Manuscript prepared for Geosci. Model Dev. Discuss.
with version 2014/05/30 6.91 Copernicus papers of the LaTeX class copernicus.cls.
Date: 23 December 2014

Pangolin v1.0, a conservative 2-D advection model towards large scale parallel calculation

A. Praga[1], **D. Cariolle**[1], **and L. Giraud**[2]

[1]Centre Européen de Recherche et de Formation Avancée en Calcul Scientifique, Toulouse, France
[2]Institut National de Recherche en Informatique et en Automatique, Talence, France

Correspondence to: A. Praga (praga@cerfacs.fr)

Abstract

To exploit the possibilities of parallel computers, we designed a large-scale bidimensional atmospheric advection model named Pangolin. As the basis for a future chemistry-transport model, a finite-volume approach for advection was chosen to ensure mass preservation and to ease parallelization. To overcome the pole restriction on time-steps for a regular latitude-longitude grid, Pangolin uses a quasi-area-preserving reduced latitude-longitude grid. The features of the regular grid are exploited to reduce the memory footprint and enable effective parallel performances; finally a custom domain decomposition algorithm is presented. To assess the validity of the advection scheme, its results are compared with state-of-the-art models on algebraic test cases. Finally, parallel performances are shown in terms of strong scaling and confirm the efficient scalability up to a few hundred of cores.

1 Introduction

Global three-dimensional chemistry-transport models (hereafter referred as CTMs) play an important role in monitoring and predicting the composition of the atmosphere (e.g. Chipperfield, 2006; Teyssèdre et al., 2007; Huijnen et al., 2010). Those models include large-scale transport, emissions and chemical transformations of trace species and sub-scale grid processes like convection and deposition. In CTMs, advection by large-scale winds is a key process that must be handled by numerical algorithms. For these algorithms, mass conservation for the considered species, monotonicity and numerical accuracy are especially important for long simulations where accumulation of errors and bias must be avoided.

In this paper, we present a conservative advection model on the sphere, which is intended to form the basic framework for a future CTM. The adopted scheme is based on a flux-form (Eulerian) tracer advection algorithm on a reduced latitude-longitude grid. A finite-volume approach was chosen as it provides an easy way to ensure mass preservation. Furthermore, parallelizing the model is then reduced to a classical domain decomposition problem.

The specificity of our model, named Pangolin[1], lies in the grid definition where the number of cells on a latitude circle is progressively decreased towards the pole in order to obtain a grid which preserves approximately the cell areas at mid- and high-latitudes. This avoids the so-called "pole problem" arising from the convergence of the meridians that severely limits the size of the time-steps for Eulerian models. Several approaches have been adopted in previous studies to cope with this issue. Finite-volume schemes on latitude-longitude grids often aggregate latitudinal fluxes near the poles (e.g. Hourdin and Armengaud, 1999) or double successively the grid size at high-latitudes (e.g. Belikov et al., 2011).

Alternatively, quasi-uniform spherical grids have been developed, such as cubed-sphere[2], composite mesh – Yin-Yang – or icosahedral grids[3]. A review of the different grids can be found in Staniforth and Thuburn (2012) or Williamson (2007). However, those approaches lose the latitudinal regularity arising from the rotation of the Earth. Furthermore, they require specific treatments at the singularities of the adopted polygons, which may also induce resolution clustering near these points. On the plus side, they allow the implementation of more accurate algorithms than the ones on reduced latitude-longitude grids. This last point is especially important for weather and climate models that solve the non-linear momentum equation but is less stringent for the two-dimensional (2-D) linear transport of trace species on the sphere.

To construct the reduced grid, one difficulty is to define a structure which avoids treating the poles as special cases, as that can impact the precision and properties of the advection algorithm. Thus we have chosen to adopt a semi-structured approach. The grid is not regular as the number of cells varies with the latitude, but the coordinates of cell interfaces can be computed algebraically. We thus avoid storing a list of neighbours as an irregular unstructured grid would normally require, hence decreasing memory costs. This was done to anticipate future parallel architectures, which may have less memory capacity per core than current systems.

Our goal is to use an adequate algorithm exploiting the grid features to achieve efficiency and scalability on massively parallel architectures. Fine control over parallelization was obtained

[1]PArallel implementatioN of a larGe scale multi-dimensiOnaL chemIstry-traNsport scheme

[2]Each face uses Cartesian coordinates and is projected gnomonically or conformally on the sphere.

[3]An icosahedra is subdivided until the desired resolution and projected on the sphere.

using the Message Passing Interface (MPI) library. In that context, the advection scheme must be chosen as to balance its accuracy vs. the volume of the required parallel communications. Furthermore, grid properties were carefully studied to improve the parallel version. Especially, a custom domain decomposition consistent with our grid was designed.

The present paper is organized as follows. Section 2 recalls the basic equations and numerical methods used to solve the advection of the chemical species. In Sect. 3, results from standard test cases for advection of tracer on the sphere are reported. Those cases were chosen from the test case suite proposed by Lauritzen et al. (2012). Section 4 gives details on the model implementation on parallel architectures and the results of parallel scalability experiments. In Sect. 5, we summarize the results obtained and discuss the possible extension of our method.

2 Numerical scheme

2.1 Finite-volume formulation

Our model is based on a finite-volume method to integrate the tracer advection equation. This is performed on a bi-dimensional discrete grid on the sphere, which is described in more detail in Sect. 2.2. In each grid cell, the tracer concentration changes according to the divergence of the fluxes at the cell boundaries. This comes from the flux-form of the continuity and tracer conservation equations:

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \boldsymbol{V}) = 0, \tag{1}$$

$$\frac{\partial \rho q}{\partial t} + \nabla \cdot (\rho q \boldsymbol{V}) = 0, \tag{2}$$

where ρ is the air density, q the tracer mixing ratio and $\boldsymbol{V}$ the winds vector field. Equations (1) and (2) are first integrated over a cell area $\mathcal{A}$. Noting $\partial\mathcal{A}$ the cell boundary and $\boldsymbol{n}$ the local normal pointing outward, the divergence theorem yields:

$$\frac{\partial m}{\partial t} = \frac{\partial}{\partial t} \int\limits_{\mathcal{A}} \rho = \int\limits_{\partial \mathcal{A}} (\rho \boldsymbol{V} \cdot \boldsymbol{n} \, \mathrm{d}S), \tag{3}$$

$$\frac{\partial m_r}{\partial t} = \frac{\partial}{\partial t} \int\limits_{\mathcal{A}} \rho q = \int\limits_{\partial \mathcal{A}} (\rho q \boldsymbol{V} \cdot \boldsymbol{n} \, \mathrm{d}S), \tag{4}$$

where m is the total air mass in a cell and m_r is the total tracer mass in a cell. The right-hand side of Eqs. (3) and (4) can be seen as the integral of all the fluxes across the cell boundaries. This formulation gives a conservative scheme when the same fluxes are used for upstream and downstream adjacent cells.

The above equations are then integrated in time during a time-step:

$$m^{n+1} = m^n - \int\limits_{t_n}^{t_{n+1}} \Big(\int\limits_{\partial \mathcal{A}} \rho \boldsymbol{V} \cdot \boldsymbol{n} \, \mathrm{d}S \Big) \mathrm{d}t, \tag{5}$$

$$m_r^{n+1} = m_r^n - \int\limits_{t_n}^{t_{n+1}} \Big(\int\limits_{\partial \mathcal{A}} \rho q \boldsymbol{V} \cdot \boldsymbol{n} \, \mathrm{d}S \Big) \mathrm{d}t. \tag{6}$$

Equation (5) can be omitted for non-divergent flows since the divergence of the mass fluxes is null and the mass inside the cells are constant: $[m]^{n+1} = [m]^n$. In this paper, we only consider non-divergent flows. As such, winds are corrected to be divergence-free in a preprocessing

step as explained in Sect. 2.3. Handling divergent flows would require minor adjustments to the scheme (removing the correction of winds and adding Eq. 5 into the scheme) but this configuration was not considered as typical use case of CTMs, where large-scale 3-D winds can be considered divergence-free.

There are many options to evaluate the tracer mass fluxes at the cell boundary. These fluxes are approximated as the air mass fluxes multiplied by the mean tracer ratio $\widehat{q}$ crossing the interface. The simplest approach to evaluate $\widehat{q}$ has been introduced by Godunov (Godunov et al., 1961), who considered $\widehat{q}$ as constant within each upstream cell. The resulting scheme is conservative, monotonicity-preserving but very diffusive. Improvements of the Godunov scheme have been introduced by van Leer in van Leer (1977), where q is now approximated by a non-constant polynomial function. Depending on the polynomial degree used, i.e., the order moments of the distribution of q inside the cell, van Leer obtained several schemes (up to six) that varied in complexity. A review of the different possible options can be found in Rood (1987) and Hourdin and Armengaud (1999). In general, accuracy is found to increase when higher-order moments are used but the price to pay lies in larger computational and memory costs. Using higher-moments requires a larger number of grid points to compute the derivatives, which increases communication volumes when domains decomposition techniques are used for parallel clusters (see Sect. 4).

For our model, we have adopted a first-order reconstruction, the van Leer scheme (noted as van Leer I in the original paper). The distribution of q in the cells is approximated by a linear function in latitudinal and meridional directions. The slope of the linear function is computed as a finite difference using the values of q within the nearest cells. In that configuration, the scheme is second-order accurate in space. To extend the algorithm to multiple dimensions, a time-splitting scheme is used. Equations (5) and (6) are first integrated in the zonal direction:

$$\begin{aligned} \widetilde{m}_i^{n+1} &= m_i^n + U_{i-1/2} - U_{i+1/2}, \\ (\widetilde{m_r})_i^{n+1} &= (m_r)_i^n + F_{i-1/2} - F_{i+1/2}, \end{aligned} \tag{7}$$

where U and F are the air and tracer fluxes respectively across the borders orthogonal to the chosen direction during a time-step. With these notation, tracer fluxes are approximated as

$F_{i+1/2} \approx \widehat{q}_{i+1/2} u_{i+1/2}$. Figure 1 illustrates the reconstruction of $\widehat{q}_{i+1/2}$. The broken line represents the tracer distribution in the 1-D case for cells i and $i+1$. Finding $\widehat{q}_{i+1/2}$ depends on the wind direction: for outward winds ($u_{i+1/2} > 0$), the grey area on the left figure will move from cell i to cell $i+1$. Then the mean tracer ratio corresponding to this flux is computed from the distribution in cell i. The same can be applied for inward fluxes, resulting into:

$$\widehat{q}_{i+1/2} = \begin{cases} q_i + \left(1 - \dfrac{u_{i+1/2}\Delta t}{\Delta x}\right)(\delta q)_i & \text{if} \quad u_{i+1/2} > 0, \\ q_{i+1} - \left(1 + \dfrac{u_{i+1/2}\Delta t}{\Delta x}\right)(\delta q)_{i+1} & \text{otherwise,} \end{cases}$$

where δq is the slope of the linear reconstruction and $u_{i+1/2}$ the wind at the interface. Δt and Δx are the time-step and cell spacing respectively.

This first advection step gives us the intermediate mass value $\widetilde{m}$ and tracer value $q' = \widetilde{m}/\widetilde{m_r}$. These new values are then used to integrate Eq. (6) in the meridional direction. As the grid is unstructured , mass and tracer fluxes in the north-south direction have to be evaluated for all the neighbours of each cell. This is detailed in Sect. 2.3.

It should be noted that other time-splitting schemes are available. Another approach is to use a zonal-meridional advection during a time-step and meridional-zonal for the next. It is also possible to compute both zonal-meridional and meridional-zonal advection and then use their mean as the final value. For the complete description of each method, see Machenhauer et al. (2009). Either way, the final bidimensional algorithm is second-order accurate. In Pangolin, all three time-splitting schemes have been tested using the numerical order of convergence test (see Sect. 3.3). It was found the choice of the time-splitting algorithm has little impact on accuracy.

To ensure monotonicity of the solution and to prevent numerical oscillations, van Leer introduced a slope limiter. The idea is to limit the slope value within a given cell such as the tracer value in that cell does not exceed the mean value of the adjacent cells. This is more restrictive than limiting the fluxes to ensure that the tracer values remain between the maximum-minimum

values of the adjacent cells but is easier to implement. The slope is given by:

$$(\delta q)_i = \min\left(\frac{1}{2}|q_{i+1} - q_{i-1}|, 2|q_{i+1} - q_i|, 2|q_i - q_{i-1}|\right) \times \text{sign}(q_{i+1} - q_i), \tag{8}$$

if q_i lies in between q_{i-1} and q_{i+1}, and $(\delta q)_i = 0$ otherwise. As discussed by Hourdin and Armengaud (1999), the slope limiter efficiently damps the numerical oscillations but introduces more diffusion of the numerical solutions. For the test cases reported in this study, the slope limiter appears to have little impact on the accuracy of the numerical solutions.

2.2 Grid

The grid used in Pangolin is completely defined by the number of cells at the north pole and the number of latitudes n_{lat} on an hemisphere. The Southern Hemisphere is simply constructed in the same way as the Northern Hemisphere. To find the number of cells at the pole, we can write the equality between cell areas at the pole and at the Equator. If we consider squared cells at the Equator, like the ones on a regular latitude-longitude grid, and small latitudinal spacing, the number of cells is approximately π. Thus we set the number of cells at the poles to 3.

At a given latitude, all cells have the same area. We can then compute the number of cells for all latitudes. Latitudinal and longitudinal spacings are no longer supposed identical, for maximum flexibility. Let us consider the area of cell (i, j), noted $\mathcal{A}_i$ as it does not depends on j. Let us note ϕ_i the colatitude and λ_{ij} the position of the south and east cell borders. As we suppose the cell spacings constant, we can write $\phi_i = i\Delta\phi_i$ and $\lambda_{ij} = j\Delta\lambda_i$. The area is defined on $\Omega_{ij} = [\lambda_j, \lambda_{j+1}] \times [\phi_{i-1}, \phi_i]$, so in spherical coordinates, we have:

$$\begin{aligned}\mathcal{A}_i &= \iint_{\Omega_{ij}} r^2 \sin\phi \,\mathrm{d}\lambda \mathrm{d}\phi \\ &= r^2 \Delta\lambda_i \big(\cos(\phi_{i-1}) - \cos(\phi_{i-1} + \Delta\phi_i)\big).\end{aligned} \tag{9}$$

Areas are preserved so $\mathcal{A}_i = \mathcal{A}_1$:

$$\Delta\lambda_i = \Delta\lambda_1 \frac{1-\cos(\Delta\phi_1)}{2\sin\left(\frac{\Delta\phi_i}{2}\right)\sin\left(\left(i-\frac{1}{2}\right)\Delta\phi_i\right)}.$$

Noting n_i the number of cells at colatitude i, we get

$$\frac{n_i}{n_1} = \left\lfloor \frac{\Delta\lambda_1}{\Delta\lambda_i} \right\rfloor = \left\lfloor \frac{2\sin\left(\frac{\Delta\phi_i}{2}\right)\sin\left(\left(i-\frac{1}{2}\right)\Delta\phi_i\right)}{1-\cos(\Delta\phi_1)} \right\rfloor.$$

Now let us assume $\forall i$, $\Delta\phi_i$ and $(i-\frac{1}{2})\Delta\phi_i$ are small enough:

$$\frac{n_i}{n_1} \approx \left\lfloor \frac{2\frac{\Delta\phi_i}{2}\left(i-\frac{1}{2}\right)\Delta\phi_i}{\frac{\Delta\phi_1^2}{2}} \right\rfloor = 2i-1.$$

Finally, we can define the number of cells for the whole grid, with $2n_{\text{lat}}$ latitudes, as:

$$n_i = \begin{cases} 3(2i-1) & \text{if } 1 \leq i \leq n_{\text{lat}}. \\ n_{2n_{\text{lat}}-i+1} & \text{otherwise.} \end{cases} \tag{10}$$

It follows that the total number of cells on the grid is $6n_{\text{lat}}^2$. As an illustration, the grid is shown on Fig. 2.

The previous formula is a sound approximation for area preservation near the poles and when latitudinal spacing is constant. In practice, we consider the approximation as reasonable up to 75°: the relative error is then less than 1 %. At lower latitudes, the error increases, with a maximum of 56 % at the Equator. So the grid used in Pangolin gives higher resolutions at the Equator than at the poles. One way around this issue is to truncate the number of cells at a given threshold. As a comparison, Fig. 3 shows the number of cells for Pangolin with the "exact" and truncated version. By "exact", we mean the number of cells comes from the area-preservation

formulae without any approximations for $\Delta\phi_i$. Furthermore, to truly preserve the cell areas, we should use a variable latitudinal spacing. However, the distortion due to a constant latitudinal spacing was found to be acceptable and much less pronounced compared with a regular latitude-longitude grid.

The formula given in Eq. (10) allows us to easily determine the coordinates of the cell neighbours in each zone. The grid used in Pangolin has four axes of symmetry: $\lambda = 0$, $\lambda = 120$, $\lambda = 240^\circ$ and the Equator, so the grid can be split into six identical *zones*, numbered from west to east and north to south as shown on Fig. 2. The following formulae to compute the position of the cell neighbours are given for the first zone – i.e, $[90^\circ, 0] \times [0, 120^\circ]$. The formulae on zones 2 and 3 are obtained by adding the proper offset. In the Southern Hemisphere (zone 4 to 6), the north and south formulae are simply inverted. The zonal neighbours for cell (i, j) are simply $(i, j-1)$ and $(i, j+1)$. For the first zone, its north and south neighbours are respectively $\big\{(i-1, j_1), \ldots, (i-1, j_2)\big\}$ and $\big\{(i+1, j_3), \ldots, (i+1, j_4)\big\}$, with:

$$
\begin{aligned}
j_1 &= \left\lfloor \frac{n_{i-1}}{n_i}(j-1) + 1 \right\rfloor, \quad j_2 = \left\lceil \frac{n_{i-1}}{n_i} j \right\rceil, \\
j_3 &= \left\lfloor \frac{n_{i+1}}{n_i}(j-1) + 1 \right\rfloor, \quad j_4 = \left\lceil \frac{n_{i+1}}{n_i} j \right\rceil.
\end{aligned} \tag{11}
$$

From that formulation, it follows that the number of meridional neighbours is not constant, even though most of cells have two north and two south adjacent cells. Special cases include the middle of each sector (one north and three south neighbours) and its extremities (one north and two south). These figures apply for the Northern Hemisphere and must be inverted in the Southern Hemisphere.

As a consequence, computing the position of the neighbours is quite efficient, involving mostly integer operations and roundings. These computations are thus performed on-the-fly to reduce the storage requirements. In a more general way, the algebraic properties of the grid are exploited as much as possible. Our parallelization strategy relies heavily on it, as shown in Sect. 4.

2.3 Adapting the scheme to the grid

For winds and tracer discretization, we adapt the Arakawa C-grid (Arakawa and Lamb, 1977) to our scheme, as shown on Fig. 4. To avoid interpolating the winds components during advection, winds are taken at the middle of the interfaces. Tracer concentrations are defined at the centres of the cells.

Due to the structure of the grid (Fig. 2), air and tracer fluxes need to be computed for all the neighbours of the cells as illustrated on Fig. 5. For each flux, the frontier between the current cell and its neighbour is computed algebraically using the cell neighbours formulae (Eq. 11). While there is no special treatment for zonal advection, meridional advection requires an interpolation to compute the meridional gradient. A linear interpolation in the zonal direction is computed to evaluate the value of the mixing ratio on the meridian passing through the center of the cell (see Fig 6). These values are used to compute the meridional gradient by finite-difference.

It is critical for mass preservation that winds have a null divergence so we correct interpolated winds (both zonal and meridional) to achieve that. The first step in our correction deals with meridional winds. If we consider all the cells on a latitude circle, the total mass variation in this "band" only comes from the north and south meridional fluxes. Now, if we consider the latitude circle containing all south meridional fluxes, it constitutes a closed contour. Therefore, the wind divergence is null and the sum of all meridional winds must be zero. Meridional winds are thus corrected by removing the mean value from the interpolated values. For future 3-D case, a preprocessing step involving vertical winds will be needed to ensure non-divergent circulation. Depending on the system of vertical coordinates used, the "mass-winds inconsistency" issue (see for example Jöckel et al. (2001)) will have to be addressed.

Then we correct zonal winds to ensure that the sum of all fluxes is locally null in each cell. As meridional winds are already corrected, only east and west zonal winds of each cell must be modified. We take zonal winds at longitude 0 as a reference and browse each cell sequentially from west to east to correct progressively each zonal winds and ensure mass preservation.

Finally, we need to take care that fluxes in a given direction do not completely empty the cells during an advection step. For each cell, a local advective Courant number restricts the time-step

to avoid this situation. As advection is performed sequentially in two different directions, we define two unidimensional local Courant numbers. Then the global Courant number C is simply defined as the most restrictive conditions on all cells:

$$C = \max_{ij} \left(\frac{u_{ij} \Delta t}{\Delta \phi_{ij}}, \frac{\Delta t \sum_{k \in V_{ij}} v_k \Delta \lambda_k}{\Delta \phi_{ij} \sum_{k \in V_{ij}} \Delta \lambda_k} \right),$$

where V_{ij} is the set of meridional neighbours for the cell (i, j) and $\Delta \lambda_k$ is the interface size between the cell and its neighbour. For the tests in this paper, we use $C_{\max} = 0.96$ as a Courant–Friedrichs–Lewy (CFL) condition.

3 Testing suite

A standard 2-D testing suite to check the accuracy and properties of a transport model was proposed in Lauritzen et al. (2012). A comparison with state-of-the-art schemes was subsequently published in Lauritzen et al. (2013), which offers a convenient benchmark to compare transport models on the sphere. From it, we have extracted a subset of the models and cases which we felt were relevant to Pangolin. In Lauritzen et al. (2013), the different grids were compared with a constant resolution at the Equator. In the present paper, we retain simulations performed with a constant total number of cells. The number of cells in each model was computed using the resolution at the Equator given in the appendix of Lauritzen et al. (2013). As a summary, Table 1 contains here the formulae used and gives an idea of the size of each grid, in comparison with Pangolin.

3.1 Models features

The models were chosen as their spatial order is similar to Pangolin. They are implemented on both regular and non-regular grids and provide a basis for a comparison between semi-Lagrangian, finite volume and wave propagation methods. A summary is given in Table 2. Other features are described below:

- FARSIGHT is a grid-point semi-Lagrangian model, running on parallel architectures,
- CLAW uses a wave propagation technique with a first-order method (donor cell upwind) in each direction,
- SLFV-ML (Slope Limited Finite Volume scheme with Method of Lines), a flux-form finite volume with simplified swept area and linear reconstruction,
- CAM-FV (Community Atmosphere Model Finite-Volume) is a finite-volume model on a regular latitude-longitude grid. It uses the Piecewise Parabolic Method (PPM), with the addition of a slope and curvature limiters,
- USICOM (UC Irvine Second-Order Moments scheme) is also a flux-form finite-volume. It uses an improved version of Prather's second order moments scheme on an Eulerian regular latitude-longitude grid.

All of these models are mass-preserving so the comparison with Pangolin is relevant. CAM-FV and UCISOM are of particular interest as they also use a directional splitting algorithm. Furthermore, all models have a shape-preserving algorithm but only FARSIGHT and Pangolin do not expand the initial tracer concentration range.

3.2 Test cases

In this paper, we only consider the case of non-divergent and time-dependent winds. Zonal and meridional winds (u and v respectively) are given by:

$$u(\lambda,\theta,t) = \frac{10R}{T}\sin^2(\lambda')\sin(2\theta)\cos\left(\frac{\pi t}{T}\right) + \frac{2\pi R}{T}\cos(\theta),$$
$$v(\lambda,\theta,t) = \frac{10R}{T}\sin(2\lambda')\cos(\theta)\cos\left(\frac{\pi t}{T}\right),$$

where θ is now the latitude, and λ the longitude, both in radians. R is the Earth radius, $T = 12\,\mathrm{days}$ the period, and $\lambda' = \lambda - 2\pi t/T$. With these winds, the tracer concentration first moves

eastwards and is then deformed into filaments up to $t = T/2$. After that, the flux is inverted and the tracer continues to move to the east until it comes back to its initial distribution at $t = T$.

These winds provide an easy way to compute the errors as the solution after a full period can be simply compared with the initial concentration. We will use the same normalized errors as in Lauritzen et al. (2012):

$$\ell_2(q) = \sqrt{\frac{\mathcal{I}((q-q_0)^2)}{\mathcal{I}(q_0^2)}} \quad \text{and} \quad \ell_\infty(q) = \frac{\max\limits_{\forall \lambda,\theta} |q - q_0|}{\max\limits_{\forall \lambda,\theta} |q_0|},$$

where $q = q(\lambda, \theta, t)$ is the tracer concentration and q_0 the initial concentration. Also, $\mathcal{I}$ is defined as the global integral:

$$\mathcal{I}(q) = \frac{1}{4\pi} \int_0^{2\pi} \int_{-\pi/2}^{\pi/2} q(\lambda, \theta, t) \cos\theta \mathrm{d}\lambda \mathrm{d}\theta.$$

For our model, the tracer mixing ratio is approximated as linear functions in a cell. Thus the mean value corresponds with the value at the middle of the cell, so the integral is approximated by $\mathcal{I}(q) = \sum_{i,j} \widehat{q}_{ij} \mathcal{A}_{ij}$ where $\widehat{q}_{ij}$ is the tracer mean value in cell (i, j) and $\mathcal{A}_{ij}$ its area given by Eq. (9).

Two initial conditions are used here: a sum of two Gaussian hills and a sum of two cosine bells. We note $(\lambda_1, \theta_1) = (5\pi/6, 0)$ and $(\lambda_2, \theta_2) = (7\pi/6, 0)$ the coordinates of the two "centres" used below. Gaussian hills are defined as:

$$q(\lambda, \theta) = h_1(\lambda, \theta) + h_2(\lambda, \theta).$$

Noting $h_{\max} = 0.95$ and $b = 5$, we have for $i = 1, 2$:

$$h_i(\lambda, \theta) = h_{\max} e^{-b((X-X_i)^2 + (Y-Y_i)^2 + (Z-Z_i)^2)}.$$

where X, Y, Z are the Cartesian coordinates of (λ, θ) and X_i, Y_i, Z_i are the Cartesian coordinates of (λ_i, θ_i) for $i = 1, 2$. Cosine bells are defined by:

$$q(\lambda,\theta) = \begin{cases} b + c \times h_1(\lambda,\theta) & \text{if} \quad r_1 < r, \\ b + c \times h_2(\lambda,\theta) & \text{if} \quad r_2 < r, \\ b \quad \text{otherwise,} \end{cases}$$

with the background value $b = 0.1$ and amplitude $c = 0.9$. Noting $h_{\text{max}} = 1$ and $r = R/2$, we also have:

$$\forall i = 1, 2, \quad h_i(\lambda,\theta) = \frac{h_{\text{max}}}{2}\left(1 + \cos\left(\pi \frac{r_i}{r}\right)\right).$$

where the r_i are the great-circle distances to (λ_i, θ_i) on the sphere:

$$r_i(\lambda,\theta) = R \arccos(\sin\theta_i \sin\theta + \cos\theta_i \cos(\lambda - \lambda_i)).$$

Pangolin results for the Gaussian hills and cosine bells test cases are shown in Fig. 7 at $t = 0$, half the period and after a full period. The shape of the tracer is distribution is well-preserved but numerical diffusion contributes to decrease the tracer maxima, as it appear at $t = T/2$ and $t = T$. To compute the numerical order of convergence in Sect. 3.3, results at $t = 0$ and $t = T$ will be used, while the preservation of filaments in Sect. 3.4 is computed using the results at $t = 0$ and $t = T/2$.

3.3 Numerical order of convergence

This test aims to check the rate at which numerical error decreases when resolution increases. Ideally, this rate should be close to the theoretical order of convergence. The Gaussian hills test case is used here, as it provides an infinitely smooth function. Results are plotted on Fig. 8 where ℓ_2 and ℓ_∞ are plotted with a varying number of cells. When available, the impact of shape-preserving filters is also represented on the plot, with the exception of UCISOM. For

this choice of models, it does not reduce errors in a significant way as it can be expected from lower-order models.

For errors at low- and mid-resolutions, Pangolin is quite close to the other models, with the exception of UCISOM. However, the errors with a large number of cells are lower for models other than Pangolin. One possible explanation is the loss of accuracy due to the interpolation when computing the meridional gradient. In general, the order of convergence of Pangolin is lower. To quantify that, we use numerical optimal order of convergences corresponding to the errors ℓ_2 and ℓ_∞. They result from a least-square linear regression on the errors plotted vs. the resolution at the Equator:

$$\log(\ell_i) = -\kappa_i \log(\Delta\lambda) + b_i \quad \text{with} \quad i = 2, \infty.$$

For Pangolin, the regression is applied after the optimal convergence was reached. This corresponds to longitudinal resolutions at the Equator in the range $[0.75°, 0.1875°]$. The final numerical convergence rates are shown in Fig. 9.

This selection of models and parameters show that theoretical order is not achieved for all models. For most of them, using a different Courant number does not improve the convergence speedup, with the exception of FARSIGHT and CAM-FV. Using a CN of 10.4 (resp. 1.2) greatly improves the result of FARSIGHT (resp. CAM-FV) with a CN of 1.4 (resp. 0.2).

Furthermore, we can see the numerical order of convergence of Pangolin is lower than other models. This is not surprising when comparing with similar finite-volume schemes as CAM-FV and UCISOM which use higher-order schemes in one or more directions. We studied this issue further using solid-body rotation test cases (described in Williamson et al. (1992)) and found that the accuracy was limited by the linear interpolation done for the meridional gradient. When the axis of the solid-body rotation matches the polar axis, accuracy is close to second-order, whereas the level of accuracy decreases when the axis is in the equatorial plane. In practice, Pangolin needs finer resolutions as shown in the following test cases to match the accuracy of other models.

3.4 Preservation of filaments

Realistic distributions will most likely be deformed into filaments when the tracer material is stretched and gradients are increased. For some applications, it is important to check how well these filaments are preserved. In the cosine bells test case, the initial concentration is deformed into thin filaments up to $t = T/2$, before being advected to the initial position. Diagnostics are thus computed at $T/2$.

Let us consider the area where the tracer is greater than a given threshold τ. For non-divergent flows and for all thresholds, if this area is not preserved at all times, it suggests filaments are degraded. This leads to the definition of the following diagnostic:

$$\ell_{\mathrm{f}} = \begin{cases} 100 \times \dfrac{A(\tau,t)}{A(\tau,0)} & \text{if } A(\tau,0) \neq 0, \\ 0 \quad \text{otherwise,} \end{cases}$$

where $A(\tau,t)$ is the surface area for which the tracer ratio at time t is greater or equal to τ. For Pangolin and other Eulerian schemes, $A(\tau,t)$ is defined as the sum of the cell areas where $q(t) \geq \tau$. Thus the closest ℓ_{f} is to 100, the better the filaments are preserved.

Results are shown in Fig. 10. All models except Pangolin are compared with different numbers of cells but with the same resolution of 0.75° at the Equator. Pangolin is shown with two different numbers of cells. The first case (shown as light solid line) has a resolution of approximately 0.75° at the Equator. The second one (shown as dark solid line) is a higher-resolution version (0.1°at the Equator) which approaches the results of more accurate models.

The behaviour of the models on non latitude-longitude grids (all models, except UCISOM and CAM-FV) is typical of diffusive schemes. They tend to diffuse the base of the distribution and reduce the maxima, thus leading to an increase of ℓ_{f} for small τ and a decrease of ℓ_{f} for large τ. Another information we can extract from this is the alteration of gradients. Here CAM-FV can be seen to have $\ell_{\mathrm{f}} > 100$ for large τ, which results from gradient steepening. On the other hand, schemes on non-regular grids have a smooth and decreasing profile, showing the scheme diffusion is also smooth and continuous.

Furthermore, the shape from the diagnostic for Pangolin is quite close to models using non regular latitude-longitude grids (CLAW, FARSIGHT, SLFV-ML). The behaviour is typical of diffusive schemes, where ℓ_{f} is increased for lower threshold values and decreased for higher τ. Pangolin is less accurate than these models but similar accuracy can be achieved using a finer resolution. Nevertheless, the models using a regular latitude-longitude grid, namely UCISOM and CAM-FV, are the most accurate for this test.

3.5 Preservation of preexisting relations

When advecting several correlated tracers, numerical transport schemes should preserve the relations between them. However, errors due to numerical diffusion can modify these relations and introduce a "mixing". This is by no means an unphysical feature as real-life tracers can undergo some mixing too, either by chemical reactions or diffusion. This test aims at assessing the amount of unphysical mixing. To that end, we follow Lauritzen et al. (2012) and use a first tracer with the cosine bells initial condition $q_{\cos}(t=0)$ and a second tracer correlated to the first:

$$q_{\mathrm{corr}}(t=0) = -0.8 q_{\cos}^2(t=0) + 0.9. \tag{12}$$

After a half-period of advection using the non-divergent winds for each case, we plot $q_{\mathrm{corr}}(t = T/2)$ against $q_{\cos}(t = T/2)$ as a scatter plot. Depending on the position of the points, we can then check the mixing level. An illustration of the different zones is given in Fig. 11. The convex area shown in dark on the figure corresponds to "real" mixing as it contains all the lines between two points on the curve corresponding to Eq. (12). The light gray area is not a physical mixing but is still in the initial range. Everything outside the box is overshooting, which may result in unphysical concentrations such as negative values.

Results are shown in Figs. 12 and 13. All unlimited schemes present some overshooting for low value of $q_{\cos}$, which is then removed with a shape-preserving filter. Also, all schemes, with the exception of UCISOM, present real mixing. For this selection of models, only FARSIGHT and CAM-FV do have range-preserving unmixing. For FARSIGHT, that can be removed with a larger Courant number. In that case, its accuracy is on the same level as UCISOM. Concerning

accuracy, the closer the points lie to the curved defined in Eq. (12), the more the scheme preserves the initial relation. In that respect, UCISOM is the most accurate models of this selection, followed by Pangolin and CAM-FV.

3.6 Comparing parallel performances

Pangolin was designed with scalability and parallel performances in mind, thus leading to the choice a smaller stencil than the models presented here. From the results presented above, it is clear Pangolin can match the accuracy of other models using finer resolution. Unfortunately, comparing the parallel performances in terms of running time on multi-core architectures is difficult. Some models provide technical details about the performances — see White and Dongarra (2011) for FARSIGHT, Dennis et al. (2011) for CAM-FV or Erath and Nair (2014); Guba et al. (2014) for some other state-of-the art schemes — but there is not enough data for a thorough comparison. We thus provide as much details as possible on the parallel performances of Pangolin alone. In particular, Sec. 4.3 highlights the smallest size of subdomains needed for a reasonable efficiency for 2-D parallel advection.

4 Parallelization

For large scale numerical simulations, using only sequential computations is no longer affordable. To use a parallel approach, we need to split and balance the computational effort among a set of computing units. For partial different equations-based simulations where the computational cost is evenly shared among the cells, a natural and widely used approach is to partition the computational domain into connex subdomains of similar sizes. Each subdomain is handled by a different computing unit that leads a well-balanced parallel calculation.

The original objectives in designing this model were twofold. First, we intended to have a discretization with cells of equal areas so that the CFL condition is not penalized by the smallest cells. Second, we targeted a semi-structured grid to avoid managing complex data structures as well as an extra-tool to generate it. This leads us to define the grid detailed in Sect. 2.2 where

computing the neighbours of grid cells is fully algebraic. In a parallel framework, this grid has an additional asset as it enables a custom algebraic partitioning. Otherwise, mesh splitting often requires sophisticated mesh partitioning tools such as those developed by Karypis and Kumar (1995) or Pellegrini (2012).

4.1 Partitioning

In order to perform the partitioning, we first exploit the grid symmetries. The grid is composed of six identical zones as described in Sect. 2.2. We then focus on partitioning one of these zones, which contains n_{lat}^2 cells, where n_{lat} is the number of latitudes in the hemisphere. A perfect work balance occurs when there are p^2 subdomains with p dividing n_{lat}. In this case, each subdomain contains exactly $(n_{\mathrm{lat}}/p)^2$ cells. The most natural way to gather cells to form the subdomains is to use the same algorithm as the one to build the grid. The p^2 subdomains are set on p bands, where the kth band contains $2k-1$ subdomains ($k>0$). Applying the same decomposition to the remaining five zones leads to the structure shown in Fig. 14.

For the sake of flexibility, this optimal situation can be slightly degenerated to accommodate any number of subdomains. For these sub-optimal situations, we consider the closest lower square p'^2 on a zone, with p' dividing n_{lat}. These p'^2 subdomains are set according to the previous strategy. The remaining $p^2-p'^2$ cores are associated with on a special band, with less cells and thus without preserving the partition size.

These results are given for one zone only. For the complete grid, we find the optimal number of subdomains is $6p^2$ with p dividing n_{lat}. Otherwise, the model can manage any number of subdomains on one third of the grid (between longitude 0 and 120). The total number of subdomains in these optimal cases is then a multiple of 3. It is worthwhile noticing that White and Dongarra (2011) need a condition similar to our perfect case for their parallel version: the number of subdomains must be of the form $6p^2$, with p dividing the number of cells on a cube edge.

This algebraic partitioning uses the regular topology of the grid to create subdomains with a regular shape. This feature ensures regular data access and allows for possible optimizations by anticipation strategies such as pre-fetching to improve faster data access by loading data into

the cache before it is actually needed. To highlight this regularity and for the sake of comparison, we display in Fig. 15 the partition computed by the mesh partitioner Scotch (Pellegrini, 2012) for the same grid as in Fig. 14. One can observe that this general purpose tool does not succeed in preserving the regular shape of the subdomains. In addition, our partitioning reduces the number of neighbours for the subdomains and consequently the number of messages exchanged. The total number of neighbours of our partitioning is at least divided by two when compared with Scotch, even in sub-optimal cases.

4.2 Parallel implementation

Our parallel implementation first targets distributed memory architectures. Therefore, we consider a message passing parallel implementation on top of the MPI library, where each subdomain is assigned to a different computing unit. To update the tracer ratio for all the cells in a subdomain, most of the information required to compute the fluxes is already available in the subdomain. However, some communications need to be performed to exchange information along the interfaces generated by the partitioning. For example, zonal fluxes need to exchange concentration and gradient data with the east and west neighbouring subdomains. In that respect, we introduce ghost cells which store data received from the neighbours via message exchanges. It should be noted that due to the shape of the subdomains meridional advection requires communications with the north, south, east and west neighbours. This is illustrated in Fig. 16, where the ghost cells are shown as hatched cells.

In order to improve the parallel performance of the code and hide the communication time as much as possible, non-blocking communications are used. This avoids waiting for the completion of communications and allows to optimize computations. In practice, we first post the communication requests, then we perform the calculation on the interior cells, which do not need data outside the current subdomain. Then we finalize data reception and eventually compute the new quantities on the boundary cells using the values received in the ghost cells. This approach is a rather classical implementation to overlap computation and communication in large scale simulations. The specificity of Pangolin comes rather from the algebraic features of the decomposition so the neighbours of a subdomains and the cells inside it are computed

on-the-fly. The model was also designed to have only one layer of ghost-cells as to decrease the communication volume.

Each advection step can be decomposed into several tasks: gradient computing, fluxes computing and mass update. The first two tasks require some communication and the non-blocking approach is used for each one of them. It should be noted that meridional advection requires an extra communication as it needs a zonal gradient to perform an interpolation (see Sect. 2.3). The different tasks are shown in detail in the algorithms shown on Fig. 17. The combination of boundary and interior cells for all these tasks is shown in Fig. 16.

The boundary zone is much larger for meridional advection as data needs to be exchanged with its four neighbours. More precisely, computation of the zonal gradient requires communication with the east and west neighbours of the subdomain. Furthermore, computing the meridional gradient and fluxes requires access to the north and south neighbours. Due to the semi-structured layout of the grid, this results in the "stair"-like structure for boundary cells in the meridional advection.

Most of the computation is performed during meridional advection. First, due to the extra step of computing the zonal gradient, meridional advection requires three messages exchanges (vs. two for zonal advection). Also, the number of boundary cells is larger, thus increasing the communication volume. Finally, computation in a cell is more expensive as the number of neighbours is four on average (vs. two in the zonal case).

4.3 Performances

In this section, we investigate the parallel scalability of our implementation of the numerical scheme. Tests were done on the Bull cluster at CERFACS, whose features are shown in Table 3. We consider a strong speed-up study where the size of the global grid is fixed and the number of computing cores is increased. Ideally, the parallel elapsed time should be reduced proportionally to the number of cores selected for the parallel simulation. For our experimental study, we consider the elapsed time on three cores as the reference time so that the speed-up is defined by

$$S(p) = \frac{T(3)}{T(p)}$$

where $T(p)$ is the parallel elapsed time observed when performing the calculation on p cores.

In strong speed-up studies, the number of cells per subdomain decreases when the number of subdomains increases. Even though we attempt to overlap the communication and the calculation, communication volume tends to grow when the number of subdomains increases. A direct consequence is that the observed speed-ups gradually depart from the ideal speed-up when the number of cores increases as it might be observed in Fig. 18 (left-hand side). On the right plot, we report experiments for the 2-D advection scheme using various grid sizes ranging from $1.13^\circ \times 0.75^\circ$ to $0.28^\circ \times 0.19^\circ$ when the number of cores varies from 3 to 128. As expected, the larger the grid, the better the parallel performance since we can better overlap calculation and computation. The speed-up curves exhibit steps, with significant gaps when an optimal number of cores (6, 24, 54, 96) is used. In between, using more cores does not translate to an improvement in performance as the work-load of the largest sub-domain is not reduced.

The final version of Pangolin will be combined with chemistry modelling. As the chemistry computation is fully local, we can estimate the performances of the chemistry-advection model. Figure 18 (right-hand side) shows the estimated speed-ups of the complete chemistry advection simulation on the finest grid (i.e., $0.28^\circ \times 0.19^\circ$). We assumed the chemistry cost was constant across all cells and the chemical time-step was similar to the advection time-step. These assumptions are not valid in practice and only give an upper bound on the speed-up. Nevertheless they give some insight on the final performances. The chemical time-step was obtained from a new solver developed by D. Cariolle (personal communication, 2014) called ASIS with 90 species. As a reference, we use its implementation by P. Moinat, using the GMRES method (personal communication, 2014). As a result, adding the chemistry greatly increases the computational load in a subdomain and thus improves the scalability. On the other hand, communication volumes only increase linearly as a function of the number of tracers, as expected.

Comparing performances between parallel models is not an easy task. A meaningful comparison would require to compile and run all the models on the same cluster as hardware and

software performances are paramount in such studies. Such tests are out-of-scope for the current paper. However, we examine the limits of our parallelization strategy, an additional strong-scaling test was run with a rather coarse resolution ($2.25 \times 1.75°$): the number of cores was increased until the subdomains became extremely small. At that point, the computational load inside the subdomains is not enough to cover the communication costs. These results are shown on Fig. 19 where the efficiency is plotted against the number of cores. Here, the efficiency is defined as

$$E(p) = \frac{T(3)}{pT(p)},$$

so ideal performances should be close to 1. We can consider the parallel performances "break down" at 294 cores. The size of the subdomains is then 5×5, an non-realistic configuration where the subdomains are so small that the communication cost can no longer be hidden. From this test, we have estimated the size of subdomains needed for an efficiency of 0.75 was 18×18. In practice, this allowed us to estimate the number of cores needed for the same efficiency at different resolutions. Results are shown in Table 4.

5 Conclusions

In this paper, we have presented a parallel scalable algorithm for 2-D-advection on the sphere. We focused on enabling the model to be as parallel as possible. Pangolin uses a reduced latitude-longitude grid, which overcomes the pole issue, and a finite-volume formulation that ensures local conservation of the tracer mass. Grid features were carefully exploited to minimize memory requirements on one hand, and provide maximal efficiency on parallel architectures on the other hand. The accuracy of the scheme was also chosen as to minimize the impact on message-passing. It was found that the approximations made for computing the meridional gradients near the poles limits the accuracy of the model. Therefore, to reach the accuracy of other second-order models, resolution must be increased. This can be easily achieved without large computation penalty due to the good scalability of Pangolin.

We expect further improvement in terms of parallelism when chemistry is added. An ongoing work addresses real-case atmospheric situations using a linear scheme (Cariolle and Teyssèdre (2007)), which used to model the evolution of stratospheric ozone on an isentropic surface. In future versions, vertical advection will be added, requiring a more advanced correction of the winds for mass preservation. A complex chemistry will also be added using the ASIS solver and the RACMOBUS scheme (Dufour et al. (2005)). This chemistry will most likely perturb the load balancing. One mitigation strategy would be to use multi-threading in the subdomains. To conclude, Pangolin is a practical model, aiming at taking advantage of present and future parallel architectures for large-scale atmospheric transport.

Code availability

The code is copyrighted to the CERFACS laboratory. The documentation is available as a user manual and as the code documentation at http://cerfacs.fr/~praga/pangolin/index.html. To request access to either the source code or documentation, please send a mail to A. Praga (praga@cerfacs.fr) or D. Cariolle (cariolle@cerfacs.fr). The data and scripts for the plots of this paper are also available as a Supplement.

The Supplement related to this article is available online at doi:10.5194/gmdd-0-1-2014-supplement.

Acknowledgements. Financial support from the DGAC (Direction Générale de l'Aviation Civile) through the IMPACT project is gratefully acknowledged. A. Praga was supported by a PhD grant from Météo-France.

References

Arakawa, A. and Lamb, V. R.: Computational design of the basic dynamical processes of the UCLA general circulation model, Methods in computational physics, 17, 173–265, 1977.

Belikov, D., Maksyutov, S., Miyasaka, T., Saeki, T., Zhuravlev, R., and Kiryushov, B.: Mass-conserving tracer transport modelling on a reduced latitude-longitude grid with NIES-TM, Geoscientific Model Development, 4, 207–222, doi:10.5194/gmd-4-207-2011, 2011.

Cariolle, D. and Teyssèdre, H.: A revised linear ozone photochemistry parameterization for use in transport and general circulation models: multi-annual simulations, Atmospheric Chemistry and Physics Discussions, 7, 1655–1697, doi:10.5194/acpd-7-1655-2007, 2007.

Chipperfield, M. P.: New version of the TOMCAT/SLIMCAT off-line chemical transport model: Intercomparison of stratospheric tracer experiments, Quarterly Journal of the Royal Meteorological Society, 132, 1179–1203, doi:10.1256/qj.05.51, 2006.

Collins, W. and Rasch, P. J.: Description of the NCAR community atmosphere model (CAM 3.0), NCAR Tech. Note . . . , 2004.

Dennis, J. M., Edwards, J., and Evans, K.: CAM-SE: A scalable spectral element dynamical core for the Community Atmosphere Model, International Journal of High Performance Computing Applications, http://hpc.sagepub.com/content/early/2011/11/12/1094342011428142.abstract, 2011.

Dufour, a., Amodei, M., Ancellet, G., and Peuch, V.-H.: Observed and modelled "chemical weather" during ESCOMPTE, Atmospheric Research, 74, 161–189, doi:10.1016/j.atmosres.2004.04.013, http://linkinghub.elsevier.com/retrieve/pii/S0169809504001498, 2005.

Erath, C. and Nair, R. D.: A conservative multi-tracer transport scheme for spectral-element spherical grids, Journal of Computational Physics, 271, 244–260, doi:10.1016/j.jcp.2014.04.008, http://linkinghub.elsevier.com/retrieve/pii/S0021999114002629, 2014.

Godunov, S. K., Zabrodin, A. V., and Prokopov, G. P.: A computational scheme for two-dimensional nonstationary problems of gas dynamics and calculation of the flow from a shock wave approaching a stationary state, Zhurnal Vychislitel'noi . . . , 1, 1020—-1050, 1961.

Guba, O., Taylor, M., and St-Cyr, A.: Optimization-based limiters for the spectral element method, Journal of Computational Physics, 267, 176–195, doi:10.1016/j.jcp.2014.02.029, http://linkinghub.elsevier.com/retrieve/pii/S0021999114001491, 2014.

Hourdin, F. and Armengaud, A.: The use of finite-volume methods for atmospheric advection of trace species. Part I: Test of various formulations in a general circulation model, Monthly weather review, 127, 822–837, 1999.

Huijnen, V., Williams, J., van Weele, M., van Noije, T., Krol, M., Dentener, F., Segers, A., Houweling, S., Peters, W., de Laat, J., Boersma, F., Bergamaschi, P., van Velthoven, P., Le Sager, P., Eskes, H., Alkemade, F., Scheele, R., Nédélec, P., and Pätz, H.-W.: The global chemistry transport model TM5: description and evaluation of the tropospheric chemistry version 3.0, Geoscientific Model Development, 3, 445–473, doi:10.5194/gmd-3-445-2010, 2010.

Jöckel, P., von Kuhlmann, R., Lawrence, M. G., Steil, B., Brenninkmeijer, C. A. M., Crutzen, P. J., Rasch, P. J., and Eaton, B.: On a fundamental problem in implementing flux-form advection schemes for tracer transport in 3-dimensional general circulation and chemistry transport models, Quarterly Journal of the Royal Meteorological Society, 127, 1035–1052, doi:10.1002/qj.49712757318, http://doi.wiley.com/10.1002/qj.49712757318, 2001.

Karypis, G. and Kumar, V.: Metis-unstructured graph partitioning and sparse matrix ordering system, version 2.0, Citeseer, http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.38.376, 1995.

Lauritzen, P. H., Skamarock, W. C., Prather, M. J., and Taylor, M. a.: A standard test case suite for two-dimensional linear transport on the sphere, Geoscientific Model Development Discussions, 5, 189–228, doi:10.5194/gmdd-5-189-2012, http://www.geosci-model-dev.net/5/887/2012/, 2012.

Lauritzen, P. H., Ullrich, P. a., Jablonowski, C., Bosler, P. a., Calhoun, D., Conley, a. J., Enomoto, T., Dong, L., Dubey, S., Guba, O., Hansen, a. B., Kaas, E., Kent, J., Lamarque, J.-F., Prather, M. J., Reinert, D., Shashkin, V. V., Skamarock, W. C., Sø rensen, B., Taylor, M. a., and Tolstykh, M. a.: A standard test case suite for two-dimensional linear transport on the sphere: results from a collection of state-of-the-art schemes, Geoscientific Model Development, 6, 105–145, doi:10.5194/gmdd-6-4983-2013, http://www.geosci-model-dev-discuss.net/6/4983/2013/http://www.geosci-model-dev.net/7/105/2014/, 2013.

LeVeque, R. J.: Finite volume methods for hyperbolic problems, Cambridge University Press, 2002.

Machenhauer, B., Kaas, E., and Lauritzen, P. H.: Finite volume methods in meteorology, ... Methods for the Atmosphere ..., M, 3–120, doi:10.1016/S1570-8659(08)00201-9, 2009.

Miura, H.: An Upwind-Biased Conservative Advection Scheme for Spherical Hexagonal–Pentagonal Grids, Monthly Weather Review, 135, 4038–4044, doi:10.1175/2007MWR2101.1, 2007.

Pellegrini, F.: PT-Scotch and LibPTScotch 0.6 User's Guide, Tech. rep., INRIA Bordeaux Sud-Ouest, 2012.

Prather, M. J.: Numerical advection by conservation of second-order moments, Journal of Geophysical Research, 91, 6671—-6681, 1986.

Rood, R. B.: Numerical advection algorithms and their role in atmospheric transport and chemistry models, Reviews of geophysics, http://onlinelibrary.wiley.com/doi/10.1029/RG025i001p00071/full, 1987.

Staniforth, A. and Thuburn, J.: Horizontal grids for global weather and climate prediction models: a review, Quarterly Journal of the Royal . . . , 138, 1–26, doi:10.1002/qj.958, 2012.

Teyssèdre, H., Michou, M., Clark, H., Josse, B., Karcher, F., Olivié, D., Peuch, V., Saint-Martin, D., Cariolle, D., Attié, J., Nédélec, P., Ricaud, P., Thouret, V., van der A, R., and Volz-Thomas, A.: A new tropospheric and stratospheric Chemistry and Transport Model MOCAGE-Climat for multi-year studies: evaluation of the present-day climatology and sensitivity, Atm. Chem. Phy, 7, 5815–5860, 2007.

van Leer, B.: Towards the ultimate conservative difference scheme. IV. A new approach to numerical convection, Journal of computational physics, 23, 276—-299, http://www.sciencedirect.com/science/article/pii/002199917790095X, 1977.

White, J. B. and Dongarra, J. J.: High-performance high-resolution semi-Lagrangian tracer transport on a sphere, Journal of Computational Physics, 230, 6778–6799, doi:10.1016/j.jcp.2011.05.008, 2011.

Williamson, D. L.: The evolution of dynamical cores for global atmospheric models, Meteorological Society Of Japan, 85, 241–269, 2007.

Williamson, D. L., Drake, J. B., Hack, J. J., Jakob, R., and Swarztrauber, P. N.: A standard test set for numerical approximations to the shallow water equations in spherical geometry, Journal of Computational Physics, 102, 211–224, 1992.

Table 1. For a given resolution at the equator, we compare the total number of cells of each model n_{model} vs. the total number of cells of Pangolin n_{pangolin}.

Model	$n_{\text{model}}/n_{\text{pangolin}}$	n_{model}
FARSIGHT	2	$6 \cdot (90/\Delta\lambda)^2$
CLAW	0.68	$2 \cdot (90/\Delta\lambda)^2$
SLFV-ML	2.17	N.A
CAM-FV	2.7	$\lfloor 360/\Delta\lambda \rfloor \cdot \lfloor 180/\Delta\lambda \rfloor$
UCISOM	2.7	$\lfloor 360/\Delta\lambda \rfloor \cdot \lfloor 180/\Delta\lambda \rfloor$
Pangolin	1	$6 \cdot \lceil 0.5 \cdot \lceil 120/\Delta\lambda \rceil + 1 \rceil^2$

Table 2. Summary of the models used as a comparison: name, implementation grid, the total number of cells vs. Pangolin, along with the time scheme.

Model	Grid	Formal accuracy	Time scheme	Reference
FARSIGHT	Gnomonic cubed-sphere	2	3rd order	White and Dongarra (2011)
CLAW	Two-patch sphere grid	2	Euler-forward	LeVeque (2002)
SLFV-ML	Icosahedral-hexagonal	2	Runge–Kutta 3rd order TVD	Miura (2007)
CAM-FV	Regular lat-lon	2	Euler-forward	Collins and Rasch (2004)
UCISOM	Regular lat-lon	2	Euler-forward	Prather (1986)
Pangolin	Pangolin	2	Euler-forward	This paper

Table 3. Configuration for one of the 158 SandyBridge nodes.

– 2 Intel Sandy Bridge 8-cores CPUs (peak performance of 330 GFlops s^{-1})
– 32 GB of memory
– 31 KB L1 cache, 256 KB L2 cache for each core
– 20 MB L3 cache, shared by the 8 cores of each CPU

Table 4. Estimation of the number of cores needed for an efficiency of 0.75 for 2-D advection at several resolutions (lat $\times$ lon).

Resolution	Nb cores
1.5×1.0	150
0.75×0.5	486
0.15×0.1	6144

Table 5. Some Pangolin configurations, with the number of latitudes on an hemisphere n_{lat}, total number of cells n_{pangolin} and resolution at the Equator in degrees.

n_{lat}	n_{pangolin}	$\Delta\phi \times \Delta\lambda$
20	2400	4.5×3.08
90	48 600	1.0×0.67
320	614 400	0.28×0.188

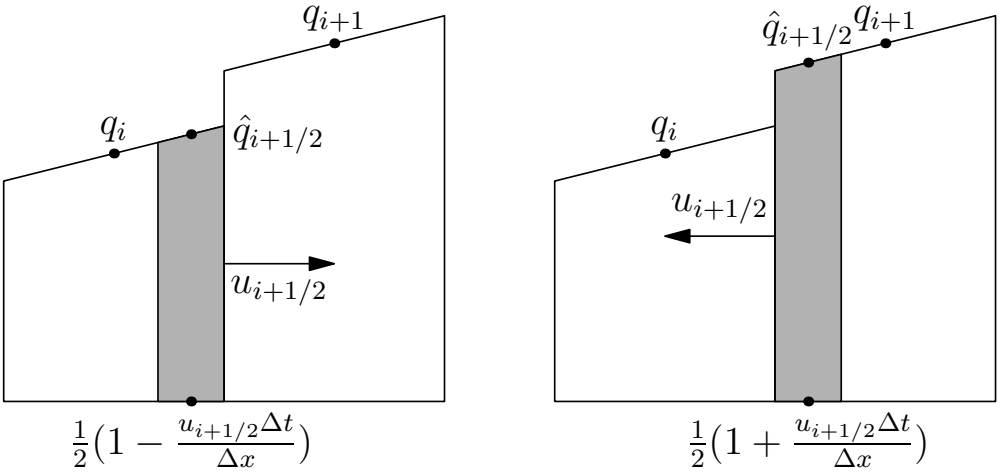

Figure 1. van Leer scheme for positive (left) and negative (rights) winds. The distribution of the tracer is shown as broken lines. The grey area is the quantity of tracer passing through the interface during a time-step.

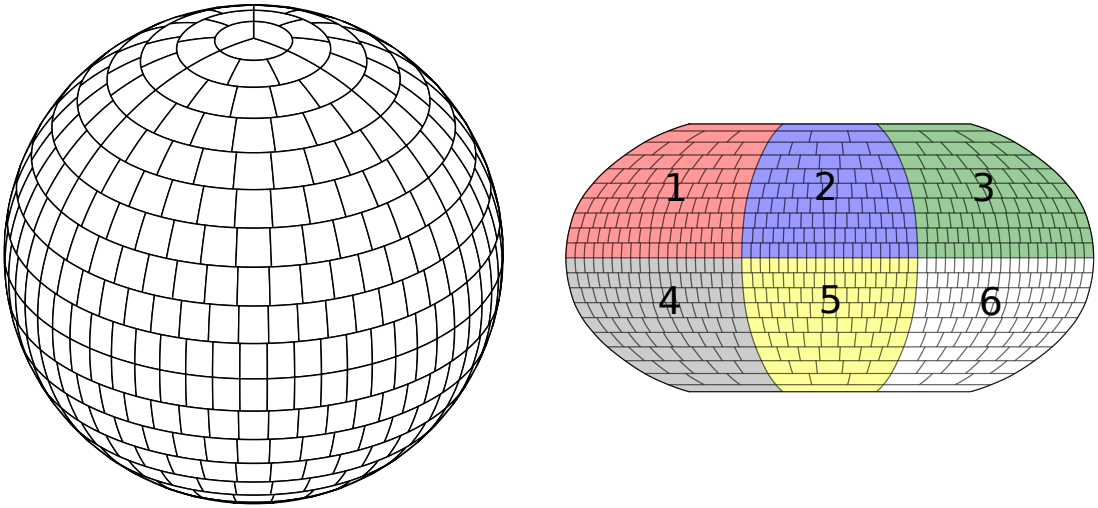

Figure 2. Grid used in Pangolin with 20 latitudes: orthographic projection (left) and Robinson projection, with the 6 identical zones highlighted (right).

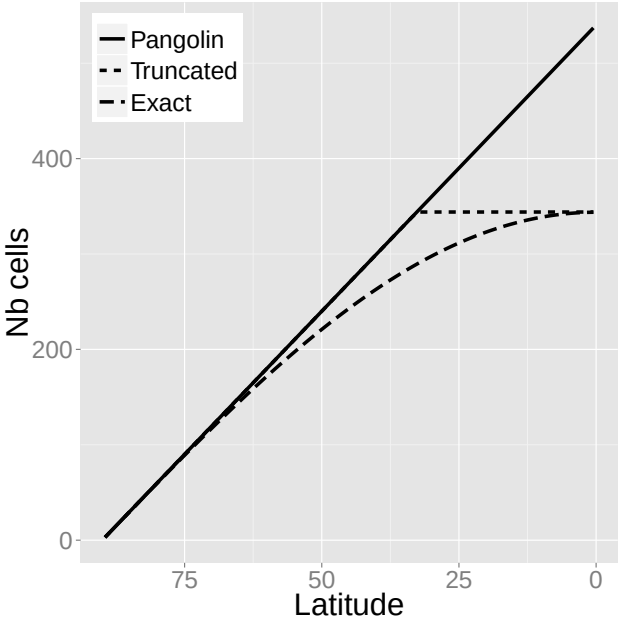

Figure 3. Number of cells for the grid used by Pangolin on one hemisphere with 90 latitudes (solid line). The truncated and "exact" version are shown as dotted and dashed lines respectively.

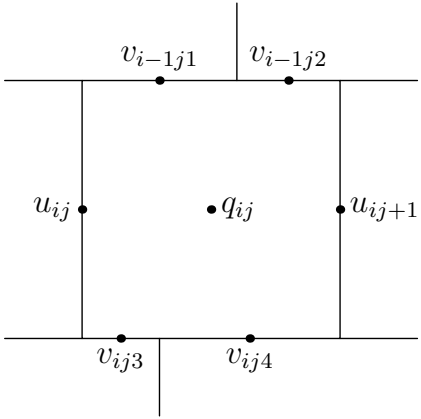

Figure 4. Discretization for zonal and meridional winds (u and v respectively) and tracer mixing ratio q.

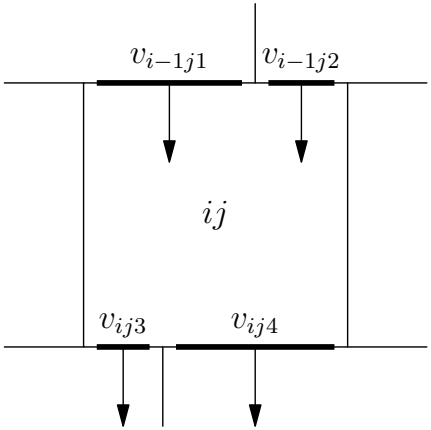

Figure 5. Meridional interfaces (bold lines) and fluxes (dark arrows) for cell (i, j).

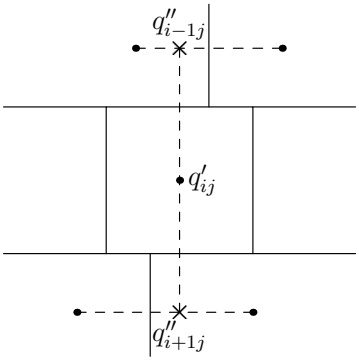

Figure 6. Zonal interpolation to compute the meridional gradient of q'_{ij}.

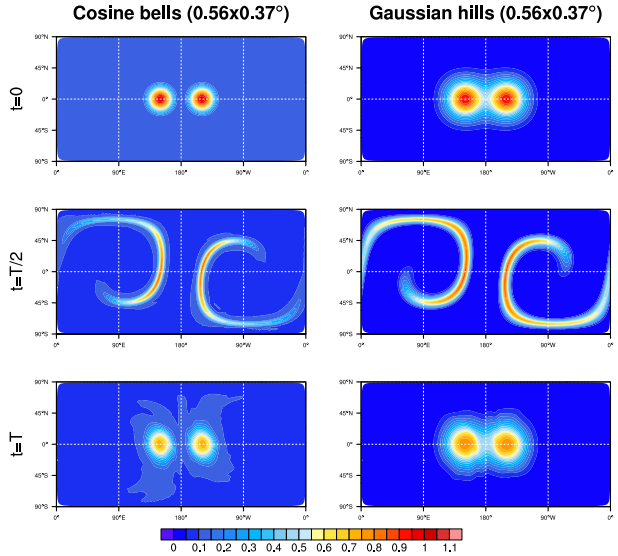

Figure 7. Cosine bells and Gaussian hills results for $t = 0$, $t = T/2$, $t = T$ (top to bottom) with Pangolin. The initial distribution is first deformed into filaments and then advected back to its initial position. The Equator resolution is $0.56° \times 0.37°$. For these plots, Pangolin data is interpolated to a regular latitude-longitude grid.

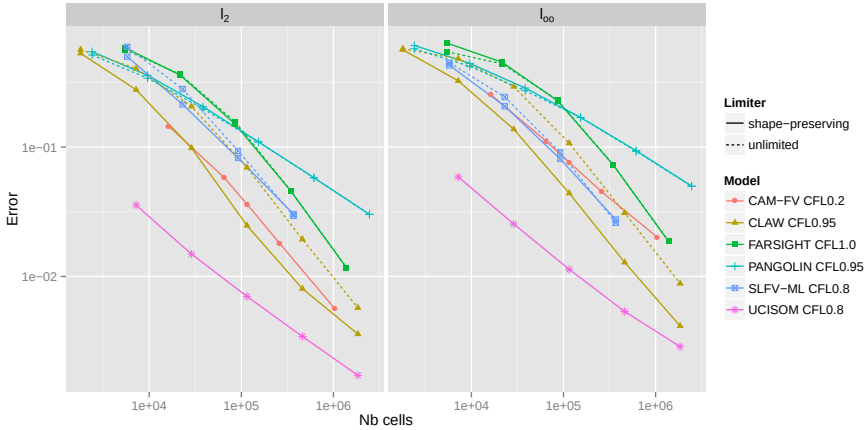

Figure 8. Numerical order of convergence for both error measures. There are computed using Gaussian hills after a full rotation. When available, the models are shown with and without the shape-preserving limiters.

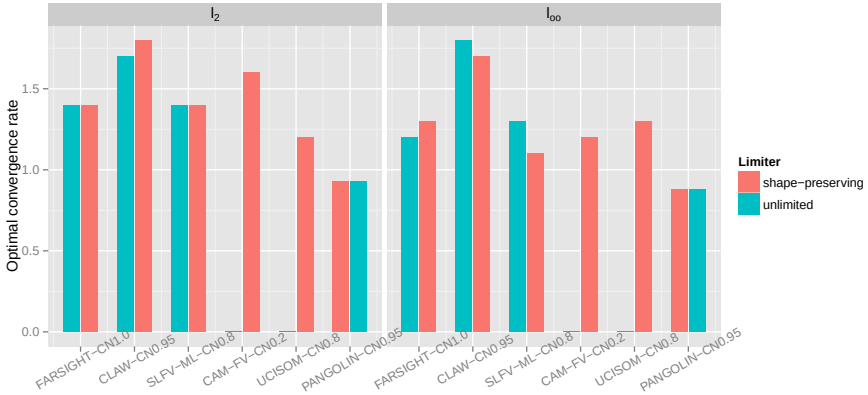

Figure 9. Optimal order of convergence computed by a least-square regression on data from Fig. 8. Some models did not offer data without shape-preserving limiters (CAM-FV, UCISOM).

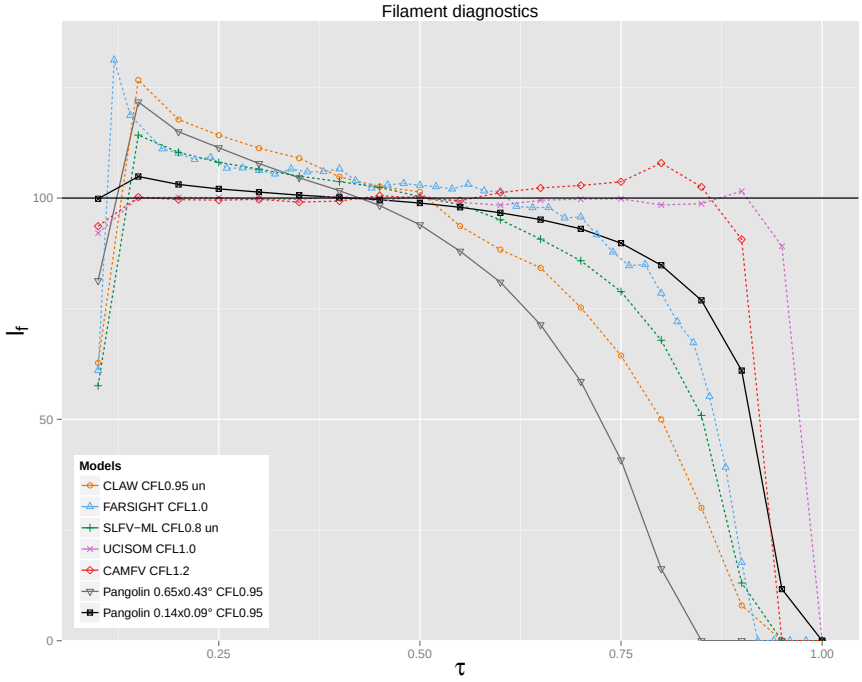

Figure 10. Filament diagnostics between for Pangolin (solid line) and other models (dashed line). By default, the shape-preserving version is used. Only CLAW and SLFV-ML use an "unlimited" version.

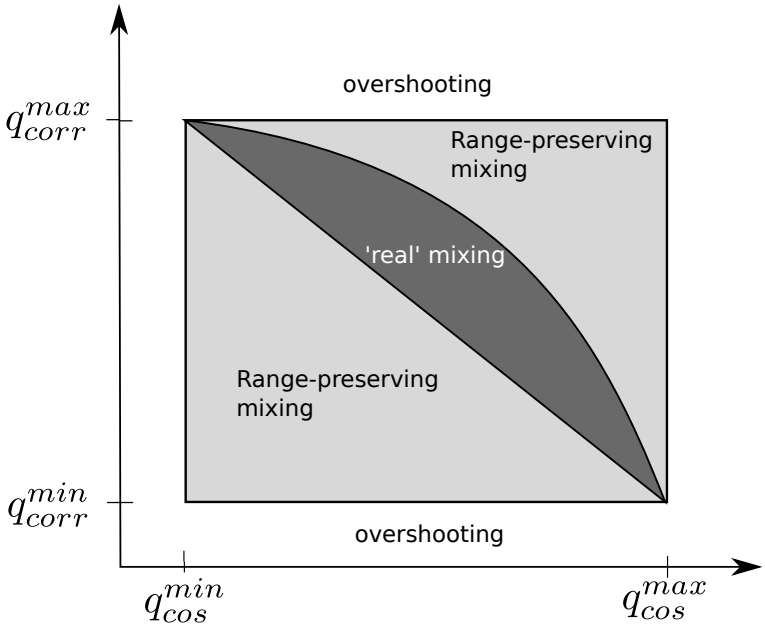

Figure 11. Different types of mixing when plotting the correlated concentration against the cosine bells distribution.

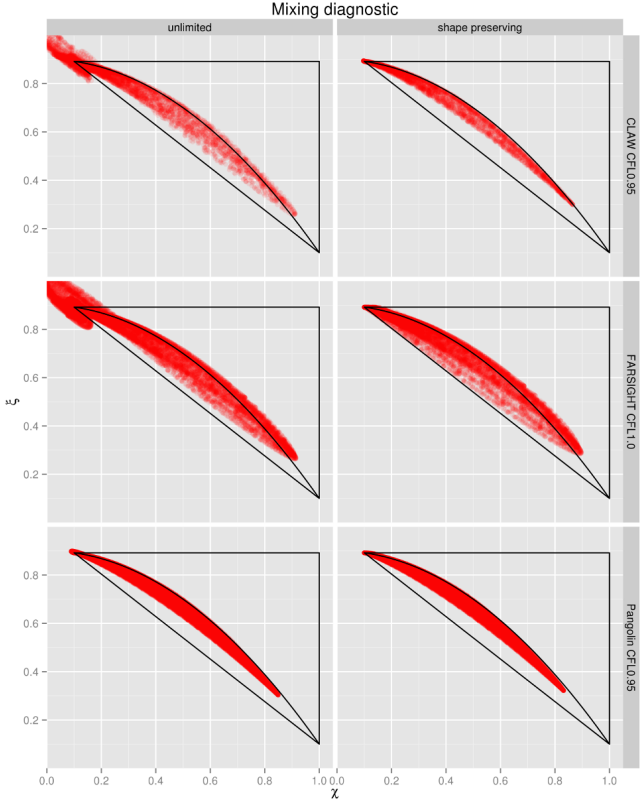

Figure 12. Mixing plots for both unlimited and shape-preserving versions. The resolution is set at $0.75°$ at the Equator, except for Pangolin, which has $0.376°$.

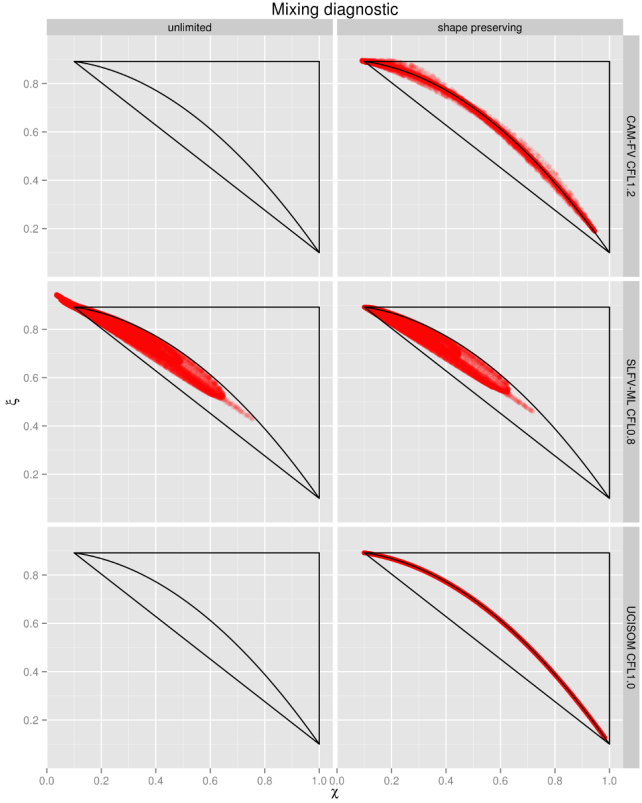

Figure 13. Mixing plots for both unlimited and shape-preserving versions. The resolution is set at $0.75°$ at the Equator, except for Pangolin, which has $0.376°$.

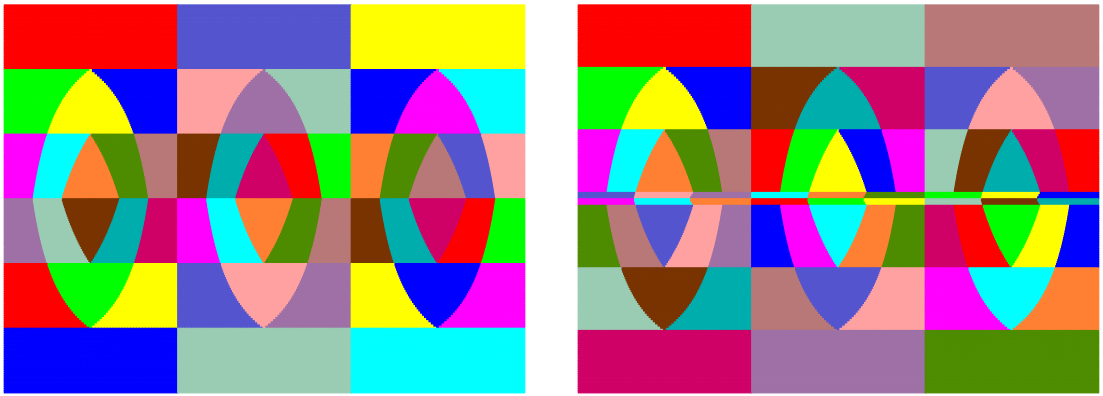

Figure 14. Algebraic mesh partitioning for an optimal case (54 domains, left) and sub-optimal case (74 partitions, left). Each color corresponds to a subdomain (the same color can be used for different subdomains). The grid contains 48600 cells and is shown in latitude-longitude coordinates.

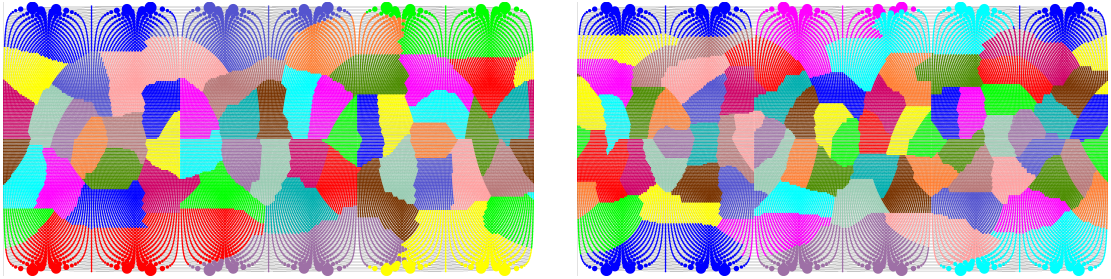

Figure 15. Mesh partitioning computed by the general purpose mesh partitioner Scotch for 54 domains (optimal case for Pangolin, left) and 72 domains (sub-optimal case for Pangolin, right). Each color corresponds to a subdomain (the same color can be used for different partitions). The grid contains 48600 cells and is shown in latitude-longitude coordinates.

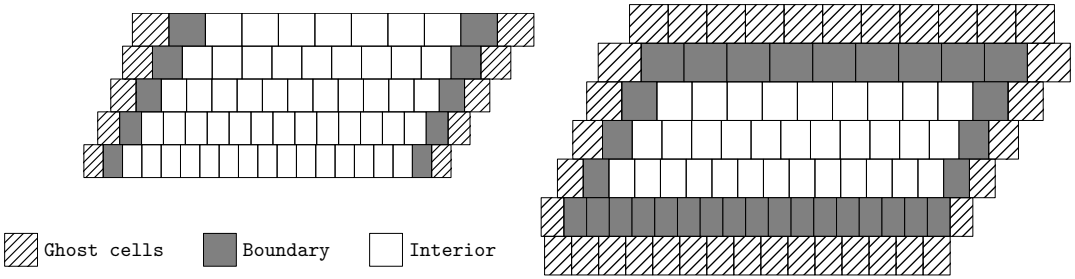

Figure 16. Ghost, boundary and interior cells for zonal (left) and meridional (right) advection.

```
1: subroutine meridional_advection
2:     Starts ratio exchange for ghost cells
3:     Compute zonal gradient on the interior
4:     Wait for end of communications
5:     Compute zonal gradient on the boundary
6:
7:     Starts zonal gradient exchange for ghost cells
8:     Compute meridional gradient on the interior
9:     Wait for end of communications
10:    Compute meridional gradient on the boundary
11:
12:    Starts meridional gradient exchange for ghost cells
13:    Compute meridional fluxes on the interior
14:    Wait for end of communications
15:    Compute meridional fluxes on the boundary
16:
17:    Update all ratios
18: end subroutine
```

```
1: subroutine zonal_advection
2:     Starts ratio exchange for ghost cells
3:     Compute zonal gradient on the interior
4:     Wait for end of communications
5:     Compute zonal gradient on the boundary
6:
7:     Starts zonal gradient exchange for ghost cells
8:     Compute zonal fluxes on the interior
9:     Wait for end of communications
10:    Compute zonal fluxes on the boundary
11:
12:    Update all ratios
13: end subroutine
```

Figure 17. Main steps of the algorithm for zonal (left) and meridional (right) advection.

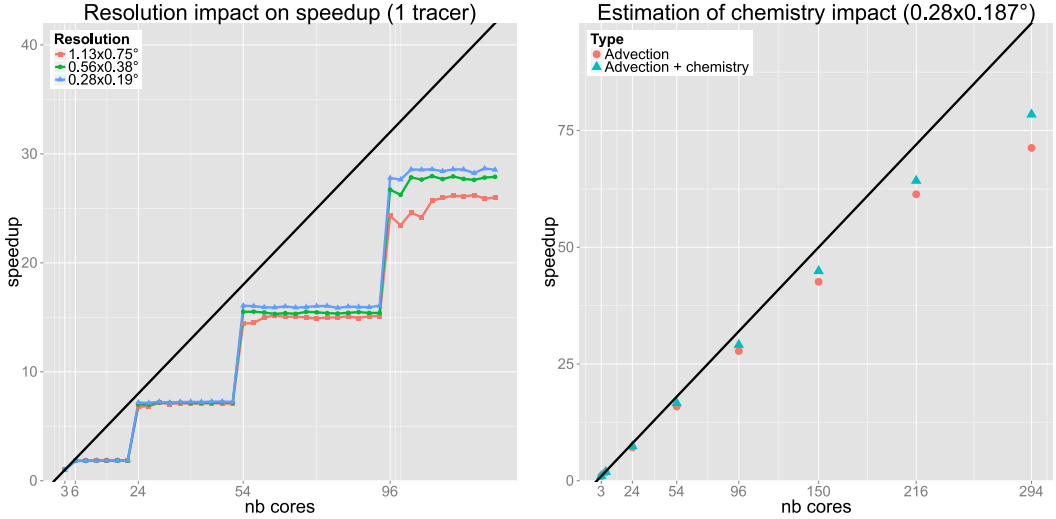

Figure 18. Speedup up to 126 cores (left) and 294 cores (right). The left plot shows the impact of grid resolution for smaller configurations. The right plot shows both the advection performances and an estimation of the chemistry impact on scalability. Resolutions used are: $1.125° \times 0.75°$, $0.56° \times 0.376°$, $0.28° \times 0.188°$. Both figures use non-divergent winds from Sect. 3 over a full period with a CFL of 0.96.

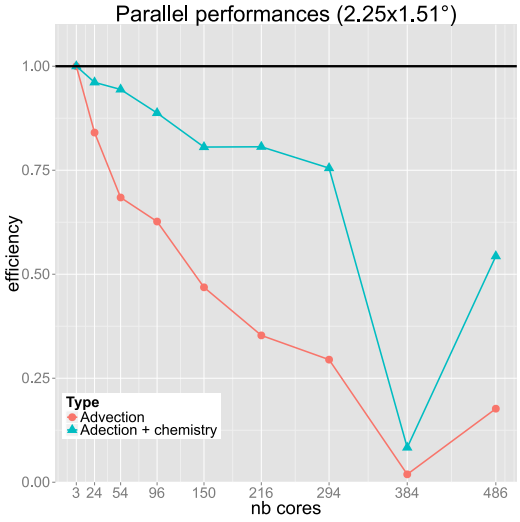

Figure 19. Efficiency up to 486 cores. A resolution of $2.25 \times 1.51°$ was used to examine the limit of the parallelization performances for 2-D advection. For this test, the Airain cluster was used, which has 9504 cores Intel SandyBridge and where each node has 16 cores at 2.7Ghz and 4GB shared by the core.