ICCM2014
28-30th July, Cambridge, England

Two computational approaches for the simulation of fluid problems in rotating spherical shells

*Ferran Garcia[1,2], Emmanuel Dormy[2], Juan Sánchez[1], and Marta Net[1]

[1] Dept. de Física Aplicada, Universitat Politècnica de Catalunya, Jordi Girona Salgado s/n. Campus Nord. Mòdul B4, 08034 Barcelona, Spain.
[2] MAG, LRA, Département de Physique, Ecole Normale Supérieure, 24 rue Lhomond, 75005 Paris, France.

*Corresponding author: ferran@fa.upc.edu

Abstract

Many geophysical and astrophysical phenomena such as magnetic fields generation, or the differential rotation observed in the atmospheres of the major planets are studied by means of numerical simulations of the Navier-Stokes equations in rotating spherical shells. Two different computational codes, spatially discretized using spherical harmonics in the angular variables, are presented. The first code, PARODY, solves the magneto-hydrodynamic anelastic convective equations with finite difference discretization in the radial direction. This allows the parallelization on distributed memory computers to perform massive numerical simulations of second order in time. It is mainly designed to perform direct numerical simulations. The second code, SPHO, solves the fully spectral Boussinesq convective equations, and its variationals, parallelized on shared memory architectures and it uses optimized linear algebra libraries. High-order time integration methods are implemented to allow the use of dynamical systems tools for the study of complex dynamics.

Introduction

Due to the increase of computing power in the last decades, many geophysical and astrophysical phenomena, such as magnetic fields generation, or the differential rotation observed in the atmosphere of the major planets, are studied by means of three-dimensional numerical simulations of the magneto-hydrodynamic

or thermal convection equations in rotating spherical geometries. The introductory sections of [8, 18], among others, provide good reviews of the state of the art on this subject. The difficulties related to the experimental studies, such as the radial gravity which can only be reproduced by means of either an electrostatic radial field or approximated by the centrifugal force, enhance the importance of the numerical approach in these fields. However, non-stationary tridimensional waves arise at the onset of convection due to the boundary curvature, and thus finding a solution requires very high resolutions. Frequently, as in [21], and [22], a two-dimensional annular geometry is used to approximate the real problem. For this reason the development and improvement of the numerical techniques provides the basis for such research.

Several numerical codes to simulate these type of problems were developed independently by different research groups and benchmarked in [5]. A common feature of these codes is that the velocity and magnetic fields are expressed in terms of poloidal and toroidal scalars following the formulation of [3]. For the spatial discretization of the equations on the sphere, many of these codes use pseudo-spectral methods based on spherical harmonics basis functions in the angular variables, which provide highly accurate solutions with relatively few grid points [2]. These methods are based on transformations from the spectral to the physical space [19]. The calculation of the quadratic terms, appearing in the truncated equations, is performed in the physical space. The main differences between the codes arise in the discretization along the radial direction, in the implementation of the boundary conditions and in the time-stepping procedures. There exist however other approaches such as that of [16] that use finite differences in all directions.

Most of the current tridimensional studies consist of direct numerical simulations of periodic, quasi-periodic, and even turbulent flows to study the variation of the time-averaged physical properties in the parameter space and to obtain scaling laws [4, 20]. These numerical simulations are performed with second order time integration semi-implicit schemes which thread only the diffusive terms implicitly. For a deeper understanding of the origin of the laminar flows and their dependence on parameters, pseudoarclength continuation methods [25, 26], and the linear stability analysis of the time dependent solutions [18, 13] have been successfully applied thanks to the use of high-order time integration methods which provide accurate enough solutions. On the other hand, high-order time integration can also be useful for evolving turbulent flows efficiently [10].

In this paper two different computational parallel codes, spatially discretized using spherical harmonics in the angular variables, are presented and their applicability for studying geophysical and astrophysical problems is discussed. Also, their parallel performance on the high performance computing center MesoPSL (http://www.mesopsl.fr) is analyzed and possible improvements of the codes are suggested.

The first code, PARODY, solves the magneto-hydrodynamic anelastic convective equations, although in this paper we only comment the Boussinesq implementation, with finite difference discretization in the radial direction. This

allows the parallelization on distributed memory computers to perform massive numerical simulations of second order in time. It is mainly designed to perform direct numerical simulations and it has been widely used by many researchers, see for instance [7, 23, 27, 28].

The second code solves the fully spectral Boussinesq convective equations, and its variationals, parallelized on shared memory architectures and it uses optimized linear algebra libraries. High-order time integration methods [10, 11, 12] are implemented to allow the use of dynamical systems tools, such as that of [25, 26], for the study of complex dynamics.

The model and the equations

The thermal convection and magnetic field generation of a spherical electrically conducting fluid shell differentially heated, rotating about an axis of symmetry with constant angular velocity $\mathbf{\Omega} = \Omega\mathbf{k}$, and subject to radial gravity $\mathbf{g} = -\gamma\mathbf{r}$, where γ is a constant, and $\mathbf{r}$ the position vector, is implemented in the code PARODY. The mass, momentum, energy and induction equations are written by using an usual formulation and non-dimensional units (see [5, 6, 7, 27] for details). The units are the gap width, $d = r_o - r_i$, for the distance, ΔT for the temperature, d^2/ν for the time, and $(\rho\mu\eta\Omega)^{1/2}$ for the magnetic field, ν being the kinematic viscosity, μ the magnetic permeability, η the magnetic diffusivity and r_i and r_o the inner and outer radii, respectively. With these units the equations governing the dynamics of the fluid in the rotating frame of reference are

$$E\Big(\partial_t\mathbf{v} + (\mathbf{v}\cdot\nabla)\,\mathbf{v} - \nabla^2\mathbf{v}\Big) = -2\mathbf{\Omega}\times\mathbf{v} - \nabla p + Ra\frac{r}{r_o}T + P_m^{-1}\,(\nabla\times\mathbf{B})\times\mathbf{B}, \tag{1}$$

$$\nabla\cdot\mathbf{v} = 0, \tag{2}$$

$$\partial_t T + \mathbf{v}\cdot\nabla T = P_r^{-1}\nabla^2 T, \tag{3}$$

$$\partial_t\mathbf{B} = \nabla\times(\mathbf{v}\times\mathbf{B}) + P_m^{-1}\nabla^2\mathbf{B}, \tag{4}$$

$$\nabla\cdot\mathbf{B} = 0. \tag{5}$$

The governing parameters are the modified Rayleigh number Ra, the Prandtl number Pr, the magnetic Prandtl number P_m, the Ekman number E, and the radius ratio χ. They are defined by

$$Ra = \frac{g_0\alpha\Delta T d}{\nu\Omega}, \quad E = \frac{\nu}{\Omega d^2}, \quad P_r = \frac{\nu}{\kappa}, \quad P_m = \frac{\nu}{\eta}, \quad \chi = \frac{r_i}{r_o},$$

where κ is the thermal diffusivity, and ΔT the difference of temperature between the inner and outer boundaries.

The boundary conditions for the velocity field can be either no slip or stress free at both boundaries, or mixed boundary conditions with no slip at the inner and stress free at the outer sphere. For the magnetic field, conducting or

insulating inner core can be imposed, although only the insulating case will be considered in this paper [6, 27]. The temperature is fixed at both boundaries.

The solenoidal velocity field, $\mathbf{v}$, is expressed in terms of toroidal, u_t, and poloidal, u_p, scalar potentials $\mathbf{v} = \nabla \times (u_t \mathbf{r}) + \nabla \times \nabla \times (u_p \mathbf{r})$. With the same expression for the magnetic field and by applying the operators $\mathbf{r} \cdot \nabla \times$ and $\mathbf{r} \cdot \nabla \times \nabla \times$ to the Navier-Stokes equation (Eq. 1), and $\mathbf{r} \cdot$ and $\mathbf{r} \cdot \nabla \times$ to the induction equation (Eq. 4), the equations for the potentials can be deduced. Finally, the functions $X = (u_t, u_p, b_t, b_p, T)$ are expanded in spherical harmonic series up to degree L in the angular variables, namely

$$X(t, r, \theta, \varphi) = \sum_{m=-L}^{L} \sum_{l=|m|}^{L} X_l^m(r, t) Y_l^m(\theta, \varphi), \tag{6}$$

with $X_l^{-m} = \overline{X_l^m}$, and $[u_t]_0^0 = [u_p]_0^0 = [b_t]_0^0 = [b_p]_0^0 = 0$ to uniquely determine the four scalar potentials, and $Y_l^m(\theta, \varphi) = P_l^m(\cos\theta) e^{im\varphi}$, P_l^m being the normalized associated Legendre functions of degree l and order m. As $X_l^{-m} = \overline{X_l^m}$ only the $m \geq 0$ amplitudes are retained. With the latter expansion, the equations can be written in terms of their complex coefficients $X_l^m = X_l^m(t, r)$ which are functions of time and radius. The coefficients of the nonlinear terms of Eqs. (1-5) are obtained following [6].

A similar model is implemented in the code SPHO without the induction equation Eq. (4). The energy equation Eq. (3) is written in terms of the temperature perturbation $\Theta = T - T_c$ from the conductive state $\mathbf{v} = \mathbf{0}$, $T_c = T_c(r)$, with $r = ||\mathbf{r}||_2$. The unit for the temperature is $\nu^2/\gamma\alpha d^4$. The main difference between the codes arise in the radial discretization of the amplitudes $X_l^m(t, r)$, in the time-stepping techniques, and in the parallel strategy used to solve the equations. All these issues are addressed in the following section.

Parallel implementation

The PARODY code

Finite differences are used on a non-uniform mesh of $N_r + 1$ points, stretched near the boundaries to cope with thin Ekman-Hartmann layers. Although finite differences are local and less accurate with respect to other discretizations such as global collocation methods, they are suitable for a parallel implementation on distributed computers in the way we now describe. The radial grid is partitioned among the processors, $p_i, \quad i = 1, .., N_p$, each one having all the spherical harmonic amplitudes at $r_{d_i}, ..., r_{d_i+n_i}$ consecutive $n_i + 1$ radial points. The radial derivative operators are of second order except in the case of the poloidal scalar velocity which is of fourth order. If centered finite differences are used, to apply the derivative operators each processor p_i has to communicate all the amplitudes at r_{d_i} with p_{i-1}, and all the amplitudes at $r_{d_i+n_i}$ with p_{i+1}. In the case of the poloidal scalar velocity the amplitudes at r_{d_i+1} and at $r_{d_i+n_i-1}$ must also be send to processors p_{i-1} and p_{i+1}, respectively. This parallelization is

suitable because the evaluation of the nonlinear terms is the most demanding task and it is performed separately by each processor with the only need of communication for two vectors.

Once the original equations Eqs. (1-5) are discretized a large system of $N = 2(L^2 + 2L)N_r + (3L^2 + 6L + 1)(N_r - 1)$ ordinary differential equations must be advanced in time. For time-integration, semi-implicit methods are used, namely, only the diffusive terms are threaded implicitly with a Crank-Nicholson scheme, and the rest of the terms which include the non-linear and the Coriolis terms are treaded explicitly with an Adams-Bashforth method. Thus the linear systems of equations to be solved at every step can be separated into spherical harmonic components, which can be solved independently, so that only a set of small linear systems must be solved at each time step. These linear systems are pentadiagonal in the case of the poloidal velocity and tridiagonal for the other scalars. More specifically, the pentadiagonal matrix comes from the radial discretization of $(\frac{\partial}{\partial t} - \Delta)\Delta$, while the tridiagonal matrices come from the radial discretization of $\frac{\partial}{\partial t} - \beta\Delta$, where $\beta = 1$ in the case of the toroidal velocity potential, $\beta = 1/Pr$ in the case of the temperature equation, and $\beta = 1/P_m$ for the equations of the magnetic field potentials.

The linear systems in PARODY are usually [7] solved with the parallel implementation of the LU factorization described in [17]. The main drawback of this solver is that becomes sequential when decreasing the number of radial points of each processor and increasing the number of processors significantly. In the current parallel LU implementation a minimum of 4 radial points are needed for each processor. An implementation of a parallel Krylov iterative solver [1, 24] could improve the solution of the linear systems. More precisely the IBiCGStab (Improved Stabilized version of BiConjugate Gradient Squared) method is an alternative form to BiCGStab which only involves a single global reduction operation instead of the usual 3 (or 4) [30]. This solver allows to assign only one radial point at each processor. Although this method is highly parallelizable because it only makes use of matrix products, its performance (number of iterations) depends strongly on the condition number of the matrix, which in our case is mainly influenced by the number of radial points and the time step used in the time integration. Thus several tests, with different N_r and time steps corresponding to different physical regimes, must to be performed to compare the performance of both solvers. Preliminary results addressing this issue will be shown later.

SPHO code

In contrast to PARODY, this code employs a collocation method on a Gauss-Lobatto mesh of $N_r + 1$ points ($N_r - 1$ being the number of inner points). With this global discretization the radial grid can not be partitioned into several processors of a distributed memory cluster for an efficient parallelization. Thus the parallelization of the code is performed in the angular variables assuming shared memory architectures to avoid communications. The linear discretized operators of the equations for the spherical harmonics amplitudes are decou-

pled with respect the order m. The same occurs for the Legendre transforms needed for the computation of the nonlinear terms. Then, the triangular grid $\{X_l^m, \quad m = 0, ..., L, \quad l = m, ..., L\}$ is partitioned among the processors by assigning a set of amplitudes with consecutive order $m_{d_i}, ..., m_{d_i+n_i}$ at each processor. In this case, the number of orders, $n_i + 1$, assigned to each processor increases as m_{d_i} increase, to maintain a similar number of amplitudes X_l^m. Finally, the fast Fourier transforms and the computations in the physical space needed for evaluating the nonlinear terms are also parallelized by evenly partitioning the colatitude physical grid among the processors.

Once the thermal convection equations have been discretized a large system of ordinary differential equations of size $N = (3L^2 + 6L + 1)(N_r - 1)$ must be integrated in time. Notice the smaller number of equations with respect the PARODY code. In SPHO the induction equation is not considered. If N_v variational equations are integrated the size of the systems becomes $N_v N + N$.

Two classes of high order (up to five) time integration methods are implemented in SPHO. The first class of methods are the implicit-explicit (or fully implicit) backward differentiation formulas (IMEX-BDF) methods [11, 12]. The IMEX methods thread the nonlinear terms explicitly in order to avoid solving nonlinear equations at each time step. The Coriolis term is treated either semi-implicitly or fully implicitly, giving rise to different algorithms. The use of *matrix-free* Krylov methods (GMRES in our case) for the linear systems facilitates the implementation of a suitable order and time stepsize control. In contrast to PARODY, the matrices of linear systems to be solved in SPHO have dense blocks of dimension $O(N_r)$ (see [11] for details on the structure of these matrices). A second alternative implementation for the time stepping is the so called exponential Rosenbrock methods proposed in [15]. A wide range of numerical simulations has shown that such exponential methods are more accurate by at least one order of magnitude than the equivalent order IMEX scheme [10]. This is especially true when they are employed with large time steps and at small Ekman number.

Performance of the codes in MesoPSL

In this section we investigate the performance of PARODY and SPHO codes on the high performance computer MesoPSL, which consist of an array of 92 nodes with 16 cores and 64 Gb of memory ram each one. More precisely, each node is a bi-processor with 8-cores Intel E5-2670 at 2,60 Ghz and the nodes are interconnected with infiniband QDR.

PARODY

Three different dynamo test cases, corresponding to different physical regimes with the same geometry ($\chi = 0.35$) and Prandtl number ($Pr = 1$), have been considered for studying the behavior of the iterative solver. The first test case, T_1, corresponds to a laminar dynamo with $P_m = 5$ at relatively high $E = 10^{-3}$

and weakly supercritical modified Rayleigh number $Ra = 100$. This is the benchmark case 1 of [5]. The radial resolution is $N_r = 160$ and the spherical harmonics truncation parameter is $L = 64$. The time integration is performed with a time step of $\Delta t = 10^{-4}$. The second case, T_2, corresponds to a chaotic dynamo with $P_m = 0.5$ at $E = 10^{-4}$ and $Ra = 700$. The radial resolution is $N_r = 256$, the spherical harmonics truncation parameter is $L = 80$ and the time step is $\Delta t = 10^{-6}$. Finally in the third case, T_3, the complexity of the dynamo with $P_m = 0.25$ is increased because of the lower $E = 3 \times 10^{-5}$ and higher $Ra = 2 \times 10^3$. The radial resolution is $N_r = 320$, the spherical harmonics truncation parameter is $L = 112$, and the time step is $\Delta t = 3 \times 10^{-7}$.

As commented previously, the iterative history of the IBiCGStab solver depends strongly on the condition number of the matrices A_2 and A_1 coming from the discretization of $(\frac{\partial}{\partial t} - \Delta)\Delta$ and of $\frac{\partial}{\partial t} - \beta\Delta$, respectively. These matrices depend on the time step, but also on the degree l of the spherical harmonic amplitudes. The condition number of both matrices decreases with increasing the degree l, thus we have only computed the condition numbers of the case $l = 1$. For an easier implementation of the iterative solver we solve all the linear systems for all X_l^m as a single linear system, i.e, we perform the same number of iterations for each l. Then, as the condition number decreases with l, the residuals for the amplitudes decrease with increasing l.

When decreasing the time step, the matrix A_1 becomes close to a multiple of the identity and the matrix A_2 have always larger condition number than A_1. For the latter we have used a diagonal preconditioner to improve the convergence (see table 1) while minimizing the number of communications. For the former we have used a little bit more complicated preconditioner that we explain below. In both cases left preconditioning is better than right preconditioning.

Consider the matrix D coming from the discretization of the laplacian Δ with the appropriate boundary conditions. The preconditioning matrix for A_2 is $M_i = P_i Q_i$ where P_i and Q_i are the matrices corresponding to i Jacobi iterations with matrices D and A_1, respectively (see [1, 24] for further details on preconditioning techniques). In all the cases we have set $i = 2$ which reduces significantly the condition number and for which the preconditioning operation only requires one additional communication. See table 2 for the dependence of the condition numbers on the type of preconditioning, radial resolution and time step.

In figure 1 the run times for performing one time step when using the LU and IBiCGStab solvers is plotted versus the number of MPI tasks for each of the cases considered. In all the cases the tolerance for the IBiCGStab is set in a way that the mean physical properties (such as volume averaged kinetic energy densities or the Nusselt number) differ by less than 3% with respect that obtained with the LU solver when starting the integration from an initial condition as in [5]. The solution at the previous time instant has been chosen as initial seed for starting the iterations.

In the case T_1 (Fig. 1(a)) due to the relatively large time step $\Delta t = 10^{-4}$ the matrices are ill conditioned and the IBiCGStab solver requires at least 100 iterations for obtaining a residual of order 10^{-6} when solving the linear systems

N_r	40	80	160	250	350	500
A_1	5.6	25.3	112.8	295.7	595.3	1260.8
M_1A_1	3.6	13.0	50.4	122.9	239.3	489.1
M_2A_1	1.5	3.8	13.1	31.2	60.3	122.8
M_3A_1	1.5	4.4	16.8	41.0	79.8	163.0
M_4A_1	1.1	2.2	6.8	15.9	30.4	61.6

Table 1: Condition number dependence on the radial resolution and the type of preconditioner with $\Delta t = 10^{-4}$. M_iA_1, mean left preconditioner where M_i is the Jacobi preconditioner with i iterations. For $i = 1$ is the diagonal preconditioner.

with the matrix A_2. With this number of iterations the iterative solver requires considerably much more computing time than the direct one. For the case T_2 (Fig. 1(b)), the number of iterations is about 50 and thus the difference between the LU and IBiCGStab curves decreases. Finally, for the case T_3 (Fig. 1(c)) only 20 iterations are needed to achieve a residual of order 10^{-4} which has been found enough for obtaining good time-averaged values.

Notice in the slopes of the curves of Figs. 1(b,c) that the IBiCGStab solver has slightly better scalability when using a larger number of processors. In this figure a degradation of the scalability is also evident when using 16 processors because of the architecture of the computer (each node has 16 cores there is thus competition for a memory access). To address this issue in figure 1(d) the run time is plotted versus the number of nodes when using 16 MPI task for the test case T_3. It is clear that is better not to use all the cores of each node to avoid memory access competition, in this way, the computing time can be halved.

SPHO

In this section we describe the performance of the code SPHO parallelized using OpenMP directives and optimized by using basic linear algebra public libraries (GOTO [14] and ATLAS [29]) and the FFTW3 library for the fast Fourier transforms [9]. A test for the integration of the variational equations is also performed.

In figure 2(a) the run time for advancing one time step obtained with one core divided by the run time obtained by p cores ($p \leq 16$) is plotted versus the number of equations for several representative resolutions which are shown in Table 3. The run time for advancing one time step with a fixed time step integration method is basically that for computing the nonlinear terms and for solving the linear systems which are solved by an LU method. Because a direct solver is used, the physical regime play no role and the performance depends only on the discretization mesh.

When the number of equations is relatively small (up to 2×10^6) the performance degradates when using more than 8 cores because of the access memory competition, however, as the number of equations is increased there is more computational work and the competition for the memory decreases increasing

$\Delta t = 10^{-4}$

N_r	40	80	160	250	350	500
A_2	5.3×10^4	1.2×10^6	2.3×10^7	1.6×10^8	6.3×10^8	2.8×10^9
M_1A_2	4.3×10^2	4.1×10^3	5.5×10^4	3.2×10^5	1.2×10^6	5.0×10^6
M_2A_2	8.8×10^1	5.5×10^2	6.2×10^3	3.4×10^4	1.3×10^5	5.3×10^5
M_3A_2	8.3×10^1	6.0×10^2	6.5×10^3	3.6×10^4	1.4×10^5	5.6×10^5
M_4A_2	6.4×10^1	3.4×10^2	2.8×10^3	1.5×10^4	5.4×10^4	2.2×10^5

$\Delta t = 10^{-7}$

N_r	40	80	160	250	350	500
A_2	8.3×10^3	4.5×10^4	2.2×10^5	6.8×10^5	1.7×10^6	5.1×10^6
M_1A_2	2.3×10^2	9.6×10^2	4.0×10^3	1.1×10^4	2.3×10^4	5.5×10^4
M_2A_2	5.9×10^1	2.4×10^2	9.8×10^2	2.4×10^3	4.9×10^3	1.1×10^4
M_3A_2	7.7×10^1	3.2×10^2	1.3×10^3	3.2×10^3	6.3×10^3	1.3×10^4
M_4A_2	2.9×10^1	1.2×10^2	5.0×10^2	1.3×10^3	2.6×10^3	6.0×10^3

Table 2: Condition number dependence on the radial resolution and the type of preconditioner. M_iA_2 mean left preconditioners where $M_i = P_iQ_i$ is the preconditioner. P_i and Q_i are the matrices corresponding to i Jacobi iterations with matrices D and A_1, respectively.

the performance. We obtain speed ups $S_p = 9.1$ for $p = 16$ for the high resolution mesh $N_r = 170$ and $L = 320$, more specifically we obtain $S_p = 1.53p^{0.64}$. Notice that the slope of the p-curves of Fig. 2(a) increases with increasing p.

As commented before, when using a collocation method to radially discretized the equations, all the radial operators of the original equations are substituted by dense matrices. When the evaluation of an operator is required all similar computations are grouped to call efficient implementations of the matrix-matrix product subroutine DGEMM of BLAS. Also the Legendre transforms needed for the evaluation of the nonlinear terms are implemented with this subroutine. In figure 2(b) the sequential run time for advancing one time step obtained with the basic BLAS library divided by the sequential run time obtained with the ATLAS and GOTO optimized libraries is plotted versus the number of equations for the same resolutions as in Fig. 2(a). Important savings can be obtained with the GOTO library for the larger number of equations where the run time can nearly be halved with respect the basic BLAS library.

Finally, a test for the integration of the variational equations is performed in the following. Assume that the evolution equation for $u \in \mathbb{R}^N$, where u is the vector of all the unknowns of the discretized equations, is

$$\partial_t u = L_0^{-1}(Lu + B(u, u)), \tag{7}$$

and let $u(t) = \phi_t(u_0)$ be its solution with initial condition $u(0) = u_0$ at $t = 0$. In the latter equation, L_0 and L are linear and B nonlinear operators of the

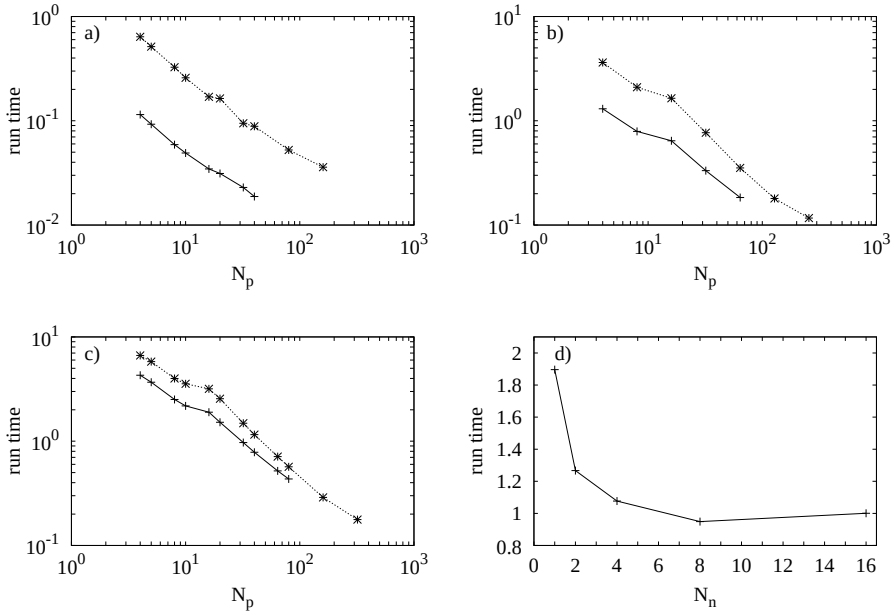

Figure 1: (a) Run time for advancing one time step plotted versus the number of MPI tasks for the test case T_1. (b) and (c) are as (a) but for the test cases T_2 and T_3, respectively. (d) Same as (a) but plotted versus the number of nodes when using 16 MPI task for the test case T_3. The symbols and types of lines indicate: iterative solver ($*$, dotted line) and direct solver ($+$, solid line).

thermal convection equations. The variational equations along $u(t)$ are

$$\partial_t u = L_0^{-1}(Lu + B(u,u)), \tag{8}$$

$$\partial_t v = L_0^{-1}\left(Lv + \frac{1}{2}\left(B(u+v,u+v) - B(u-v,u-v)\right)\right) \tag{9}$$

with $w(t) = (u(t), v(t)) \in \mathbb{R}^{2N}$ the solution with initial condition $w(0) = (u_0, v_0)$. The property $D\phi_t(u_0)v_0 = v(t, v_0)$ allows us to check the numerical integration of Eqs. (8-9):

$$v(t,v_0) = D\phi_t(u_0)v_0 \approx \frac{\phi_t(u_0 + hv_0) - \phi_t(u_0 - hv_0)}{2h} = \tilde{v}(t,v_0) \tag{10}$$

Algorithm

- Initialise $u_0, v_0 \in \mathbb{R}^N$ and $t, h > 0$
- Integrate t time units the variational equations Eqs. (8-9) with initial condition (u_0, v_0)

- Integrate t time units the original system Eq. 7 with initial conditions u_0+hv_0 and u_0-hv_0 to obtain $\phi_t(u_0+hv_0)$ and $\phi_t(u_0-hv_0)$, respectively.
- Compute $\tilde{v}(t,v_0)=\frac{\phi_t(u_0+hv_0)-\phi_t(u_0-hv_0)}{2h}$
- Check the error

$$\varepsilon=\frac{||\tilde{v}(t,v_0)-v(t,v_0)||_2}{||v(t,v_0)||_2}. \tag{11}$$

Notice that $\varepsilon=\varepsilon(t,u_0,v_0,h,tol)$, where $tol=\varepsilon^a=\varepsilon^r$ is the tolerance for the Q-implicit VSVO time integration code fully described in [11] (ε^a is the absolute and ε^r the relative error tolerance).

To check the time integration of the variational equations we will consider a case where the Ekman number is $E=10^{-4}$, the Prandtl number is $\sigma=0.1$ and the radius ratio is $\eta=0.35$. More precisely, a modulated travelling wave with azimuthal wave number $m_d=6$ which is stable at the weak supercritical Rayleigh number $Ra_e=2.59929964\times10^5$ ($Ra_e=\frac{\gamma\alpha\Delta T d^4}{\kappa\nu}$) is considered. This is a quasiperiodic resonant orbit which has two frequencies $f_1=60.21680$ and $f_2=26.75897$. They satisfy the relation $(4f_1-9f_2)/f_2=O(tol)$, where tol is the tolerance of the time integration method used to obtain the initial condition u_0.

The initial conditions of Eqs. (8-9) are $u_0=v_0$ being u_0 the initial condition of the quasiperiodic orbit and the final time of the time integration is $t\approx 1/f_2$. We compute $\tilde{v}(t,v_0)$ for several values of the finite difference tolerance h and we integrate Eqs. (8-9) with several time integration tolerances tol. The results are shown in Fig. 2 (c), where the relative error ε of Eq. (11) is plotted versus the finite difference tolerance h for three different tolerances $tol=10^{-3},10^{-6},10^{-9}$ of the VSVO time integration code. In this figure the error due to the time integration and that due to the truncation can be identified. The latter is exibited for $h>10^{-2}$ where the curve has an slope of 2. The error due to time integration appear for $h<10^{-2}$.

N_r	24	32	38	50	60	72	80	88	94	106	120	130	150	170
L	42	54	70	84	106	128	150	172	194	230	256	280	300	320

Table 3: Radial resolution, N_r, and spherical truncation parameter, L, used in figure 2.

Discussion

Two different approach for solving fluid problems in rotating spherical shells are studied in this paper. In the first approach a finite differences radial discretization is used to allow the parallelization with MPI directives by partitioning the shell in the radial direction into different processors. This is suitable because

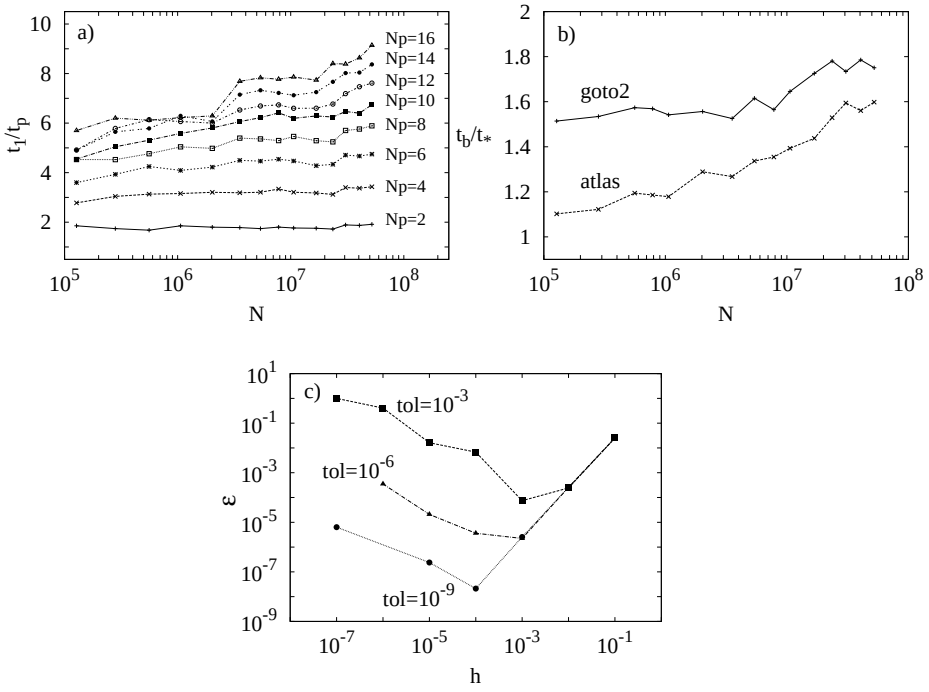

Figure 2: (a) The ratios t_1/t_p, where t_p means the run time obtained with p processors, plotted versus the number of equations. (b) Sequential run time for advancing one time step obtained with the basic BLAS library divided by the sequential run time obtained with the ATLAS and GOTO optimized libraries plotted versus the number of equations. (c) Test for the variational equations: the relative error, ε, plotted versus the centred finite difference approximation tolerance h for three tolerances (labeled on the curves) of the VSVO time integration code.

several types of architectures can be used to run the code. The implementation of the improved version of the BiCGStab Krylov solver could improve the efficiency of the code in certain physical regimes which need very small time steps for their integration in time. With this iterative solver a larger number of processors can be used to minimize the computing time for obtaining time-averaged physical properties of chaotic and turbulent dynamo models.

In the second approach the parallelization is performed by partitioning the triangular mesh of spherical harmonics and by using OpenMP directives. The code can only be executed on shared memory architectures. The implementation of the code is performed in a way to exploit the use of matrix-matrix products with the DGEMM subroutine of the BLAS library. In this case the code is fully spectral, integrates the variational equations, and the time integration schemes

are of high order to obtain high accurate solutions which are needed when using dynamical systems tools for a deep study of the physical system.

In certain architectures, such as that of MesoPSL (a cluster of nodes with several cores each one) it is better not to use all the cores to avoid competition for a memory access. Notice that the approach followed in SPHO can be also performed in PARODY by assigning one MPI task at each node and using the cores of it to parallelize the computations on the spherical harmonics mesh using OpenMP directives. The use, as much as possible, of the DGEMM subroutine can also improve the code.

Possible slightly improvement of the SPHO code with MPI directives will consist on separating independent computations on different nodes. For instance one node could compute the velocity field and other node the vorticity field needed for the evaluation of the nonlinear terms.

Acknowledgments

We would like to thank Dr. L. Petitdemange who kindly supplied us the data for the tests cases and R. Raynaud for useful discussions during the implementation of the iterative solver in the PARODY code. This research has been supported by Spain Ministerio de Ciencia e Innovación, and Generalitat de Catalunya under projects MTM2010-16930 and 2009-SGR-67, respectively. The first author is supported by the Fondation Sciences Mathmatiques de Paris (FSMP) and by a public grant overseen by the French National Research Agency (ANR) as part of the *Investissements d'Avenir* program (reference: ANR-10-LABX-0098). Also this work was granted access to the HPC resources of MesoPSL financed by the Region Ile de France and the project Equip@Meso (reference ANR-10-EQPX-29-01) of the programme Investissements dAvenir supervised by the Agence Nationale pour la Recherche.

References

[1] R. Barrett, M. Berry, T. F. Chan, J. Demmel, J. M. Donato, J. Dongarra, V. Eijkhout, R. Pozo, C. Romine, and H. Van Der Vorst. *Templates for the Solution of Linear Systems: Building Blocks for Iterative Methods*. SIAM, 1994.

[2] C. Canuto, M. Y. Hussaini, A. Quarteroni, and T. A. Zang. *Spectral Methods in Fluid Dynamics*. Springer, 1988.

[3] S. Chandrasekhar. *Hydrodynamic And Hydromagnetic Stability*. Dover, New York, 1981.

[4] U. Christensen and J. Aubert. Scaling properties of convection-driven dynamos in rotating spherical shells and application to planetary magnetic fields. *Geophys. J. Int.*, 166:97–114, 2006.

[5] U.R. Christensen, J. Aubert, P. Cardin, E. Dormy, S. Gibbons, G.A. Glatzmaier, E. Grote, Y. Honkura, C. Jones, M. Kono, M. Matsushima, A. Sakuraba, F. Takahashi, A. Tilgner, J. Wicht, and K. Zhang. A numerical dynamo benchmark. *Phys. Earth Planet. Inter.*, 128:25–34, 2001.

[6] E. Dormy. *Modelisation numerique de la dynamo terrestre*. Ph.D. thesis, Institut de physique du globe de Paris, 1997.

[7] E. Dormy, P. Cardin, and D. Jault. MHD flow in a slightly differentially rotating spherical shell, with conducting inner core, in a dipolar magnetic field. *EPSL*, 160:15–30, 1998.

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

[9] Matteo Frigo and Steven G. Johnson. The design and implementation of FFTW3. *Proceedings of the IEEE*, 93(2):216–231, 2005. special issue on "Program Generation, Optimization, and Platform Adaptation".

[10] F. Garcia, L. Bonaventura, M. Net, and J. Sánchez. Exponential versus imex high-order time integrators for thermal convection in rotating spherical shells. *J. Comput. Phys.*, 264:41–54, 2014.

[11] F. Garcia, M. Net, B. García-Archilla, and J. Sánchez. A comparison of high-order time integrators for thermal convection in rotating spherical shells. *J. Comput. Phys.*, 229:7997–8010, 2010.

[12] F. Garcia, M. Net, and J. Sánchez. A comparison of high-order time integrators for highly supercritical thermal convection in rotating spherical shells. In J.S. Hesthaven M. Azaïez, H. El Fekih, editor, *Proc. of International Conference on Spectral and High Order Methods for Partial Differential Equations-ICOSAHOM 2013, Lecture Notes in Computational Science and Engineering*, volume 95, 2014.

[13] F. Garcia, J. Sánchez, and M. Net. Antisymmetric polar modes of thermal convection in rotating spherical fluid shells at high Taylor numbers. *Phys. Rev. Lett.*, 101:194501–(1–4), 2008.

[14] Kazushige Goto and Robert A. van de Geijn. Anatomy of high-performance matrix multiplication. *ACM Trans. Math. Softw.*, 34(3):1–25, 2008.

[15] M. Hochbruck, C. Lubich, and H. Selhofer. Exponential integrators for large systems of differential equations. *SIAM J. Sci. Comput.*, 19(5):1552–1574, 1998.

[16] A. Kageyama and T. Sato. The complexity simulation group, computer simulation of a magnetohydrodynamic dynamo. *Phys. Plasma*, 2:1421–1431, 1995.

[17] S. Lakshmivarahan and K. D. Sudarshan. *Analysis and design of parallel algorithms: Arithmetic and matrix problems*. Series in supercomputing and parallel processing. McGraw-Hill, 1990.

[18] M. Net, F. Garcia, and J. Sánchez. On the onset of low-prandtl-number convection in rotating spherical shells: non-slip boundary conditions. *J. Fluid Mech.*, 601:317–337, 2008.

[19] S. A. Orszag. Transform method for calculation of vector-coupled sums: Application to the spectral form of the vorticity equation. *J. Atmos. Sci.*, 27:890–895, 1970.

[20] L. Oruba and E. Dormy. Predictive scaling laws for spherical rotating dynamos. *Geophys. J. Int.*, Accepted, 2014.

[21] D. Pino, I. Mercader, and M. Net. Thermal and inertial modes of convection in a rapidly rotating annulus. *Phys. Rev. E*, 61(2):1507–1517, 2000.

[22] E. Plaut and F. H. Busse. Multicellular convection in rotating annuli. *J. Fluid Mech.*, 528:119–133, 2005.

[23] R. Raynaud and E. Dormy. Intermittency in spherical couette dynamos. *Phys. Rev. E*, 87:1–6, 2013.

[24] Y. Saad. *Iterative methods for sparse linear systems*. PWS pub. company, New York, 1996.

[25] J. Sánchez, M. Net, B. García-Archilla, and C. Simó. Newton-Krylov continuation of periodic orbits for Navier-Stokes flows. *J. Comput. Phys.*, 201(1):13–33, 2004.

[26] J. Sánchez, M. Net, and C. Simó. Computation of invariant tori by Newton-Krylov methods in large-scale dissipative systems. *Physica D*, 239:123–133, 2010.

[27] M. Schrinner, L Petitdemange, and E. Dormy. Dipole collapse and dynamo waves in global direct numerical simulations. *AJ*, 752:1–12, 2012.

[28] M Schrinner, L. Petitdemange, R. Raynaud, and E. Dormy. Topology and field strength in spherical anelastic dynamo simulations. *Astronomy & Astrophysics*, 564:1–13, 2014.

[29] R. C. Whaley, A. Petitet, and J. Dongarra. Automated empirical optimization of software and the ATLAS project. Technical Report UT-CS-00-448, Netlib, Lapack working notes, September 2000.

[30] L.T. Yang and R. Brent. The improved bicgstab method for large and sparse unsymmetric linear systems on parallel distributed memory architectures. *Proceedings of the Fifth International Conference on Algorithms and Architectures for Parallel Processing, IEEE.*, 2002.