Geophysical Journal International

Geophys. J. Int. (2024) **238,** 631–645
Advance Access publication 2024 May 17
GJI General Geophysical Methods

https://doi.org/10.1093/gji/ggae175

A generalized curvilinear solver for spherical shell Rayleigh–Bénard convection

Souvik Naskar [1,2] Karu Chongsiripinyo,[3] Siddhant Mishra,[1] Anikesh Pal [1] and Akshay Jananan [1]

[1] *Department of Mechanical Engineering, Indian Institute of Technology, Kanpur* 208016*, India. E-mail: pala@iitk.ac.in*
[2] *School of Earth and Environment, University of Leeds, Leeds, LS29JT, United Kingdom*
[3] *Department of Mechanical Engineering, Chulalongkorn University, Bangkok* 10330*, Thailand*

Accepted 2024 May 13. Received 2024 May 4; in original form 2023 June 19

SUMMARY

A 3-D finite-difference solver has been developed and implemented for Boussinesq convection in a spherical shell. The solver transforms any complex curvilinear domain into an equivalent Cartesian domain using Jacobi transformation and solves the governing equations in the latter. This feature enables the solver to account for the effects of the non-spherical shape of the convective regions of planets and stars. Apart from parallelization using MPI, implicit treatment of the viscous terms using a pipeline alternating direction implicit scheme and HYPRE multigrid accelerator for pressure correction makes the solver efficient for high-fidelity direct numerical simulations. We have performed simulations of Rayleigh–Bénard convection at two Rayleigh numbers $Ra = 10^5$ and 10^7 while keeping the Prandtl number fixed at unity ($Pr = 1$). The average radial temperature profile and the Nusselt number match very well, both qualitatively and quantitatively, with the existing literature. Closure of the turbulent kinetic energy budget, apart from the relative magnitude of the grid spacing compared to the local Kolmogorov scales, ensures sufficient spatial resolution.

Key words: Core; Numerical modelling; Planetary interiors.

1 INTRODUCTION

Turbulent thermal convection is ubiquitous in nature as a primary driving mechanism for atmospheric and oceanic circulations (Hartmann *et al.* 2001). Such convective motions in Earth's outer core or in the solar convective zone, for example provide energy to sustain global-scale magnetic fields in planets and stars (Rüdiger & Hollerbach 2006; Roberts & King 2013). Such flow phenomena are further enriched due to the presence of global rotation, external or self-generated magnetic fields, chemical reactions, phase change, the porosity of the medium and particle suspension (Chillà & Schumacher 2012). Furthermore, the design of heat exchangers, cooling systems for electronics and indoor air circulation systems requires a fundamental understanding of thermal convection (Incropera 1988; Incropera *et al.* 1996). Rayleigh–Bénard convection (RBC) is a simple model of thermal convection, where a fluid layer between two parallel plates is heated from below and cooled from above. Such a plane layer geometry can be considered, for example, as a local approximation of the tangent cylinder region of Earth's outer core, which is situated between the top and bottom surfaces of the solid inner core and extending towards the north and south poles, respectively, up to the core-mantle boundary. Simulations in the plane layer geometry can reproduce the basic force balance and heat transfer behaviour that can be validated from well-designed laboratory experiments. Therefore, this flow configuration has been extensively studied, with the individual or combined effect of global rotation and magnetic fields (Naskar & Pal 2022a, b) to model various geophysical and astrophysical turbulent flows (Ahlers *et al.* 2009).

In the geophysical and astrophysical context, however, a spherical shell geometry is more pertinent to modelling planetary cores or stellar convective zones. The most extensive body of literature in this geometry focuses on 'geodynamo' simulations that attempt to model Earth's outer core convection and the associated geomagnetic field originating from it (Jones 2011). Mantle convection (Wolstencroft *et al.* 2009), rapidly rotating convection (Aurnou *et al.* 2015; Gastine *et al.* 2016; Mound & Davies 2017; Long *et al.* 2020), RBC without rotation and magnetic field (Gastine *et al.* 2015), deep convection in gas giants (Yadav & Bloxham 2020; Yadav *et al.* 2020) and solar convection (Korre & Featherstone 2021) are among the other prolific areas of research where spherical shell models are implemented. The superiority of these models lies in their capability to model many essential dynamic features of planetary atmospheres, such as thermal winds, strong shear layers, magnetic buoyancy,

meridional circulations and large-scale flows. They can also incorporate important geometric constraints, such as tangent cylinders and curvature effects near the boundaries, whose combined or individual influence can not be accounted for in a local Cartesian plane layer configuration (Rincon 2019).

The local plane layer and the global spherical shell simulations differ primarily in the direction of gravity, which is generally kept vertically downwards in the local Cartesian models. In contrast, the direction is radially inwards in global spherical shell models. Additionally, rotating convection in spherical shells exhibits distinct scales in the radial, axial and azimuthal directions (Dormy *et al.* 2004), whereas, for the local Cartesian model, we need to consider only two spatial scales: the horizontal scale of convection and the vertical scale over which convection occurs. For both geometries, the governing non-dimensional parameters are the Rayleigh numbers (Ra), which is a non-dimensional measure of the thermal forcing and the Prandtl number (Pr), representing the viscous to thermal diffusivity ratio. Apart from this, the flow properties may also depend on the aspect ratio $\Gamma = W/H$ (where W and H are the horizontal and vertical extents of the domain) and the radius ratio $\Gamma = r_i/r_o$ in-plane layer and spherical shell geometries, respectively. The important global diagnostic quantities are the Nusselt number Nu and the Reynolds number Re, representing the non-dimensional heat transfer and flow speed.

An intriguing question in this research direction is the scaling relation between such a diagnostic quantity with a governing input parameter, such as Nu, as a function of Ra. The thermal convection in planets and stars occurs at parameter values that are several orders of magnitude away from the reach of state-of-the-art numerical simulations and experiments. Therefore, these scaling relations are valuable tools to extrapolate the results of these experiments and simulations to planetary and stellar convective regimes. For plane layer geometry, a Nusselt number scaling of $Nu \sim Ra^{2/7}$ is found for moderate thermal forcing ($Ra \leq 10^{10}$), whereas, for higher thermal forcing, a scaling relation of $Nu \sim Ra^{1/3}$ has been widely reported (Iyer *et al.* 2020). A systematic investigation has been reported by (Gastine *et al.* 2015), who found the same scaling laws for the Nusselt number in the spherical geometry. It should be noted here that though the global diagnostic quantities exhibit similar behaviour, the local properties, such as the thickness of the viscous and thermal boundary layers, are markedly different in the two geometries. For example, the effect of curvature and a radially varying gravitational acceleration (as appropriate in Earth's core) results in asymmetric boundary layers in the spherical geometry, in contrast to the symmetric boundary layers in a plane layer geometry.

Experimental difficulties related to the radial direction of gravity make the advances in spherical shell convection almost entirely dependent on massively parallel numerical simulations. A comparative study among the existing solvers to address these issues has been reported by Matsui *et al.* (2016). Existing solvers primarily use spherical harmonic decomposition of the flow variables in the azimuthal and latitudinal directions, while the Chebyshev collocation method (Glatzmaier 1984; Wicht 2002; Simitev & Busse 2005; Sasaki *et al.* 2011; Featherstone & Hindman 2016) or finite-difference schemes(Dormy *et al.* 1998; Hollerbach 2000; Willis *et al.* 2007; Jiang & Kuang 2008; Takahashi 2012; Marti 2012; Matsui *et al.* 2014; Schaeffer *et al.* 2017), are used in the radial direction. Compared to these pseudo-spectral methods, locally discretized methods that use finite element (Matsui & Okuda 2004; Ribeiro *et al.* 2015), finite volume (Vantieghem *et al.* 2016), or finite difference (Santelli *et al.* 2021) are less popular as they require more resolution to achieve similar accuracy (Matsui *et al.* 2016). However, these local methods are much more suitable for massively parallel computations. Also, some of these local methods can handle non-spherical boundary topography (Fournier *et al.* 2004; Vantieghem *et al.* 2016), unlike pseudo-spectral methods that use spherical harmonic decomposition. Therefore, the effect of the ellipticity of the core–mantle boundary (Forte *et al.* 1995) on the azimuthal and latitudinal variation of radial heat flux can be accounted for by the local methods. Furthermore, magnetic field generation in some exoplanets and the moon has been found to be significantly dependent on the boundary topology (Dwyer *et al.* 2011; Le Bars *et al.* 2011; Cébron *et al.* 2012), making the use of such local methods indispensable. Among the local methods for modelling planetary convection, the spectral element model of Fournier *et al.* (2004) can model non-spherical shapes, which are symmetric about the axis of rotation. This flexibility allows it to model axisymmetric containers of any shape that can be used in laboratory experiments. Recently, a finite volume formulation with unstructured grids has been used by Vantieghem *et al.* (2016) that can be used to model convection in any complicated topology.

In this paper, we report on the development, implementation and validation of a new finite-difference solver for studying spherical shell convection. The capability to account for any effect of the non-spherical boundaries is the primary motivation for developing the present code. The solver can map any 3-D curvilinear geometry to a computational Cartesian domain using the Jacobi transformation. This enables us to solve the conservation equations in Cartesian coordinates, which are much simpler than their spherical coordinate counterpart, even after their modification by the Jacobi, elongation and stiffness matrix coefficients. The solver uses second-order central spatial discretization, while temporal discretization is achieved with the fractional step method (Chongsiripinyo 2019). In order to avoid the stiffness induced by the fine resolution near the boundary layers, the viscous terms have been treated implicitly, while the other terms are marched explicitly. The fractional step marches the velocity field into an intermediate field by a combination of the alternating direction implicit (ADI) method, the Crank–Nicolson (CN) method and the third-order low-storage Runge–Kutta (RKW3) method (Chongsiripinyo 2019). The remaining procedure in the fractional step method is to remove the divergence residual from the velocity field after the end of each RKW3 step, which in turn is achieved by pressure correction. We use the multigrid HYPRE module to accelerate the pressure correction. The rest of the article is structured as follows. Section 2 discusses the governing equation used. The numerical scheme is described in Section 3. Results are presented in Section 5 and summarized in Section 6.

2 GOVERNING EQUATIONS

We aim to investigate Rayleigh–Bénard convection of an incompressible, Newtonian, Boussinesq fluid in a spherical shell geometry as illustrated in Fig. 1. The spherical shell has an inner radius r_i and an outer radius r_o kept at constant temperatures T_i and T_o, respectively. The shell gap $d = r_o - r_i$, the temperature difference $\Delta T = T_i - T_o$ and the free-fall velocity $u_f = \{g_0\alpha(T_i - T_o)d\}^{1/2}$ have been used as the characteristics scale for length, temperature and velocity, respectively, to non-dimensionalize the governing equations. Here, g_0 is the gravitational acceleration at the outer radius. The relevant fluid properties are the kinematic viscosity (ν), thermal diffusivity (κ) and thermal expansion coefficient (α). The non-dimensional governing equations are expressed below using a Cartesian coordinate system.

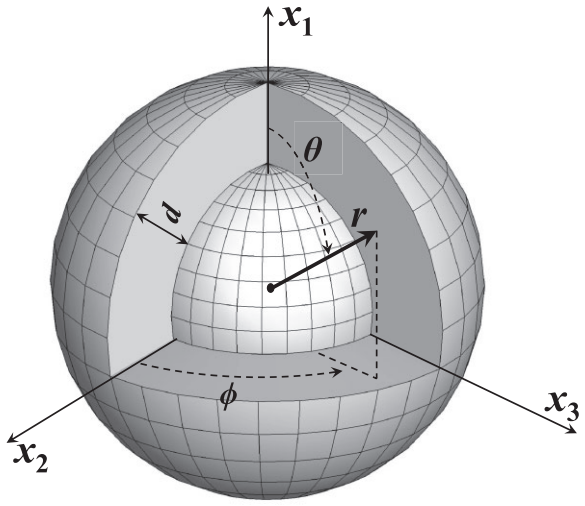


Figure 1. Spherical shell geometry.

$$\frac{\partial u_j}{\partial x_j} = 0, \tag{1}$$

$$\frac{\partial u_i}{\partial t} + \frac{\partial u_i u_j}{\partial x_j} = -\frac{\partial p}{\partial x_i} + gT\delta_{in} + \sqrt{\frac{Pr}{Ra}}\frac{\partial^2 u_i}{\partial x_j \partial x_j}, \tag{2}$$

$$\frac{\partial T}{\partial t} + u_j\frac{\partial T}{\partial x_j} = \frac{1}{\sqrt{Ra\,Pr}}\frac{\partial^2 T}{\partial x_j \partial x_j}, \tag{3}$$

where $g = (r_o/r)^2$ is the radial variation of gravitational acceleration and $\delta_{\text{in}} = \cos\theta\,\delta_{i1} + \sin\theta\,\cos\phi\,\delta_{i2} + \sin\theta\,\sin\phi\,\delta_{i3}$. Here θ and ϕ are the colatitude and longitude as shown in Fig. 1. The non-dimensional temperature difference is defined as $T = (T_f - T_o)/(T_i - T_o)$, where T_f is the temperature of the fluid. The non-dimensional parameters in these equations are the Rayleigh number and the Prandtl number defined below.

$$Ra = \frac{g_0\alpha\Delta T d^3}{\kappa\nu},\ Pr = \frac{\nu}{\kappa}. \tag{4}$$

In the subsequent section, we will use a coordinate transformation to convert the spherical domain to a Cartesian domain.

3 NUMERICAL ALGORITHMS

3.1 Coordinate transformation

To solve the governing eqs (1)–(3) in a generalized curvilinear coordinate system we perform coordinate transformation. The basic idea behind a coordinate transformation is to transform a set of physical laws written in Cartesian coordinates x_1, x_2, x_3 into an alternative form based on generalized curvilinear coordinates ζ, η, ξ (Chongsiripinyo 2019).

Physical law written in the Cartesian system
Physical curvilinear grid
↓ Grid transformation
Physical law written in the generalized system
Computational Cartesian grid, Jacobi terms

Such a transformation (as depicted in Fig. 2) will result in the inclusion of additional coefficients in the space derivatives in the governing equations, and the relation of this transformation between the Cartesian and the generalized curvilinear coordinate system is stored in a Jacobi matrix (J). The continuity, momentum and energy equations after the transformation are expressed below.

$$\frac{\partial\left[C_{nj}u_j\right]}{\partial\zeta_n} = 0 \tag{5}$$

$$\frac{\partial\left|J^{-1}\right|u_i}{\partial t} + \frac{\partial\left[C_{nj}u_j\right]u_i}{\partial\zeta_n} = -\frac{\partial C_{ni}P}{\partial\zeta_n} + \left|J^{-1}\right|gT\delta_{in} + \sqrt{\frac{Pr}{Ra}}\frac{\partial}{\partial\zeta_n}\left(G_{nj}\frac{\partial u_i}{\partial\zeta_j}\right) \tag{6}$$

$$\frac{\partial T}{\partial t} + \frac{\partial\left[C_{nj}u_j\right]T}{\partial\zeta_n} = \frac{1}{\sqrt{Ra\,Pr}}\frac{\partial}{\partial\zeta_n}\left(G_{nj}\frac{\partial T}{\partial\zeta_j}\right). \tag{7}$$

Here, x_i denotes the coordinate i of the Cartesian system and ζ_i denotes the coordinate i of the generalized system. The notations $x_i = (x_1, x_2, x_3) = (x, y, z)$ and $\zeta_i = (\zeta, \eta, \xi) = (\zeta_1, \zeta_2, \zeta_3)$ have been used interchangeably. After the transformation, the transformed governing equations are solved as if in a Cartesian system. In this context, grid transformation is often synonymously used with coordinate transformation as the curvilinear domain (i.e. a spherical shell domain in our case) is transformed into a new computational Cartesian domain. Here J^{-1}, C_{ij} and G_{ij} are

$$J^{-1} = \begin{bmatrix} \partial x_1/\partial\zeta & \partial x_2/\partial\zeta & \partial x_3/\partial\zeta \\ \partial x_1/\partial\eta & \partial x_2/\partial\eta & \partial x_3/\partial\eta \\ \partial x_1/\partial\xi & \partial x_2/\partial\xi & \partial x_3/\partial\xi \end{bmatrix} =: \left[\partial x_i/\partial\zeta_j\right] \tag{8}$$

$$C_{ij} = \left|J^{-1}\right|\frac{\partial\zeta_i}{\partial x_j} \quad G_{ij} = \left|J^{-1}\right|\frac{\partial\zeta_i}{\partial x_k}\frac{\partial\zeta_j}{\partial x_k} \tag{9}$$

The determinant $|J^{-1}|$ is the volume ratio of the original cell to the transformed cell, whereas C_{ij} and G_{ij} are grid elongation and skewness coefficients, respectively. The side length, and consequently the side area and the total volume, of a transformed cell, is chosen to be unity (Rosenfeld *et al.* 1991). Note that a recent and similar finite difference implementation by Santelli *et al.* (2021) uses a special transformation of variables that is equivalent to the contravariant velocity components (multiplied by cell volume) as done in eq. (6) (Rosenfeld *et al.* 1991). Such special treatments are required for a local discretization scheme near the pole to avoid singularities (see Fournier *et al.* (2004) for an example). In the present solver, the singularities near the poles can be easily circumnavigated by setting the transformation matrix coefficients to zero at the pole (x_1-axis in Fig. 1).

3.2 Jacobi terms

Fig. 3 demonstrates a cell (i, j, k) in a transformed computational domain. The Jacobi terms, J^{-1}, C_{pq} and G_{pq}, as expressed in eqs (8) and (9) are stored at the cell's faces. The calculation of J^{-1}, C_{pq} and G_{pq} is given below.

(i) J^{-1} is computed at every cell face, denoted by $J^{-1,fc}$; where fc indicates cell face (1–3), by calculating all the nine components in J^{-1}. For instance, we can compute the components of $J^{-1,2}$ of a cell (i,j,k) as follows:

$$(\partial\vec{x}/\partial\zeta)|_{i,j,k} = 0.125 * (+\vec{x}|_{i+1,j+1,k} + \vec{x}|_{i+1,j,k} + \vec{x}|_{i+1,j+1,k+1} + \vec{x}|_{i+1,j,k+1} - \vec{x}|_{i-1,j+1,k} - \vec{x}|_{i-1,j,k} - \vec{x}|_{i-1,j+1,k+1} - \vec{x}|_{i-1,j,k+1})$$

$$(\partial\vec{x}/\partial\eta)|_{i,j,k} = 0.5 * (+\vec{x}|_{i,j+1,k} + \vec{x}|_{i,j+1,k+1} - \vec{x}|_{i,j,k} - \vec{x}|_{i,j,k+1})$$

$$(\partial\vec{x}/\partial\xi)|_{i,j,k} = 0.5 * (+\vec{x}|_{i,j+1,k+1} + \vec{x}|_{i,j,k+1} - \vec{x}|_{i,j,k} - \vec{x}|_{i,j+1,k}).$$

(ii) Calculate $det(J^{-1,fc})$, denoted by $|J^{-1,fc}|$.

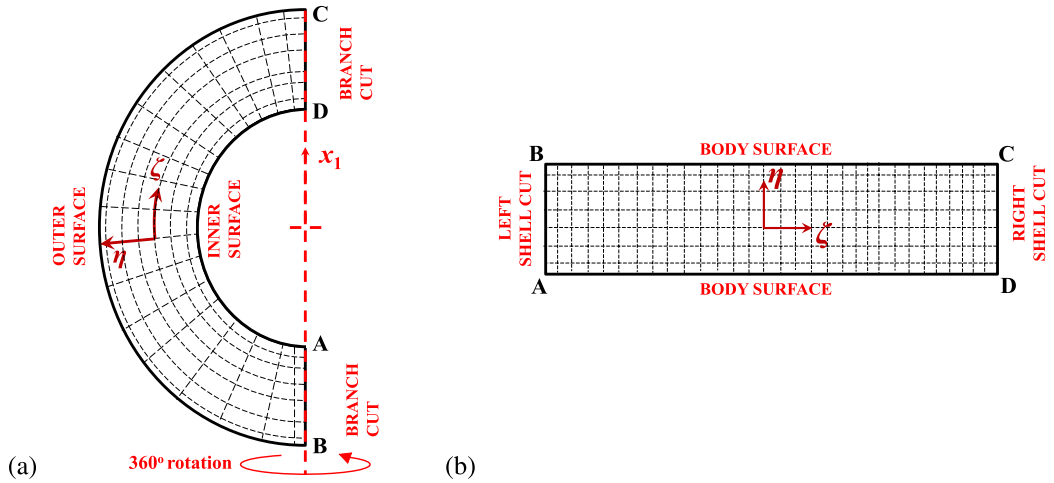


Figure 2. (a) Physical curvilinear domain depicting the left half of the $x_3 = 0$ plane in the spherical geometry with the corresponding (b) transformed Cartesian computational domain.

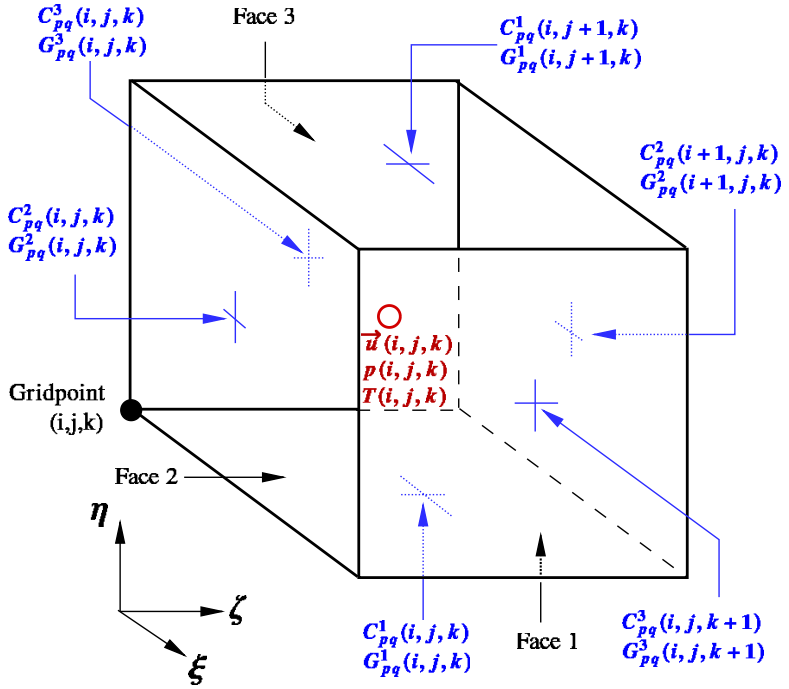


Figure 3. A transformed computational cell associated with the gridpoint (i, j, k); where i, j, k are the integer indices used to identify discrete space in the ζ, η and ξ directions, respectively.

(iii) The variable $|J^{-1}|$ in eq. (6) is an averaged value at the cell centre calculated from the six surrounding faces:

$$|J^{-1}|_{i,j,k} = \frac{1}{6}\left(\sum_{fc=1}^{3} |J^{-1,fc}|_{i,j,k} + |J^{-1,1}|_{i,j+1,k} + |J^{-1,2}|_{i+1,j,k} + |J^{-1,3}|_{i,j,k+1}\right).$$

(iv) Compute $J^{fc} = [\partial\zeta_i/\partial x_j]$ simply by the straight-forward inversion, $J^{-1,fc}$:
$\{J^{-1,fc}\}^{-1} = det(J^{-1,fc})^{-1}\{cof(J^{-1})\}^T$

(v) Calculate C_{pq} and G_{pq} at face fc, denoted by C^{fc}_{pq} and G^{fc}_{pq} from J^{fc} using eq. (9).

3.3 Spatial discretization

The spatial derivatives in eqs (6) and (7) are discretized using a second-order central finite difference scheme. Fig. 4 illustrates the stencils used to discretize the term eq. (10) using this scheme.

$$\frac{\partial}{\partial\zeta_p}\left[G_{pq}\frac{\partial\phi}{\partial\zeta_q}\right] \tag{10}$$

Eq. (10) consists of 9 terms. We present the discretization of term 1 (p=1 and q=1), term 2 (p=1 and q=2) and term 3 (p=1 and q = 3) as examples.

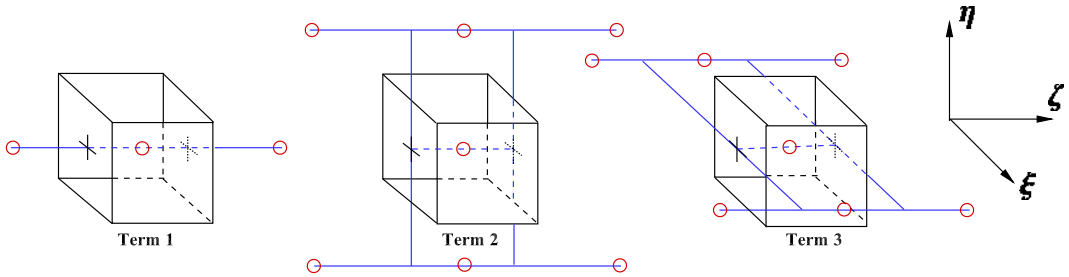


Figure 4. Stencils used for computing eqs (11), (12) and (13).

$$\left(\frac{\delta}{\delta\zeta_1}\left[G_{11}\frac{\delta\phi}{\delta\zeta_1}\right]\right)_{i,j,k} = \left[G_{11}\frac{\delta\phi}{\delta\zeta_1}\right]_{i+1/2,j,k} - \left[G_{11}\frac{\delta\phi}{\delta\zeta_1}\right]_{i-1/2,j,k}$$
$$= +G^2_{11}|_{i+1,j,k}[\phi|_{i+1,j,k} - \phi|_{i,j,k}] - G^2_{11}|_{i,j,k}\ [\phi|_{i,j,k} - \phi|_{i-1,j,k}] \quad (11)$$

$$\left(\frac{\delta}{\delta\zeta_1}\left[G_{12}\frac{\delta\phi}{\delta\zeta_2}\right]\right)_{i,j,k} = \left[G_{12}\frac{\delta\phi}{\delta\zeta_2}\right]_{i+1/2,j,k} - \left[G_{12}\frac{\delta\phi}{\delta\zeta_2}\right]_{i-1/2,j,k}$$
$$= +\frac{G^2_{12}|_{i+1,j,k}}{2}\left[+\frac{\phi|_{i,j+1,k}+\phi|_{i+1,j+1,k}}{2} - \frac{\phi|_{i,j-1,k}+\phi|_{i+1,j-1,k}}{2}\right] - \frac{G^2_{12}|_{i,j,k}}{2}\left[+\frac{\phi|_{i,j+1,k}+\phi|_{i-1,j+1,k}}{2} - \frac{\phi|_{i,j-1,k}+\phi|_{i-1,j-1,k}}{2}\right] \quad (12)$$

$$\left(\frac{\delta}{\delta\zeta_1}\left[G_{13}\frac{\delta\phi}{\delta\zeta_3}\right]\right)_{i,j,k} = \left[G_{13}\frac{\delta\phi}{\delta\zeta_3}\right]_{i+1/2,j,k} - \left[G_{13}\frac{\delta\phi}{\delta\zeta_3}\right]_{i-1/2,j,k}$$
$$= +\frac{G^2_{13}|_{i+1,j,k}}{2}\left[+\frac{\phi|_{i,j,k+1}+\phi|_{i+1,j,k+1}}{2} - \frac{\phi|_{i,j,k-1}+\phi|_{i+1,j,k-1}}{2}\right] - \frac{G^2_{13}|_{i,j,k}}{2}\left[+\frac{\phi|_{i,j,k+1}+\phi|_{i-1,j,k+1}}{2} - \frac{\phi|_{i,j,k-1}+\phi|_{i-1,j,k-1}}{2}\right] \quad (13)$$

3.4 Temporal discretization

For temporal discretization, a fractional step method is used where a velocity field is sequentially advanced in multiple substeps. We use a combination of the Alternating Direction Implicit method (ADI), the Crank-Nicolson method (CN) and the third-order low-storage Runge-Kutta method (RKW3) to march to an intermediate field as described below (Chongsiripinyo 2019).

3.4.1 ADI method

The ADI method has been used to treat the viscous term implicitly while marching in one direction at a time. We demonstrate the method with a 2-D diffusion equation as shown in eq. (14). To solve eq. (14) using the Euler method, the procedure is to perform implicit Euler in the x-direction with explicit Euler in the y-direction for the first half ($\Delta t/2$) and vice versa for the second half ($\Delta t/2$) as shown in eqs (15) and (16).

$$\frac{\partial\phi}{\partial t} = \alpha\left[\frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2}\right] \quad (14)$$

Table 1. RKW3 parameters.

Substep	h	β	Π
1	$8\Delta t/15$	1	0
2	$2\Delta t/15$	25/8	$-17/8$
3	$1\Delta t/3$	9/4	$-5/4$

$$\frac{\phi^{n+\frac{1}{2}} - \phi^n}{\Delta t/2} = \alpha\left[\frac{\partial^2\phi^{n+\frac{1}{2}}}{\partial x^2} + \frac{\partial^2\phi^n}{\partial y^2}\right] \quad (15)$$

$$\frac{\phi^{n+1} - \phi^{n+\frac{1}{2}}}{\Delta t/2} = \alpha\left[\frac{\partial^2\phi^{n+\frac{1}{2}}}{\partial x^2} + \frac{\partial^2\phi^{n+1}}{\partial y^2}\right]. \quad (16)$$

3.4.2 CN method

The CN method splits the right-hand side into two equal parts, the implicit and the explicit, as demonstrated in eqs (17) and (18).

$$\frac{\partial\phi}{\partial t} = \alpha\frac{\partial^2\phi}{\partial x^2} \quad (17)$$

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} = \frac{\alpha}{2}\left[\frac{\partial^2\phi^{n+1}}{\partial x^2} + \frac{\partial^2\phi^n}{\partial x^2}\right] \quad (18)$$

3.4.3 RKW3 method

The RKW3 method uses only two storage variables. Marching is accomplished in three substeps, briefly summarized here. Given an equation for ϕ,

$$\frac{\partial\phi}{\partial t} = \mathrm{R}(\phi). \quad (19)$$

RKW3 is implemented in the following manner,

$$\frac{\phi^{rk} - \phi^{rk-1}}{h^{rk}} = \beta^{rk}\mathrm{R}(\phi^{rk-1}) + \Pi^{rk}\mathrm{R}(\phi^{rk-2}). \quad (20)$$

Here, rk goes from substep 1 to substep 3, and the values of h, β and Π are given in Table 1.

3.4.4 The ADI-CN-RKW3 combined marching scheme

The above-mentioned algorithms (ADI, CN and RKW3) are combined to march the governing equations to an intermediate state temporally (Fig. 5). The right-hand side of eq. (6) is split into explicit and implicit terms as indicated by the subscripts *ex* and *im* in eq. (21). Depending on the grid skewness G_{ij}, the diagonal components of the viscous terms are susceptible to the stiffness of the

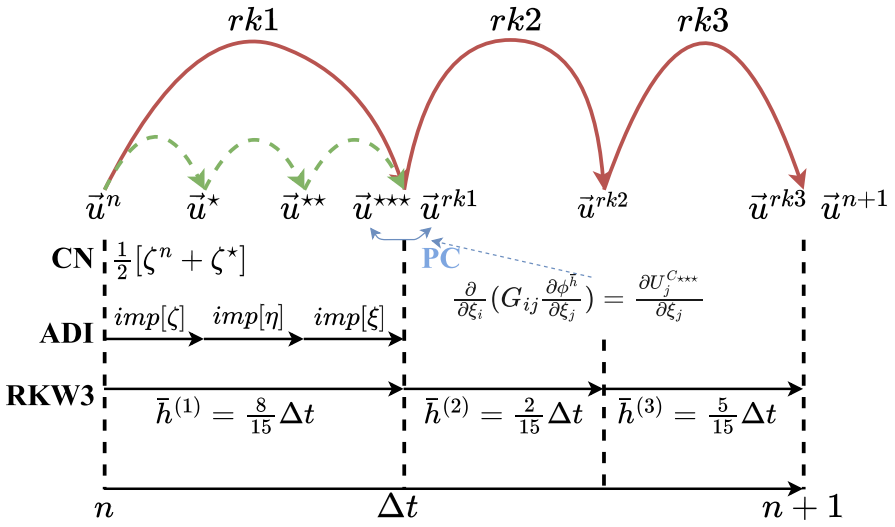


Figure 5. ADI-CN-RKW3 combined marching scheme. PC denotes the pressure correction.

discretized systems and are, therefore, marched implicitly. The ADI scheme is used since there are three viscous terms containing G_{11}, G_{22} and G_{33}. At a given time, they are split into two parts using CN. These steps are shown in eqs (22), (23) and (24) as an example for substep 1 of the RKW3 marching scheme.

$$\frac{\partial J^{-1}u_i}{\partial t} = \left[-\frac{\partial C_{ni}P}{\partial\zeta_n} - \frac{\partial\left[C_{nj}u_j\right]u_i}{\partial\zeta_n} + \sqrt{\frac{Pr}{Ra}}\left(\frac{\partial}{\partial\zeta_n}G_{nj}\frac{\partial u_i}{\partial\zeta_j}\right)_{n\neq j} + gT\delta_{in}\right]_{ex} + \left[\sqrt{\frac{Pr}{Ra}}\left(\frac{\partial}{\partial\zeta_n}G_{nj}\frac{\partial u_i}{\partial\zeta_j}\right)_{n=j}\right]_{im} \tag{21}$$

$$J^{-1}u_i^\star = J^{-1}u_i^n + \beta^{(1)}h^{(1)}\Box + \sqrt{\frac{Pr}{Ra}}\frac{h^{(1)}}{2}\left(\frac{\partial}{\partial\zeta}\left[G_{11}\frac{\partial u_i^n}{\partial\zeta}\right] + \frac{\partial}{\partial\zeta}\left[G_{11}\frac{\partial u_i^\star}{\partial\zeta}\right]\right), + \sqrt{\frac{Pr}{Ra}}h^{(1)}\frac{\partial}{\partial\eta}\left[G_{22}\frac{\partial u_i^n}{\partial\eta}\right] + \sqrt{\frac{Pr}{Ra}}h^{(1)}\frac{\partial}{\partial\xi}\left[G_{33}\frac{\partial u_i^n}{\partial\xi}\right], \tag{22}$$

Here, $\Box$ represents all the terms to be marched explicitly.

$$J^{-1}u_i^{\star\star} = J^{-1}u_i^\star - \sqrt{\frac{Pr}{Ra}}\frac{h^{(1)}}{2}\frac{\partial}{\partial\eta}\left[G_{22}\frac{\partial u_i^n}{\partial\eta}\right] + \sqrt{\frac{Pr}{Ra}}\frac{h^{(1)}}{2}\frac{\partial}{\partial\eta}\left[G_{22}\frac{\partial u_i^{\star\star}}{\partial\eta}\right] \tag{23}$$

$$J^{-1}u_i^{\star\star\star} = J^{-1}u_i^{\star\star} - \sqrt{\frac{Pr}{Ra}}\frac{h^{(1)}}{2}\frac{\partial}{\partial\xi}\left[G_{33}\frac{\partial u_i^n}{\partial\xi}\right] + \sqrt{\frac{Pr}{Ra}}\frac{h^{(1)}}{2}\frac{\partial}{\partial\xi}\left[G_{33}\frac{\partial u_i^{\star\star\star}}{\partial\xi}\right] \tag{24}$$

The intermediate velocity fields $u_i^\star$, $u_i^{\star\star}$, $u_i^{\star\star\star}$ are obtained by solving a set of tridiagonal matrices that result from the spatial discretization of the eqs (22), (23) and (24) in the ζ, η and ξ directions respectively. The intermediate velocity $u_i^{\star\star\star}$ is the first step in the fractional-step scheme. We use the Thomas algorithm with pipelining, as described in the next section, to solve the tridiagonal system eqs (22)–(24) to obtain $u^\star$, $u^{\star\star}$ and $u^{\star\star\star}$.

3.5 Thomas algorithm

Let us consider solving $A\psi = g$ for ψ, which is the outcome of the spatial discretization of, for instance, eq. (22). Here A is a tridiagonal matrix given as

$$\begin{bmatrix} b_0 & c_0 & & & & & \\ a_1 & b_1 & c_1 & & & & \\ & a_2 & b_2 & c_2 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & a_{n-1} & b_{n-1} & c_{n-1} \\ & & & & & a_n & b_n \end{bmatrix}\begin{bmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \psi_{n-1} \\ \psi_n \end{bmatrix} = \begin{bmatrix} g_0 \\ g_1 \\ g_2 \\ \cdot \\ \cdot \\ g_{n-1} \\ g_n \end{bmatrix}. \tag{25}$$

The first two relations in eq. (25) are,

$$b_0\psi_0 + c_0\psi_1 = g_0 \tag{26}$$

$$a_1\psi_0 + b_1\psi_1 + c_1\psi_2 = g_1. \tag{27}$$

Substituting ψ_0 from eq. (26) into ψ_0 in eq. (27) gives

$$b_1'\psi_1 + c_1\psi_2 = g_1' \tag{28}$$

where $b_1' = \left[b_1 - a_1b_0^{-1}c_0\right]$ and $g_1' = \left[g_1 - a_1b_0^{-1}g_0\right]$. The algorithm involves two stages, forward sweeping and backward substitution. The subdiagonal elements $a_1 - a_n$ are removed using Gaussian elimination during the forward sweeping step. Therefore, eq. (25) takes the form:

$$\begin{bmatrix} b_0 & c_0 & & & & & \\ 0 & b_1' & c_1 & & & & \\ & 0 & b_2' & c_2 & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & 0 & b_{n-1}' & c_{n-1} \\ & & & & & 0 & b_n' \end{bmatrix}\begin{bmatrix} \psi_0 \\ \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \psi_{n-1} \\ \psi_n \end{bmatrix} = \begin{bmatrix} g_0 \\ g_1' \\ g_2 \\ \cdot \\ \cdot \\ g_{n-1}' \\ g_n' \end{bmatrix}. \tag{29}$$

At the end of the forward sweep, we can solve for $\psi_n = g_n'/b_n'$ in eq. (29). Subsequently, we solve for ψ_{n-1} ψ_0 (equations $n-1$ until 0) as in $\psi_i = (g_i' - c_i\psi_{i+1})/b_i'$. For the grid in Section 4, periodic boundary conditions are enforced in the ξ direction. Therefore, the above-mentioned Thomas algorithm is modified as follows. Consider the discretized system $A\psi = g$ with periodicity as in eq. (30)

where $\psi_1 = \psi_{n-1}$ and $\psi_2 = \psi_n$.

$$\begin{bmatrix} b_1 & c_1 & & & & & a_1 \\ a_2 & b_2 & c_2 & & & & \\ & \cdot & \cdot & \cdot & & & \\ & & \cdot & \cdot & \cdot & & \\ & & & \cdot & \cdot & \cdot & \\ & & & & a_{n-1} & b_{n-1} & c_{n-1} \\ c_n & & & & & a_n & b_n \end{bmatrix} \begin{bmatrix} \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \psi_{n-1} \\ \psi_n \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \cdot \\ \cdot \\ \cdot \\ g_{n-1} \\ g_n \end{bmatrix} \tag{30}$$

The first step includes separating eq. (30) into a tridiagonal system 31 with an additional eq. (32).

$$\begin{bmatrix} b_1 & c_1 & & & & \\ a_2 & b_2 & c_2 & & & \\ & \cdot & \cdot & \cdot & & \\ & & \cdot & \cdot & \cdot & \\ & & & \cdot & \cdot & \\ & & & & a_{n-1} & b_{n-1} \end{bmatrix} \begin{bmatrix} \psi_1 \\ \psi_2 \\ \cdot \\ \cdot \\ \cdot \\ \psi_{n-1} \end{bmatrix} = \begin{bmatrix} g_1 \\ g_2 \\ \cdot \\ \cdot \\ \cdot \\ g_{n-1} \end{bmatrix} + \begin{bmatrix} -a_1 \\ 0 \\ 0 \\ \cdot \\ \cdot \\ -c_{n-1} \end{bmatrix} \psi_n \tag{31}$$

$$c_n\psi_1 + a_n\psi_{n-1} + b_n\psi_n = g_n. \tag{32}$$

The tridiagonal matrix on the left-hand side is defined as $[A_c]$ and $[g]$ as the g-column matrix on the right-hand side. Let

$$[\psi] = [\psi 1] + [\psi 2]\psi_n \tag{33}$$

be the solution of the system eq. (31) where

$$[\psi 1] = [A_c]^{-1}[g] \tag{34}$$

$$[\psi 2] = [A_c]^{-1}[-a_1 0 \cdots -c_{n-1}]^T. \tag{35}$$

Substituting ψ in eq. (33) into ψ_1 and ψ_{n-1} in eq. (32) gives

$$c_n(\psi 1_1 + \psi 2_1\psi_n) + a_n(\psi 1_{n-1} + \psi 2_{n-1}\psi_n) + b_n\psi_n = g_n. \tag{36}$$

Rearrange eq. (36) for ψ_n

$$\psi_n = \frac{g_n - c_n\psi 1_1 - a_n\psi 1_{n-1}}{b_n + c_n\psi 2_1 + a_n\psi 2_{n-1}}. \tag{37}$$

In summary, to solve the system eq. (30), we use the following steps:

(i) Construct eqs (34) and (35).
(ii) Solve eqs (34) and (35) for $[\psi 1]$ and $[\psi 2]$ from index 1 to index $n-1$.
(iii) Substitute $[\psi 1]$ and $[\psi 2]$ into eq. (37) and solve for ψ_n.
(iv) Calculate $[\psi]$ from eq. (33) using $[\psi 1]$, $[\psi 2]$ and ψ_n.

3.5.1 *Parallel algorithm*

$$\begin{array}{l|cccccccc|c} \text{CPU}^i & b_s'^{i-1} & c_s^{i-1} & & & & & & & \cdot \\ & 0 & b_s'^i & c_s^i & & & & & & g_s'^i \\ & & 0 & b_{s+1}'^i & c_{s+1}^i & & & & & g_{s+1}'^i \\ & & & \cdot & \cdot & \cdot & & & & \cdot \\ & & & & 0 & b_e'^i & c_e^i & & & g_e'^i \\ \hline \text{CPU}^{i+1} & & & & a_s^{i+1} & b_s^{i+1} & c_s^{i+1} & & & g_s^{i+1} \\ & & & & & a_{s+1}^{i+1} & b_{s+1}^{i+1} & c_{s+1}^{i+1} & & g_{s+1}^{i+1} \\ & & & & & & \cdot & \cdot & \cdot & \cdot \\ & & & & & & & a_e^{i+1} & b_e^{i+1} & c_e^{i+1} & g_e^{i+1} \\ \hline & & & & & & & & a_s^{i+1} & \cdot & \cdot & \cdot \end{array} \tag{38}$$

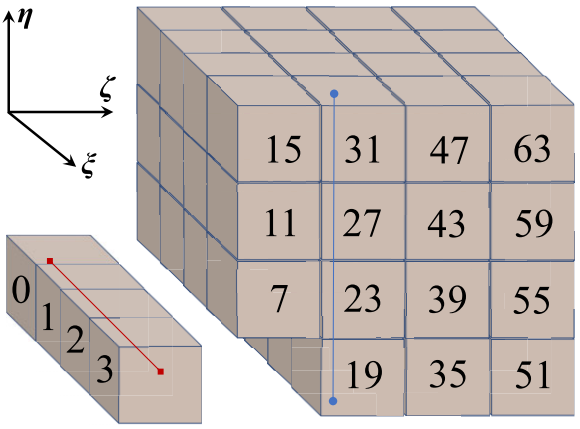


Figure 6. Example of a $4 \times 4 \times 4$ CPU topology in a single computational domain. Numbers in the figure represent an individual CPU's rank (id).

$$\begin{array}{l|cccccccc|c} \text{CPU}^i & b_s'^{i-1} & c_s^{i-1} & & & & & & & \cdot \\ & & b_s'^i & c_s^i & & & & & & g_s'^i \\ & & & b_{s+1}'^i & c_{s+1}^i & & & & & g_{s+1}'^i \\ & & & & \cdot & \cdot & & & & \cdot \\ & & & & & b_e'^i & c_e^i & & & g_e'^i \\ \hline \text{CPU}^{i+1} & & & & & b_s'^{i+1} & c_s^{i+1} & & & \psi_s^{i+1} \\ & & & & & & b_{s+1}'^{i+1} & c_{s+1}^{i+1} & & \psi_{s+1}^{i+1} \\ & & & & & & & \cdot & \cdot & \cdot \\ & & & & & & & & b_e'^{i+1} & c_e^{i+1} & \psi_e^{i+1} \\ \hline & & & & & & & & & \cdot & \cdot & \end{array} \tag{39}$$

Fig. 6 demonstrates an example $4 \times 4 \times 4$ central processing unit (CPU) topology in a single computational domain. The spatial discretization of eqs (22), (23) or (24) over several CPUs results in tridiagonal matrices and vectors. The tridiagonal matrices constructed from the discretization of eq. (24) span over the entire ξ space index, for example over CPU^{0-3} illustrated by the solid red line in Fig. 6. Similarly, the tridiagonal matrices constructed from the discretization of eq. (23) span over the entire η space index (e.g. over CPU^{19-31}), as indicated by a solid blue line. Considering eq. (38), forward sweeping starts at CPU^0. Once the sweeping reaches the interface between CPU^i and CPU^{i+1}, CPU^i sends $b_e'^i$, c_e^i and $g_e'^i$ to CPU^{i+1}. Then, CPU^{i+1} continues to carry out the sweeping by sending data $b_e'^{i+1}$, c_e^{i+1} and $g_e'^{i+1}$ to CPU^{i+2} and so on. After the forward sweeping is finalized, backward substitution starts and reverse sweeping is performed, as shown in eq. (39). However, the only information being sent from CPU^{i+1} to CPU^i is ψ_s^{i+1}.

For a periodic system, we use the following steps:

(i) Construct eqs (34) and (35).
(ii) Use the parallel Thomas subroutine to solve eqs (34) and (35) for $[\psi 1]$ and $[\psi 2]$.
(iii) CPU^0 owning the first block (contains node 1), sends $\psi 1_1$ and $\psi 2_1$ to CPU^N that owns the last block (contains node n).
(iv) CPU^N calculates ψ_n and broadcasts ψ_n to every CPU that owns a subsystem of eq. (30).
(v) Every CPU calculates $[\psi]$ from eq. (33).

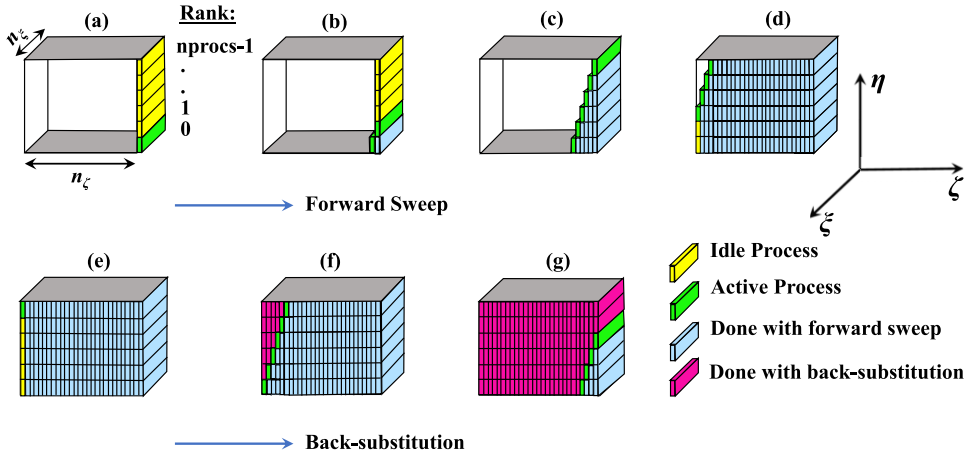


Figure 7. Illustration of pipelining with the parallel Thomas algorithm following Fig. VI.16 of Taylor (2008). Forward sweep is achieved in steps (a–d), while steps (e-g) depict back-substitution.

Note that by splitting eq. (30) into eqs (31) and (32), CPU^N solves the tridiagonal system eq. (33) which has size one element less than the others.

3.5.2 *Pipelining*

In the previous subsection, we summarize how to solve a tridiagonal system in parallel. It is done simply by completing the forward/backward sweep and sending data to the proper neighbour in order to continue marching. CPU^i that finishes the forward sweep sends data to CPU^{i+1} until the last block is reached. Generally, each CPU can be responsible for thousands of tridiagonal subsystems contained in a single subdomain (or a 'block'). This subsection summarizes how to solve such a big system efficiently.

Consider a computational domain containing (n_ζ, n_η, n_ξ) grid-points. For the sake of simplicity, the domain is equally decomposed only in the η direction into NJ blocks so that CPU^0 occupies block 0, CPU^1 occupies block 1 and so on. Thus, each CPU owns a block of size (n_ζ, n_η/N_η, n_ξ); given that n_η/N_η is, by design, an integer. Supposing that we choose to perform an implicit marching in the η direction, the resulting tridiagonal matrix is subdivided into N_η sections. The easiest, though the least efficient, way to solve these systems is to let CPU^0 solve all of its subsystems across the (n_ζ, n_η/N_η, n_ξ) grid before sending data to CPU^1. That is, CPU^0 performs a forward sweep at cell (1, 1 $\rightarrow$ n_η/N_η, 1), at cell (1, 1 $\rightarrow$ n_η/N_η, 2) and so on until cell (n_ζ, 1 $\rightarrow$ n_η/N_η, n_ξ). Next, CPU^0 packs the plane data with $n_\zeta * n_\xi * 3$ elements (recall $b_e^{'i}$, c_e^i and $g_e^{'i}$ in the previous subsection) at (1: n_ζ, n_η/N_η, 1: n_ξ) and sends it to CPU^1. Following the same process for the subsequent CPUs until CPU$^{N_\eta-1}$ is reached, the backward substitution is carried out in the same way from CPU$^{N_\eta-1}$ to CPU^0. The obvious drawback is that only one CPU operates at a given time, and the whole process will be even slower than the serial version since there is additional communication overhead.

Pipelining is used in an attempt to minimize the number of idle CPUs while optimizing communication overhead. In essence, rather than sweeping across the (n_ζ, n_η/N_η, n_ξ) grid all at once, each CPU performs the sweeps only for a portion of the grid and shares data with its neighbouring CPU in the sweep direction downstream in a forward sweep and upstream for a backward sweep. A portion of the grid can be chosen for the first CPU, with the others obeying the same portion. We give an example of pencil-type pipelining. Consider Fig. 7 and the following steps:

(i) CPU^0, process rank 0 in the figure, performs the forward sweep in the η direction at cell (i, k) = (1, 1) from (i, j, k) = (1, 1, 1) to (i, j, k) = (1, n_η/N_η, 1); here i and k are dummy indices pointing to a grid location in ζ and ξ directions, respectively. CPU^0 then repeats the forward sweep until (i, k) = (1, n_ξ). Notice that the forward sweep is in the η direction, but the 'pencil' aligns in the ξ direction. At this point, CPU^0 packs and passes data to CPU^1. The data is of size $n_\xi * 3$ elements containing $b'^0_{n_\eta/N_\eta}$, $c^0_{n_\eta/N_\eta}$ and $g'^0_{n_\eta/N_\eta}$ for each $k \in [1, n_\xi]$ (with 1-element width in the ζ direction, hence the word 'pencil').

(ii) CPU^1 continues the forward sweep while CPU^0 starts solving the new tridiagonal system by shifting 1 step from the first block in the ζ, which is the 'slide' direction. The 'slide' and 'pencil' directions can be swapped.

(iii) CPU^1 passes data to CPU^2 for the sliding index $i = 1$, receives data from CPU^0 at the sliding index $i = 2$ and continues the forward sweep.

(iv) The same process is carried out until CPU$^{N_\eta-1}$ reaches the slide index $i = n_\zeta$.

(v) CPU$^{N_\eta-1}$ starts the backward sweep at the sliding index $i = n_\zeta$, shares data of size $n_\xi * 1$-element containing $\zeta_1^{N_\eta-1}$ for each $k \in [1, n_\xi]$ with CPU$^{N_\eta-2}$, and starts the backward sweep at the sliding index $i = n_\zeta - 1$.

(vi) The backward sweeping process is carried out in the same way as the forward sweep.

(vii) Solving the system of tridiagonal matrices is finalized after CPU^0 finishes the backward sweep at the sliding index $i = 1$.

3.5.3 *Handling shell cut*

Using the parallel Thomas algorithm with pipelining, we have been able to solve eqs (22), (23) and (24) in parallel for $u^\star$, $u^{\star\star}$ and $u^{\star\star\star}$. The grid used for the solver before it is rotated about the x_1-axis is shown in Fig. 2(a). The directions parallel and perpendicular to the body surface are denoted by ζ and η, respectively. Fig. 2(b) represents the transformed coordinate that is obtained using Jacobi transformation. The top and bottom edges of the domain, parallel to x_1-axis in Fig. 2(a), are indicated by the phrase 'Branch cut' (AB and

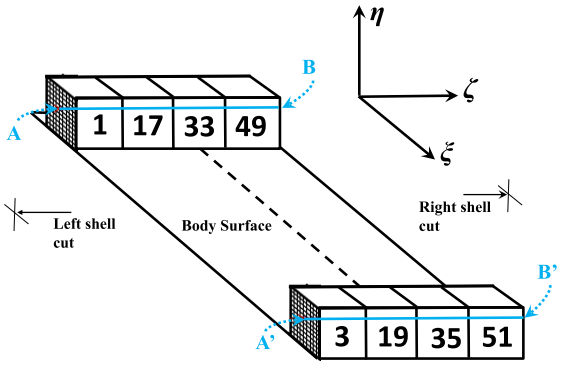


Figure 8. Solving a tridiagonal system across a shell cut. Here (A, A') and (B, B') are the two pairs of surfaces across the left' and right shell cuts, respectively, as per the arrangement of processors shown in Fig. 6.

CD). They correspond to the left and right sides of the transformed domain, which are parallel to the η direction, indicated by the word 'shell cut'. The body surface seen in the curvilinear domain in Fig. 2(a) is transformed to the top and bottom body surface of the transformed domain Fig. 2(b). The words 'shell cut' or 'Branch cut' represent a shared interface among CPUs that cuts through the centreline. A tridiagonal system in the ζ direction created by discretizing eq. (22) is interrupted by the shell cut on both the left- and right-hand sides. To handle the shell cut, the tridiagonal system from one side of the cut is merged with the system on the opposite side. Therefore, the resulting system is twice as large as the system without the cut.

For example, consider solving a system in the ζ direction in Fig. 8. In the forward sweeping, CPU^1 with point A starts solving from this point A and then passes the information to CPU^{17} until the forward sweeping reaches CPU^{49}. Then, CPU^{49}, with point B passes the information to CPU^{51} that has point B$'$ on the other side of the cut. CPU^{51} keeps performing the forward sweep by sending data to CPU^{35} and so on until the sweep reaches A' owned by CPU^3. The backward substitution follows the same procedure by starting from point A$'$ and marching until the substitution reaches back to point A. Forward and backward sweeping are done using the pencil-type pipeline Thomas algorithm explained previously.

3.6 Pressure correction

The generalized curvilinear solver uses a combination of the ADI-CN-RKW3 methods to obtain u^{***}, the intermediate velocity. The remaining procedure in the fractional step method is to remove the divergence residual from the projected velocity u^{***} at the end of each sub-RKW3 step (denoted as PC in Fig. 5). This step requires correcting the pressure to account for the divergenceless field. Rewriting eq. (6) as eq. (40); where $\Box$ represents the advection, the diffusion and the baroclinic terms. Eq. (40) is temporally discretized into eqs (41) and (42). Here, u_i^{***} denotes velocity at the third step of the ADI, and h is a subtime step of RKW3.

$$\frac{\partial J^{-1}u_i}{\partial t} = \Box - \frac{\partial C_{ji}P}{\partial \zeta_j} \tag{40}$$

$$\frac{J^{-1}u_i^{***} - J^{-1}u_i^{n}}{h} = \Box^n - \frac{\partial C_{ji}P^n}{\partial \zeta_j} \tag{41}$$

$$\frac{J^{-1}u_i^{n+h} - J^{-1}u_i^{n}}{h} = \Box^n - \frac{\partial C_{ji}P^{n+h}}{\partial \zeta_j} \tag{42}$$

eqs (42)–(41) gives:

$$\frac{J^{-1}u_i^{n+h} - J^{-1}u_i^{***}}{h} = -\frac{\partial C_{ji}\delta P^h}{\partial \zeta_j} \tag{43}$$

Taking divergence of eq. (43) gives eq. (44). Note that $\partial_i u_i^{n+h} = 0$.

$$\frac{1}{h}\frac{\partial}{\partial \zeta_j}\left[\frac{\partial \zeta_j}{\partial x_i}J^{-1}u_i^{***}\right] = \frac{\partial}{\partial \zeta_k}\frac{\partial \zeta_k}{\partial x_i}\left[\frac{\partial}{\partial \zeta_j}J^{-1}\frac{\partial \zeta_j}{\partial x_i}\delta P^h\right] \tag{44}$$

This yields the Poisson eq. (45) for pressure correction δP^h

$$\frac{\partial}{\partial \zeta_i}\frac{\partial}{\partial \zeta_j}\left[G_{ij}h\delta P^h\right] = \frac{\partial}{\partial \zeta_j}\left[C_{ji}u_i^{***}\right] \tag{45}$$

Removing divergence from the u^{***} field is done by solving eq. (45) for $\delta P^{\bar{h}}$ and computing $u_i^{n+\bar{h}}$ using eq. (46).

$$u_i^{n+\bar{h}} = u_i^{***} - \frac{1}{J^{-1}}\frac{\partial}{\partial \zeta_j}\left[C_{ji}\bar{h}P^{\bar{h}}\right] \tag{46}$$

The divergence-free field $u^{n+h^{(1)}}$ marks the end of RKW3 first sub step. We follow the same procedure until $u^{n+h^{(1)}+h^{(2)}+h^{(3)}} = u^{n+1}$ is obtained. Fig. 5 illustrates the entire process. HYPRE is a library of scalable linear solvers and multigrid methods as detailed in Falgout & Jones (2000) and Falgout *et al.* (2002). Generalized curvilinear solver utilizes two solvers provided by HYPRE: (1) SMG, a parallel semi-coarsening multigrid (SCM) solver for linear systems (Brown *et al.* 2000) and (2) BoomerAMG, a parallel implementation of the algebraic multigrid method (Ruge & Stüben 1987).

4 SIMULATION DETAILS

We have performed simulations for two Rayleigh numbers, $Ra = 10^5$ and 10^7, keeping the Prandtl number constant at $Pr = 1$. The radius ratio is also kept constant at $\Gamma = 0.6$, and an inverse square-law profile of gravity with the radius $g = (r_o/r)^2$ is assumed. These parameter choices aim to facilitate comparison with Gastine *et al.* (2015). The number of gridpoints used in each direction, along with the output diagnostic quantities for the two Ra is given in Table 2. The physical curvilinear grid is clustered in the radial direction near the boundaries to resolve the boundary layers near the solid surfaces before performing the Jacobi transformation. The clustering function is given below.

$$r(j) = \frac{\tanh\left[rx_2\left(\frac{j-1}{nx_2} - \frac{1}{2}\right)\right]}{2\tanh\left(\frac{rx_2}{2}\right)}, \tag{47}$$

here, rx_2 is the stretching factor, and nx_2 is the number of grid divisions in the radial direction.

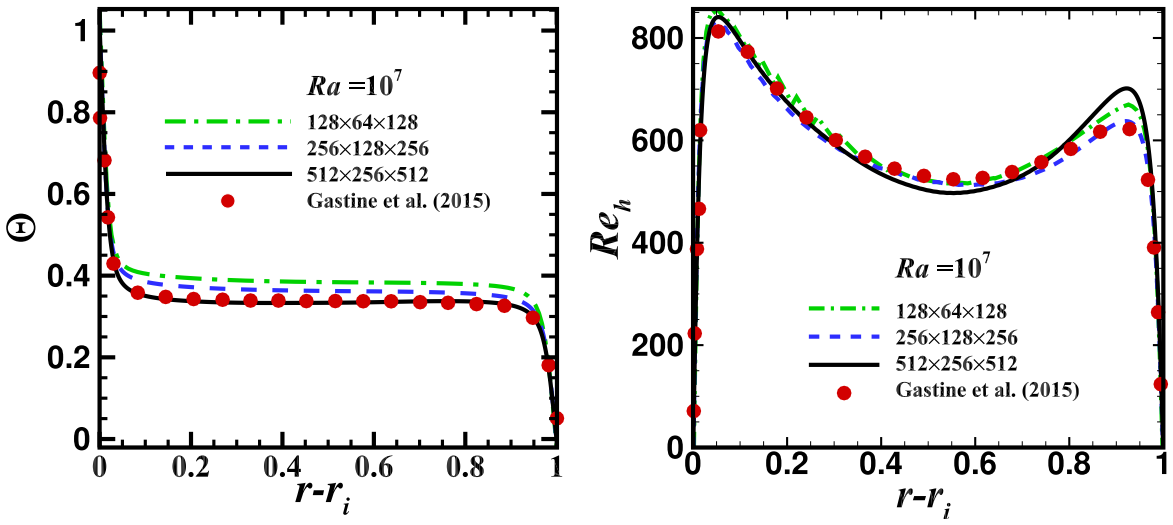
After the transformation, all the grid spacings are unity, and the information about grid stretching is provided effectively through the elongation matrix. At the bottom and top surfaces, a no-slip boundary condition is used ($u_1 = u_2 = u_3 = 0$), while the temperatures are fixed at the bottom ($T = 1$) and top ($T = 0$) surfaces to impose an unstable gradient for maintaining thermal convection. The periodic boundary condition is used for all the variables in the ξ directions. The 'shell-cut' boundary conditions are used in the ζ direction, as explained before in Section 3.5.3. All simulations are started with $u_i = 0$ and small random perturbations in the temperature field.

We use the following notations for the surface, volume and time-averaged quantities.

$$\langle f\rangle_s = \frac{1}{4\pi}\int_0^{2\pi}\int_0^{\pi} f\,\sin\theta\,d\theta\,d\phi, \tag{48}$$

Table 2. Summary table for different Ra along with the grid used for each case. Here, Prandtl number $Pr = 1$, gravity profile, $g = (r_o/r)^2$ and radius ratio $\Gamma = 0.6$ has been used for all the cases.

Ra	Grid ($n_\zeta \times n_\eta \times n_\xi$)	Nu (present DNS)	Nu (Gastine *et al.* 2015)	Re (present DNS)	Re (Gastine *et al.* 2015)	δ_i^T/δ_o^T	δ_i^u/δ_o^u
10^5	$64 \times 64 \times 64$	4.70	4.71	83.5	82.3	0.105/0.115	0.045/0.052
10^7	$512 \times 256 \times 512$	17.96	17.07	816.9	790.4	0.02/0.032	0.013/0.024



(a) (b)

Figure 9. Comparison of surface and time-averaged (a) non-dimensional radial temperature and (b) Re_h profile for the case $Ra = 10^7$.

$$\langle f \rangle = \frac{1}{V} \int_{r_i}^{r_o} \int_0^{2\pi} \int_0^{\pi} f\, r^2 \, \sin\theta \; d\theta \; d\phi \; dr, \tag{49}$$

$$\overline{f} = \frac{1}{\tau} \int_{t_0}^{t_0+\tau} f \; d\tau, \tag{50}$$

where $V = \frac{4}{3}\pi(r_o^3 - r_i^3)$. All the statistical quantities are averaged in time for at least 100 free fall time ($t_f = d/u_f$) units after the simulation reaches a steady state.

The average diagnostic quantities such as r.m.s. temperature, horizontal velocity, Reynolds number and horizontal Reynolds number are defined below.

$$T_{rms}(r) = \sqrt{\overline{\langle (T - \overline{\langle T \rangle_s})^2 \rangle}} \tag{51}$$

$$u_h(r) = \sqrt{\langle u_\theta^2 + u_\phi^2 \rangle_s} \tag{52}$$

$$Re = \frac{u_{rms} d}{\nu} \tag{53}$$

$$Re_h(r) = \frac{u_h d}{\nu} \tag{54}$$

Here $u_{rms} = \sqrt{\overline{\langle u_r^2 + u_\theta^2 + u_\phi^2 \rangle}}$ is the r.m.s. velocity. Further, the heat transport in the spherical shell is quantified by the Nusselt number Nu, which is defined as

$$Nu = \frac{\overline{\langle u_r T \rangle_s} - \frac{1}{\sqrt{RaPr}} \frac{\mathrm{d}\Theta}{\mathrm{d}r}}{-\frac{1}{\sqrt{RaPr}} \frac{\mathrm{d}T_c}{\mathrm{d}r}} = -\Gamma \frac{\mathrm{d}\Theta}{\mathrm{d}r}(r = r_i) = -\frac{1}{\Gamma} \frac{\mathrm{d}\Theta}{\mathrm{d}r}(r = r_o), \tag{55}$$

where $\Theta(r) = \overline{\langle T \rangle_s}$, $\langle \rangle_s$ represents the average over the spherical surface (eq. 48) and the overbar represents the time average (eq. 50). Here, Tc eq. (57). The thermal conduction equation for a spherical shell with isothermal boundary condition is given by

$$\frac{\mathrm{d}}{\mathrm{d}r}\left(r^2 \frac{\mathrm{d}T_c}{\mathrm{d}r}\right) = 0, \quad T_c(r = r_i) = 1, \quad T_c(r = r_o) = 0, \tag{56}$$

which yields

$$T_c(r) = \frac{\Gamma}{(1-\Gamma)^2} \frac{1}{r} - \frac{\Gamma}{1-\Gamma}. \tag{57}$$

5 RESULTS

This section summarizes the results of the simulations at two Rayleigh numbers $Ra = 10^5$ and 10^7, as listed in Table 2. We validate our results with those of Gastine *et al.* (2015) and further demonstrate the closure of the turbulent kinetic energy budget.

5.1 Validation

We compare the non-dimensionalized radial temperature profile for $Ra = 10^7$ in Fig. 9(a) against the profile reported by Gastine *et al.* (2015). We also compare the horizontal velocity profile (non-dimensionalized as Reynolds number, Re_h as defined in eq. (54)) in Fig. 9(b). Additionally, we compare the Nu as defined in eq. (55) with the values reported by Gastine *et al.* (2015) at the same Ra in Table 2. The Nu in the present simulation also matches very well with the values reported in Gastine *et al.* (2015).

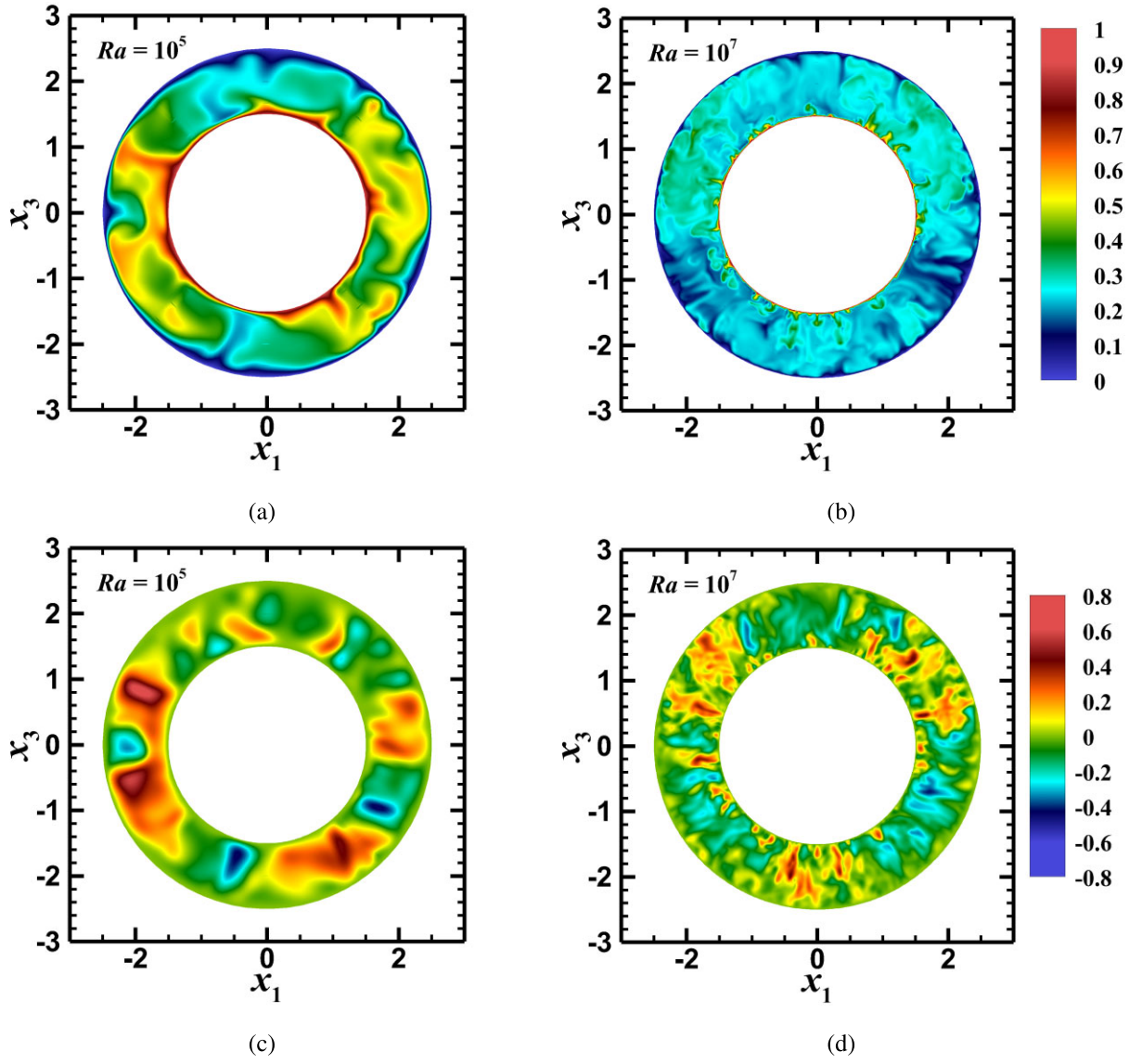


Figure 10. Instantaneous snapshots of (a, b) the thermal field and (c, d) radial velocity field observed at Rayleigh numbers (a, c) $Ra = 10^5$ and (b, d) $Ra = 10^7$.

Table 3. *Nu* obtained for $Ra = 10^7$ with increasing grid resolution.

Ra	Grid ($n_\zeta \times n_\eta \times n_\xi$)	*Nu*
10^7	$128 \times 64 \times 128$	19.88
	$256 \times 128 \times 256$	19.10
	$512 \times 256 \times 512$	17.96

A convergence test has been performed at $Ra = 10^7$ to evaluate the sensitivity of the Nusselt number estimate to spatial resolution. The *Nu* values obtained (using eq. 55), with increasing grid resolution are listed in Table 3. We can observe that the *Nu* values from our simulations approaches that of Gastine *et al.* (2015) ($Nu = 17.07$), with increasing resolution.

5.2 Flow visualization

A comparison of the qualitative features of the instantaneous flow and the thermal field with an increase of *Ra* from 10^5 to 10^7 is presented in Fig. 10. For lower thermal forcing at $Ra = 10^5$, the plumes generated from the boundaries span the radial extent of the domain, as seen from Fig. 10(a). However, $Ra = 10^7$, as shown in Fig. 10(b), the plumes are much smaller with a well-mixed interior. With the increase in *Ra*, turbulence increases, accompanied by higher mixing and generation of smaller scales. The higher *Ra* case will have a negligible temperature gradient in bulk due to enhanced mixing. In Fig. 10(c), the alternating regions with positive and negative radial velocities indicate the presence of structures similar to convective rolls, while Fig. 10(d) exhibit the presence of small-scale plumes near the boundary at higher *Ra*.

5.3 Boundary layer asymmetry

The thermal boundary layer thicknesses (δ_i^T, inner δ_o^T, outer) are defined as the distance of the local maximums in the $T_{\rm rms}$ (eq. 51) profile from the inner and the outer walls, respectively. The velocity boundary layers thicknesses (δ_i^u, inner δ_o^u, outer) are evaluated by the slope method, which defines them as the distance from the respective boundaries where the linear fit to Re_h (eq. 54) at the boundary intersects the horizontal line passing through

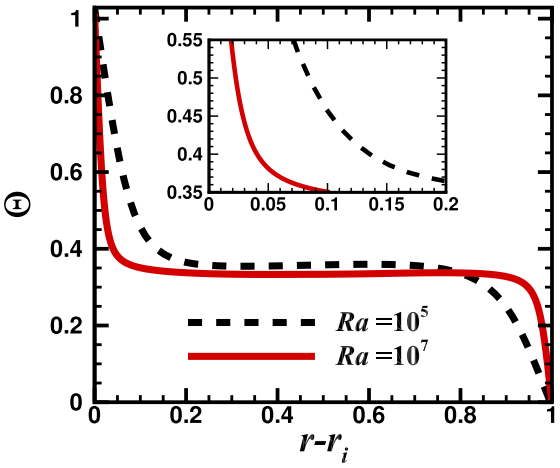


Figure 11. Radial temperature profiles for the two *Ra*. An enlarged view in the inset demonstrates the increase in gradient with the increase in *Ra*.

the maximum horizontal velocity (Gastine *et al.* 2015). From the values of the boundary layer thickness at the inner and the outer boundary from Table 2, it is apparent that there is an asymmetry in the temperature profile between the inner and the outer radius. The total heat flowing in through the inner surface should flow out from the outer surface for thermal equilibrium. In conjunction with the inner spherical shell area being less than the outer spherical shell area, the temperature drop is higher at the inner boundary than at the outer boundary (Gastine *et al.* 2015). We also observe a steepening of the temperature profile near the boundaries that occurs with an increase in *Ra*, as shown in the insets of Fig. 11.

5.4 Turbulent kinetic energy budget

In this section, we discuss the turbulent kinetic energy (*t.k.e.*) budget in RBC to check the energy balance and the adequacy of the resolution of the present simulations. The *t.k.e.* budget can be expressed as follows:

$$\left\langle \frac{\mathrm{d}\mathcal{K}}{\mathrm{d}t} \right\rangle = \langle \mathcal{B} \rangle - \langle \epsilon_v \rangle \,, \tag{58}$$

where

$$\langle \mathcal{K} \rangle = \left\langle \frac{1}{2} u_i u_i \right\rangle, \ \langle \mathcal{B} \rangle = \langle g u_r T \rangle \,, \ \langle \epsilon_v \rangle = \sqrt{\frac{Pr}{Ra}} \left\langle (\nabla \times u)^2 \right\rangle. \tag{59}$$

In the RHS of eq. (58), the buoyancy flux $\langle \mathcal{B} \rangle$ is the source term that converts the available potential energy to turbulent kinetic energy to drive the convective motions. This *t.k.e.* is converted to internal energy by the viscous dissipation term $\langle \epsilon_v \rangle$, (Tennekes & Lumley 1972), which acts as a sink. The buoyancy flux averaged over the whole spherical volume can be expressed as

$$\langle \mathcal{B} \rangle = \frac{4\pi}{V} \int_{r_i}^{r_o} g r^2 \overline{\langle u_r T \rangle_s} \, dr. \tag{60}$$

After substituting *Nu* from eq. (55) and T_c from eq. (57), we obtain,

$$\langle \mathcal{B} \rangle = \frac{3}{1+\Gamma+\Gamma^2} \frac{1}{\sqrt{RaPr}} (Nu - 1) = \sqrt{\frac{Pr}{Ra}} \overline{\left\langle (\nabla \times u)^2 \right\rangle}, \tag{61}$$

$$\chi_{\epsilon_v} = \frac{\overline{\langle \epsilon_v \rangle}}{\overline{\langle \mathcal{B} \rangle}} \tag{62}$$

Table 4. The turbulent kinetic energy budget terms and the viscous dissipation ratio, χ_{ϵ_v} with increasing grid resolution demonstrating grid convergence.

Ra	Grid	$\overline{\langle \mathcal{B} \rangle}$	$\overline{\langle \epsilon_v \rangle}$	χ_{ϵ_v}
10^5	$64 \times 64 \times 64$	1.87×10^{-2}	1.57×10^{-2}	0.84
10^5	$128 \times 128 \times 128$	1.96×10^{-2}	1.83×10^{-2}	0.93
10^5	$256 \times 256 \times 256$	2.04×10^{-2}	1.98×10^{-2}	0.97
10^5	$512 \times 256 \times 512$	2.27×10^{-2}	2.22×10^{-2}	0.98
10^5	$512 \times 512 \times 512$	2.36×10^{-2}	2.34×10^{-2}	0.99
10^7	$512 \times 256 \times 512$	8.50×10^{-3}	7.67×10^{-3}	0.90

The evolution of the *t.k.e.* budget terms in eq. (58) for the case $Ra = 10^5$ is shown in Fig. 12(a). We evaluate the volume-averaged *t.k.e.* budget terms for three grid resolutions $64 \times 64 \times 64$ (grid1), $128 \times 128 \times 128$ (grid2) and $256 \times 256 \times 256$ (grid3). The balance term signifies the difference between the left- and right-hand sides of eq. (58). As the dissipation increases with resolution, the balance becomes smaller than 1 per cent of $\langle \mathcal{B} \rangle$ for grid2 and grid3, indicating sufficient resolution achieved in these simulations to dissipate all the kinetic energy. To further quantify the spatial resolution of the numerical model, the viscous dissipation ratio (χ_{ϵ_v}) defined by eq. (62) is also tested for its closeness to unity (Gastine *et al.* 2015). The Table 4 shows that χ_{ϵ_v} approaches unity with increasing grid resolution, signifying grid convergence for $Ra = 10^5$. For $Ra = 10^7$, we obtain $\chi_{\epsilon_v} = 0.9$, and it is possible that the value of χ_{ϵ_v} will also approach 1, with an increase in the number of grids as we observed for $Ra = 10^5$. To test the adequacy of the resolution further, the radial grid spacing is compared against the Kolmogorov scale (l_η) defined by, $l_\eta = (Pr/Ra)^{\frac{3}{8}} (1/\overline{\epsilon_v})^{\frac{1}{4}}$. As seen from the Fig. 12(b), the radial grid spacing normalized by l_η is near unity for all the *Ra* cases near the walls, indicating appropriate wall resolution. As seen in Fig. 12(b), the normalized spacing stays below 2 for all the cases, which is sufficient for accurate calculation of second-order correlations (Brucker & Sarkar 2010; Naskar & Pal 2022a, b; Singh & Pal 2023).

5.5 Strong scaling test

We have performed a strong scaling test following Matsui *et al.* (2016) to test the performance of the numerical methods implemented in the solver. All computations are performed on Intel Xeon Platinum 8268 CPUs. To standardize our tests, we exclude initialization and all data IO operations. The tests were run at $Ra = 10^5$, using saved data (i.e. from a simulation run that achieved a statistically stationary state) as an initial condition, keeping the time-step size (Δt in Fig. 5) at a constant value. For this strong scaling test, the grid size is kept constant at $256 \times 256 \times 256$ while the number of processing cores increases gradually. We measured the time step per iteration, which remains approximately constant over the iterations. The variation of time-step per iteration with the number of cores is plotted in Fig. 13(a). We fit a power law of $Y = aX^n$, where Y is the time per iteration and X is the number of processing cores. Ideally, a strong scaling test should demonstrate an exponent of $n = -1$ if the time/iteration decreases proportionally with the increase in the number of cores (Matsui *et al.* 2016). We get a scaling exponent of $n = -1.035$ up to 128 cores, after which the scaling deviates from its ideal value for 256 cores owing to increased communication time. We further compute the scaling efficiency, following Matsui *et al.* (2016), as defined below.

$$\epsilon = \frac{N_{ref}}{N_{core}} \frac{t_{ref}}{t_{core}} \tag{63}$$

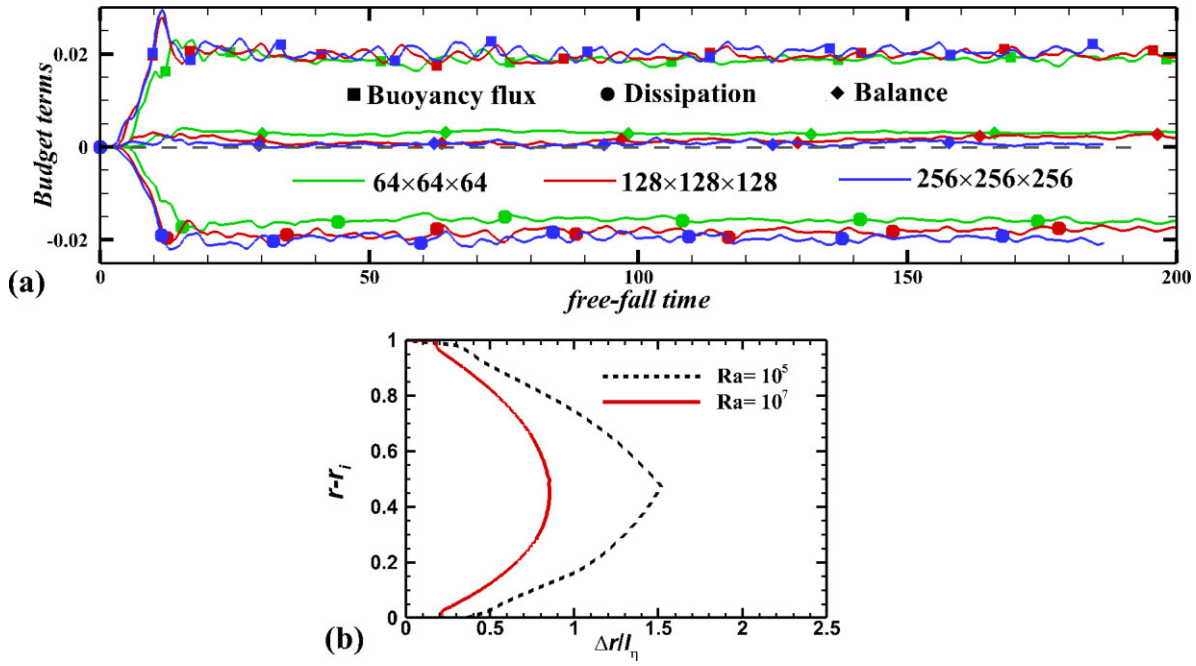


Figure 12. (a) The *t.k.e.* budget terms for $Ra = 10^5$ and (b) the radial variation of grid spacing (Δr) normalized by the Kolmogorov scale (l_η) as estimated from the spatially averaged dissipation ($\langle \epsilon_v \rangle_s$).

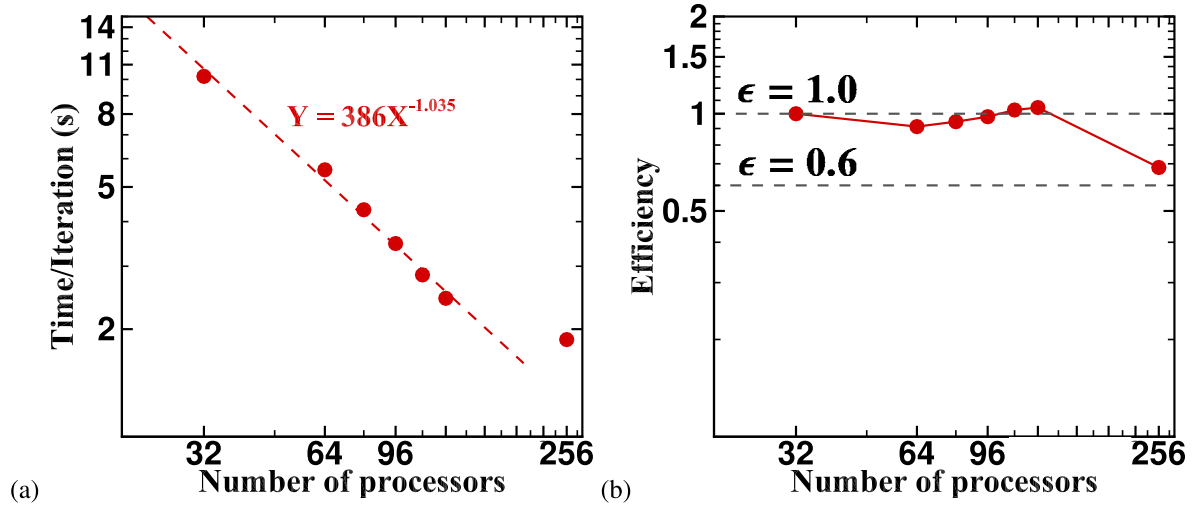


Figure 13. A comparison of the time per iteration as a function of the number of CPUs for different grids.

Here, N_{ref} and t_{ref} are the reference number of processors and the corresponding time taken per interaction. Here, time per iteration t_{core} should decrease proportionally as the number of processors, N_{core} increases. The scaling efficiency stays near unity up to 128 and falls to 0.68 for 256 cores. Matsui *et al.* (2016) further defined the parallelization limit as $\epsilon = 0.6$ to compare the scaling behaviour of contemporary solvers (see fig. 6 in their paper). The present solver stays above this limit up to the highest number of cores studied here (i.e. 256). Therefore, the scaling behaviour of the present solver is comparable to that of the contemporary solvers.

6 CONCLUSION

This present investigation discusses the development of a generalized curvilinear solver for spherical Rayleigh–Bénard convection. Using the Jacobi transformation, the solver transforms a curvilinear domain into a Cartesian domain, and a set of modified governing equations are solved in the Cartesian domain. The solver uses a second-order central differencing scheme for spatial discretization, while for temporal discretization, a combined marching scheme of ADI-CN-RKW3 is implemented. A parallel Thomas algorithm with pipelining is utilized to solve the tridiagonal system, which is more efficient and faster as it reduces the idle time for CPUs. To remove

the divergence residual from the projected velocity in the intermediate field of the fractional step method, a SMG routine from the HYPRE library for the pressure correction is used.

We have performed simulations at two Rayleigh numbers ($Ra = 10^5$ and 10^7). The primary emphasis is given to the ability of the solver to predict the heat transfer, quantified by the Nusselt number, *Nu*. Comparing *Nu* obtained from the numerical simulation with the expected value is not a reliable criterion to assess its validity because even the under-resolved schemes show good closeness with the *Nu* while producing temperature fields with strong non-physical oscillations (Kooij *et al.* 2018). Due to this fact, the solver is not only validated for *Nu* but also with the radial temperature and velocity profiles from Gastine *et al.* (2015). The radial temperature profile and *Nu* obtained from our solver match the results from Gastine *et al.* (2015). To further test the spatial resolution, we check on the viscous dissipation ratio's closeness to unity and turbulent kinetic energy budget closure. The *t.k.e*. budget exhibits better closure with increasing resolution. With increased *Ra*, the temperature profile near the boundaries becomes steeper. This is because the fluid near the boundaries is subjected to strong thermal gradients, generating a large buoyancy force that drives the flow. Therefore, the steepening of the temperature profile near the boundaries is the evidence of the buoyancy-induced strong convective flow with increased *Ra*. For a particular *Ra*, the temperature profile shows asymmetry due to the difference in area between the spherical inner and outer shell.

The majority of the computational methods developed for spherical shell Rayleigh–Bénard convection use spherical harmonic decomposition of the solution variables in the angular coordinates (θ, ϕ) while using finite difference or Chebyshev polynomials in the radial direction (Busse *et al.* 1998; Christensen *et al.* 1998, 1999; Dormy *et al.* 1998; Glatzmaier 1984; Sakuraba & Kono 1999; Tilgner 1999). A notable exception to this rule is reported in the work by Kageyama *et al.* (1995). However, all these methods were developed for perfectly spherical geometries. The novelty of the present solver lies in its capability to account for the effects of non-spherical geometries in planetary core convection. A strong scaling test suggests that our solver has comparable scalability to contemporary codes for simulating spherical shell convection.

Our ongoing work is focused primarily on extending the present solver to include the effects of rotation and magnetic field. Future extensions, with further model improvements, should reveal the possible effects of a non-spherical geometry on the convective patterns and the self-generated magnetic field in global numerical dynamo simulations.

ACKNOWLEDGMENTS

We thank the support and the resources provided by PARAM Sanganak under the National Supercomputing Mission, Government of India, at the Indian Institute of Technology Kanpur.

DATA AVAILABILITY

The data set generated from this study is available at Naskar *et al.* (2023).

REFERENCES

Ahlers, G., Grossmann, S. & Lohse, D., 2009. Heat transfer and large scale dynamics in turbulent Rayleigh–Bénard convection, *Rev. Mod. Phys.,* **81**(2), doi:10.1103/RevModPhys.81.503.

Aurnou, J.M., Calkins, M.A., Cheng, J.S., Julien, K., King, E.M., Nieves, D., Soderlund, K.M. & Stellmach, S., 2015. Rotating convective turbulence in earth and planetary cores, *Phys. Earth planet. Inter.,* **246,** 52–71.

Brown, P., Falgout, R., Jones, J. & Comput, S., 2000. Semicoarsening multigrid on distributed memory machines, *SIAM J. Sci. Comput.,* **21**(5), doi:10.1137/S1064827598339141.

Brucker, K. & Sarkar, S., 2010. A comparative study of self-propelled and towed wakes in a stratified fluid, *J. Fluid Mech.,* **652,** 373–404.

Busse, F., Grote, E. & Tilgner, A., 1998. On convection driven dynamos in rotating spherical shells, *Stud. Geophys. Geod.,* **42**(3), 211–223.

Cébron, D., Le Bars, M., Moutou, C. & Le Gal, P., 2012. Elliptical instability in terrestrial planets and moons, *Astron. Astrophys.,* **539,** doi:10.1051/0004-6361/201117741.

Chillà, F. & Schumacher, J., 2012. New perspectives in turbulent Rayleigh–Bénard convection, *Eur. Phys. J. E,* **35,** 1–25.

Chongsiripinyo, K., 2019. *Decay of Stratified Turbulent Wakes Behind a Bluff Body,* University of California.

Christensen, U., Olson, P. & Glatzmaier, G.A., 1998. A dynamo model interpretation of geomagnetic field structures, *Geophys. Res. Lett.,* **25**(10), 1565–1568.

Christensen, U., Olson, P. & Glatzmaier, G., 1999. Numerical modelling of the geodynamo: a systematic parameter study, *J. geophys. Int.,* **138**(2), 393–409.

Dormy, E., Cardin, P. & Jault, D., 1998. Mhd flow in a slightly differentially rotating spherical shell, with conducting inner core, in a Dipolar magnetic field, *Earth planet. Sci. Lett.,* **160**(1–2), 15–30.

Dormy, E., Soward, A.M., Jones, C.A., Jault, D. & Cardin, P., 2004. The onset of thermal convection in rotating spherical shells, *J. Fluid Mech.,* **501,** 43–70.

Dwyer, C., Stevenson, D. & Nimmo, F., 2011. A long-lived lunar dynamo driven by continuous mechanical stirring, *Nature,* **479**(7372), 212–214.

Falgout, R. & Jones, J., 2000. Multigrid on massively parallel architectures, in *Multigrid Methods VI. Lecture Notes in Computational Science and Engineering,* Vol. **14,** eds Dick, E., Riemslagh, K. & Vierendeels, J., Springer, Berlin, Heidelberg.

Falgout, R., Jones, J. & Yang, U., 2002. The design and implementation of *hypre*, a library of parallel high performance preconditioners, in *Numerical Solution of Partial Differential Equations on Parallel Computers. Lecture Notes in Computational Science and Engineering,* Vol. **51,** eds Bruaset, A. M. & Tveito, A.& Springer, Berlin, Heidelberg.

Featherstone, N. & Hindman, B., 2016. The spectral amplitude of stellar convection and its scaling in the high-Rayleigh-number regime, *Astrophys. J.,* **818**(1), doi:10.3847/0004-637X/818/1/32.

Forte, A.M., Mitrovica, J.X. & Woodward, R.L., 1995. Seismic-geodynamic determination of the origin of excess ellipticity of the core-mantle boundary, *Geophys. Res. Lett.,* **22**(9), 1013–1016.

Fournier, A., Bunge, H.-P., Hollerbach, R. & Vilotte, J.-P., 2004. Application of the spectral-element method to the axisymmetric Navier–Stokes equation, *J. geophys. Int.,* **156**(3), 682–700.

Gastine, T., Wicht, J. & Aurnou, J.M., 2015. Turbulent Rayleigh–Bénard convection in spherical shells, *J. Fluid Mech.,* **778,** 721–764.

Gastine, T., Wicht, J. & Aubert, J., 2016. Scaling regimes in spherical shell rotating convection, *J. Fluid Mech.,* **808,** 690–732.

Glatzmaier, G.A., 1984. Numerical simulations of stellar convective dynamos. I. The model and method, *J. Comput. Phys.,* **55**(3), 461–484.

Hartmann, D.L., Moy, L.A. & Fu, Q., 2001. Tropical convection and the energy balance at the top of the atmosphere, *J. Clim.,* **14**(24), 4495–4511.

Hollerbach, R., 2000. A spectral solution of the magneto-convection equations in spherical geometry, *Int. J. Numer. Methods Fluids,* **32**(7), 773–797.

Incropera, F.P., 1988. Convection heat transfer in electronic equipment cooling, *J. Heat Transfer.,* **110**(4b), 1097–1111.

Incropera, F.P., DeWitt, D.P., Bergman, T.L., Lavine, A.S. *et al.*, 1996. *Fundamentals of Heat and Mass Transfer,* Vol. **6,** Wiley.

Iyer, K., Scheel, J., Schumacher, J. & Sreenivasan, K., 2020. Classical 1/3 scaling of convection holds up to $Ra = 10^{15}$, *Proc. Natl. Acad. Sci. U.S.A.,* **117**(14), 7594–7598.

Jiang, W. & Kuang, W., 2008. An MPI-based Mosst core dynamics model, *Phys. Earth planet. Inter.,* **170**(1–2), 46–51.
Jones, C.A., 2011. Planetary magnetic fields and fluid dynamos, *Ann. Rev. Fluid Mech.,* **43,** 583–614.
Kageyama, A., Sato, T. & Groupa, C.S., 1995. Computer simulation of a magnetohydrodynamic dynamo. II, *Phys. Plasmas,* **2**(5), 1421–1431.
Kooij, G.L. *et al.*, 2018. Comparison of computational codes for direct numerical simulations of turbulent Rayleigh–Bénard convection, *Comput. Fluids,* **166,** 1–8.
Korre, L. & Featherstone, N.A., 2021. On the dynamics of overshooting convection in spherical shells: effect of density stratification and rotation, *Astrophys. J.,* **923**(1), doi:10.3847/1538-4357/ac2dea.
Le Bars, M., Wieczorek, M.A., Karatekin, Ö., Cébron, D. & Laneuville, M., 2011. An impact-driven dynamo for the early moon, *Nature,* **479**(7372), 215–218.
Long, R.S., Mound, J.E., Davies, C.J. & Tobias, S.M., 2020. Scaling behaviour in spherical shell rotating convection with fixed-flux thermal boundary conditions, *J. Fluid Mech.,* **889,** doi:10.1017/jfm.2020.67.
Marti, P.D., 2012. Convection and boundary driven flows in a sphere, *PhD thesis*, ETH Zurich.
Matsui, H. & Okuda, H., 2004. Development of a simulation code for MHD dynamo processes using the geofem platform, *Int. J. Comput. Fluid Dyn.,* **18**(4), 323–332.
Matsui, H., King, E. & Buffett, B., 2014. Multiscale convection in a geodynamo simulation with uniform heat flux along the outer boundary, *Geochem. Geophys. Geosyst.,* **15**(8), 3212–3225.
Matsui, H. *et al.*, 2016. Performance benchmarks for a next generation numerical dynamo model, *Geochem. Geophys. Geosyst.,* **17**(5), 1586–1607.
Mound, J.E. & Davies, C.J., 2017. Heat transfer in rapidly rotating convection with heterogeneous thermal boundary conditions, *J. Fluid Mech.,* **828,** 601–629.
Naskar, S. & Pal, A., 2022a. Direct numerical simulations of optimal thermal convection in rotating plane layer dynamos, *J. Fluid Mech.,* **942,** doi:10.1017/jfm.2022.402.
Naskar, S. & Pal, A., 2022b. Effects of kinematic and magnetic boundary conditions on the dynamics of convection-driven plane layer dynamos, *J. Fluid Mech.,* **951,** doi:10.1017/jfm.2022.841.
Naskar, S., Chongsiripinyo, K., Mishra, S., Pal, A. & Jananan, A., 2023. doi:10.5281/zenodo.11198071.
Ribeiro, A., Fabre, G., Guermond, J.-L. & Aurnou, J.M., 2015. Canonical models of geophysical and astrophysical flows: turbulent convection experiments in liquid metals, *Metals,* **5**(1), 289–335.
Rincon, F., 2019. Dynamo theories, *J. Plasma Phys,* **85**(4), doi:10.1017/S0022377819000539.
Roberts, P. & King, E., 2013. On the genesis of the earth's magnetism, *Rep. Prog. Phys.,* **76**(9), doi:10.1088/0034-4885/76/9/096801.
Rosenfeld, M., Kwak, D. & Vinokur, M., 1991. A fractional step solution method for the unsteady incompressible navier-stokes equations in generalized coordinate systems, *J. Comput. Phys.,* **94**(1), 102–137.
Rüdiger, G. & Hollerbach, R., 2006. *The Magnetic Universe: Geophysical and Astrophysical Dynamo Theory,* John Wiley & Sons.
Ruge, J. & Stüben, K., 1987. Algebraic multigrid, in *Multigrid Methods,* pp. 73–130, SIAM.
Sakuraba, A. & Kono, M., 1999. Effect of the inner core on the numerical solution of the magnetohydrodynamic dynamo, *Phys. Earth planet. Inter.,* **111**(1–2), 105–121.
Santelli, L., Orlandi, P. & Verzicco, R., 2021. A finite–difference scheme for three-dimensional incompressible flows in spherical coordinates, *J. Comput. Phys.,* **424,** doi:10.1016/j.jcp.2020.109848.
Sasaki, Y., Takehiro, S.-I., Kuramoto, K. & Hayashi, Y.-Y., 2011. Weak-field dynamo emerging in a rotating spherical shell with stress-free top and no-slip bottom boundaries, *Phys. Earth planet. Inter.,* **188**(3–4), 203–213.
Schaeffer, N., Jault, D., Nataf, H.-C. & Fournier, A., 2017. Turbulent geodynamo simulations: a leap towards Earth's core, *J. geophys. Int.,* **211**(1), 1–29.
Simitev, R. & Busse, F.H., 2005. Prandtl-number dependence of convection-driven dynamos in rotating spherical fluid shells, *J. Fluid Mech.,* **532,** 365–388.
Singh, N. & Pal, A., 2023. The onset and saturation of the Faraday instability in miscible fluids in a rotating environment, *J. Fluid Mech.,* **973,** doi:10.1017/jfm.2023.744.
Takahashi, F., 2012. Implementation of a high-order combined compact difference scheme in problems of thermally driven convection and dynamo in rotating spherical shells, *Geophys. Astrophys. Fluid Dyn.,* **106**(3), 231–249.
Taylor, J., 2008. *Numerical simulations of the stratified oceanic bottom boundary layer publication date, PhD dissertation*, University of California, San Diego.
Tennekes, H. & Lumley, J., 1972. *A First Course in Turbulence,* The MIT Press.
Tilgner, A., 1999. Spectral methods for the simulation of incompressible flows in spherical shells, *Int. J. Numer. Methods Fluids,* **30**(6), 713–724.
Vantieghem, S., Sheyko, A. & Jackson, A., 2016. Applications of a finite-volume algorithm for incompressible MHD problems, *J. geophys. Int.,* **204**(2), 1376–1395.
Wicht, J., 2002. Inner-core conductivity in numerical dynamo simulations, *Phys. Earth planet. Inter.,* **132**(4), 281–302.
Willis, A.P., Sreenivasan, B. & Gubbins, D., 2007. Thermal core–mantle interaction: exploring regimes for 'locked' dynamo action, *Phys. Earth planet. Inter.,* **165**(1–2), 83–92.
Wolstencroft, M., Davies, J.H. & Davies, D.R., 2009. Nusselt–Rayleigh number scaling for spherical shell earth mantle simulation up to a Rayleigh number of 10^9, *Phys. Earth planet. Inter.,* **176**(1–2), 132–141.
Yadav, R.K. & Bloxham, J., 2020. Deep rotating convection generates the polar hexagon on Saturn, *Proc. Natl. Acad. Sci. U.S.A.,* **117**(25), 13 991–13 996.
Yadav, R.K., Heimpel, M. & Bloxham, J., 2020. Deep convection–driven vortex formation on Jupiter and Saturn, *Sci. Adv.,* **6**(46), doi:10.1126/sciadv.abb9298.