [33] for general saddle point problems. Subsequently, the generalized, modified and inexact modified variants were proposed in [10, 16, 31] for the generalized saddle point problems. In the meanwhile, for the same type of sparse linear systems, the relaxed triangular and block-triangular splitting preconditioners as well as a new relaxed splitting preconditioner modeled as the multiplication of block lower and upper triangular matrices were reported in [17, 32, 59]. The lattermost and most recent one for solving (2.4) has the form

$$\tilde{\mathbf{P}} = \begin{bmatrix} A_R & O & O \\ O & I_\pi & O \\ D_{ER} & \tilde{\alpha} D_{EI} & I_\pi \end{bmatrix} \begin{bmatrix} I_\pi & O & \tilde{\alpha} D_{RE} \\ O & A_I & D_{IE} \\ O & O & \tilde{\beta} I_\pi + A_E \end{bmatrix},$$

where I_π represents the identity matrix of suitable size. Another important point is that the positive parameters $\tilde{\alpha}$ and $\tilde{\beta}$ were simply determined by the trial-and-error procedure in [32], namely, they were chosen to be the experimentally optimal values in one significant digit, however, at a large extra computational cost. To go a little further, it can easily be seen that the relaxed splitting preconditioner

$$\hat{\mathbf{P}} = \begin{bmatrix} A_R & O & O \\ O & I_\pi & O \\ D_{ER} & \hat{\alpha} D_{EI} & I_\pi \end{bmatrix} \begin{bmatrix} I_\pi & O & \hat{\alpha} D_{RE} \\ O & A_I & D_{IE} \\ O & O & \hat{S}_E \end{bmatrix}, \tag{3.1}$$

which will be taken for a comparison study in Section 4, would behave better than $\tilde{\mathbf{P}}$ defined above since there is only one positive parameter to be determined and the respective difference $\hat{\mathbf{P}} - \mathbf{A}$ takes on fewer nonzero sub-blocks than $\tilde{\mathbf{P}} - \mathbf{A}$, where

$$\hat{S}_E = A_E - \hat{\alpha}(D_{ER}D_{RE} + D_{EI}D_{IE}) = A_E - \hat{\alpha}\left(\sum_{g=1}^{G} D_{Eg}D_{gE} + D_{EI}D_{IE}\right)$$

possesses the same nonzero pattern as A_E.

3.2. Preconditioner constitution

In this subsection, a new-type preconditioning algorithm is proposed for solving the flux-limited MGD system of linear equations (2.4). Utilizing the special block structure of $\mathbf{A}$, we are able to split it into

$$\mathbf{A}=\begin{bmatrix} A_R & O & D_{RE} \\ O & O & O \\ D_{ER} & O & O \end{bmatrix}+\begin{bmatrix} O & O & O \\ O & A_I & D_{IE} \\ O & D_{EI} & A_E \end{bmatrix}=:\mathbf{A}_1+\mathbf{A}_2.$$

Consider now the following two splittings of $\mathbf{A}$:

$$\mathbf{A}=(\mathbf{\Lambda}_1+\mathbf{A}_1)-(\mathbf{\Lambda}_1-\mathbf{A}_2)\quad\text{and}\quad\mathbf{A}=(\mathbf{\Lambda}_2+\mathbf{A}_2)-(\mathbf{\Lambda}_2-\mathbf{A}_1) \tag{3.2}$$

with two selective relaxations

$$\mathbf{\Lambda}_1=\begin{bmatrix} O & O & O \\ O & \beta I_\pi & O \\ O & O & \alpha I_\pi \end{bmatrix}\quad\text{and}\quad\mathbf{\Lambda}_2=\begin{bmatrix} \gamma I_\pi & O & O \\ O & O & O \\ O & O & O \end{bmatrix},$$

using three positive parameters β,α and γ. The matrices $\mathbf{\Lambda}_1+\mathbf{A}_1$ and $\mathbf{\Lambda}_2+\mathbf{A}_2$ are obviously nonsingular. On account of (3.2) and a practical use of the two-sweep alternating direction implicit idea, we establish the following iteration scheme:

$$\begin{aligned}&(\mathbf{\Lambda}_1+\mathbf{A}_1)\mathbf{u}^{(k+1/2)}=(\mathbf{\Lambda}_1-\mathbf{A}_2)\mathbf{u}^{(k)}+\mathbf{f},\\&(\mathbf{\Lambda}_2+\mathbf{A}_2)\mathbf{u}^{(k+1)}=(\mathbf{\Lambda}_2-\mathbf{A}_1)\mathbf{u}^{(k+1/2)}+\mathbf{f},\end{aligned}\qquad k=0,1,2,\ldots$$

for a given initial guess vector $\mathbf{u}^{(0)}$. Removing the intermediate variable $\mathbf{u}^{(k+1/2)}$ from the above two formulae, yields

$$\begin{aligned}\mathbf{u}^{(k+1)}=&\underbrace{(\mathbf{\Lambda}_2+\mathbf{A}_2)^{-1}(\mathbf{\Lambda}_2-\mathbf{A}_1)(\mathbf{\Lambda}_1+\mathbf{A}_1)^{-1}(\mathbf{\Lambda}_1-\mathbf{A}_2)}_{:=\mathbf{G}}\mathbf{u}^{(k)}\\&+\underbrace{(\mathbf{\Lambda}_2+\mathbf{A}_2)^{-1}(\mathbf{\Lambda}_2+\mathbf{\Lambda}_1)(\mathbf{\Lambda}_1+\mathbf{A}_1)^{-1}}_{:=\mathbf{P}^{-1}}\mathbf{f},\end{aligned}$$

where $\mathbf{G}$ is the iteration matrix and can be written as

$$\mathbf{G}=[\mathbf{I}_\pi-(\mathbf{\Lambda}_2+\mathbf{A}_2)^{-1}\mathbf{A}][\mathbf{I}_\pi-(\mathbf{\Lambda}_1+\mathbf{A}_1)^{-1}\mathbf{A}]=\mathbf{I}_\pi-\mathbf{P}^{-1}\mathbf{A}$$

with $\mathbf{I}_\pi\in\mathbb{R}^{(G+2)n\times(G+2)n}$ being the identity matrix, together with our selectively relaxed splitting (SRS) preconditioning matrix

$$\mathbf{P}=(\mathbf{\Lambda}_1+\mathbf{A}_1)(\mathbf{\Lambda}_2+\mathbf{\Lambda}_1)^{-1}(\mathbf{\Lambda}_2+\mathbf{A}_2)=\begin{bmatrix} A_R & \dfrac{1}{\alpha}D_{RE}D_{EI} & \dfrac{1}{\alpha}D_{RE}A_E \\ O & A_I & D_{IE} \\ D_{ER} & D_{EI} & A_E \end{bmatrix} \tag{3.3}$$

deduced from the definitions of $\mathbf{\Lambda}_1,\mathbf{A}_1,\mathbf{\Lambda}_2$ and $\mathbf{A}_2$.

Remark 3.1. It is worthwhile to point out that positive parameters β and γ both vanish when constructing the preconditioner $\mathbf{P}$, i.e. they have no effect on the specific preconditioning behavior.

3.3. An algebraic selection strategy for determining the involved parameter

This subsection is devoted to deriving an algebraic, quasi-optimal and feasible estimation formula for the involved positive parameter α in (3.3), using the fact that a good choice of α would make the preconditioner $\mathbf{P}$ quite close to the coefficient matrix $\mathbf{A}$, which results in a clustered spectrum of the left- or right-preconditioned matrix as well as a fast and smooth convergence behavior of the preconditioned flexible restarted GMRES solver. To put it simply, the parameter α must be selected so that the difference between $\mathbf{P}$ and $\mathbf{A}$ vanishes almost everywhere, namely,

$$\mathbf{R}=\mathbf{P}-\mathbf{A}=\begin{bmatrix} O & \frac{1}{\alpha}D_{RE}D_{EI} & D_{RE}\left(\frac{1}{\alpha}A_E-I_\pi\right) \\ O & O & O \\ O & O & O \end{bmatrix}\approx O, \tag{3.4}$$

which, exploiting a similar idea developed in [28], is equivalent to minimizing the objective function

$$\eta(\alpha)=\|\mathbf{R}\|_F^2=\operatorname{trace}(\mathbf{R}\mathbf{R}^\top)=\operatorname{trace}\left(\frac{1}{\alpha^2}D_{RE}D_{EI}^2D_{RE}^\top+D_{RE}\left(\frac{1}{\alpha}A_E-I_\pi\right)^2D_{RE}^\top\right).$$

Then a straightforward calculation yields

$$\eta(\alpha)=\frac{1}{\alpha^2}k_1-\frac{2}{\alpha}k_2+k_3=k_1\left(\frac{1}{\alpha}-\frac{k_2}{k_1}\right)^2+k_3-\frac{k_2^2}{k_1},$$

where

$$k_1=\operatorname{trace}\big(D_{RE}(D_{EI}^2+A_E^2)D_{RE}^\top\big),\quad k_2=\operatorname{trace}\big(D_{RE}A_ED_{RE}^\top\big),\quad k_3=\operatorname{trace}\big(D_{RE}D_{RE}^\top\big).$$

Thus we have the quasi-optimal expression for the involved positive parameter

$$\alpha^*=\frac{k_1}{k_2}=\frac{\sum_{g=1}^G\operatorname{trace}\big(D_{gE}(D_{EI}^2+A_E^2)D_{gE}\big)}{\sum_{g=1}^G\operatorname{trace}\big(D_{gE}A_ED_{gE}\big)}. \tag{3.5}$$

Remark 3.2. It can be analogously shown that the quasi-optimal choice of the positive parameter $\hat{\alpha}$ in (3.1) is

$$\hat{\alpha}^*=\frac{\operatorname{trace}\big(\sum_{g=1}^G D_{gE}D_{gE}A_R+D_{EI}A_ID_{EI}\big)}{\operatorname{trace}\big(A_R\sum_{g=1}^G D_{gE}D_{gE}A_R+D_{EI}A_I^2D_{EI}\big)}. \tag{3.6}$$

3.4. Spectral properties of the preconditioned matrix

Regarding the SRS left-preconditioned matrix $\mathbf{P}^{-1}\mathbf{A}$, or equivalently, the SRS right-preconditioned variant $\mathbf{A}\mathbf{P}^{-1}$, we have the eigenvalue and eigenvector distribution results stated in the following theorem.

Theorem 3.1. *Assume that sub-matrices $A_R, D_{RE}, A_I, D_{IE}, D_{ER}, D_{EI}$ and A_E are defined by* (2.4) *and α is an arbitrarily given positive parameter. Then the eigenvalues of $\mathbf{P}^{-1}\mathbf{A}$ (or equivalently, $\mathbf{A}\mathbf{P}^{-1}$) are given by* 1 *with algebraic multiplicity at least $(G+1)n$ and $1-\mu_i$ with μ_i being the i-th eigenvalue of the real $n \times n$ matrix*

$$Z_\alpha = \left(\frac{1}{\alpha} D_{EI} A_I^{-1} D_{IE} - \frac{1}{\alpha} A_E + I_\pi\right) S_E^{-1} D_{ER} S_R^{-1} D_{RE}, \tag{3.7}$$

where

$$S_R = A_R - \frac{1}{\alpha} D_{RE} D_{ER}, \quad S_E = A_E - D_{EI} A_I^{-1} D_{IE}. \tag{3.8}$$

Moreover, $(v_R^\top, v_I^\top, v_E^\top)^\top$ is an eigenvector of $\mathbf{P}^{-1}\mathbf{A}$ associated with the eigenvalue one if

$$\frac{1}{\alpha} S_R^{-1} D_{RE} D_{EI} v_I + S_R^{-1} D_{RE} \left(\frac{1}{\alpha} A_E - I_\pi\right) v_E = \mathbf{0} \tag{3.9}$$

and v_R can be chosen arbitrarily; concerning a non-unitary eigenvalue, v_R is the eigenvector of the matrix

$$S_R^{-1} D_{RE} \left(\frac{1}{\alpha} S_E - I_\pi\right) S_E^{-1} D_{ER} + I_\pi \tag{3.10}$$

associated with the same eigenvalue while

$$v_I = A_I^{-1} D_{IE} S_E^{-1} D_{ER} v_R, \quad v_E = -S_E^{-1} D_{ER} v_R. \tag{3.11}$$

Proof. With taking advantage of the undermentioned block tri-factorization

$$\mathbf{P} = \begin{bmatrix} S_R & O & \frac{1}{\alpha} D_{RE} \\ O & I_\pi & O \\ O & O & I_\pi \end{bmatrix} \begin{bmatrix} I_\pi & O & O \\ O & I_\pi & O \\ D_{ER} & D_{EI} A_I^{-1} & I_\pi \end{bmatrix} \begin{bmatrix} I_\pi & O & O \\ O & A_I & D_{IE} \\ O & O & S_E \end{bmatrix}, \tag{3.12}$$

it is easy to verify that

$$\mathbf{P}^{-1} = \begin{bmatrix} I_\pi & O & O \\ O & A_I^{-1} & -A_I^{-1} D_{IE} S_E^{-1} \\ O & O & S_E^{-1} \end{bmatrix} \begin{bmatrix} I_\pi & O & O \\ O & I_\pi & O \\ -D_{ER} & -D_{EI} A_I^{-1} & I_\pi \end{bmatrix} \begin{bmatrix} S_R^{-1} & O & -\frac{1}{\alpha} S_R^{-1} D_{RE} \\ O & I_\pi & O \\ O & O & I_\pi \end{bmatrix}$$

and

$$\mathbf{P}^{-1}\mathbf{R} = \begin{bmatrix} O & \frac{1}{\alpha} S_R^{-1} D_{RE} D_{EI} & S_R^{-1} D_{RE} \left(\frac{1}{\alpha} A_E - I_\pi\right) \\ O & \frac{1}{\alpha} A_I^{-1} D_{IE} S_E^{-1} D_{ER} S_R^{-1} D_{RE} D_{EI} & A_I^{-1} D_{IE} S_E^{-1} D_{ER} S_R^{-1} D_{RE} \left(\frac{1}{\alpha} A_E - I_\pi\right) \\ O & -\frac{1}{\alpha} S_E^{-1} D_{ER} S_R^{-1} D_{RE} D_{EI} & -S_E^{-1} D_{ER} S_R^{-1} D_{RE} \left(\frac{1}{\alpha} A_E - I_\pi\right) \end{bmatrix} \tag{3.13}$$

using the formula (3.4) as well as the definitions of S_R and S_E in (3.8). Observe from the block structure of $\mathbf{P}^{-1}\mathbf{R}$ that its eigenvalues are 0 with algebraic multiplicity Gn and those of the bottom right $2n \times 2n$ sub-matrix

$$Y_\alpha = \begin{bmatrix} \frac{1}{\alpha}A_I^{-1}D_{IE}S_E^{-1}D_{ER}S_R^{-1}D_{RE}D_{EI} & A_I^{-1}D_{IE}S_E^{-1}D_{ER}S_R^{-1}D_{RE}\left(\frac{1}{\alpha}A_E - I_\pi\right) \\ -\frac{1}{\alpha}S_E^{-1}D_{ER}S_R^{-1}D_{RE}D_{EI} & -S_E^{-1}D_{ER}S_R^{-1}D_{RE}\left(\frac{1}{\alpha}A_E - I_\pi\right) \end{bmatrix}.$$

Additionally, it is easy to show that

$$Y_\alpha = \underbrace{\begin{bmatrix} A_I^{-1}D_{IE}S_E^{-1}D_{ER}S_R^{-1}D_{RE} \\ -S_E^{-1}D_{ER}S_R^{-1}D_{RE} \end{bmatrix}}_{:=U} \underbrace{\begin{bmatrix} \frac{1}{\alpha}D_{EI} & \frac{1}{\alpha}A_E - I_\pi \end{bmatrix}}_{:=V},$$

which has the same nonzero eigenvalues as Z_α defined in (3.7) and computed by the formula $Z_\alpha = VU$ while the other eigenvalues are 0 with algebraic multiplicity at least n. Then the first desired result can be directly verified through the relation

$$\mathbf{P}^{-1}\mathbf{A} = \mathbf{P}^{-1}(\mathbf{P} - \mathbf{R}) = \mathbf{I}_\pi - \mathbf{P}^{-1}\mathbf{R}$$

and the similarity transformation $\mathbf{A}\mathbf{P}^{-1} = \mathbf{P}(\mathbf{P}^{-1}\mathbf{A})\mathbf{P}^{-1}$ between $\mathbf{A}\mathbf{P}^{-1}$ and $\mathbf{P}^{-1}\mathbf{A}$.

The second part of the theorem can now be proved by expanding $\mathbf{P}^{-1}\mathbf{A}\mathbf{v} = (\mathbf{I}_\pi - \mathbf{P}^{-1}\mathbf{R})\mathbf{v} = \lambda\mathbf{v}$ into

$$v_R - \frac{1}{\alpha}S_R^{-1}D_{RE}D_{EI}v_I - S_R^{-1}D_{RE}\left(\frac{1}{\alpha}A_E - I_\pi\right)v_E = \lambda v_R, \tag{3.14a}$$

$$\left(I_\pi - \frac{1}{\alpha}A_I^{-1}D_{IE}S_E^{-1}D_{ER}S_R^{-1}D_{RE}D_{EI}\right)v_I - A_I^{-1}D_{IE}S_E^{-1}D_{ER}S_R^{-1}D_{RE}\left(\frac{1}{\alpha}A_E - I_\pi\right)v_E = \lambda v_I, \tag{3.14b}$$

$$\frac{1}{\alpha}S_E^{-1}D_{ER}S_R^{-1}D_{RE}D_{EI}v_I + \left[I_\pi + S_E^{-1}D_{ER}S_R^{-1}D_{RE}\left(\frac{1}{\alpha}A_E - I_\pi\right)\right]v_E = \lambda v_E \tag{3.14c}$$

from exploiting (3.13), where $\mathbf{v} = (v_R^\top, v_I^\top, v_E^\top)^\top$ is an eigenvector of $\mathbf{P}^{-1}\mathbf{A}$ associated with the eigenvalue λ. We first assume that $\lambda = 1$. Then, the two formulas (3.14b) and (3.14c) are readily satisfied when the relation (3.9) is true, which just happens to be derived from (3.14a). Under the assumption that $\lambda \neq 1$, the expression

$$(\lambda - 1)v_I = (\lambda - 1)A_I^{-1}D_{IE}S_E^{-1}D_{ER}v_R$$

can be obtained by substituting (3.14a) into (3.14b) while

$$(\lambda - 1)v_E + (\lambda - 1)S_E^{-1}D_{ER}v_R = \mathbf{0}$$

upon substituting (3.14a) into (3.14c). Consequently, the two relations in (3.11) hold. Substituting them into (3.14a) yields

$$S_R^{-1}D_{RE}\left(\frac{1}{\alpha}A_E - I_\pi - \frac{1}{\alpha}D_{EI}A_I^{-1}D_{IE}\right)S_E^{-1}D_{ER}v_R = (\lambda-1)v_R,$$

which indicates that v_R is the eigenvector of (3.10) associated with λ by exploiting the second relation in (3.8). This completes the proof. □

A low degree minimal polynomial in its left- or right-preconditioned matrix is a well-known sufficient condition for searching a good preconditioning strategy [47]. We can deduce the following result from utilizing Theorem 3.1 in a similar way to [5, Proposition 2.2] and [22, Theorem 4].

Theorem 3.2. *The degree of the minimal polynomial of* $\mathbf{P}^{-1}\mathbf{A}$ *or* $\mathbf{A}\mathbf{P}^{-1}$*, namely, the dimension of the Krylov subspace* $\mathscr{K}(\mathbf{P}^{-1}\mathbf{A}, \mathbf{r}_0)$ *or* $\mathscr{K}(\mathbf{A}\mathbf{P}^{-1}, \mathbf{r}_0)$ *for an arbitrary vector* $\mathbf{r}_0$*, is at most* $n+1$.

Remark 3.3. It is easily seen from Theorem 3.2 that at most $n+1$ iterations of the full GMRES algorithm are required for convergence, applying the preconditioner $\mathbf{P}$ from the left or right, when solving the flux-limited MGD linear system (2.4).

4. Practical Implementation and Performance Evaluation

4.1. Sequential implementation issue

The SRS preconditioner, defined by (3.3), is applied frequently to accelerate the convergence rate of the FGMRES(m) solver. Nonetheless, the resulting solver involves the solutions of a suite of the generalized residual equations

$$\mathbf{P}\mathbf{w} \equiv \begin{bmatrix} A_R & \frac{1}{\alpha}D_{RE}D_{EI} & \frac{1}{\alpha}D_{RE}A_E \\ O & A_I & D_{IE} \\ D_{ER} & D_{EI} & A_E \end{bmatrix}\begin{bmatrix} w_R \\ w_I \\ w_E \end{bmatrix} = \begin{bmatrix} b_R \\ b_I \\ b_E \end{bmatrix} \equiv \mathbf{b}, \tag{4.1}$$

where $\mathbf{w}$ and $\mathbf{b}$ are the outgoing Krylov solution and an arbitrarily incoming Krylov vector, respectively. Utilizing the block tri-factorization (3.12), the residual equation (4.1) can be treated by solving the system

$$\begin{bmatrix} S_R & O & \frac{1}{\alpha}D_{RE} \\ O & I_\pi & O \\ O & O & I_\pi \end{bmatrix}\begin{bmatrix} u_R \\ u_I \\ u_E \end{bmatrix} = \begin{bmatrix} b_R \\ b_I \\ b_E \end{bmatrix},$$

first. After that we solve the system

$$\begin{bmatrix} I_\pi & O & O \\ O & I_\pi & O \\ D_{ER} & D_{EI}A_I^{-1} & I_\pi \end{bmatrix}\begin{bmatrix} v_R \\ v_I \\ v_E \end{bmatrix} = \begin{bmatrix} u_R \\ u_I \\ u_E \end{bmatrix},$$

and finally the system

$$\begin{bmatrix} I_\pi & O & O \\ O & A_I & D_{IE} \\ O & O & S_E \end{bmatrix} \begin{bmatrix} w_R \\ w_I \\ w_E \end{bmatrix} = \begin{bmatrix} v_R \\ v_I \\ v_E \end{bmatrix}.$$

Therefore, a specific implementation procedure for computing the generalized residual solution vector **w** is as follows:

1. Radiation segment. Solve w_R from $S_R w_R = b_R - (1/\alpha) D_{RE} b_E$, which is approximated by
$$\left(A_g - \frac{1}{\alpha} D_{gE} D_{Eg}\right) w_g = b_g - \frac{1}{\alpha} D_{gE} b_E, \quad g = 1, \ldots, G. \tag{4.2}$$
2. Solve $\tilde{v}_I$ from $A_I \tilde{v}_I = b_I$ and set
$$v_E = b_E - \sum_{g=1}^{G} D_{Eg} w_g - D_{EI} \tilde{v}_I.$$
3. Electron segment. Solve w_E from $S_E w_E = v_E$, which is approximatively accomplished by
$$(A_E - D_{EI} \Lambda_I^{-1} D_{IE}) w_E = v_E$$
with Λ_I representing the row-Schur norm (a type of diagonal approximation) of A_I defined by
$$\Lambda_I = \operatorname{diag}\left(\sqrt{\sum_{j=1}^{n} \left(a_{1j}^I\right)^2}, \cdots, \sqrt{\sum_{j=1}^{n} \left(a_{nj}^I\right)^2} \right), \tag{4.3}$$
where a_{ij}^I is the (i,j)-th entry of A_I.
4. Ion segment. Compute $p_I = b_I - D_{IE} w_E$ and solve w_I from $A_I w_I = p_I$.

It is easy to see from the above procedure that there are $G+3$ sparse linear subsystems in the same nonzero structure needed to solve, either by a designated number of best practices algebraic multigrid V-cycles [14] or by iterating till a prescribed relative tolerance is reached.

Remark 4.1. A significant advantage of diagonal approximations (4.2) over the full block matrix S_R is that they can be carried out in embarrassingly parallel, which may lead to a slight increase in the iteration count of the FGMRES(m) solver.

Remark 4.2. It should be mentioned that Step 4 is mathematically equivalent to

$4^\dagger$. (The corrected ion segment). Compute $p_I = D_{IE} w_E$, solve w_I from $A_I w_I = p_I$ and set $w_I = \tilde{v}_I - w_I$,

which is exactly our practical implementation of this step.

Remark 4.3. It has been revealed by the comparison of numerical results in [56] that, when constructing block preconditioning approaches, the row-Schur norm approximation (4.3) is not inferior to and often outperforms other popular choices, such as the classical diagonal, row-maximum and row-infinity approximations.

Remark 4.4. For an application of the relaxed splitting preconditioner $\hat{\mathbf{P}}$ defined by (3.1), i.e. to perform the preconditioning operation $\mathbf{w} = \hat{\mathbf{P}}^{-1}\mathbf{b}$, the solution vector $\mathbf{w}$ can be computed by the following procedure:

1. Solve $v_R = (v_1^\top, \cdots, v_G^\top)^\top$ from $A_g v_g = b_g$ for $g = 1, \ldots, G$.
2. Calculate $v_E = b_E - D_{ER} v_R - \hat{\alpha} D_{EI} b_I$.
3. Electron segment. Solve w_E from $\hat{S}_E w_E = v_E$.
4. Ion segment. Compute $p_I = b_I - D_{IE} w_E$ and solve w_I from $A_I w_I = p_I$.
5. Radiation segment. Set $w_R = v_R - \hat{\alpha} D_{RE} w_E$.

In this situation, it is clear that we need to operate $G + 2$ sparse sub-matrix inverses, one fewer inverse than that of $\mathbf{P}$, however, with more expensive manipulations and communications (in parallel) to determine $\hat{\alpha}^*$ by the formula (3.6). In addition, the FGMRES(m) solver preconditioned by $\mathbf{P}$ would be more efficient when its iteration count is less than that of $\hat{\mathbf{P}}$.

4.2. Parallel implementation details

A critical issue when investigating and surveying the thermal radiation transport process, such as the hydrodynamic Rayleigh-Taylor instability during the deceleration phase of a laser-driven spherical implosion [18], is the supersized computational load driven by an ultrahigh mesh resolution, which imposes a large requirement for the software and hardware resources. The increase in hardware resource demand is quite justifiable with the ever-increasing computational availability of supercomputers. Meanwhile, an appropriately designed distributed and parallel algorithm, which can be scalable to a great deal of processors and ensures robustness in relation to different physical model parameters and spatio-temporal mesh step sizes, is required for effective utilization of the modern powerful hardware resources. It should particularly be emphasized that the algorithmic parallelism achieved via supercomputers, which are interconnected by means of a high-speed network and exchange messages exploiting the message passing interface (MPI) library, is the de-facto standard, whose particular concern is the cost of communication. In our MPI-parallel C codes, the two-level parallelization strategy is chosen so that the communication overhead can be reduced as much as possible to efficiently make use of the existing supercomputer resources:

- At first, we break up the global communicator containing a total number of $(G+2)q$ parallel processor cores into $G + 2$ communication sub-groups, which are differentiated by their owned 'color' value, namely, COMM_R (which is further spanided into

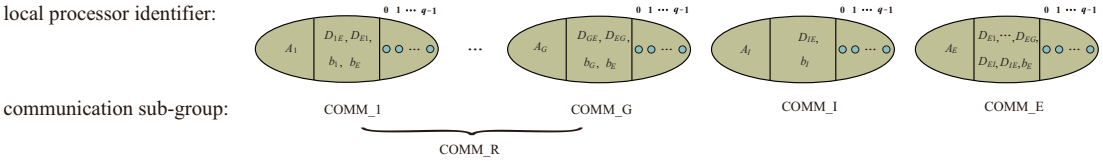

Figure 1: An abridged general view of the $G+2$ communication sub-groups stemmed from a decomposition of the global communicator containing $(G+2)q$ processors, each of them associated with the coefficient matrix (left), the right-hand side vector and some additional vectors (middle) and q parallel processor cores (right).

COMM_1, $\cdots$, COMM_G), COMM_I and COMM_E. This procedure is invariably accomplished by invoking the MPI function 'MPI_Comm_split' and illustrated in Fig. 1, which aims at apportioning sparse linear subsystems accompanied by some additional vectors among communication sub-groups in a consistent one-to-one matching manner. Furthermore, each communication sub-group is comprised of q processors labeled with local continuous index — i.e. 'key', values, starting from 0, as shown in Fig. 1.

- The second-level parallelization is embodied in the distribution of degrees of freedom (i.e., rows) of the sub-matrix A_g, $g = 1,\ldots,G,I,E$ and sub-vectors (including off-diagonal sub-blocks $D_{gg'}$ and some auxiliary vectors) onto different processors within the communication sub-group COMM_g in a roughly uniform mode (i.e., at most one more row is fallen into a certain processor compared with the other processor cores). Concretely speaking, A_g is stored in the parallel CSR (compressed sparse row) matrix format, namely, the ParCSRMatrix container in the hypre (high performance preconditioners and solvers featuring multigrid) software package [15], while sub-vectors are in the parallel vector storage scheme (i.e., the ParVector container in hypre).

Note that the communication mechanism for exchanging messages among these communication sub-groups is rather simple: communications only take place between processors with the same 'key' value — i.e. local identifier.

Remark 4.5. The 'row-wise distribution' in the second-level parallelization phase is equivalent to the direct row-wise decomposition of (2.4) onto the global communicator only when n can be divisible by q.

Remark 4.6. The positive parameters α^* defined by (3.5) and $\hat{\alpha}^*$ defined by (3.6) must be determined at the cost of the computation of all diagonal entries of one and $G+1$ (however, in embarrassingly parallel) ParCSRMatrix-by-ParCSRMatrix multiplications, respectively, which are easy to implement via certain modifications to the subroutine 'hypre_ParMatmul' in hypre to simply generate the first — i.e. the diagonal element of each row, which means that we do not have to calculate the entire product.

We present the parallel operation flowchart and the data communication network topology on an individual application call of the proposed preconditioner **P** in Fig. 2, whose

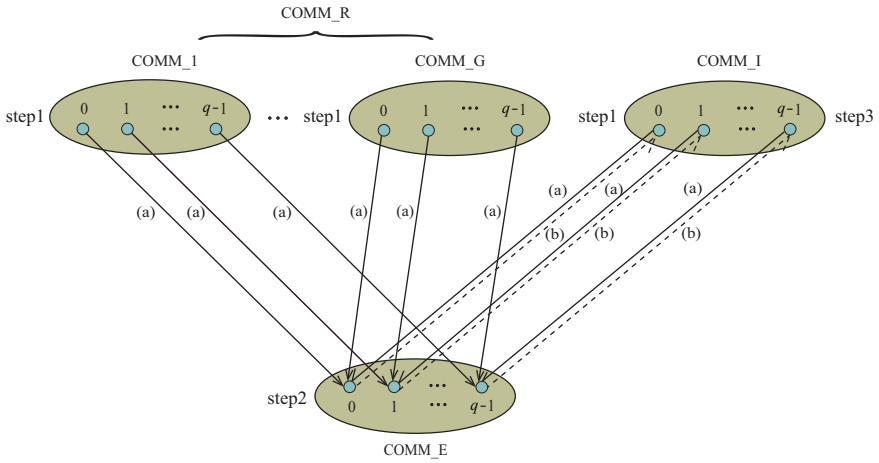

Figure 2: The diagrammatic drawing of the parallel operation flowchart and the data communication mode of a single application of the proposed preconditioner **P**.

detailed interpretation is as follows:

1. Parallel operation (step1). Within the communication sub-group COMM_g, $g = 1,\ldots,G$, solve for w_g from $(A_g - (1/\alpha)D_{gE}D_{Eg})w_g = b_g - (1/\alpha)D_{gE}b_E$ via BoomerAMG — a frequently-used and prestigious parallel implementation of classical algebraic multigrid in hypre [24] in a specified number of iterations $n_g^{\max}$ and a prescribed relative tolerance δ_g while in parallel, inside the communication sub-group COMM_I, seek for the approximate solution $\tilde{v}_I$ provided by $A_I\tilde{v}_I = b_I$ using BoomerAMG with $n_I^{\max}$ and δ_I prescribed.

2. Data transfer (a). Send the real arrays w_g, $g = 1,\ldots,G$ and $\tilde{v}_I$ from COMM_g and COMM_I, respectively, to COMM_E among processors of the same 'key' value.

3. Parallel operation (step2). Within COMM_E, receive the data packets, update $b_E \leftarrow b_E - \sum_{g=1}^{G} D_{Eg}w_g - D_{EI}\tilde{v}_I$ and determine the numerical solution w_E from $(A_E - D_{EI}\Lambda_I^{-1}D_{IE})w_E = b_E$ by exploiting BoomerAMG with $n_E^{\max}$ and δ_E prescribed.

4. Data backhaul (b). Among processors of the same 'key' value, send the resulting real array w_E from COMM_E to COMM_I.

5. Parallel operation (step3). Within COMM_I, after the piece of data is received, generate $p_I = D_{IE}w_E$, numerically solve $A_Iw_I = p_I$ via BoomerAMG to obtain w_I and update $w_I \leftarrow \tilde{v}_I - w_I$.

It is important to mention that the computation-communication overlap technique[†] is used in the above algorithm and the setup phase of **P** including the generation of the quasi-optimal parameter α and the row-Schur norm Λ_I (stored in a real array) defined by (3.5)

[†]The computation-communication overlap is to interlard as many computations as possible between a non-blocking communication call — i.e. the MPI function 'MPI_Isend' and its blocking wait primitive 'MPI_Wait'.

and (4.3), respectively, and the 'setup' phase of BoomerAMG on matrices $A_g - (1/\alpha)D_{gE}D_{Eg}$, A_I and $A_E - D_{EI}\Lambda_I^{-1}D_{IE}$. Also, the generation of Λ_I occurs within COMM_I, and then the array is sent to the processor with the same local identifier within COMM_E while the generation of α requires a bit complicated procedures:

1. Within COMM_E, calculate the auxiliary array trace(A_E^2) while in parallel, inside COMM_I, compute D_{EI}^2.

2. Send trace(A_E^2) and diag(A_E) from COMM_E together with D_{EI}^2 from COMM_I to COMM_g, $g = 1, \ldots, G$ among processors of the same 'key' value.

3. Compute the numerator and denominator of formula (3.5) via the MPI reduction operation 'MPI_Reduce' and invoke the MPI broadcast function 'MPI_Bcast' to copy the required parameter α in the root processor to all leaf processors.

4.3. Experimental setup

We take advantage of three unstructured (adaptive) computational grids (namely, $\mathscr{M}_0$, $\mathscr{M}_1$ and $\mathscr{M}_2$) with different numbers of mesh cells under the JAUMIN (J adaptive unstructured meshes applications infrastructure) framework [35] to investigate the grid-independent convergence and seven flux-limited MGD linear systems (i.e., the 3rd, 2nd, 1st, 3rd, 1st, 2nd and 1st nonlinear iterations at the 27th, 86th, 35th, 92nd, 41st, 113rd and 174th time levels, which are symbolized by U27-3, U86-2, U35-1, U92-3, U41-1, U113-2 and U174-1) to research the robustness under different physical parameters. Fig. 3 shows two side elevations of the coarsest unstructured computational grid $\mathscr{M}_0$. Two other unstructured meshes, $\mathscr{M}_1$ and $\mathscr{M}_2$, are created on the trot by refining $\mathscr{M}_0$ once and twice, respectively. Initial and boundary value conditions of the mentioned real-world twenty-group ($G = 20$) simulations are: the initial radiation energy densities are all computed through the Plank interpolation formula [8] with the contained radiation temperature being 3.0×10^{-4} while the initial ion and electron temperatures are also set to 3.0×10^{-4}; concerning the ion and electron temperature variables, we impose the zero-flow condition at all physical boundaries, while, regarding the radiation energy density variables, the zero-flow condition is imposed at the angle direction and spherical center as well as the inflow condition at the outer radius.

Numerical tests are performed on the Tianhe-2A supercomputer at China's National Supercomputer Center in Guangzhou (NSCC-GZ), currently ranked 16th in the June 2024 TOP500 list — cf. `https://www.top500.org/lists/top500/list/2024/06/`, which employs Kylin Linux operating system and delivers 100.68 petaflops peak performance in theory and 61.44 petaflops Linpack performance with 17,792 compute nodes. Each of these nodes is assembled with dual 12-core Intel Xeon E5-2692v2 central processing units (24 compute cores in total) clocked at 2.2 gigahertz and 64.0 gigabytes of DDR3 main memory as well as the Matrix-2000 processor for performance acceleration. The proprietary high-speed TH Express-2 interconnect network topology is an opto-electronic hybrid and hierarchical fat tree. We utilize the Intel C compiler (icc) with Tianhe's self-optimized

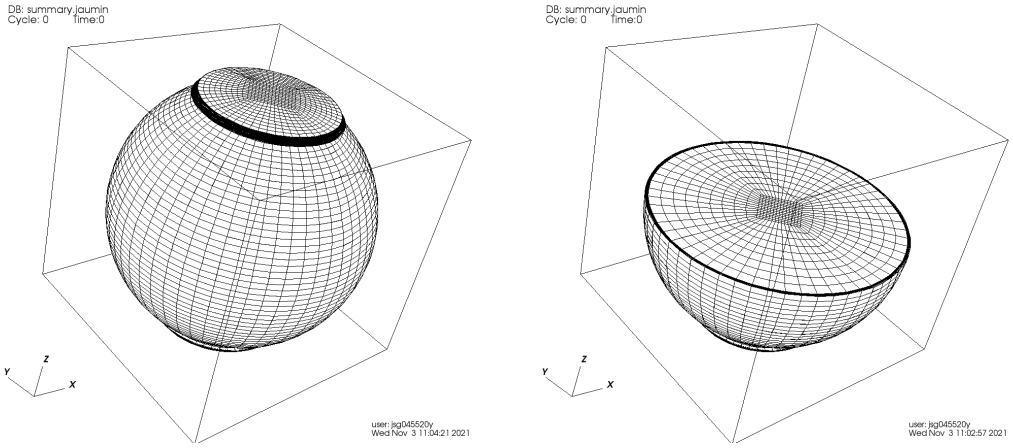

Figure 3: External and internal side elevations of the coarsest computational grid $\mathscr{M}_0$ with 372,708 mesh cells.

mpich-3.2 in MPI-only mode (i.e., the 'configure' script is carried out with '`-with-MPI`' and '`-without-openmp`' options and the pure MPI implementation is investigated) and the optimization flag '-O3 -mavx' as our practical compilation tool, and link all C codes to Intel MKL composer_xe_2015.1.133. All of the 24 MPI processes (i.e., physical cores) are made use of running parallel codes on each compute node.

In the next subsection we numerically inspect the actual convergence, performance and scalability of the FGMRES(m) solvers preconditioned by BoomerAMG [24], the relaxed splitting preconditioner $\hat{\mathbf{P}}$ defined by (3.1), the SRS preconditioner $\mathbf{P}$ defined by (3.3), and the physical-variable based coarsening two-level (PCTL) preconditioner [27,50,57], which are effectively implemented in the hypre [15] and JXPAMG (parallel algebraic multigrid solvers developed by JiuSuo and XTU) [52] software libraries, respectively. What is particularly noteworthy is that, for the solution of all the solitary sparse linear subsystems derived from discrete linear scalar second-order reaction-diffusion operators and involved in practical implementations of $\hat{\mathbf{P}}$, $\mathbf{P}$ and PCTL algorithms, a single BoomerAMG V(1,1)-cycle with 1 pre- and 1 post-smoothing step is simply utilized — i.e. $n_g^{\max} = 1$ and $\delta_g = 10^{-4}$ (which, usually, cannot be reached) in the parallel operations (step1), (step2) and (step3) for the index $g = 1,\ldots,G,I,E$. All of the invoked applications of BoomerAMG are in its best practices scenario [14], i.e. using commonly recommended by the hypre configuration parameters developers — viz.

- In the setup phase, a strength-of-connection measure of 0.25 in the HMIS coarsening strategy (coarsen_type: 10), the aggressive coarsening scheme on the finest level (agg_num_levels: 1), the 'extended+i' interpolation strategy (interp_type: 6) followed by a truncation to no more than five nonzero entries per row (P_max_elmts: 5), the coarse-grid operator (i.e., the triple sparse matrix product) determined algebraically via the Galerkin approach and the termination of the coarsening process when the coarse-grid size (i.e., the number of current degrees of freedom) is less than 100 (max_coarse_size: 100).

- In the solve phase, the Gaussian elimination (grid_relax_type[3]: 9) as the smoother at the coarsest grid level while, at the other grid levels, one sweep of the hybrid ℓ_1 Gauss-Seidel smoothing in the 'Coarse-Fine' forward ordering on the down cycle (relax_type: 13) and 'Fine-Coarse' backward ordering on the up cycle (relax_type: 14).

For the numerical results on real-world test cases described above in the next subsection, we choose the restart parameter $m = 30$, use a relative convergence tolerance of 10^{-8} in Euclidean norms of the current residual vector and the right-hand side vector (since the initial guess vector is zero) and the maximum number of the preconditioned FGMRES(m) iteration steps $iter_{\max} = 200$ as our stopping criterion, and further report the realistic number of iteration steps $iter_{np}$ and total elapsed time-to-solution $time_{np}^{tot}$ in seconds (averaged over 10 test runs and measured via the MPI function 'MPI_Wtime') that the preconditioned FGMRES(m) solver would require to converge when our MPI parallel code is executed across np MPI tasks. Also, the parallel weak or strong efficiency $efcy_p^k$ evaluated by $time_p^{tot}/time_{kp}^{tot}$ or $time_p^{tot}/(k \cdot time_{kp}^{tot})$ [41] is provided below.

4.4. Results and discussions

4.4.1. Convergence and efficiency comparisons

Here, of particular practical interests are the convergence performance and computational efficiency on one processor of the four integrated preconditioned FGMRES(m) solvers when dealing with the seven twenty-group linear systems mentioned above over two unstructured adaptive computational meshes $\mathscr{M}_0$ and $\mathscr{M}_1$ with 8,199,576 and 65,596,608 unknowns.

As reported in Table 1, we first see overall similar and good enough robustness and high convergence rate (in terms of the number of iterations) of the FGMRES(30) solvers preconditioned with $\hat{\mathbf{P}}$ defined by (3.1) and $\mathbf{P}$ defined by (3.3) under the computational grid refinement and the physical parameter variation, namely, their convergence are both achieved with roughly the same number of iterations (in no more than 11 steps) regardless of the problem size and numeration.

Furthermore, BoomerAMG does not iterate robustly in regard to the spatial mesh size and physical parameters — e.g. radiative free paths of possibly rather different scales, and exhibits a low convergence rate for U86-2, U92-3, U113-2 and U174-1, while two test cases — viz. U92-3 and U113-2, are not solvable by the PCTL algorithm within the maximum number of iteration steps allowed. It is important to mention that, for these two problems, PCTL exhibits a pathological convergence behavior — e.g. for $\mathscr{M}_0$ and $\mathscr{M}_1$ the relative residual norm does not decrease after 16 and 18 and after 21 and 24 iterations, respectively. However, PCTL solves U27-3, U86-2, U35-1, U41-1 and U174-1, with a slightly lower convergence rate than that of preconditioners $\hat{\mathbf{P}}$ and $\mathbf{P}$, in a nearly robust manner.

Taken together, the fastest rate of reduction in the relative residual norm — i.e. the best preconditioning behavior, is obtained by the preconditioner $\mathbf{P}$ defined by (3.3). Moreover, it results in an average of 13.79 and 1.08 times faster than BoomerAMG and the preconditioner $\hat{\mathbf{P}}$ defined by (3.1) on the unstructured adaptive mesh $\mathscr{M}_0$, while, on the refined

Table 1: Iteration counts and elapsed time-to-solutions of four types of right-preconditioned FGMRES(30) solvers applied to solving seven twenty-group linear systems over two unstructured adaptive computational meshes $\mathscr{M}_0$ (top) and $\mathscr{M}_1$ (bottom). Dashed entries (-) indicate the numerical solutions failed to converge after 200 iteration steps while values in parentheses represent the order of magnitude of the final relative residual norm.

	$\mathscr{M}_0$							
	BoomerAMG		$\hat{\mathbf{P}}$ defined by (3.1)		$\mathbf{P}$ defined by (3.3)		PCTL	
	$iter_1$	$time_1^{tot}$	$iter_1$	$time_1^{tot}$	$iter_1$	$time_1^{tot}$	$iter_1$	$time_1^{tot}$
U27-3	17	60.48	5	7.66	5	7.79	12	13.73
U86-2	54	127.73	8	9.81	6	8.54	14	14.84
U35-1	23	71.41	6	8.37	5	7.76	13	14.26
U92-3	69	155.04	9	10.52	7	9.28	- (10^{-5})	-
U41-1	21	67.78	5	7.68	4	7.04	13	14.28
U113-2	80	175.02	8	9.84	7	9.31	- (10^{-4})	-
U174-1	76	167.65	9	10.55	8	10.12	15	15.39
	$\mathscr{M}_1$							
	BoomerAMG		$\hat{\mathbf{P}}$ defined by (3.1)		$\mathbf{P}$ defined by (3.3)		PCTL	
	$iter_1$	$time_1^{tot}$	$iter_1$	$time_1^{tot}$	$iter_1$	$time_1^{tot}$	$iter_1$	$time_1^{tot}$
U27-3	28	815.45	6	68.53	5	64.54	13	137.73
U86-2	81	1645.83	9	82.79	7	74.48	17	154.35
U35-1	35	925.07	7	73.28	6	69.57	15	146.29
U92-3	90	1787.86	8	78.04	7	74.61	- (10^{-4})	-
U41-1	32	878.14	7	73.25	5	64.53	16	150.48
U113-2	97	1896.42	10	87.52	8	79.45	- (10^{-3})	-
U174-1	94	1849.51	11	92.30	9	84.42	19	165.61

mesh $\mathscr{M}_1$, the speedup ratios are 19.15 and 1.09; it runs averagely 1.79 and 2.17 times faster than PCTL on $\mathscr{M}_0$ and $\mathscr{M}_1$, respectively, for the five test problems that can be tackled by PCTL.

4.4.2. Parallel strong and weak scaling performance

A practical and important aspect of distributed parallel solvers is their inherent strong and weak scaling properties. The strong scalability investigation measures the intrinsic parallel performance by taking advantage of a fixed problem with increasing MPI processes while the actual performance of parallel solvers under weak scaling is evaluated with a fixed problem size — i.e. the number of the degrees of freedom, per processor core.

A strong scaling analysis is carried out over the seven test problems considered in this study on the unstructured adaptive mesh $\mathscr{M}_1$ while the number of parallel processor cores used is altered from 88 to 704, doubling every time. Accordingly, 745,416, 372,708, 186,354 and 93,177 degrees of freedom are distributed on each physical core. It is crucial to point out that PCTL is not considered in this subsection for its hyperslow convergence be-

Table 2: Iteration counts, elapsed time-to-solutions and parallel efficiencies of three right-preconditioned FGMRES(30) solvers in a strong scalability investigation.

$\mathscr{M}_1$	BoomerAMG										
	np=88		np=176			np=352			np=704		
	$iter_{88}$	$time_{88}^{tot}$	$iter_{176}$	$time_{176}^{tot}$	$efcy_{88}^{2}$	$iter_{352}$	$time_{352}^{tot}$	$efcy_{176}^{2}$	$iter_{704}$	$time_{704}^{tot}$	$efcy_{352}^{2}$
U27-3	31	126.36	33	71.09	88.9%	34	42.39	83.8%	37	28.58	74.2%
U86-2	86	261.13	89	140.87	92.7%	93	82.14	85.7%	95	52.70	77.9%
U35-1	39	145.94	41	81.06	90.0%	43	48.45	83.6%	46	32.32	75.0%
U92-3	96	285.62	99	153.34	93.1%	101	87.52	87.6%	103	56.03	78.1%
U41-1	35	136.14	37	76.11	89.5%	39	45.75	83.2%	41	30.25	75.6%
U113-2	103	302.75	105	160.73	94.2%	109	92.91	86.5%	112	59.77	77.7%
U174-1	97	288.07	101	155.82	92.4%	103	88.87	87.7%	104	56.44	78.6%
$\mathscr{M}_1$	$\hat{\mathbf{P}}$ defined by (3.1)										
	np=88		np=176			np=352			np=704		
	$iter_{88}$	$time_{88}^{tot}$	$iter_{176}$	$time_{176}^{tot}$	$efcy_{88}^{2}$	$iter_{352}$	$time_{352}^{tot}$	$efcy_{176}^{2}$	$iter_{704}$	$time_{704}^{tot}$	$efcy_{352}^{2}$
U27-3	6	10.26	6	6.04	84.9%	6	3.75	80.5%	6	2.51	74.8%
U86-2	10	13.59	10	7.76	87.6%	10	4.72	82.3%	11	3.27	72.2%
U35-1	7	11.10	8	6.91	80.4%	8	4.24	81.5%	8	2.81	75.3%
U92-3	9	12.76	9	7.33	87.1%	10	4.69	77.7%	10	3.12	75.7%
U41-1	8	11.93	8	6.88	86.3%	8	4.22	81.4%	9	2.96	71.4%
U113-2	12	15.26	12	8.64	88.5%	13	5.45	79.2%	13	3.57	76.2%
U174-1	11	14.43	11	8.19	88.0%	11	4.96	82.6%	12	3.42	72.5%
$\mathscr{M}_1$	$\mathbf{P}$ defined by (3.3)										
	np=88		np=176			np=352			np=704		
	$iter_{88}$	$time_{88}^{tot}$	$iter_{176}$	$time_{176}^{tot}$	$efcy_{88}^{2}$	$iter_{352}$	$time_{352}^{tot}$	$efcy_{176}^{2}$	$iter_{704}$	$time_{704}^{tot}$	$efcy_{352}^{2}$
U27-3	5	9.58	5	5.67	84.3%	5	3.55	80.1%	5	2.38	74.5%
U86-2	8	12.19	8	7.03	86.7%	9	4.56	77.2%	9	3.01	75.6%
U35-1	6	10.45	7	6.58	79.4%	8	4.30	76.4%	8	2.86	75.4%
U92-3	7	11.32	8	7.01	80.5%	8	4.32	81.7%	9	2.99	71.4%
U41-1	6	10.43	6	6.13	85.2%	7	4.05	75.6%	7	2.70	75.1%
U113-2	8	12.17	8	7.05	86.5%	8	4.33	81.5%	8	2.84	75.3%
U174-1	9	13.06	10	7.93	82.3%	10	4.81	82.4%	11	3.33	72.2%

havior when solving certain flux-limited MGD linear systems. The parallel results tabulated in Table 2 demonstrate that BoomerAMG, $\hat{\mathbf{P}}$ defined by (3.1) and $\mathbf{P}$ defined by (3.3) right-preconditioned FGMRES(30) solvers, in the strong sense, all exhibit excellent numerical and parallel scaling properties, that is, their iteration counts and elapsed time-to-solutions required for convergence with the specified tolerance remain primarily unchanged and appropriately decreased in relation to the number of parallel processor cores: their average strong parallel efficiencies are 91.5%, 85.4% and 76.7% for BoomerAMG, 86.1%, 80.7% and 74.0% for $\hat{\mathbf{P}}$ defined by (3.1) while 83.6%, 79.3% and 74.2% for $\mathbf{P}$ defined by (3.3), respectively. Despite the fact that $\mathbf{P}$ defined by (3.3) is numerically inferior to the other two preconditioners, it reveals the lowest amplitude of relative decrease (actually, 16.2%, 14.1% and 11.2% for these three preconditioners with the number of MPI tasks varied from

Table 3: A weak scalability investigation on the number of iterations, elapsed wall-clock time and parallel efficiencies of three right-preconditioned FGMRES(30) solvers.

	BoomerAMG							
	np=44, $\mathscr{M}_0$		np=352, $\mathscr{M}_1$			np=2,816, $\mathscr{M}_2$		
	$iter_{44}$	$time_{44}^{tot}$	$iter_{352}$	$time_{352}^{tot}$	$efcy_{44}^{8}$	$iter_{2,816}$	$time_{2,816}^{tot}$	$efcy_{352}^{8}$
U27-3	29	33.32	34	42.39	78.6%	40	59.22	71.6%
U86-2	87	76.02	93	82.14	92.5%	104	107.76	76.2%
U35-1	38	39.94	43	48.45	82.4%	51	67.66	72.6%
U92-3	94	81.17	101	87.52	92.7%	109	111.54	78.5%
U41-1	32	35.53	39	45.75	77.6%	45	63.10	72.5%
U113-2	100	85.59	109	92.91	92.1%	117	117.92	78.8%
U174-1	95	81.91	103	88.87	92.2%	108	110.88	80.1%
	$\hat{\mathbf{P}}$ defined by (3.1)							
	np=44, $\mathscr{M}_0$		np=352, $\mathscr{M}_1$			np=2,816, $\mathscr{M}_2$		
	$iter_{44}$	$time_{44}^{tot}$	$iter_{352}$	$time_{352}^{tot}$	$efcy_{44}^{8}$	$iter_{2,816}$	$time_{2,816}^{tot}$	$efcy_{352}^{8}$
U27-3	6	3.14	6	3.75	83.6%	6	5.17	72.6%
U86-2	9	3.77	10	4.72	79.9%	10	6.45	73.2%
U35-1	8	3.56	8	4.24	83.8%	9	6.13	69.2%
U92-3	10	3.98	10	4.69	84.7%	11	6.76	69.3%
U41-1	7	3.35	8	4.22	79.3%	8	5.82	72.5%
U113-2	11	4.19	13	5.45	76.9%	14	7.78	70.0%
U174-1	10	3.95	11	4.96	80.2%	12	7.11	69.8%
	$\mathbf{P}$ defined by (3.3)							
	np=44, $\mathscr{M}_0$		np=352, $\mathscr{M}_1$			np=2,816, $\mathscr{M}_2$		
	$iter_{44}$	$time_{44}^{tot}$	$iter_{352}$	$time_{352}^{tot}$	$efcy_{44}^{8}$	$iter_{2,816}$	$time_{2,816}^{tot}$	$efcy_{352}^{8}$
U27-3	5	2.96	5	3.55	83.3%	5	4.89	72.6%
U86-2	8	3.62	9	4.56	79.4%	10	6.57	69.4%
U35-1	7	3.39	8	4.30	78.9%	8	5.90	72.9%
U92-3	8	3.64	8	4.32	83.7%	9	6.23	69.3%
U41-1	6	3.18	7	4.05	78.4%	7	5.57	72.7%
U113-2	8	3.59	8	4.33	83.6%	9	6.25	69.2%
U174-1	9	3.84	10	4.81	79.8%	11	6.92	69.6%

88 to 704). Besides, it should be mentioned that $\hat{\mathbf{P}}$ defined by (3.1) and $\mathbf{P}$ defined by (3.3) both show an insensitivity toward problem characteristics and $\mathbf{P}$ defined by (3.3) achieves an average speedup of 15.7 and 1.08 over BoomerAMG and $\hat{\mathbf{P}}$ defined by (3.1) when using 704 MPI ranks.

A weak scalability investigation is proceeded to compare BoomerAMG, $\hat{\mathbf{P}}$ defined by (3.1) and $\mathbf{P}$ defined by (3.3) by exploiting 44, 352 and 2,816 parallel processor cores, octupling every time on the strength of 186,354 degrees of freedom per physical core. More exactly, three particular cases $np = 44$, $np = 352$ and $np = 2,816$ correspond to the coarsest, intermediate and finest computational meshes $\mathscr{M}_0$, $\mathscr{M}_1$ and $\mathscr{M}_2$, respectively. It serves the purpose of inspecting the numbers of these three right-preconditioned FGMRES(30) iterations and their elapsed wall-clock time against the number of processor cores. By inspecting Table 3, we observe that the average parallel efficiencies in this weak scaling

examination for BoomerAMG, $\hat{\mathbf{P}}$ defined by (3.1) and $\mathbf{P}$ defined by (3.3) are 75.8%, 70.9% and 70.8%, respectively, when using 2,816 MPI ranks, in which case we further notice 15.1 and 1.1 times higher computational performance for $\mathbf{P}$ defined by (3.3) than for BoomerAMG and $\hat{\mathbf{P}}$ defined by (3.1). It is obvious that the two proposed preconditioners $\hat{\mathbf{P}}$ defined by (3.1) and $\mathbf{P}$ defined by (3.3) both weakly scale well from the viewpoint of numerical and parallel scalabilities (with respect to the number of iterations and elapsed wall-clock time, respectively) and have the same desirable weak scaling property.

5. Conclusion and Outlook

We have proposed an improved relaxed splitting preconditioner and a selectively relaxed splitting preconditioner both with algebraic multigrid subsolves and their easy-to-implement algebraic estimations on the respective contributing parameter for efficient monolithic solutions of three-dimensional flux-limited MGD equations. The spectral distribution and the minimal polynomial of the latter preconditioned matrix have been rigorously proved. We have highlighted their stable behaviors to different physical parameters and spatial mesh lengths, remarkable overall computational efficiency and excellent numerical and parallel scalabilities in strong and weak senses. We foresee the extension of the present work in four ways: (1) mixed precision preconditioning and solution algorithms [1], (2) Krylov subspace recycling strategy [38], (3) Gaussian process regression technique based on the Bayesian inference to predict the optimal parameter [29] and (4) computations on heterogeneous architectures consisting of central and graphics processing devices for explorations of the MPI+OpenMP/CUDA-based parallelism.

Acknowledgments

The authors sincerely appreciate the editor and two anonymous referees for their valuable comments and constructive revision suggestions that have greatly improved the original manuscript.

This work was supported in part by the National Natural Science Foundation of China (Grant No. 12371373), by the Hunan National Applied Mathematics Center (Grant No. 2020ZYT003) and by the Research Foundation of Education Bureau of Hunan (Grant No. 21B0162).

References

[1] A. Abdelfattah, H. Anzt, E.G. Boman, E. Carson, T. Cojean, J. Dongarra, A. Fox, M. Gates, N.J. Higham, X.S. Li, J. Loe, P. Luszczek, S. Pranesh, S. Rajamanickam, T. Ribizel, B.F. Smith, K. Swirydowicz, S. Thomas, S. Tomov, Y.M. Tsai and U.M. Yang, *A survey of numerical linear algebra methods utilizing mixed-precision arithmetic*, Int. J. High Perform. Comput. Appl. **35**, 344–369 (2021).

[2] P.R. Amestoy, A. Buttari, J.Y. L'Excellent and T. Mary, *Performance and scalability of the block*

low-rank multifrontal factorization on multicore architectures, ACM Trans. Math. Software **45**, 2 (2019).

[3] H.B. An, Z.Y. Mo, X.W. Xu and X.W. Jia, *Operator-based preconditioning for the 2-D 3-T energy equations in radiation hydrodynamics simulations*, J. Comput. Phys. **385**, 51–74 (2019).

[4] Z.Z. Bai, G.H. Golub and M.K. Ng, *Hermitian and skew-Hermitian splitting methods for non-Hermitian positive definite linear systems*, SIAM J. Matrix Anal. Appl. **24**, 603–626 (2003).

[5] Z.Z. Bai and M.K. Ng, *On inexact preconditioners for nonsymmetric matrices*, SIAM J. Sci. Comput. **26**, 1710–1724 (2005).

[6] C. Baldwin, P.N. Brown, R. Falgout, F. Graziani and J. Jones, *Iterative linear solvers in 2D radiation-hydrodynamics code: Methods and performance*, J. Comput. Phys. **154**, 1–40 (1999).

[7] M. Benzi, M. Ng, Q. Niu and Z. Wang, *A relaxed dimensional factorization preconditioner for the incompressible Navier-Stokes equations*, J. Comput. Phys. **230**, 6185–6202 (2011).

[8] T.H. Boyer, *Derivation of the Planck radiation spectrum as an interpolation formula in classical electrodynamics with classical electromagnetic zero-point radiation*, Phys. Rev. D **27**, 2906–2911 (1983).

[9] P.N. Brown and C.S. Woodward, *Preconditioning strategies for fully implicit radiation diffusion with material-energy transfer*, SIAM J. Sci. Comput. **23**, 499–516 (2001).

[10] Y. Cao, S.X. Miao and Y.S. Cui, *A relaxed splitting preconditioner for generalized saddle point problems*, Comput. Appl. Math. **34**, 865–879 (2015).

[11] R. Chauvin, S. Guisset, B. Manach-Perennou and L. Martaud, *A colocalized scheme for three-temperature grey diffusion radiation hydrodynamics*, Commun. Comput. Phys. **31**, 293–330 (2022).

[12] A.J. Cleary, R.D. Falgout, V.E. Henson, J.E. Jones, T.A. Manteuffel, S.F. McCormick, G.N. Miranda and J.W. Ruge, *Robustness and scalability of algebraic multigrid*, SIAM J. Sci. Comput. **21**, 1886–1908 (2000).

[13] T.A. Davis, *Algorithm 832: UMFPACK–an unsymmetric-pattern multifrontal method with a column preordering strategy*, ACM Trans. Math. Software **30**, 196–199 (2004).

[14] R.D. Falgout and J.B. Schroder, *Non-Galerkin coarse grids for algebraic multigrid*, SIAM J. Sci. Comput. **36**, C309–C334 (2014).

[15] R.D. Falgout and U.M. Yang, *hypre: A library of high performance preconditioners*, Lect. Notes Comput. Sci. **2331**, 632–641 (2002).

[16] H.T. Fan and X.Y. Zhu, *A modified relaxed splitting preconditioner for generalized saddle point problems from the incompressible Navier-Stokes equations*, Appl. Math. Lett. **55**, 18–26 (2016).

[17] H.T. Fan, X.Y. Zhu, Y.J. Li and B. Zheng, *A variant of relaxed triangular splitting preconditioners for generalized saddle point problems from Navier-Stokes equations*, Appl. Math. Comput. **362**, 124495 (2019).

[18] Z.F. Fan, S.P. Zhu, W.B. Pei, W.H. Ye, M. Li, X.W. Xu, J.F. Wu, Z.S. Dai and L.F. Wang, *Numerical investigation on the stabilization of the deceleration phase Rayleigh-Taylor instability due to alpha particle heating in ignition target*, EPL **99**, 65003 (2012).

[19] P. Ghysels, X.S. Li, F.H. Rouet, S. Williams and A. Napov, *An efficient multicore implementation of a novel HSS-structured multifrontal solver using randomized sampling*, SIAM J. Sci. Comput. **38**, S358–S384 (2016).

[20] R. Glowinski and J. Toivanen, *A multigrid preconditioner and automatic differentiation for non-equilibrium radiation diffusion problems*, J. Comput. Phys. **207**, 354–374 (2005).

[21] M. González, N. Vaytet, B. Commerçon and J. Masson, *Multigroup radiation hydrodynamics with flux-limited diffusion and adaptive mesh refinement*, A&A **578**, A12 (2015).

[22] L. Grigori, Q. Niu and Y.X. Xu, *Stabilized dimensional factorization preconditioner for solving incompressible Navier-Stokes equations*, Appl. Numer. Math. **146**, 309–327 (2019).

[23] P. Hénon, P. Ramet and J. Roman, *PaStiX: A high-performance parallel direct solver for sparse symmetric definite systems*, Parallel Comput. **28**, 301–321 (2002).
[24] V.E. Henson and U.M. Yang, *BoomerAMG: A parallel algebraic multigrid solver and preconditioner*, Appl. Numer. Math. **41**, 155–177 (2002).
[25] M.R. Hestenes and E. Stiefel, *Methods of conjugate gradients for solving linear systems*, J. Res. Nat. Bur. Stand. **49**, 409–436 (1952).
[26] Q.Y. Hu and L. Zhao, *Domain decomposition preconditioners for the system generated by discontinuous Galerkin discretization of 2D-3T heat conduction equations*, Commun. Comput. Phys. **22**, 1069–1100 (2017).
[27] S.L. Huang, X.Q. Yue and X.W. Xu, *αSetup-PCTL: An adaptive setup-based two-level preconditioner for sequence of linear systems of three-temperature energy equations*, Commun. Comput. Phys. **32**, 1287–1309 (2022).
[28] Y.M. Huang, *A practical formula for computing optimal parameters in the HSS iteration methods*, J. Comput. Appl. Math. **255**, 142–149 (2014).
[29] K. Jiang, X.H. Su and J. Zhang, *A general alternating-direction implicit framework with Gaussian process regression parameter prediction for large sparse linear systems*, SIAM J. Sci. Comput. **44**, A1960–A1988 (2022).
[30] L.M. Kačanov, *Variational methods of solution of plasticity problems*, J. Appl. Math. Mech. **23**, 880–883 (1959).
[31] Y.F. Ke and C.F. Ma, *An inexact modified relaxed splitting preconditioner for the generalized saddle point problems from the incompressible Navier-Stokes equations*, Numer. Algorithms **75**, 1103–1121 (2017).
[32] Y.F. Ke and C.F. Ma, *A new relaxed splitting preconditioner for the generalized saddle point problems from the incompressible Navier-Stokes equations*, Comput. Appl. Math. **37**, 515–524 (2018).
[33] C.X. Li, S.L. Wu and A.Q. Huang, *A relaxed splitting preconditioner for saddle point problems*, J. Numer. Math. **23**, 361–368 (2015).
[34] X.S. Li, *An overview of SuperLU: Algorithms, implementation, and user interface*, ACM Trans. Math. Software **31**, 302–325 (2005).
[35] Q.K. Liu, Z.Y. Mo, A.Q. Zhang and Z. Yang, *JAUMIN: A programming framework for large-scale numerical simulation on unstructured meshes*, CCF Trans. High Perform. Comput. **1**, 35–48 (2019).
[36] V.A. Mousseau and D.A. Knoll, *New physics-based preconditioning of implicit methods for non-equilibrium radiation diffusion*, J. Comput. Phys. **190**, 42–51 (2003).
[37] C.C. Paige and M.A. Saunders, *Solution of sparse indefinite systems of linear equations*, SIAM J. Numer. Anal. **12**, 617–629 (1975).
[38] M.L. Parks, E. de Sturler, G. Mackey, D.D. Johnson and S. Maiti, *Recycling Krylov subspaces for sequences of linear systems*, SIAM J. Sci. Comput. **28**, 1651–1674 (2006).
[39] R. Rodríguez, G. Espinosa and J.M. Gil, *MIXKIP/RAPCAL: A computational package for integrated simulations of large-scale atomic kinetics and radiation transport in non-local thermodynamic equilibrium plasmas*, Commun. Comput. Phys. **30**, 602–634 (2021).
[40] Y. Saad and M.H. Schultz, *GMRES: A generalized minimal residual algorithm for solving non-symmetric linear systems*, SIAM J. Sci. Stat. Comput. **7**, 856–869 (1986).
[41] S. Sahni and V. Thanvantri, *Performance metrices: Keeping the focus on runtime*, IEEE Parall. Distrib. **4**, 43–56 (1996).
[42] O. Schenk, K. Gärtner, W. Fichtner and A. Stricker, *PARDISO: A high-performance serial and parallel sparse linear solver in semiconductor device simulation*, Future Gener. Comput. Syst. **18**, 69–78 (2001).

[43] S. Shu, M.H. Liu, X.W. Xu, X.Q. Yue and S.G. Li, *Algebraic multigrid block triangular preconditioning for multidimensional three-temperature radiation diffusion equations*, Adv. Appl. Math. Mech. **13**, 1203–1226 (2021).
[44] S. Su, H.Z. Tang and J.M. Wu, *An efficient positivity-preserving finite volume scheme for the nonequilibrium three-temperature radiation diffusion equations on polygonal meshes*, Commun. Comput. Phys. **30**, 448–485 (2021).
[45] N.B. Tan, T.Z. Huang and Z.J. Hu, *A relaxed splitting preconditioner for the incompressible Navier-Stokes equations*, J. Appl. Math. **2012**, 402490 (2012).
[46] H.A. van der Vorst, *Bi-CGSTAB: A fast and smoothly converging variant of Bi-CG for the solution of nonsymmetric linear systems*, SIAM J. Sci. Stat. Comput. **13**, 631–644 (1992).
[47] A.J. Wathen, *Preconditioning*, Acta Numer. **24**, 329–376 (2015).
[48] J. Xia, S. Chandrasekaran, M. Gu and X.S. Li, *Superfast multifrontal method for large structured linear systems of equations*, SIAM J. Matrix Anal. Appl. **31**, 1382–1411 (2009).
[49] X.W. Xu and Z.Y. Mo, *Algebraic interface-based coarsening AMG preconditioner for multiscale sparse matrices with applications to radiation hydrodynamics computation*, Numer. Linear Algebra Appl. **24**, e2078 (2017).
[50] X.W. Xu, Z.Y. Mo and H.B. An, *Algebraic two-level iterative method for 2-D 3-T radiation diffusion equations*, Chin. J. Comput. Phys. **26**, 1–8 (2009).
[51] X.W. Xu, Z.Y. Mo, X.Q. Yue, H.B. An and S. Shu, *αSetup-AMG: An adaptive-setup-based parallel AMG solver for sequence of sparse linear systems*, CCF Trans. High Perform. Comput. **2**, 98–110 (2020).
[52] X.W. Xu, X.Q. Yue, R.Z. Mao, Y.T. Deng, S.L. Huang, H.F. Zou, X. Liu, S.L. Hu, C.S. Feng, S. Shu and Z.Y. Mo, *JXPAMG: A parallel algebraic multigrid solver for extreme-scale numerical simulations*, CCF Trans. High Perform. Comput. **5**, 72–83 (2023).
[53] X.Q. Yue, J.M. He, X.W. Xu, S. Shu and L.B. Wang, *Two physics-based Schwarz preconditioners for three-temperature radiation diffusion equations in high dimensions*, Commun. Comput. Phys. **32**, 829–849 (2022).
[54] X.Q. Yue, S. Shu, J.X. Wang and Z.Y. Zhou, *Substructuring preconditioners with a simple coarse space for 2-D 3-T radiation diffusion equations*, Commun. Comput. Phys. **23**, 540–560 (2018).
[55] X.Q. Yue, S. Shu, X.W. Xu and Z.Y. Zhou, *An adaptive combined preconditioner with applications in radiation diffusion equations*, Commun. Comput. Phys. **18**, 1313–1335 (2015).
[56] X.Q. Yue, C.Q. Wang, X.W. Xu, L.B. Wang and S. Shu, *A new relaxed splitting preconditioner for the multidimensional multi-group radiation diffusion equations*, East Asian J. Appl. Math. **12**, 163–184 (2022).
[57] X.Q. Yue, S.L. Zhang, X.W. Xu, S. Shu and W.D. Shi, *Algebraic multigrid block preconditioning for multi-group radiation diffusion equations*, Commun. Comput. Phys. **29**, 831–852 (2021).
[58] X.Q. Yue, Z.K. Zhang, X.W. Xu, S.Y. Zhai and S. Shu, *An algebraic multigrid-based physical factorization preconditioner for the multi-group radiation diffusion equations in three dimensions*, Numer. Math. Theory Methods Appl. **16**, 410–432 (2023).
[59] S.W. Zhou, A.L. Yang and Y.J. Wu, *A relaxed block-triangular splitting preconditioner for generalized saddle-point problems*, Int. J. Comput. Math. **94**, 1609–1623 (2017).