A Highly Efficient Dynamical Core of Atmospheric General Circulation Model based on Leap-Format

Hang Cao[1,2], Liang Yuan[1], He Zhang[3,✉], Baodong Wu[1,6], Shigang Li[5], Pengqi Lu[1,2], Yunquan Zhang[1], Yongjun Xu[4], and Minghua Zhang[3]

[1]State Key Laboratory of Computer Architecture, Institute of Computing Technology, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
[3]Institute of Atmospheric Physics, Chinese Academy of Sciences
[4]Institute of Computing Technology, Chinese Academy of Sciences
[5]Department of Computer Science, ETH Zurich
[6]Sensetime Research
[✉]Corresponding author: zhanghe@mail.iap.ac.cn

***Abstract*—The finite-difference dynamical core based on the equal-interval latitude-longitude mesh has been widely used for numerical simulations of the Atmospheric General Circulation Model (AGCM). Previous work utilizes different filtering schemes to alleviate the instability problem incurred by the unequal physical spacing at different latitudes, but they all incur high communication and computation overhead and become a scaling bottleneck. This paper proposes a new leap-format finite-difference computing scheme. It generalizes the usual finite-difference format with adaptive wider intervals and is able to maintain the computational stability in the grid updating. Therefore, the costly filtering scheme is eliminated. The new scheme is parallelized with a shifting communication method and implemented with fine communication optimizations based on a 3D decomposition. With the proposed leap-format computation scheme, the communication overhead of the AGCM is significantly reduced and good load balance is exhibited. The simulation results verify the correctness of the new leap-format scheme. The new scheme achieves the speed of 16.6 simulation-year-per-day (SYPD) and up to 3.3x speedup over the latest implementation.**

***Index Terms*—dynamical core, leap-format finite-difference, shifting communication scheme, polar regions, filtering module**

I. Introduction

The Atmospheric General Circulation Model (AGCM) has always been an important research tool in the study of climate change. With the urgent need for climate modeling, the numerical simulation of AGCM is facing a great challenge with respect to scalability and simulation performance. The atmospheric general circulation model mainly consists of two modules: the dynamical core and the physical process [1]. And as one of the most time-consuming modules, the dynamical core refers to the formulation of the atmospheric hydrodynamic equations along with the numerical algorithms to solve them.

In recent decades, many related atmospheric models are developed, including CAM5 [2] from the National Center for Atmospheric Research (NCAR) [3], ECHAM-5 [4] from the Max Planck Institute for Meteorology, HadCM from the UK Met office Hadley Center, and IAP-AGCM [5] from the Institute of Atmospheric Physics, Chinese Academy of Sciences. The dynamical core of the above AGCMs are basically solved by two types of mesh, namely the quasi-uniform polygonal mesh and the equal-interval latitude-longitude mesh. CAM-SE of NCAR adopts the first type of mesh, while CAM-FV and IAP-AGCM use the second one. CAM-SE, with the spectral element dynamical core implementation, is known for the good scalability and parallel efficiency. CAM-FV (finite volume implementation) and IAP-AGCM, a finite-difference dynamical core, are both based on the equal-interval latitude-longitude mesh. Comparing to the quasi-uniform polygonal mesh based dynamical cores, latitude-longitude mesh based models have advantages in aspects of preserving energy conservation, dealing with complex terrains and moistures, and coupling with other climate system components.

The atmospheric component of the Chinese Academy of Sciences' Earth System Model (CAS-ESM), as known as the IAP-AGCM4, adopts a finite-difference dynamical core with a terrain-following σ coordinate vertically, and a latitude–longitude grid with C grid staggering in the horizontal discretization [1]. Despite the aforesaid merits IAP-AGCM's dynamical core has, it's still difficult to improve the parallel scalability and maintain the computation stability in the meantime. Wu et al. [6] have developed a scalable finite-difference dynamical core based on the latitude-longitude mesh using a 3D decomposition method. This method released parallelism in all three dimensions and chose an alternate filtering scheme to overcome the shortcomings of IAP-AGCM4. However, the overheads of filtering and MPI communication remain quite high. Also, small time steps must be used to alleviate computational instability.

The computation problem at the polar regions, as known as the pole problem [7], is usually solved by longitudinal filtering. Due to the rapid decrease of zonal mesh interval in high latitudes, the filtering is required to damp the high frequency effects of the shortwave and then maintain the computing stability. In the original dynamical core of IAP-AGCM4, a Fast Fourier Transformation (FFT) filtering module is adopted in the 2D decomposition dynamical core, and a 13-point Gaussian filtering scheme is adopted in the 3D decomposition model, both of which may scale poorly as the model resolution increases. In this paper, we present a new optimized finite-difference computing method to replace the costly filtering module of the dynamical core in AGCM, which highly reduces the filtering runtime and maintains the computational stability. The following are the major contributions of our work:

- We propose a new leap-format finite-difference computation scheme. It is able to maintain the computational stability in the grid updating and eliminates additional filtering requirements at the high latitudes and polar regions. Thus the overall communication overhead is significantly reduced and the load balance of the model is improved.
- We design a novel shifting communication window concept for parallelizing the new format. It is further optimized with the communication aggregation. Our new implementation achieves 6.4x speedup for the filtering module and 3.3x speedup for the whole dynamical core on average over the original implementation.

This paper is organized as follows. The background is described in the next section. Section III introduces the leap-format computation scheme and the design of the parallelled leap-format communication. Experimental results and performance evaluations are presented in Section IV. The final Section V contains the conclusion.

II. Background

A. Model Description

Our work targets on the dynamical core of IAP-AGCM4, the fourth generation of global atmospheric general circulation model developed by the Institute of Atmospheric Physics, CAS [5]. It has been used to simulate the air temperature, summer precipitation, and circulations related to monsoons in the long-run atmospheric circulations and climate change [8]–[11].

The IAP-AGCM4 adopts a finite-difference dynamical core using a latitude–longitude grid with C grid staggering in the horizontal discretization. With the subtraction of standard stratification, IAP transform and the terrain-following vertical coordinate, the model equations based on the baroclinic primitive equations can be written as follows:

$$\begin{cases} \frac{\partial U}{\partial t} = -\sum_{m=1}^{3} \alpha^* L_m(U) - \beta^* P_\lambda - \gamma^* f^* V \\ \frac{\partial V}{\partial t} = -\sum_{m=1}^{3} \alpha^* L_m(V) - \beta^* P_\theta + \gamma^* f^* U \\ \frac{\partial \Phi}{\partial t} = -\sum_{m=1}^{3} \alpha^* L_m(\Phi) + (1-\delta_p) \\ \qquad \cdot \left[b(1+\delta_c) + \delta \cdot \kappa \Phi / P\right] \cdot \beta^* \widetilde{\Omega} \\ \frac{\partial}{\partial t}\left(p'_{sa}/p_0\right) = -\beta^* \widetilde{P}(W) + \kappa^* D_{sa}/P_0 \end{cases} \tag{1}$$

where the $U, V, \varPhi, p'_{sa}$ and ϕ', W are the forcast variables and prognostic variables, respectively. The partial derivatives represent the calculations of the variables' tendencies.

The large-scale motion in the dynamical core of the atmosphere is conventionally divided into the advection process and the adaption process. For the purpose of simplicity and energy conservation, the Governing Equations (1) can be written as follows:

$$\frac{\partial F}{\partial t} = -L_F + A_F,\ where\ F = \left(U, V, \varPhi, p'_{sa}\right) \tag{2}$$

In Equation (2), L is an operator representing the advection term, and A indicates the adaption term. In the time integration scheme, the two processes have different time scales. The advection process is 10x faster than the adaption. Therefore, the two processes are implemented seperately and dominate the overall execution cost.

B. Filtering and Parallelization

The finite-difference method on the latitude–longitude grid leads to unequal longitudinal distances. As the meridians tend to converge to the north and south poles, the physical distance of the equal-interval mesh will reduce rapidly [12], [13]. According to the Courant-Friedrichs-Lewy (C.F.L) condition [14], which is a necessary condition for the computational stability of the partial differential equations, the atmospheric model needs to satisfy

$$\varDelta t \leqslant \varDelta x / U$$

where U is the maximum characteristic velocity, Δt and Δx are the time step and the space interval, respectively. As a consequence of the inconsistent mesh interval, the time step of simulation should be small enough, otherwise the computational instability is inevitable [15]. To allow a larger time step and reduce the computation cost, a filtering module is used to preserve the computational stability. In previous IAP AGCM implementations, an FFT filtering is used on the tendencies of $U, V, \varPhi,$ and p'_{sa} to dump out the short-wave modes poleward of $\pm 70°$. It is well known that the parallelization of FFT requires all-to-all communications and the parallel efficiency improvement can be very challenging. Therefore former dynamical core designs including IAP-AGCM4 choose to leave the X dimension executed sequentially [16]. However, as the computing resources of supercomputers grow rapidly, the traditional 2D decomposition method is no longer effective enough to utilize the rich computing resources efficiently. This is mainly because only the parallelism of the Y and Z dimensions is exploited [17], while the X dimension, which contains the most number of mesh points among the three dimensions, is serialized. Thus, the total degree of parallelism of the 2D decomposition is not enough, which hinders the parallel scalability. For example, the state-of-the-art finite-volume dynamical core based on the latitude-longitude mesh can only scale up to 1664 MPI processes (1664 MPI processes × 4 OpenMP threads = 6656 cores) at the resolution of 0.5° × 0.5° [14]. For IAP AGCM-4, the dynamical core can only

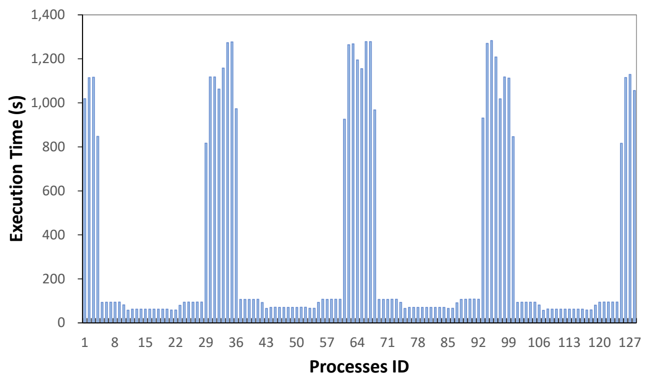


Fig. 1. Load imbalance is visible in the varying filtering runtime in 128 different processes.

scale up to 1024 MPI processes at the resolution of 0.5° × 0.5° [12], with 64 processes along the Y dimension and 16 processes along the Z dimension. Wu et al. [6] propose a novel 3D decomposition method. With all the advantages 3D decomposition method has, the computation instability and filtering parallelism in the high-latitude and polar regions remain pivotal problems. They further propose a new adaptive Gaussian filtering scheme implemented in 3D decomposition method has alleviate the difficulties of parallelization along the X direction.

C. Discussion

In the numerical simulation of IAP-AGCM, a Fast Fourier Transformation (FFT) filtering scheme is adopted in the 2D decomposition, and a 13-point Gaussian filtering scheme is employed in the 3D decomposition. For both AGCM2D and AGCM3D, the runtime ratio of the filtering module in the whole dynamical core is considerable. Table II shows the runtime percentages of various filtering schemes for the processes that compute the points at poles. For the 2D decomposition model, the filtering overhead ratio decreases as the parallelism increases. The reason is that the filtering only occurs along the X dimension, which is not parallelized.

Another disadvantage of the Gaussian filter is that high latitudes require more neighbor points or multiple calls to enhance the computational stability. For example, the IAP AGCM3D needs 241 neighbor points for filtering at poles. Although the Gaussian filtering incurs an easier parallelized neighbor communication pattern than the all-to-all communication pattern caused by the FFT filtering, it still needs a large amount of communication volume and has a tremendous influence on the performance. The last row in Table I shows that the communication of the Gaussian filter still dominates the overall execution time.

TABLE I
THE RUNTIME RATIO OF FILTERING MODULES

num of procs	128	256	512	1024	2048	4096	8192
FFT (2D)	28.8%	17.3%	6.9%	8.3%	N/A	N/A	N/A
Gaussian (2D)	58.1%	40.4%	21.3%	27.2%	N/A	N/A	N/A
Gaussian (3D)	69.6%	81.9%	90.7%	94.6%	96.1%	95.3%	91.1%

Finally, accompany with the further decrease of the zonal grid size in the high resolution model, the FFT filtering and Gaussian filtering will be more costly in the iteration of model simulation, which leads to serious load imbalance. Fig. 1 exhibits the filtering costs of 128 processes with a 32×4 2D decomposition along the Y and Z dimensions. The processes at the high-latitudes incur much more computation costs. Therefore, the dynamical core can be more difficult to be parallelized and scale up to larger scale computing systems due to the load imbalance of filtering.

III. Leap-Format Difference Computation

In this section, we will introduce our new approach to the high-latitude and polar problems. We first discuss the motivation and then propose the new leap-format finite-difference computing method. Finally, we present the parallelization and communication optimization utilized in the 3D decomposition.

A. Motivation

Our key observation is that the conventional filtering methods for the pole problem often distinguish different latitude zones. The reason is that the short wave impact is more serious at high-latitudes as mentioned above. Therefore, stronger filters like the FFT filtering [18] can only target variables at high latitudes and simple filters are adequate for low and mid latitudes. For example, in IAP-AGCM4, the computation grids are divided into three latitudinal bands. For the low latitude regions ($|\varphi| < 38°$), a simplified filter is used to get rid of the waves of double mesh spacing. And a 3-point recursive operator [19] is applied at the midlatitudes ($38° \leqslant |\varphi| \leqslant 70°$). For the high latitude regions ($|\varphi| > 70°$), the zonal FFT or Gaussian filters are added to stabilize the tendency computations of U, V, Φ, and $p_{sa}^{'}$, etc.

Another observation is that the polar zones offer a complementary property that the finite-difference format design is flexible. In particular, it permits a finite-difference with larger spacing along the X dimension. Our approach seeks to improve the finite-difference calculation by incorporating the filtering function directly and getting rid of additional filtering demands at high latitudes. From the performance perspective, the filter for the high latitudes leads to load imbalance and damage the execution speed. Since simple filters for low latitudes cause far less overhead, this new approach is expected to boost the performance significantly.

B. Leap-Format Design

The dynamical core mainly comprises two parts: the advection process and the adaption process. For both processes, the model uses a latitude-longitude grid with Arakawa's C grid staggering in the horizontal discretization. The calculations of variables are performed in three dimensions, i.e. the longitude,

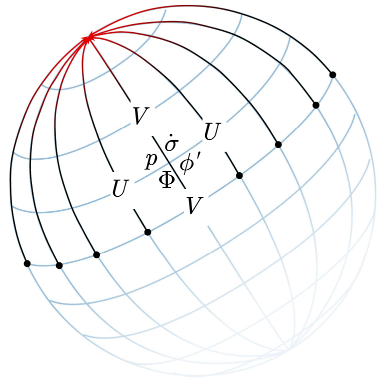


Fig. 2. Distribution of variables in C grid.

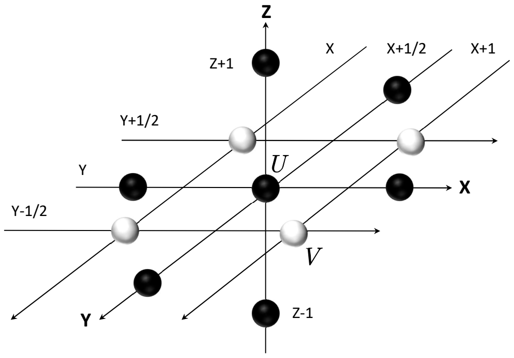


Fig. 3. Stencil computation for 3D variables.

latitude and level dimensions which are denoted as X, Y and Z dimensions, respectively. In the Z dimensions, the vertical distribution of every forecast variables or prognostic variables is set on the integer layer or the semi-integer layer. The Arakawa's C grid staggers in the horizontal (X and Y) discretization. The forecast variable zonal wind U is located at $(x+\frac{1}{2}, y, z)$, i.e. the semi-integer index layer along the X dimension and the integer index layer along the Y dimension. The meridional wind V is located at $(x, y+\frac{1}{2}, z)$ while other forecast variables are set at (x, y, z). Fig. 2 illustrates the forecast and prognostic variables' distribution.

The calculations of these variables are 3D star stencil computations [20]. Take one difference term regarding to the forecast variable zonal wind U as an example, as shown in Fig. 3. A two-dimensional central difference form of U is as follows:

$$\left(\frac{\partial U}{a\sin\theta\partial\lambda}\right)_{x,y,z} = \frac{U_{x+\frac{1}{2},y,z} - U_{x-\frac{1}{2},y,z}}{a\sin\theta_y \Delta\lambda} \tag{3}$$

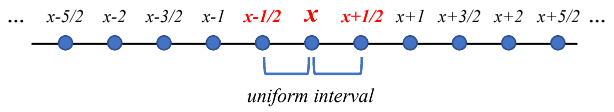
where θ denotes the colatitude ($90^\circ - latitude$) of the grid point, $\Delta\lambda$ is the longitudinal grid spacing, a is the radius of the earth and the subscript x, y and z denote the index of longitudinal and latitudinal direction, respectively.

To avoid the drawbacks induced by the filtering and universal finite-difference format, we propose a new leap-format finite-difference computing method. The fundamental technique is to increase the grid-size at high latitudes. Take Equation (3) as an example again, the spacing interval used in the central difference is extended to a wider size for an exact high latitude. The subscripts can be generalized to $U_{x+N_{leap}/2,y,z}$ and $U_{x-N_{leap}/2,y,z}$, where N_{leap} denotes the extended new central difference interval of U in the longitudinal direction. Accordingly, the grid-size changes from Δx to $\Delta x * N_{leap}$. The new leap-format central difference form of U is written as follows:

$$\left(\frac{\partial U}{a\sin\theta\partial\lambda}\right)_{x,y,z} = \frac{U_{x+N_{leap}/2,y,z} - U_{x-N_{leap}/2,y,z}}{a\sin\theta_y \Delta\lambda * N_{leap}} \tag{4}$$

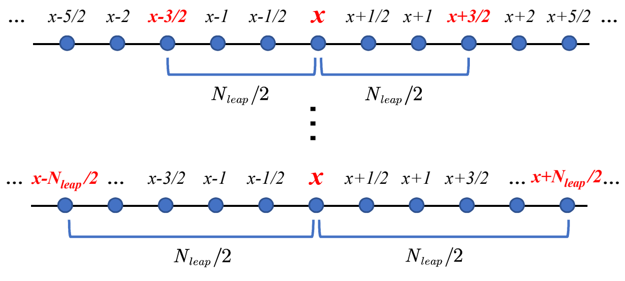
Equation (4) degrades to the central difference when N_{leap} equals 1. Since the filter is only required along the zonal circle, the difference terms and grid spacing in other dimensions remain unchanged. Based on the difference latitudes of various grid points, the value N_{leap} can be chosen as difference integer values. Fig. 4(a) shows the original difference scheme with a uniform interval. Fig. 4(b) illustrates the possible leap intervals of the new central difference scheme. Remember that variables U locate at semi-integer points in the X dimension, thus N_{leap} must be odd integers.

One critical problem is the zonal grid size ($\Delta x = a\sin\theta\Delta\lambda$) shrinks quickly with the decrease of the colatitude θ. Table II lists the zonal grid sizes Δx of u-grids and v-grids on the equator and at poles with various resolutions. The interval of V is approximately half of that of U at poles since V is located at the semi-integer layer, as shown in Figure 2 and $\sin\theta \approx \theta$ for small θ values. Take the horizontal resolution of $1.4^\circ \times 1.4^\circ$ for example, the physical distance at the equator is approximately 155.7 km, while the grid size at the poles is 3.8 km. Furthermore, the difference at the polar regions and low latitude regions will be even bigger as the horizontal resolution of the model increases. For example, the ratio of the interval at poles to that on the equator is $222.4/3.9 \approx 58$



... x-5/2 x-2 x-3/2 x-1 x-1/2 x x+1/2 x+1 x+3/2 x+2 x+5/2 ...
uniform interval

(a) Original central difference scheme.



... x-5/2 x-2 x-3/2 x-1 x-1/2 x x+1/2 x+1 x+3/2 x+2 x+5/2 ...
$N_{leap}/2$
$N_{leap}/2$
... x-$N_{leap}/2$ ... x-3/2 x-1 x-1/2 x x+1/2 x+1 x+3/2 ... x+$N_{leap}/2$...
$N_{leap}/2$
$N_{leap}/2$

(b) New central difference scheme with leap-format.

Fig. 4. Transformation of central difference scheme from fixed interval to leap-format.

with the resolution 2° while it increases to $55.6/0.25 \approx 222$ with the resolution 0.5°.

To improve the adaptivity of the new format, the leap interval N_{leap} is automatically adjusted with the latitude. We choose the interval size at mid-latitude 45° as a standard and every interval in higher latitude is adjusted to an equivalent physical size with it. Specifically, in the spherical coordinate system the zonal distance of the mesh interval can be calculated by $2*a*\arcsin(\cos\alpha \times \sin res)$, where α is the current latitude, res is the difference of longitudes (resolution in X dimension), and a is the radius of earth. Therefore, N_{leap} is defined by the ratio of referenced threshold (45°) and the grid size of current latitude (colatitude) θ_y.

$$N_{leap} = \lceil \frac{\arcsin(\cos 45^\circ \times \sin res)}{\arcsin(\cos(90^\circ - \theta_y) \times \sin res)} \rceil \tag{5}$$

Fig. 5 shows the N_{leap} values for resolution of 0.5°. As the colatitude approaches 0, the number of leap points can reach as high as 41 or 82. Note that other difference terms may contain various formats and similar physical interval adjusting scheme is required in those cases.

Theoretically, the equivalent physical interval to the lower latitudes for the high latitude regions permits an increased time step for the model's simulation. The effect is similar to the filtering modules. In other words, no additional filters are needed at high-latitudes ($|\varphi| > 70^\circ$). And the far less costly simple filter for the low latitudes ($|\varphi| < 38^\circ$) and the 3-point recursive operator for the mid latitudes($38^\circ \leqslant |\varphi| \leqslant 70^\circ$) remain the same. Therefore, the leap-format difference scheme implementation can bring down the overall runtime of the whole dynamical core and improve the load balance.

C. Parallelization

To parallelize the leap-format computation and incorporate it in the 3D decomposition model, we need to consider the concrete values of the leap grid points, namely $N_{leap}(j)$ for each latitude, as shown in Fig. 5. The 3D decomposition brings in an extra communication domain along the X dimension, i.e. the latitudinal circle direction. With the widely varied number of leap points, it is obvious that the neighbor communication along the X dimension fails to fullfill the demands of leap-format difference computation in high latitudes.

For the variables to adopt leap-format difference computations, multiple point to point communications are required to transfer the required leap grid points from the current process to the relevant process. Let N_x, N_y and N_z be the number of mesh points along the three dimensions. The numbers of processes assigned in the three communication domains are denoted as P_x, P_y and P_z. For the difference computations in zonal direction, which corresponds to the X dimension of the communication domain, the related variables are split into local arrays based on the value of P_x. Each process i ($1 \leqslant i \leqslant P_x$) holds the data on a block of longitudes, whose length is refereed to as NB_i. Note that if N_x is not a multiple of P_x, NB_i may be equal to $\lceil N_x/P_x \rceil$ or $\lfloor N_x/P_x \rfloor$. Fig. 6 also plots the start index IB_i and end index IE_i of each process and it is obvious that $NB_i = IE_i - IB_i + 1$.

We propose a shifting leap-format communication algorithm to apply the 3D parallelization of the designed leap-format finite-difference computation. The basic idea is to determine the position and length of the required data, which is refereed to as the communication window. According to the definition, the start of a communication window is easily located by $IE_i + N_{leap}/2$. But both N_{leap} and NB_i affect the length of the window $W(N_{leap}, NB_i)$. We further explore two cases according to whether one process depends only on its neighbor process or not. We only study the communication direction of

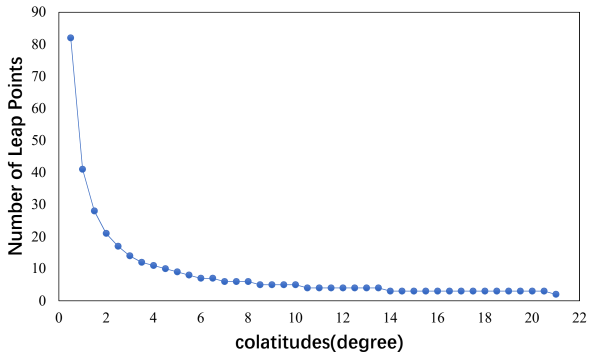


Fig. 5. Numbers of leap points in different colatitudes.

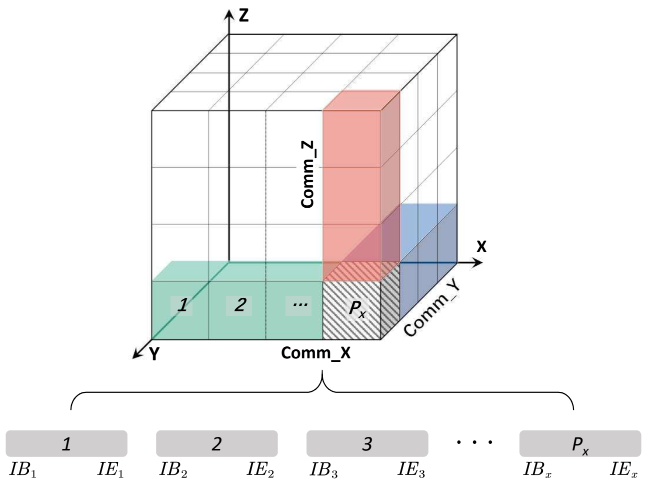


Fig. 6. Local arrays in the X dimension.

TABLE II
THE ZONAL GRID SIZE IN DIFFERENT RESOLUTIONS

Horizontal Resolution	$\Delta x_{\text{equator}}$ (km)	Δx_{poles} (km) of p, u	Δx_{poles} (km) of v
$2^\circ \times 2^\circ$	222.4	7.8	3.9
$1.4^\circ \times 1.4^\circ$	155.7	3.8	1.9
$1^\circ \times 1^\circ$	111.2	1.9	1.0
$0.5^\circ \times 0.5^\circ$	55.6	0.5	0.25

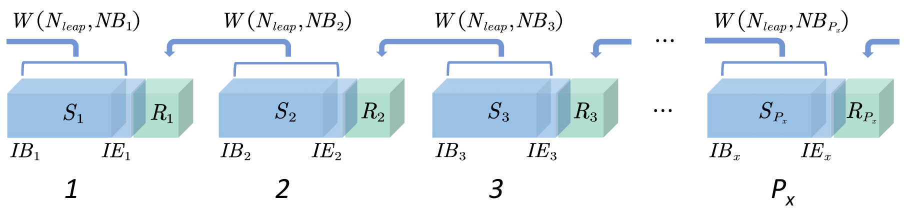


Fig. 7. Shifting leap-format communication for neighbor processes.

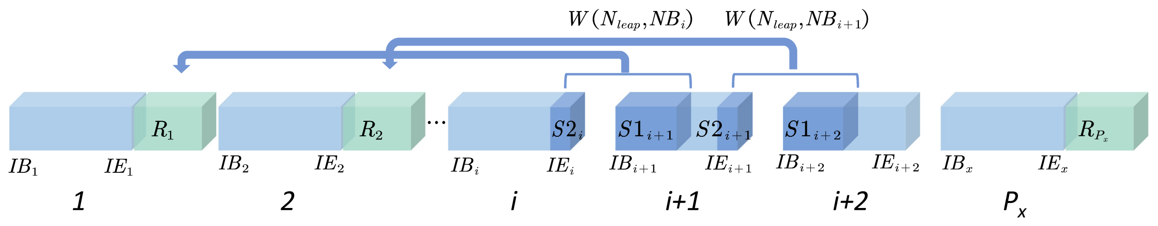


Fig. 8. Shifting leap-format communication for crossed processes.

TABLE III
LEAP-FORMAT COMMUNICATION SCHEME

Number of leap points	Communication volume along X	Participant processes
$1 < N_{leap} \leqslant NB_{i+1}$	N_{leap}	Neighbors
$N_{leap} > NB_{i+1}$	NB_i	Remote & Crossed

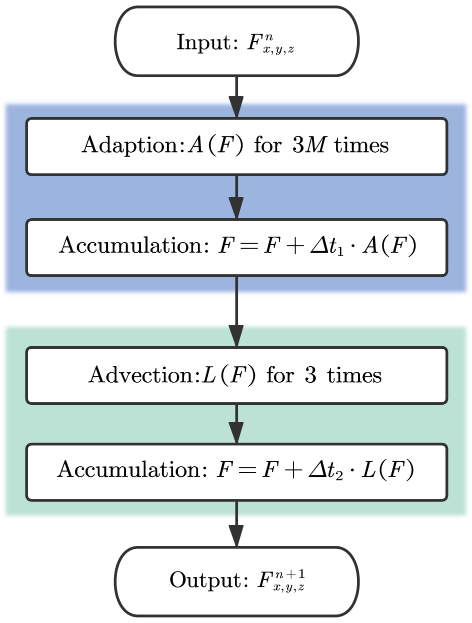
receiving data from the neighbor to the right and the opposite direction is similar. As shown in Table III, if $1 < N_{leap} \leqslant NB_{i+1}$, process i only demands the data from processor $i+1$, otherwise it incurs communication with remote processes.

Fig. 7 illustrates the neighbor communication case, where $S_1, S_2, ..., S_{P_x}$ denote the send buffers of processes, and $R_1, R_2, ..., R_{P_x}$ the receive buffers. There are one send operation and one receive operation for each process in this case.

Fig. 8 illustrates the other case where the position of the communication window demands data from at least one remote process. The size of receive buffers are constant in this case. The shifting communication windows $W(N_{leap}, NB_i)$ and $W(N_{leap}, NB_{i+1})$ are now stretched across two neighbor processes. For a process i, the send buffer is partitioned to $S1_i$ and $S2_i$. So there are two send and receive operations for each process in the group. However, there might exist the situation that the window is enclosed in a single process.

D. Communication Optimizations

The dynamical core mainly consists of two processes: the advection process (L_F) and the adaption process (A_F). In our 3D decomposition implementation, all the filtering overhead is in these two processes, so is the communication overhead of shifting leap-format communication. Specifically in the model simulation, the iteration procedure is shown in Fig. 9. In each iteration of the dynamical core, the adaption process is called



Input: $F^n_{x,y,z}$
Adaption: $A(F)$ for $3M$ times
Accumulation: $F = F + \Delta t_1 \cdot A(F)$
Advection: $L(F)$ for 3 times
Accumulation: $F = F + \Delta t_2 \cdot L(F)$
Output: $F^{n+1}_{x,y,z}$

Fig. 9. Iteration procedure in dynamical core.

for $3{*}M$ times where M identifies the speed different between the two processes, and the advection process is called for 3 times. After the calling of A_F and L_F, an accumulation process is adopt to add the tendencies to the corresponding variables. Hereby, the shifting leap-format communication for the iterations of the model can be partitioned into two parts, namely the advection part and the adaption part.

Based on the observation before, we distinguish the leap-format patterns for every variable involved in the shifting communication. As listed in Table IV is the communication consolidation scheme of variables participated in the leap-format difference computation. Due to the independence of the adaption and advection process, variables are split into two parts

TABLE IV
COMMUNICATION CONSOLIDATION OF LEAP-FORMAT DIFFERENCE VARIABLES

Process	Original difference terms	Leap-format difference gitterms	Variables
Adaption	$(x+1,\ x)$	$(x+N_{leap}, x-N_{leap}+1)$	PXW, UT
	$(x,\ x-1)$	$(x+N_{leap}-1, x-N_{leap})$	$PT, Pstar1$ $Pstar2, TT$ $deltap, GHI$
	$(x+1,\ x-1)$	$(x+2*N_{leap}-1, x-2*N_{leap}+1)$	$Pstar2$
Advection	$(x+1,\ x)$	$(x+N_{leap}, x-N_{leap}+1)$	$Ustar$
	$(x,\ x-1)$	$(x+N_{leap}-1, x-N_{leap})$	$Ustar$
	$(x+1,\ x-1)$	$(x+2*N_{leap}-1, x-2*N_{leap}+1)$	UT, VT TT

to take into consideration. And the variables with the same leap form, such as $PT, Pstar1, Pstar2, TT, deltap, GHI$ in adaption process, or UT, VT, TT in advection process, are aggregated into one send buffer to perform the shifting leap communication. In that way, the message passing for the shifting communication in the 3D dynamical core can achieve a better bandwidth usage for the MPI. Nevertheless, the aggregation of the same patterns are not unconditional, especially when the large amount of communication volume and the application of computation/communication overlap are taken into account.

IV. PERFORMANCE EVALUATION

In this section, we present the correctness verification and simulation performance of the IAP-AGCM4 dynamical core with our new leap-format scheme.

A. Experiment Setup

The platform of our simulation experiments is the Supercomputer Tianhe-2, one of the world's fastest supercomputers in recent years. Each computational node of Tianhe-2 is equipped with two Intel Xeon E5-2692 processors (total 24 cores) and 64 GB memory connected by the TH Express-2 interconnected network. The communication library is a customized MPICH-3.2.1, and the backend compiler is Intel 15.0 compiler.

For the correctness and performance evaluation of the new dynamical core, a series of idealized dry-model experiments proposed by Held and Suarez [21] are conducted. Based on the existing resolution options of the IAP-AGCM model, we set the horizontal resolution as the highest $0.5^\circ \times 0.5^\circ$, with the vertical layer $30L$. The number of mesh points involved in the simulation is $N_x \times N_y \times N_z = 720 \times 361 \times 30$ (7,797,600) in total. As listed in Table V, the group of processes is distributed in three dimensions and scale to the highest number

TABLE V
PROCESSES CONFIGURATIONS FOR ORIGINAL AND LEAP-FORMAT 3D DYNAMICAL CORE

Number of processes	Original/Leap-format $(P_x \times P_y \times P_z)$
128	$32 \times 4 \times 1$
256	$32 \times 8 \times 1$
512	$32 \times 16 \times 1$
1024	$32 \times 32 \times 1$
2048	$32 \times 64 \times 1$
4096	$32 \times 64 \times 2$
8192	$32 \times 64 \times 4$
16384	$32 \times 64 \times 8$
32768	$32 \times 64 \times 16$

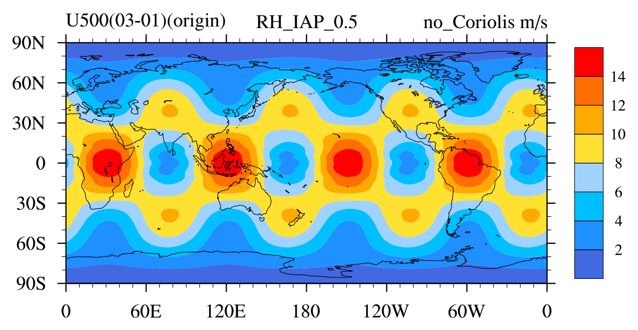
accordingly. The maximum processes used in our experiments for both the original and leap-format difference dynamical core is 32,768. To verify the feasibility of our new leap-format, we always set $P_x = 32$.

B. Correctness Verification of Simulation

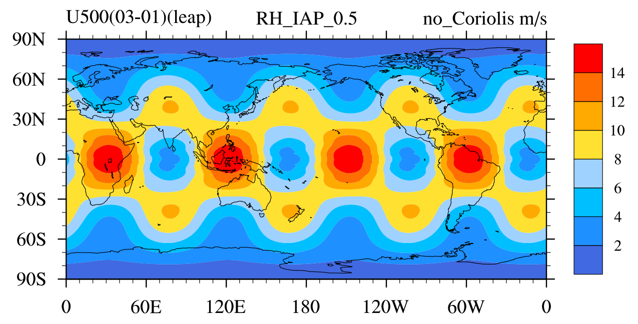
To examine the correctness of the simulation results of leap-format difference computation, we adopt the R-H test [22] for the dynamical core. R-H (Rossby-Haurwitz) wave is a closed-form expression of the spherical barotropic vorticity equation [23], the test of which is a commonly used method for IAP-AGCM. We conduct the R-H tests for both the original dynamical core and the leap-format dynamical core here. The waveform of zonal wind U(m/s) is shown in Fig. 10.

As presented in Fig. 10(a) and Fig. 10(b), in the 2-month simulations, the four R-H waveforms of leap-format difference implementation are not broken and maintained well. Compared with the original simulation results, the distribution of zonal wind is approximately identical. Also, the difference of R-H waves between the original dynamical core and the leap-format dynamical core is described in Fig. 10(c). As can be seen, the difference is very small and less than 0.1 m/s.

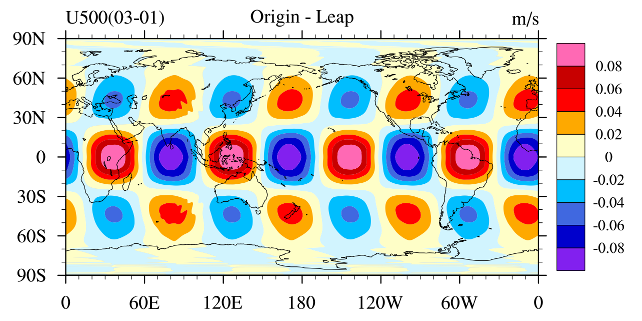
In addition, we also investigate the energy conservation for different filtering schemes in the R-H tests (Fig. 11). The red line shows the evolution of total global mean energy attenuation with FFT filtering, while the black line and blue line indicate the ones with Gaussian filtering and leap-format scheme, respectively. A good dynamical core should conserve the total energy as long as possible in R-H tests. As can be seen, both FFT filtering, leap-format scheme, and Gaussian filtering can approximately conserve the total energy with very little attenuation for 90 days. However, the energy attenuation with Gaussian filtering is about 0.5% larger than that with FFT filtering and leap-format scheme during day 90 to day 180, which indicates the accuracy of leap-format scheme is slightly better than Gaussian filtering.



(a) R-H waveform for original dynamical core.



(b) R-H waveform for leap-format dynamical core.



(c) Difference of R-H waves between original and leap-format dynamical core.

Fig. 10. R-H 4 waves test for zonal wind U. The distributions of R-H wave are derived from the output data of 2 simulated months.The test aims to examine the impact of spherical baroclinic dynamical core without moist physics.

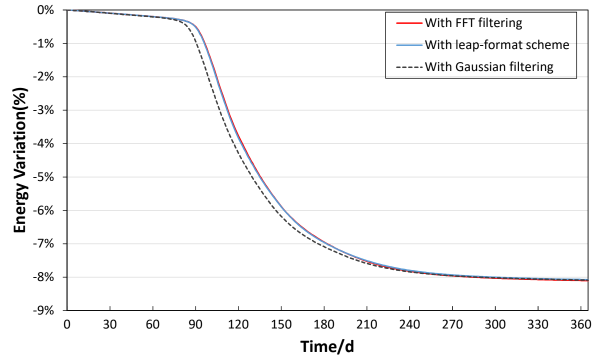


Fig. 11. The total energy conservation of three different filtering scheme within 1 model year.The total available energy consists of the kinetic energy, the available potential energy, and the available surface potential energy.

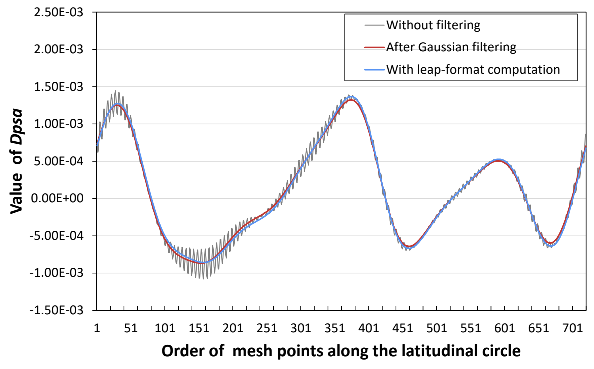


Fig. 12. The comparison of filtering effectiveness for the valuable $DPsa$ at the latitude 85°.

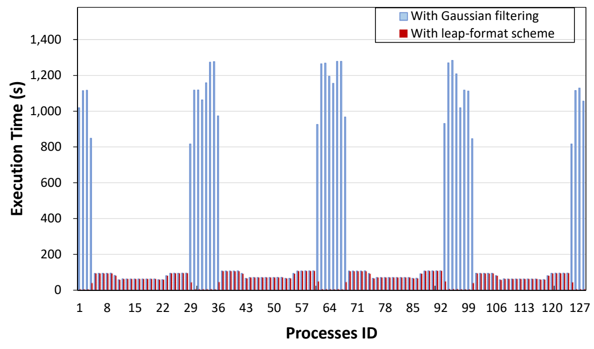


Fig. 13. Load balance for original and leap-format 3D dynamical core.

C. Feasibility and Load Balance of leap-format scheme

We compare our new leap-format implementation with the original 3D decomposition implementation of IAP-AGCM4 in this section.

To verify the practicability of our new leap-format computation, we present the effectiveness of short wave restraint at high latitude regions as in Fig. 12. Here we take the values of $DPsa$ along the zonal direction as the representation of the relative variables. All three lines are drawn based on the data at the latitude 85°. The gray line indicates the values of $DPsa$ without using any filtering scheme. The red line and blue line indicate the values after the adaptive Gaussian filtering and the dealing of leap-format computation, respectively; it is clearly that the high frequency part of the curve is well filtered by both solutions. In other words, the leap-format computation scheme achieves the exact effect as the original Gaussian filtering, despite that the two curves (red and blue) do not coincide absolutely. Moreover, larger time step can be used with the leap-format scheme to make the simulation more efficient.

As discussed in Section II-C, serious load imbalance occurs in the filtering module. Fig. 13 compares the load balance performance for the original and leap-format 3D dynamical core. The number of processes used for the test is 128, and the Y dimension is assigned 32 processes in priority. In each subdomain along the Y dimension, the execution time for the dynamical core with adaptive Gaussian filtering

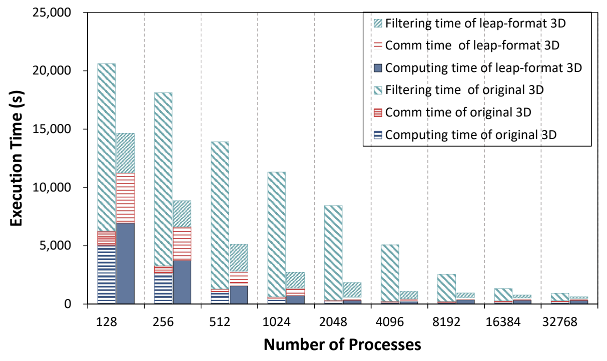


Fig. 14. Overall scaling comparison for original and leap-format 3D dynamical core.

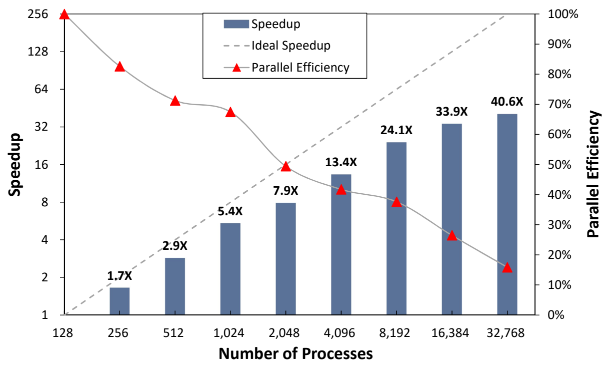


Fig. 15. Speedup and parallel efficiency of the leap-format 3D dynamical core.

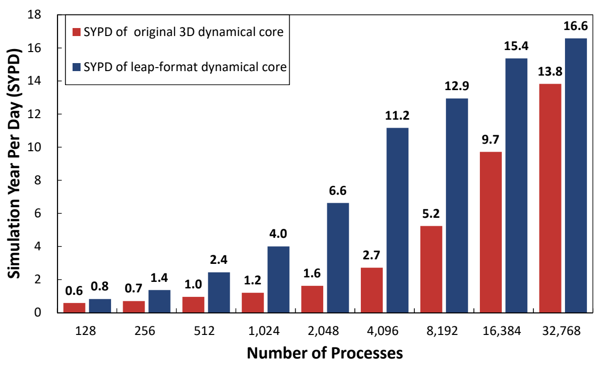


Fig. 16. Comparison of simulated-years-per-day for original and leap-format 3D dynamical core.

(blue histogram) differs dramatically in different processes, with the highest overhead of 1283 s, and the minimum of 57 s. In contrast, the overhead of leap-format computation (red histogram) for each process is more balanced due to the workload reduction at the high latitudes. Note that the runtime exhibited in Fig. 13 represent the whole cost of filtering in dynamical core. For the calls of filtering module of each forcast variable, such as U and V, there exists extra computation and communication. In general, the leap-format computation scheme achieves better performance than the original filtering module in terms of load balancing of the whole model.

D. Scalability and Overall Performance Test

Our tests of the original and leap-format 3D dynamical core on strong scaling are carried out with the configuration as in Table V. In both cases, the AGCM model is set to the resolution of $0.5° \times 0.5°$ and simulated for 2 model months. The execution time of the simulation is mainly comprised of three parts: the filtering time, the communication time, and the computing time, as shown in Fig. 14. As can be seen, for the range of process number from 128 to 32,768, the leap-format computation scheme achieves 3.3x speedup on average over the original implementation for the overall execution time. In particular, the runtime of the filtering module is decreased by 6.4x than the original adaptive Gaussian filtering module, which is the main contributor of the reduction of simulation time. For the communication and computing module, the overheads are increased by 3.3x and 1.2x compared with the original implementation, respectively. The reason for the runtime increasing is that some extra communication and computation are introduced along with the assignment and reference operation of the leaping grid points along the latitudinal circles. However, with the impressive performance improvement of the filtering module, the new leap-format computation scheme scales well up to 32,768 processes. The speedup and parallel efficiency for the leap-format based 3D dynamical core run of is shown in Fig. 15. For the strong scaling from 128 processes to 32,768 processes, the leap-format scheme achieves the speedup of 40.6x and 16% parallel efficiency.

To further analyze the simulation speed and computing throughput of the leap-format scheme, we perform experiments in terms of simulation year per computing day, namely SYPD for both the original and the leap-format 3D dynamical core. Results are presented in Fig. 16. The leap-format based dynamical core achieves the maximum simulation speed of 16.6 SYPD in comparison with the 13.8 SYPD of the original implementation and achieves on average 2.5x speedup over the original implementation.

V. Conclusion

In this work, a new optimized leap-format finite-difference computation scheme is proposed and implemented in the dynamical core of the IAP-AGCM4. The leap-format scheme generalizes the new difference format on the basis of the adaptive suitable mesh intervals and in turn stabilizes the numerical computation of simulation, which is the exact effect of a high-latitude filter takes. With the application of leap-format scheme, the costly filtering module at high latutudes and polar regions are fully eliminated. And the new scheme is parallelized with a shifting communication scheme in 3D decomposition dynamical core. In the 3D dynamical core of AGCM, the new leap-format scheme significantly reduces the overhead of filtering module and exhibits better load balance comparing to the original dynamical core with Gaussian filtering. Experiments are performed on the supercompter Tianhe-

2 with a series of case configurations from 128 processes to 32,768 processes. The feasibility and the correctness are examined. It's demonstrated that the our new leap-format computation scheme produces reasonable distribution of the involved variables, and performs better load balance than the original filtering module. As a whole, the new scheme scales the dynamical core of IAP-AGCM to 32,768 cores and achieves the speed of 16.6 simulation-year-per-day (SYPD) and up to 3.3x speedup over the latest implementation for the resolution of $0.5^\circ \times 0.5^\circ$.

We foresee our work on the new leap-format finite-difference computation scheme will achieve better scalability in higher resolution such as $0.25^\circ \times 0.25^\circ$. And it's worthy to explore the possibilities to migrate the leap-format to other modules of the earth system model (ESM), many of which come up against same pole problems with the equal-interval latitude-longitude mesh.

VI. Acknowledgments

This work is supported by the National Key R&D Program of China under Grant No. 2016YFB0200803; the Strategic Priority Research Program of Chinese Academy of Sciences, Grant No. XDC01040100; National Natural Science Foundation of China under Grant No. 61432018, Grant No. 61521092, and Grant No. 61602443; State Key Laboratory of Computer Architecture Foundation under Grant No. CARCH3504.

References

[1] H. Zhang, M. Zhang, and Q.-c. Zeng, "Sensitivity of simulated climate to two atmospheric models: Interpretation of differences between dry models and moist models," *Monthly Weather Review*, vol. 141, no. 5, pp. 1558–1576, 2013.

[2] X. Liu, R. C. Easter, S. J. Ghan, R. Zaveri, P. Rasch, X. Shi, J.-F. Lamarque, A. Gettelman, H. Morrison, F. Vitt *et al.*, "Toward a minimal representation of aerosols in climate models: Description and evaluation in the Community Atmosphere Model CAM5," *Geoscientific Model Development*, vol. 5, no. 3, p. 709, 2012.

[3] J. W. Hurrell, M. M. Holland, P. R. Gent, S. Ghan, J. E. Kay, P. J. Kushner, J.-F. Lamarque, W. G. Large, D. Lawrence, K. Lindsay *et al.*, "The community earth system model: a framework for collaborative research," *Bulletin of the American Meteorological Society*, vol. 94, no. 9, pp. 1339–1360, 2013.

[4] E. Roeckner, G. Bäuml, L. Bonaventura, R. Brokopf, M. Esch, M. Giorgetta, S. Hagemann, I. Kirchner, L. Kornblueh, E. Manzini *et al.*, "The atmospheric general circulation model ECHAM 5. PART I: Model description," 2003.

[5] H. Zhang, Z. Lin, and Q. Zeng, "The computational scheme and the test for dynamical framework of IAP AGCM-4," *Chinese J. Atmos. Sci*, vol. 33, pp. 1267–1285, 2009.

[6] B. Wu, S. Li, H. Cao, Y. Zhang, H. Zhang, J. Xiao, and M. Zhang, "Agcm3d: A highly scalable finite-difference dynamical core of atmospheric general circulation model based on 3d decomposition," in *2018 IEEE 24th International Conference on Parallel and Distributed Systems (ICPADS)*. IEEE, 2018, pp. 355–364.

[7] D. L. Williamson, "The evolution of dynamical cores for global atmospheric models," *Journal of the Meteorological Society of Japan. Ser. II*, vol. 85, pp. 241–269, 2007.

[8] D. Xiao, X. Feng, Z. He, and Z. Qing-Cun, "Evaluation of surface air temperature change over china and the globe during the twentieth century in iap agcm4. 0," *Atmospheric and Oceanic Science Letters*, vol. 5, no. 5, pp. 435–438, 2012.

[9] Y. Zheng-Bin, L. Zhao-Hui, and Z. He, "The relationship between the east asian subtropical westerly jet and summer precipitation over east asia as simulated by the iap agcm4. 0," *Atmospheric and Oceanic Science Letters*, vol. 7, no. 6, pp. 487–492, 2014.

[10] T. Su, F. Xue, and H. Zhang, "Simulating the intraseasonal variation of the east asian summer monsoon by iap agcm4. 0," *Advances in Atmospheric Sciences*, vol. 31, no. 3, pp. 570–580, 2014.

[11] M. Adeniyi, Z. Lin, and H. Zhang, "Evaluation of the performance of iap-agcm4. 1 in simulating the climate of west africa," *Theoretical and Applied Climatology*, vol. 136, no. 3-4, pp. 1419–1434, 2019.

[12] D. A. Randall, T. D. Ringler, R. P. Heikes, P. Jones, and J. Baumgardner, "Climate modeling with spherical geodesic grids," *Computing in Science & Engineering*, vol. 4, no. 5, pp. 32–41, 2002.

[13] B. Wang, H. Wan, Z. Ji, X. Zhang, R. Yu, Y. Yu, and H. Liu, "Design of a new dynamical core for global atmospheric models based on some efficient numerical methods," *Science in China Series A: Mathematics*, vol. 47, no. 1, pp. 4–21, 2004.

[14] R. Courant, K. Friedrichs, and H. Lewy, "Über die partiellen differenzengleichungen der mathematischen physik," *Mathematische annalen*, vol. 100, no. 1, pp. 32–74, 1928.

[15] Y. Zhang, R. Yu, J. Li, and H. Chen, "An implementation of a leaping-point two-step shape-preserving advection scheme in the high-resolution spherical latitude-longitude grid," *Acta Meteorol Sin*, vol. 71, no. 6, pp. 1089–1102, 2013.

[16] Y. Wang, J. Jiang, H. Zhang, X. Dong, L. Wang, R. Ranjan, and A. Y. Zomaya, "A scalable parallel algorithm for atmospheric general circulation models on a multi-core cluster," *Future Generation Computer Systems*, vol. 72, pp. 1–10, 2017.

[17] J. Xiao, S. Li, B. Wu, H. Zhang, K. Li, E. Yao, Y. Zhang, and G. Tan, "Communication-Avoiding for Dynamical Core of Atmospheric General Circulation Model," in *Proceedings of the 47th International Conference on Parallel Processing*. ACM, 2018, p. 12.

[18] K. Ito and K. Xiong, "Gaussian filters for nonlinear filtering problems," *IEEE transactions on automatic control*, vol. 45, no. 5, pp. 910–927, 2000.

[19] J. G. Charney, R. Fjörtoft, and J. Von Neumann, "Numerical integration of the barotropic vorticity equation," in *The Atmosphere—A Challenge*. Springer, 1990, pp. 267–284.

[20] L. Yuan, S. Huang, Y. Zhang, and H. Cao, "Tessellating star stencils," in *Proceedings of the 48th International Conference on Parallel Processing*. ACM, 2019, p. 43.

[21] I. M. Held and M. J. Suarez, "A proposal for the intercomparison of the dynamical cores of atmospheric general circulation models," *Bulletin of the American Meteorological Society*, vol. 75, no. 10, pp. 1825–1830, 1994.

[22] Z. Xuehong, "Dynamical framework of IAP nine-level atmospheric general circulation model," *Advances in Atmospheric Sciences*, vol. 7, no. 1, pp. 67–77, 1990.

[23] N. A. Phillips, *Numerical integration of the primitive equations on the hemisphere*. Citeseer, 1959.