A Mechanistic Spectral Primitive Equation Model using Pressure Coordinates

R. Saravanan

Department of Applied Mathematics and Theoretical Physics
University of Cambridge
Silver Street, Cambridge CB3 9EW
United Kingdom
E-mail: svn@amtp.cam.ac.uk

June 1992

References

1. Introduction

This document describes the implementation of a very simple 3-dimensional primitive equation model, using pressure coordinates as described by Lorenz (1960). The advantage of using pressure coordinates is that the equations of motion take on a particularly simple form. The main disadvantage is that we cannot easily apply a realistic boundary condition at the lower boundary, i.e., it is not really possible to include the effects of surface topography. The model we are about to describe assumes that the planetary surface is essentially smooth, and that there is sufficient drag acting near the lower boundary to make the details of the lower boundary condition quite unimportant. This model would not be suitable for "realistic" simulation of the Earth's climate (especially in the troposphere), but could be quite useful for mechanistic models of isolated phenomena, and for studying idealized planetary atmospheres.

The model is very flexible in terms of the number of pressure levels, level thicknesses, and zonal versus meridional resolution. The number of levels can be any integer ≥ 2. The levels can have any combination of thicknesses. The horizontal discretization uses spectral truncation, which is usually triangular. For simplicity, the spectral transforms are carried out all the latitudes simultaneously. Any order of triangular truncation is allowed, but very high orders may be impractical in terms of memory requirements, because of the way the spectral transforms are carried out. The range of zonal wavenumbers may be further restricted within the range of triangular truncation, allowing anisotropic integrations with only very few zonal wavenumbers being resolved. The extreme case of axially symmetric horizontal representation can also be handled, although not too efficiently. The model can also handle an arbitrary number of passive tracers, using simple spectral advection.

The model incorporates virtually no physical parameterizations of any kind. But simple forms of scale-selective damping, and Newtonian/Rayleigh friction are provided to facilitate mechanistic simulations.

One of the unusual features of this model is that the vertically integrated divergence over the whole atmosphere is constrained to be identically zero. In other words, divergent shallow water modes with barotropic vertical structure are excluded. This leads to increased computational stability of the model. This also means that the model needs virtually no other boundary conditions, and only interior forcing/damping parameters need to be specified. But this constraint can easily be relaxed to allow "geopotential forcing" at the lower boundary, as would be necessary for models of the middle atmosphere alone. (The necessary modifications are described in Appendix C, although they have not been implemented as yet.)

2. Primitive equations in pressure coordinates

The primitive equations for a dry rotating spherical shallow compressible hydrostatic atmosphere in Lagrangian form may be written in pressure coordinates as (e.g. see Holton, 1979)

$$\frac{d\mathbf{u}}{dt} = -f\mathbf{k}\times\mathbf{u} - \nabla_H\,\Phi + \mathbf{F}_u \tag{1a}$$

$$\frac{d\Theta}{dt} = F_T \tag{1b}$$

$$0 = \nabla_H\cdot\mathbf{u} + \frac{\partial\omega}{\partial p} \tag{1c}$$

$$\frac{\partial\Phi}{\partial\zeta} = -C_p\Theta \tag{1d}$$

where $\zeta = (p/p_s)^\kappa$ denotes an auxiliary vertical coordinate, Θ denotes potential temperature, $\mathbf{F}_u$ denotes mechanical forcing/damping terms, and F_T denotes thermal forcing/damping terms. Otherwise the notation is fairly standard, and the reader is referred to Appendix A for a complete definition of all symbols.

Defining the operators

$$\mathrm{curl_z}(\ ) \equiv (\mathbf{k}\times\nabla_H)\cdot(\ ); \quad \mathrm{div}(\ ) \equiv \nabla_H\cdot(\ )$$

we define the relative vorticity ξ and divergence D as

$$D \equiv \mathrm{div}\,\mathbf{u}; \quad \xi \equiv \mathrm{curl_z}\,\mathbf{u}$$

Then we can write the primitive equations in Eulerian form as follows:

$$\frac{\partial\mathbf{u}}{\partial t} = -\nabla_H\,\Phi - \nabla_H(\frac{1}{2}\mathbf{u}^2) - (f+\xi)\mathbf{k}\times\mathbf{u} - D\mathbf{u} - \frac{\partial(\omega\mathbf{u})}{\partial p} \tag{2a}$$

$$\frac{\partial\Theta}{\partial t} = -\,\mathrm{div}(\Theta\mathbf{u}) - \frac{\partial(\omega\Theta)}{\partial p} \tag{2b}$$

$$\frac{\partial\omega}{\partial p} = -D \tag{2c}$$

$$\frac{\partial\Phi}{\partial\zeta} = -C_p\Theta \tag{2d}$$

Note that we have omitted the forcing/damping terms. We shall continue to ignore these terms in our discussion of spatial discretization. We will re-introduce some of the damping terms later when we discuss the time-marching scheme.

3. Vertical discretization

a. *Vertical grid*

We assume that the domain of the model atmosphere extends upward from a reference pressure level $p = p_s$ ("surface") to the level $p = 0$ ("top" of the atmosphere). Following Lorenz (1960), we divide the pressure-interval $[0, p_s]$ into L (possibly unequal) sub-intervals, each of extent Δp_l.

$$p_s = \sum_l \Delta p_l,\ l = 1, \ldots, L$$

We may then think of various quantities as being defined in the middle of these L sub-intervals, which we refer to as *levels* (or *full-levels*). The levels are numbered, starting from the uppermost-level, as $1, \ldots, L$. The pressure p_l at the full-levels is defined by

$$p_l = \frac{1}{2}\Delta p_l + \sum_{l'=1}^{l-1} \Delta p_{l'}$$

It is then convenient to introduce the concept of *half-levels* which lie at the boundaries of the L pressure sub-intervals. There would be $L - 1$ half-levels which lie in between the L full-levels. These we will refer to as the *interior* half-levels. There would also be two more half-levels, one at the top of the atmosphere, and another at the surface. These we will refer to as the boundary half-levels. The half-levels are numbered, starting from the top, as $\frac{1}{2}, 1 + \frac{1}{2}, \ldots, L - \frac{1}{2}, L + \frac{1}{2}$.

$$p_{l+\frac{1}{2}} = \sum_{l'=1}^{l} \Delta p_{l'}; \qquad p_{\frac{1}{2}} = 0, \quad p_{L+\frac{1}{2}} = p_s.$$

We take the prognostic quantities to be the horizontal velocity $\mathbf{u}_l$ and potential temperature Θ_l defined at each of the L full-levels. The continuity equation (2c) then suggests that it would be natural to define "pressure velocity" $\omega_{l+\frac{1}{2}}$ at the half-levels. Next we introduce the *bar* and *cap* operators:

$$\overline{q}_{l+\frac{1}{2}} \equiv \frac{q_{l+1} + q_l}{2}$$

$$\widehat{q}_{l+\frac{1}{2}} \equiv \frac{q_{l+1} - q_l}{2}$$

where q is some quantity defined at full-levels. (Note that the above operators are not defined at the boundary half-levels)

We then define the vertical flux of prognostic quantities at the interior half-levels as

$$\mathbf{V}_{u,l+\frac{1}{2}} \equiv \omega_{l+\frac{1}{2}} \overline{\mathbf{u}}_{l+\frac{1}{2}}$$

$$V_{T,l+\frac{1}{2}} \equiv \omega_{l+\frac{1}{2}} \overline{\Theta}_{l+\frac{1}{2}}$$

For any vertical flux $V_{q,l+\frac{1}{2}}$, we define the vertical divergence of this flux at the full-levels as

$$\left(\frac{\partial V_q}{\partial p}\right)_l \equiv \frac{V_{q,l+\frac{1}{2}} - V_{q,l-\frac{1}{2}}}{\Delta p_l}$$

We define the *vertical integral* of a quantity q as being $\sum_{l=1}^{L} \Delta p_l \, q_l$, which is essentially a mass-weighted integral. In particular, we note that

$$\sum_{l=1}^{L} \Delta p_l \left(\frac{\partial V_q}{\partial p}\right)_l = V_{q,L+\frac{1}{2}} - V_{q,\frac{1}{2}}$$

We assume that the geopotential Φ_l is also defined at the full-levels. Defining $\zeta_l \equiv (p_l/p_s)^\kappa$, we choose to discretise the hydrostatic equation (2d) as follows:

$$\frac{\Phi_{l+1} - \Phi_l}{\zeta_{l+1} - \zeta_l} \equiv -C_p \overline{\Theta}_{l+\frac{1}{2}}$$

We can then write the vertically discretized version of (2) as

$$\frac{\partial \mathbf{u}_l}{\partial t} = -\nabla_H \Phi_l - \nabla_H E_l - \mathbf{H}_{u,l} \tag{3a}$$

$$\frac{\partial \Theta_l}{\partial t} = -H_{T,l} \tag{3b}$$

$$D_l = -\frac{\omega_{l+\frac{1}{2}} - \omega_{l-\frac{1}{2}}}{\Delta p_l} \tag{3c}$$

$$\widehat{\Phi}_{l+\frac{1}{2}} = -C_p \widehat{\zeta}_{l+\frac{1}{2}} \overline{\Theta}_{l+\frac{1}{2}} \tag{3d}$$

where

$$\mathbf{H}_{u,l} \equiv (f + \xi_l)\mathbf{k} \times \mathbf{u}_l + D_l \mathbf{u}_l + \left(\frac{\partial \mathbf{V}_u}{\partial p}\right)_l \tag{4a}$$

$$H_{T,l} \equiv \mathrm{div}(\Theta_l \mathbf{u}_l) + \left(\frac{\partial V_T}{\partial p}\right)_l \tag{4b}$$

$$E_l \equiv \frac{1}{2}\mathbf{u}_l^2 \tag{4c}$$

The auxiliary quantity $\mathbf{H}_u$ represents a part of the momentum flux divergence, H_T represents the heat flux divergence, and E represents the kinetic energy.

b. Boundary conditions

We choose to set the pressure-velocity ω to zero at the upper and lower boundaries of the atmosphere:

$$\omega_{\frac{1}{2}} = \omega_{L+\frac{1}{2}} = 0$$

This choice then leads to the constraint that the vertically integrated divergence is zero. i.e.

$$\sum_{l=1}^{L} \Delta p_l \, D_l = \omega_{\frac{1}{2}} - \omega_{L+\frac{1}{2}} = 0$$

This choice also allows us to set the vertical fluxes at the boundaries to zero. i.e

$$\mathbf{V}_{u,\frac{1}{2}} = \mathbf{V}_{u,L+\frac{1}{2}} = V_{T,\frac{1}{2}} = V_{T,L+\frac{1}{2}} = 0$$

c. Conservation properties

It should be obvious from inspecting (3) that the vertical discretization preserves the global conservation properties of absolute angular momentum $(\Omega a \cos\phi + u)\cos\phi$ and potential temperature Θ. Now we proceed to show that the global integral of total energy is also conserved (in the absence of forcing/damping). Taking the dot-product of $\mathbf{u}$ with (3a), we obtain

$$\frac{\partial E_l}{\partial t} = -\mathbf{u}_l \cdot \nabla_H \, \Phi_l - \mathbf{u}_l \cdot \nabla_H \, E_l - D{\mathbf{u}_l}^2 - \mathbf{u}_l \cdot \left(\frac{\partial \mathbf{V}_u}{\partial p} \right)_l$$

After some rearranging, this can be written as

$$\frac{\partial E_l}{\partial t} = -\operatorname{div}(E_l \mathbf{u}_l) - \operatorname{div}(\Phi_l \mathbf{u}_l) - \left(\frac{\partial V_E}{\partial p} \right)_l + \Phi_l D_l \tag{5}$$

where $V_{E,l+\frac{1}{2}} \equiv \omega_{l+\frac{1}{2}} \frac{1}{2}(\mathbf{u}_{l+1} \cdot \mathbf{u}_l)$

If we further define $V_{\Phi,l+\frac{1}{2}} \equiv \omega_{l+\frac{1}{2}} \overline{\Phi}_{l+\frac{1}{2}}$ then we get

$$\left(\frac{\partial V_\Phi}{\partial p}\right)_l = -\Phi_l D_l + \frac{\omega_{l+\frac{1}{2}}\widehat{\Phi}_{l+\frac{1}{2}} + \omega_{l-\frac{1}{2}}\widehat{\Phi}_{l-\frac{1}{2}}}{\Delta p_l}$$

If we also define $V_{\bar{\zeta}\,\overline{\Theta},l+\frac{1}{2}} \equiv \omega_{l+\frac{1}{2}}\bar{\zeta}_{l+\frac{1}{2}}\overline{\Theta}_{l+\frac{1}{2}}$ then we get

$$C_p\left(\frac{\partial V_{\bar{\zeta}\,\overline{\Theta}}}{\partial p}\right)_l = C_p\zeta_l\left(\frac{\partial V_T}{\partial p}\right)_l - \frac{\omega_{l+\frac{1}{2}}\widehat{\Phi}_{l+\frac{1}{2}} + \omega_{l-\frac{1}{2}}\widehat{\Phi}_{l-\frac{1}{2}}}{\Delta p_l}$$

This allows us to write the kinetic energy tendency equation (5) as

$$\frac{\partial E_l}{\partial t} = -\operatorname{div}\{(E_l + \Phi_l)\mathbf{u}_l\} - \left(\frac{\partial V_E}{\partial p}\right)_l - \left(\frac{\partial V_\Phi}{\partial p}\right)_l - C_p\left(\frac{\partial V_{\bar{\zeta}\,\overline{\Theta}}}{\partial p}\right)_l + C_p\zeta_l\left(\frac{\partial V_T}{\partial p}\right)_l \tag{6}$$

If we multiply the potential temperature tendency equation (3b) by $C_p\zeta$, we obtain

$$\frac{\partial C_p\zeta_l\Theta_l}{\partial t} = -\operatorname{div}(C_p\zeta_l\Theta_l\mathbf{u}_l) - C_p\zeta_l\left(\frac{\partial V_T}{\partial p}\right)_l \tag{7}$$

From (6) and (7) it is clear that the vertical truncation preserves the global conservation property of the total energy $(E + C_p\zeta\Theta)$.

Another quantity conserved by this vertical truncation is Θ^2. If we multiply (3b) by Θ, we obtain

$$\frac{\partial \frac{1}{2}\Theta_l^2}{\partial t} = -\operatorname{div}(\frac{1}{2}\Theta_l^2\mathbf{u}_l) - \left(\frac{\partial V_{\frac{1}{2}\Theta^2}}{\partial p}\right)_l \tag{8}$$

where $V_{\frac{1}{2}\Theta^2,l+\frac{1}{2}} \equiv \omega_{l+\frac{1}{2}}\frac{1}{2}\Theta_{l+1}\Theta_l$

d. Linearization about reference vertical stratification

For the purposes of the semi-implicit time-stepping scheme that we will be using, it is necessary to express the Θ distribution in terms of deviations from some reference vertical profile $\Theta^R(p)$. We express $\Theta(\lambda, \phi, p, t)$ as

$$\Theta = \Theta^R(p) + \Theta'(\lambda, \phi, p, t)$$

Then we also decompose the heat fluxes as follows:

$$
\begin{aligned}
H_T &= H_{T^R} + H_{T'} \\
H_{T',l} &= \mathrm{div}(\Theta'_l \mathbf{u}_l) + \left(\frac{\partial V_{T'}}{\partial p}\right)_l \\
H_{T^R,l} &= \Theta^R_l D_l + \left(\frac{\partial V_{T^R}}{\partial p}\right)_l \\
V_{T',l+\frac{1}{2}} &= \omega_{l+\frac{1}{2}} \overline{\Theta}'_{l+\frac{1}{2}}
\end{aligned}
$$

We can re-express H_{T^R} as

$$
\begin{aligned}
H_{T^R,l} &= \frac{\widehat{\Theta}^R_{l+\frac{1}{2}} \omega_{l+\frac{1}{2}} + \widehat{\Theta}^R_{l-\frac{1}{2}} \omega_{l-\frac{1}{2}}}{\Delta p_l} \\
&= \sum_{l'=1}^{L-1} (\delta_{l',l} + \delta_{l',l-1}) \widehat{\Theta}^R_{l'+\frac{1}{2}} \omega_{l'+\frac{1}{2}}
\end{aligned}
$$

(Here we have used the fact that $\omega_{\frac{1}{2}} = \omega_{L+\frac{1}{2}} = 0$.)

Defining a column vector of length $L-1$: $\vec{\omega} = (\omega_{1+\frac{1}{2}}, \ldots, \omega_{L-\frac{1}{2}})$, and a column vector of length L: $\vec{H}_{T^R} = (H_{T^R,1}, \ldots, H_{T^R,L})$, we can write the above equation in matrix form as

$$
\vec{H}_{T^R} = M_{\omega \to H}\, \vec{\omega}
$$

where $M_{\omega \to H}$ is an $L \times (L-1)$ matrix defined by

$$
[M_{\omega \to H}]_{l,l'} = \begin{cases} \frac{\widehat{\Theta}^R_{l'+\frac{1}{2}}}{\Delta p_l} & \text{if } l' = l \text{ or } l' = l-1; \\ 0 & \text{otherwise.} \end{cases}
$$

$$
M_{\omega \to H} = \begin{pmatrix}
\frac{\widehat{\Theta}^R_{1+\frac{1}{2}}}{\Delta p_1} & 0 & 0 & \cdots & 0 & 0 \\
\frac{\widehat{\Theta}^R_{1+\frac{1}{2}}}{\Delta p_2} & \frac{\widehat{\Theta}^R_{2+\frac{1}{2}}}{\Delta p_2} & 0 & \cdots & 0 & 0 \\
0 & \frac{\widehat{\Theta}^R_{2+\frac{1}{2}}}{\Delta p_3} & \frac{\widehat{\Theta}^R_{3+\frac{1}{2}}}{\Delta p_3} & \cdots & 0 & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\
0 & 0 & 0 & \cdots & \frac{\widehat{\Theta}^R_{L-1-\frac{1}{2}}}{\Delta p_{L-1}} & \frac{\widehat{\Theta}^R_{L-\frac{1}{2}}}{\Delta p_{L-1}} \\
0 & 0 & 0 & \cdots & 0 & \frac{\widehat{\Theta}^R_{L-\frac{1}{2}}}{\Delta p_L}
\end{pmatrix}
$$

Next we note that

$$\omega_{l'+\frac{1}{2}} = -D_{l'}\Delta p_{l'} + \omega_{l'-\frac{1}{2}} = -\sum_{l''=1}^{l'} \Delta p_{l''} D_{l''}$$

Defining a column vector of length L: $\vec{D} = (D_1, \ldots, D_L)$, we can rewrite the above equation in matrix form as follows:

$$\vec{\omega} = M_{D\to\omega}\, \vec{D}$$

where $M_{D\to\omega}$ is an $(L-1)\times L$ matrix defined by

$$[M_{D\to\omega}]_{l',l''} = \begin{cases} -\Delta p_{l''} & \text{if } l'' \le l'; \\ 0 & \text{otherwise.} \end{cases}$$

$$M_{D\to\omega} = \begin{pmatrix} -\Delta p_1 & 0 & 0 & \cdots & 0 & 0 \\ -\Delta p_1 & -\Delta p_2 & 0 & \cdots & 0 & 0 \\ -\Delta p_1 & -\Delta p_2 & -\Delta p_3 & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ -\Delta p_1 & -\Delta p_2 & -\Delta p_3 & \cdots & -\Delta p_{L-1} & 0 \end{pmatrix}$$

This allows us to define the compound matrix $M_{D\to H} = M_{\omega\to H}\, M_{D\to\omega}$.

e. Vorticity/divergence form

Rather than work with the discretised momentum equations, we prefer to work with the time-tendency equations for vorticity and divergence. This is motivated by the fact that when representing the horizontal variation of quantities using spherical harmonics, it is much easier to deal with scalars (such as vorticity) than with components of vectors. We apply the div and $\mathrm{curl_z}$ operators to (3a) to obtain the following prognostic equations

$$\frac{\partial \xi_l}{\partial t} = -\,\mathrm{curl_z}\,\mathbf{H}_{u,l} \tag{9a}$$

$$\frac{\partial D_l}{\partial t} = -\,\mathrm{div}\,\mathbf{H}_{u,l} - \nabla_H^2\, E_l - \nabla_H^2\, \Phi_l \tag{9b}$$

Since the quantities D_l are not all independent, it is convenient to work with the $L-1$ independent quantities $\widehat{D}_{l+\frac{1}{2}}$ at the interior half-levels. We can express D in terms of $\widehat{D}$ as follows:

$$D_l = 2\widehat{D}_{l-\frac{1}{2}} + D_{l-1} = \sum_{l'=1}^{l-1} 2\widehat{D}_{l'+\frac{1}{2}} + D_1$$

Now we compute the vertically integrated divergence

$$\sum_{l'=1}^{L} D_{l'} \Delta p_{l'} = \sum_{l'=1}^{L-1} 2\widehat{D}_{l'+\frac{1}{2}} (p_{L+\frac{1}{2}} - p_{l'+\frac{1}{2}}) + D_1 \, p_{L+\frac{1}{2}} = 0$$

This gives the following expression for D_1 in terms of $\widehat{D}$

$$D_1 = \sum_{l'=1}^{L-1} 2\widehat{D}_{l'+\frac{1}{2}} \left(\frac{p_{l'+\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1 \right)$$

Substituting back in the expression for D_l, we obtain

$$D_l = \sum_{l'=1}^{l-1} 2\widehat{D}_{l'+\frac{1}{2}} + \sum_{l'=1}^{L-1} 2\widehat{D}_{l'+\frac{1}{2}} \left(\frac{p_{l'+\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1 \right)$$

or

$$D_l = \sum_{l'=1}^{L-1} 2 \left(\frac{p_{l'+\frac{1}{2}}}{p_{L+\frac{1}{2}}} \right) \widehat{D}_{l'+\frac{1}{2}} - \sum_{l'=l}^{L-1} 2\widehat{D}_{l'+\frac{1}{2}}$$

Defining a column vector of length $L-1$: $\vec{\widehat{D}} = (\widehat{D}_{1+\frac{1}{2}}, \ldots, \widehat{D}_{L-\frac{1}{2}})$, we can write the above relation in matrix form as

$$\vec{D} = M_{\widehat{D}\to D} \, \vec{\widehat{D}}$$

where $M_{\widehat{D}\to D}$ is an $L \times (L-1)$ matrix defined by

$$\left[M_{\widehat{D}\to D} \right]_{l,l'} = \begin{cases} 2 \left(\frac{p_{l'+\frac{1}{2}}}{p_{L+\frac{1}{2}}} \right) & \text{if } l' < l; \\ 2 \left(\frac{p_{l'+\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1 \right) & \text{if } l' \geq l. \end{cases}$$

$$M_{\widehat{D}\to D} = \begin{pmatrix} 2\Big(\frac{p_{1+\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1\Big) & 2\Big(\frac{p_{2+\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1\Big) & \cdots & 2\Big(\frac{p_{L-\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1\Big) \\ 2\Big(\frac{p_{1+\frac{1}{2}}}{p_{L+\frac{1}{2}}}\Big) & 2\Big(\frac{p_{2+\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1\Big) & \cdots & 2\Big(\frac{p_{L-\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1\Big) \\ 2\Big(\frac{p_{1+\frac{1}{2}}}{p_{L+\frac{1}{2}}}\Big) & 2\Big(\frac{p_{2+\frac{1}{2}}}{p_{L+\frac{1}{2}}}\Big) & \cdots & 2\Big(\frac{p_{L-\frac{1}{2}}}{p_{L+\frac{1}{2}}} - 1\Big) \\ \vdots & \vdots & \ddots & \vdots \\ 2\Big(\frac{p_{1+\frac{1}{2}}}{p_{L+\frac{1}{2}}}\Big) & 2\Big(\frac{p_{2+\frac{1}{2}}}{p_{L+\frac{1}{2}}}\Big) & \cdots & 2\Big(\frac{p_{L-\frac{1}{2}}}{p_{L+\frac{1}{2}}}\Big) \end{pmatrix}$$

The above $L \times (L-1)$ matrix has a generalized left inverse which may be obtained as follows:

$$\widehat{D}_{l'+\frac{1}{2}} = \frac{1}{2} D_{l'+1} - \frac{1}{2} D_{l'} = \sum_{l''=1}^{L} \frac{1}{2} (\delta_{l'',l'+1} - \delta_{l'',l'}) D_{l''}$$

In matrix form, this may be written as

$$\vec{\widehat{D}} = M_{D\to\widehat{D}}\, \vec{D}$$

where $M_{D\to\widehat{D}}$ is an $(L-1)\times L$ matrix defined by

$$\Big[M_{D\to\widehat{D}}\Big]_{l',l''} = \begin{cases} -\frac{1}{2} & \text{if } l'' = l'; \\ +\frac{1}{2} & \text{if } l'' = l'+1; \\ 0 & \text{otherwise.} \end{cases}$$

$$M_{D\to\widehat{D}} = \begin{pmatrix} -\frac{1}{2} & +\frac{1}{2} & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{2} & +\frac{1}{2} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1}{2} & +\frac{1}{2} \end{pmatrix}$$

$M_{D\to\widehat{D}}$ has the following property: $M_{D\to\widehat{D}}\, M_{\widehat{D}\to D} = 1$.

Next we write the hydrostatic relation (3d) as

$$\widehat{\Phi}_{l'+\frac{1}{2}} = -C_p \widehat{\zeta}_{l'+\frac{1}{2}} \overline{\Theta}_{l'+\frac{1}{2}} = -C_p \sum_{l''=1}^{L} \frac{1}{2}\, \widehat{\zeta}_{l'+\frac{1}{2}} (\delta_{l'',l'+1} + \delta_{l'',l'}) \Theta_{l''}$$

Defining a column vector of length $L-1$: $\vec{\widehat{\Phi}} = (\widehat{\Phi}_{1+\frac{1}{2}}, \ldots, \widehat{\Phi}_{L-\frac{1}{2}})$, and a column vector $\vec{\Theta}$ of length L, we can write the above equation in matrix form as

$$\vec{\widehat{\Phi}} = M_{\Theta\to\widehat{\Phi}}\,\vec{\Theta}$$

where $M_{\Theta\to\widehat{\Phi}}$ is an $(L-1)\times L$ matrix defined by

$$\left[M_{\Theta\to\widehat{\Phi}}\right]_{l',l''} = \begin{cases} -\frac{1}{2}C_p\widehat{\zeta}_{l'+\frac{1}{2}} & \text{if } l''=l'+1 \text{ or } l''=l'; \\ 0 & \text{otherwise.} \end{cases}$$

$$M_{\Theta\to\widehat{\Phi}} = \begin{pmatrix} -\frac{1}{2}C_p\widehat{\zeta}_{1+\frac{1}{2}} & -\frac{1}{2}C_p\widehat{\zeta}_{1+\frac{1}{2}} & 0 & \cdots & 0 & 0 \\ 0 & -\frac{1}{2}C_p\widehat{\zeta}_{2+\frac{1}{2}} & -\frac{1}{2}C_p\widehat{\zeta}_{2+\frac{1}{2}} & \cdots & 0 & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & 0 & \cdots & -\frac{1}{2}C_p\widehat{\zeta}_{L-\frac{1}{2}} & -\frac{1}{2}C_p\widehat{\zeta}_{L-\frac{1}{2}} \end{pmatrix}$$

4. Time integration

a. Semi-implicit leap-frog scheme

We will use a leap-frog time-stepping scheme, but the terms associated with gravity waves will be treated implicitly. Using superscripts to denote the time-step, we write (for a generic prognostic quantity q)

$$q^{n+1} = q^{n-1} + 2\Delta t\left(\frac{\partial q}{\partial t}\right)^n \tag{10}$$

where Δt denotes the time-step, and $(\partial q/\partial t)^n$ denotes an appropriately defined time-tendency for q.

Let α (such that $0 \le \alpha \le 1$) denote the "implicitness" of the time-stepping scheme. We add scale-selective (∇_H^8) damping terms to (3), and define the time-tendencies as follows

$$\left(\frac{\partial\vec{\xi}}{\partial t}\right)^n = -\gamma_u\,\nabla_H^8\left(\alpha\vec{\xi}^{n+1} + (1-\alpha)\vec{\xi}^{n-1}\right) - \mathrm{curl_z}\,\vec{\mathbf{H}}_u^n \tag{11a}$$

$$\left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n = -\nabla_H^2\,M_{\Theta\to\widehat{\Phi}}\left(\alpha\vec{\Theta}^{n+1} + (1-\alpha)\vec{\Theta}^{n-1}\right) - \gamma_u\,\nabla_H^8\left(\alpha\vec{\widehat{D}}^{n+1} + (1-\alpha)\vec{\widehat{D}}^{n-1}\right) + M_{D\to\widehat{D}}(-\operatorname{div}\vec{\mathbf{H}}_u^n - \nabla_H^2\,\vec{E}^n) \tag{11b}$$

$$\left(\frac{\partial\vec{\Theta}}{\partial t}\right)^n = -M_{\widehat{D}\to H}\left(\alpha\vec{\widehat{D}}^{n+1} + (1-\alpha)\vec{\widehat{D}}^{n-1}\right) - \gamma_T\,\nabla_H^8\left(\alpha\vec{\Theta}^{n+1} + (1-\alpha)\vec{\Theta}^{n-1}\right) - \vec{H}_{T'}^n \tag{11c}$$

where $\vec{\xi}$, $\vec{\mathbf{H}}_u$, $\vec{E}$, and $\vec{H}_{T'}$ are vectors of length L, and $M_{\widehat{D}\to H} = M_{D\to H}\, M_{\widehat{D}\to D}$.

It is convenient to define the "explicit time-tendencies"

$$\begin{aligned}
\vec{X}_\xi^n &= -\operatorname{curl}_z \vec{\mathbf{H}}_u^n - \gamma_u\, \nabla_H^8\, \vec{\xi}^{n-1} \\
\vec{X}_D^n &= -\operatorname{div} \vec{\mathbf{H}}_u^n - \nabla_H^2\, \vec{E}^n - \nabla_H^2\, M_{\Theta\to\Phi} \vec{\Theta}^{n-1} - \gamma_u\, \nabla_H^8\, \vec{D}^{n-1} \\
\vec{X}_T^n &= -\vec{H}_{T'}^n - M_{D\to H} \vec{D}^{n-1} - \gamma_T\, \nabla_H^8\, \vec{\Theta}^{n-1}
\end{aligned}$$

where $M_{\Theta\to\Phi} = M_{\widehat{D}\to D} M_{\Theta\to\widehat{\Phi}}$.

This allows us to rewrite the time-tendency equations (11) as

$$\left(\frac{\partial\vec{\xi}}{\partial t}\right)^n = -\gamma_u\, \nabla_H^8\, \alpha(\vec{\xi}^{n+1} - \vec{\xi}^{n-1}) + \vec{X}_\xi^n \tag{12a}$$

$$\left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n = -\nabla_H^2\, M_{\Theta\to\widehat{\Phi}}\, \alpha(\vec{\Theta}^{n+1} - \vec{\Theta}^{n-1}) - \gamma_u\, \nabla_H^8\, \alpha(\vec{\widehat{D}}^{n+1} - \vec{\widehat{D}}^{n-1}) + M_{D\to\widehat{D}} \vec{X}_D^n \tag{12b}$$

$$\left(\frac{\partial\vec{\Theta}}{\partial t}\right)^n = -M_{\widehat{D}\to H} \alpha(\vec{\widehat{D}}^{n+1} - \vec{\widehat{D}}^{n-1}) - \gamma_T\, \nabla_H^8\, \alpha(\vec{\Theta}^{n+1} - \vec{\Theta}^{n-1}) + \vec{X}_T^n \tag{12c}$$

Assuming that we expand our variables in terms of eigenfunctions of the ∇_H^2 operator, we can define the following "implicit correction factors"

$$\begin{aligned}
\gamma_u^{corr} &\equiv \frac{1}{1 + \alpha 2\Delta t \gamma_u\, \nabla_H^8} \\
\gamma_T^{corr} &\equiv \frac{1}{1 + \alpha 2\Delta t \gamma_T\, \nabla_H^8}
\end{aligned}$$

We then use (10) to rewrite (12) as

$$\left(\frac{\partial\vec{\xi}}{\partial t}\right)^n = \gamma_u^{corr} \vec{X}_\xi^n \tag{13a}$$

$$\left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n = -\gamma_u^{corr}\, \nabla_H^2\, M_{\Theta\to\widehat{\Phi}}(\alpha 2\Delta t) \left(\frac{\partial\vec{\Theta}}{\partial t}\right)^n + \gamma_u^{corr} M_{D\to\widehat{D}} \vec{X}_D^n \tag{13b}$$

$$\left(\frac{\partial\vec{\Theta}}{\partial t}\right)^n = -\gamma_T^{corr} M_{\widehat{D}\to H}(\alpha 2\Delta t) \left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n + \gamma_T^{corr} \vec{X}_T^n \tag{13c}$$

We can now solve (13b,c) for the $\widehat{D}$ tendency

$$\left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n = (\alpha 2\Delta t)^2\gamma_u^{corr}\gamma_T^{corr}\,\nabla_H^2\,M_{\Theta\to\widehat{\Phi}}M_{\widehat{D}\to H}\left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n - \gamma_u^{corr}\gamma_T^{corr}\,\nabla_H^2\,M_{\Theta\to\widehat{\Phi}}(\alpha 2\Delta t)\vec{X}_T^n + \gamma_u^{corr}M_{D\to\widehat{D}}\vec{X}_D^n$$

or

$$\left(\frac{\partial\vec{\widehat{D}}}{\partial t}\right)^n = \left[1-(\alpha 2\Delta t)^2\gamma_u^{corr}\gamma_T^{corr}\,\nabla_H^2\,M_{\Theta\to\widehat{\Phi}}M_{\widehat{D}\to H}\right]^{-1} \left(-\gamma_u^{corr}\gamma_T^{corr}\,\nabla_H^2\,M_{\Theta\to\widehat{\Phi}}(\alpha 2\Delta t)\vec{X}_T^n + \gamma_u^{corr}M_{D\to\widehat{D}}\vec{X}_D^n\right) \tag{14}$$

Using the identity $M_{D\to\widehat{D}}\,M_{\widehat{D}\to D}=1$, we can write

$$M_{\Theta\to\widehat{\Phi}} = M_{D\to\widehat{D}}M_{\widehat{D}\to D}M_{\Theta\to\widehat{\Phi}} = M_{D\to\widehat{D}}M_{\Theta\to\Phi}$$

which allows us to rewrite (14) as

$$\left(\frac{\partial\vec{D}}{\partial t}\right)^n = M_{\widehat{D}\to D}\left[1-(\alpha 2\Delta t)^2\gamma_u^{corr}\gamma_T^{corr}\,\nabla_H^2\,M_{\Theta\to\widehat{\Phi}}\,M_{\widehat{D}\to H}\right]^{-1}M_{D\to\widehat{D}} \left(-\gamma_u^{corr}\gamma_T^{corr}\,\nabla_H^2\,M_{\Theta\to\Phi}(\alpha 2\Delta t)\vec{X}_T^n + \gamma_u^{corr}\vec{X}_D^n\right) \tag{15}$$

Equations (13a,c), along with (15) constitute the prognostic equations of the time-stepping scheme.

b. Time-filtering

Since the leap-frog scheme produces a computational mode, a Robert time-filter (e.g. see Haltiner and Williams, p. 147) is used to damp it out. The leap-frog scheme as described in the previous section is modified as follows (for a generic prognostic quantity q):

$$q^{n+1} = q_*^{n-1} + 2\Delta t\left(\frac{\partial q}{\partial t}\right)^n \tag{16}$$

where q_* denotes the filtered value of q obtained as follows

$$q_*^n = (1-2\epsilon)q^n + \epsilon(q^{n+1} + q^{n-1})$$

Here ϵ is a small positive fraction, typically of $O(0.01)$ for meteorological applications.

5. Horizontal discretization

The model uses a truncated series of spherical harmonics to represent the horizontal variation of a quantity $q(\lambda, \phi, p, t)$. The spherical harmonics $Y_{m,n}$ are defined as

$$Y_{m,n}(\lambda, \mu) = P_{m,n}(\mu)e^{im\lambda}$$

where $\mu \equiv \sin\phi$, and $P_{m,n}$ are the associated Legendre polynomials. The normalization conditions are

$$\frac{1}{2}\int_{-1}^{1} d\mu\, P_{m,n}P_{m,n'} = \delta_{n,n'}$$
$$\frac{1}{4\pi}\int_{-1}^{1} d\mu \int_{0}^{2\pi} d\lambda\, Y_{m,n}^* Y_{m',n'} = \delta_{m,m'}\delta_{n,n'}$$

Some useful expression for $P_{m,n}$ are

$$P_{0,0} = 1; \qquad P_{0,1} = \sqrt{3}\mu; \qquad P_{0,2} = \frac{\sqrt{5}}{2}(3\mu^2 - 1)$$

We expand q in terms of the spherical harmonics as

$$q(\lambda, \mu, p, t) = \sum_{n=0}^{N} \sum_{m=-\min(n,M)}^{+\min(n,M)} q_{m,n}(p,t)\, Y_{m,n}(\lambda, \mu) \tag{17a}$$

$$q_{m,n} = \frac{1}{4\pi}\int_{-1}^{1} d\mu \int_{0}^{2\pi} d\lambda\, Y_{m,n}^*\, q(\lambda, \mu, p, t) \tag{17b}$$

where M and N determine the order of the zonal and the meridional truncation respectively. The choice $M = N$ corresponds to *triangular* spectral truncation of order N. Note that for real q,

we have the property that $q_{-m,n} = (-1)^m q^*_{m,n}$. Therefore, we only need to store the values of $q_{m,n}$ for $m \geq 0$.

For all the linear terms in the prognostic equations discussed in the previous section, it is quite straightforward to obtain a separate time-tendency equation for each spectral coefficient $q_{m,n}$. Since $Y_{m,n}$ are eigenfunctions of the ∇^2_H operator, with eigenvalues $(-n(n+1)/a^2)$, it is very easy to compute the horizontal velocity $\mathbf{u}$ from ξ and D, by using the streamfunction ψ and velocity potential χ defined as follows:

$$\xi = \nabla^2_H \, \psi; \qquad D = \nabla^2_H \, \chi; \qquad \mathbf{u} = \mathbf{k} \times \nabla_H \, \psi + \nabla_H \, \chi$$

The quadratic nonlinear products that occur in the prognostic equations are evaluated using the *transform method* (e.g., see Haltiner and Williams, pp. 193–201). To summarize this method—consider a quadratic product of the form (qr). Given the spectral coefficients $q_{m,n}$ and $r_{m,n}$, we wish to evaluate the spectral coefficients of the product $(qr)_{m,n}$. We choose a (λ_j, μ_k) longitude/sine-latitude grid in physical space, where $j = 1, \ldots, K_1$, and $k = 1, \ldots, K_2$. We choose the K_1 longitudes to be equally spaced. The K_2 latitudes (referred to as "gaussian" latitudes) are chosen to be gaussian quadrature points for the associated Legendre polynomials. i.e.

$$\frac{1}{2} \int_{-1}^{1} d\mu \, P_{m,n} \, P_{m',n'} = \frac{1}{2} \sum_{k=1}^{K_2} G_k \, P_{m,n}(\mu_k) \, P_{m',n'}(\mu_k), \qquad \text{for all } n, n' \leq N;$$

where G_k are the weights associated with each quadrature point μ_k (cf. Press et al, pp. 121–126). To avoid aliasing, K_1 and K_2 must satisfy the following constraints:

$$K_1 \geq 3M + 1; \qquad K_2 \geq \frac{3N+1}{2}$$

The physical space values $q_{j,k}$ and $r_{j,k}$ at the grid-points (λ_j, μ_k) are computed using (17a). Then the quadratic term (qr) is transformed to spectral space as follows:

$$(qr)_{m,n} = \frac{1}{2K_1} \sum_{k=1}^{K_2} G_k \sum_{j=1}^{K_1} Y^*_{m,n}(\lambda_j, \mu_k) \, q_{j,k} \, r_{j,k}$$

Thus the transforms between spectral and physical representations essentially involve a discrete Legendre transform combined with a discrete Fourier transform.

6. Miscellaneous details

a. Passive tracers

A conserved passive tracer q is assumed to be governed by the equation

$$\frac{\partial q}{\partial t} = -\operatorname{div}(q\mathbf{u}) - \frac{\partial(\omega q)}{\partial p}$$

It is discretised in a manner very similar to the Θ equation, and the prognostic equation is written as

$$\left(\frac{\partial q_l}{\partial t}\right)^n = \gamma_T^{corr}\left[-\operatorname{div}(q_l^n \mathbf{u}_l^n) - \left(\frac{\partial V_q^n}{\partial p}\right)_l - \gamma_T\,\nabla_H^8\, q_l^{n-1}\right]$$

b. Choice of reference stratification

For numerical stability, $\Theta^R(p)$ should preferably be chosen such that the reference values of static stability $\{-(\partial\Theta^R/\partial p)\}$ are typically larger than the values of static stability likely to be encountered during time-integrations with the model (Simmons et al, 1978). In addition to $\Theta^R(p)$, it is useful define a "standard" vertical profile $\Theta^S(p)$. This "standard" profile may be some representation of the globally averaged vertical stratification such as the U.S. Standard Atmosphere. The standard profile Θ^S is then used to define a standard height z^S as follows:

$$gz_L^S = C_p(1-\zeta_L)\Theta_l^S; \qquad gz_l^S = 2C_p\widehat{\zeta}_{l+\frac{1}{2}}\overline{\Theta}^S_{l+\frac{1}{2}} + gz_{l+1}$$

The standard height values are used in the formulation of viscous damping terms.

c. Forcing/damping terms

We had previously introduced the scale-selective ∇_H^8 damping term during our discussion of the semi-implicit scheme. Now we discuss that and other damping terms in some more detail. We redefine the "explicit time-tendencies" as follows:

$$X^n_{\xi,l} = \cdots - \gamma(\nabla_H^2 + \frac{2}{a^2})^4\xi_l^{n-1} - \eta_{u,l}\xi_l^{n-1} + \operatorname{curl}_z\left(\frac{\partial \mathbf{V}^{n-1}_{\nu u}}{\partial p}\right)_l$$

$$X^n_{D,l} = \cdots - \gamma(\nabla_H^2 + \frac{2}{a^2})^4 D_l^{n-1} - \eta_{u,l}D_l^{n-1} + \operatorname{div}\left(\frac{\partial \mathbf{V}^{n-1}_{\nu u}}{\partial p}\right)_l$$

$$X^n_{T,l} = \cdots - \gamma(\nabla^2_H)^4\Theta^{n-1}_l - \eta_{T,l}(\Theta^{n-1}_l - \Theta^M_l) + \frac{1}{\zeta_l}\left(\frac{\partial V^{n-1}_{\nu T}}{\partial p}\right)_l$$

where $(\cdots)$ denotes the adiabatic explicit tendencies, i.e., without any damping at all; γ is the scale-selective ∇^8_H damping coefficient (usually expressed in units of a^8/sec), η_u is a pressure-dependent Rayleigh friction coefficient, η_T is a pressure-dependent Newtonian cooling coefficient. Θ^M is a specified "mean" potential temperature distribution that the Newtonian cooling relaxes Θ back to.

Note that there is a subtle difference in the way that the ∇^8_H damping acts on ξ or D as compared to Θ, i.e., we have made the following substitutions: $\gamma_u \nabla^8_H \rightarrow \gamma(\nabla^2_H + 2a^{-2})^4$ and $\gamma_T \nabla^8_H \rightarrow \gamma(\nabla^2_H)^4$. The above scheme, which is motivated by the properties of horizontal viscous mixing on a sphere, ensures that a state of solid-body rotation is not affected by the horizontal scale-selective damping. Also note that all the damping terms involve values of the prognostic quantity at time step $n-1$. The damping terms are not treated using a leap-frog scheme because it would lead to numerical instabilities.

$\mathbf{V}_{\nu u}$ is the vertical viscous stress, defined as follows:

$$\begin{aligned}
\mathbf{V}_{\nu u,\frac{1}{2}} &= 0 \\
\mathbf{V}_{\nu u,L+\frac{1}{2}} &= -\left(\frac{\frac{1}{2}\Delta p_L}{z^S_L}\right)^2 \nu_L \frac{\mathbf{u}_L}{\frac{1}{2}\Delta p_L} \\
\mathbf{V}_{\nu u,l+\frac{1}{2}} &= \left(\frac{\frac{1}{2}\Delta p_{l+1} + \frac{1}{2}\Delta p_l}{z^S_{l+1} - z^S_l}\right)^2 \nu_l \frac{\mathbf{u}_{l+1} - \mathbf{u}_l}{\frac{1}{2}\Delta p_{l+1} + \frac{1}{2}\Delta p_l}
\end{aligned}$$

where ν_l is the value of "vertical" kinematic viscosity at level $l + \frac{1}{2}$, expressed in m^2/s. Since our vertical discretization is in terms of pressure, we multiply the kinematic viscosity by $(\partial p/\partial z)^2$ when computing the viscous stresses in terms of vertical "shears". Note that ν_L effectively acts like a surface drag coefficient, and ν_0 does not need to be specified because the viscous stress is assumed to vanish at the top of the model.

$V_{\nu T}$ is the vertical diffusive heat "flux", defined as follows:

$$\begin{aligned}
V_{\nu T,\frac{1}{2}} &= \mathbf{V}_{\nu T,1+\frac{1}{2}} \\
V_{\nu T,L+\frac{1}{2}} &= \mathbf{V}_{\nu T,L-\frac{1}{2}} \\
V_{\nu T,l+\frac{1}{2}} &= \left(\frac{\frac{1}{2}\Delta p_{l+1} + \frac{1}{2}\Delta p_l}{z^S_{l+1} - z^S_l}\right)^2 \sigma^{-1}\, \nu_l \frac{T_{l+1} - T_l}{\frac{1}{2}\Delta p_{l+1} + \frac{1}{2}\Delta p_l}
\end{aligned}$$

σ^{-1} is the inverse of the Prandtl number, which is defined as $\sigma \equiv \nu/\kappa$, where κ is the thermal diffusivity (expressed in m^2/s). Thermal diffusion acts on temperature T, not on potential temperature Θ. Note that the temperatures in the uppermost and the lowermost levels are not affected by thermal diffusion, and that σ is assumed to be independent of height.

7. Fortran implementation

a. Overview

The model described so far has been implemented as set of Fortran77 subroutines. The code has been written with simplicity, rather than efficiency, in mind. A simple storage scheme is used for the spectral coefficients, which leads to about 50% redundancy in memory usage. The execution speed should not be too bad, because an effort has been made to ensure that the major "innermost" loops of the spectral transform routines automatically vectorize on vector-processors. One important feature of the time-marching routines in module *prognos* is that they carry out spectral transforms over all the latitudes at the same time. This keeps the code simple and efficient, but the table of Legendre polynomials needed for this procedure becomes rather large at high orders of spectral truncation.

The fortran source code is spread out over the following files:

splib.F, prognos.F, diagnos.F, spcons.h, spgrid.h, sppoly.h, spfftb.h, mcons.h, mgrid.h, tmarch.h, and *diacom.h*

The three "*.F*" files contain Fortran source code interspersed with C preprocessor directives. (On UNIX systems, the *f77* command automatically invokes the C preprocessor on such files). There are three source modules: *splib, prognos,* and *diagnos.* The module *splib* deals exclusively with spectral transforms, and is self-contained (except for the NCAR-FFT routines, which are supplied separately with this distribution). The module *prognos* implements the time-stepping scheme for the model. *Prognos* uses several routines from module *splib*, and a single routine from the NAG library. (In case the NAG routine is not available, a substitute routine is supplied separately with this distribution.) The module *diagnos* computes run-time diagnostics, and uses several routines from module *splib.*

The "*.h*" files are "include" files used by the "*.F*" modules. There are two types of include files— C-parameter include files and Fortran declarative include files. The former contain definitions of C preprocessor parameters, and need to be included only once in the whole file, preferably at the very beginning of the file. The latter type of include files usually contain declarations of Fortran COMMON variables, and would need need to be included in the declarative part of every Fortran function/subroutine that needs to use them.

There are two C-parameter include files: *spcons.h* and *mcons.h*. The file *mcons.h* contains important C preprocessor definitions for module *prognos*. Therefore, any program that uses routines from module *prognos* should begin with the following line—

#include "mcons.h"

Similarly, any program using routines from module *splib* should "include" the file *spcons.h* at the very beginning, unless *mcons.h* has already been included. (Including *mcons.h* automatically causes *spcons.h* to be included.)

The remaining include files contain Fortran declarations. The include file *mgrid.h* contains several useful COMMON variables of the primitive equation model, which are described in the preface to module *prognos*. Similarly, the include file *spgrid.h* contains several useful variables related to the horizontal truncation and the spectral transforms, which are described in the preface to module *splib*. (Including *mgrid.h* automatically causes *spgrid.h* to be included.)

The declarative include file *tmarch.h* contains COMMON variables dealing with the time-stepping scheme, but they would rarely be needed by other programs. The declarative include file *diacom.h* contains COMMON variables dealing with run-time diagnostics. The include files *sppoly.h,* and *spfftb.h* are meant primarily for internal use by module *splib*.

Appendix B contains a Fortran to Symbols "dictionary," which lists the symbolic equivalents of most Fortran variable names.

b. Module splib

Module *splib* contains numerous routines dealing with spectral transforms. Only the most commonly used routines, viz., the ones used by module *prognos*, are described below. To use these routines, the following conventions must be used:

i) A spectral space variable $q_{m,n}$ should be declared as

   COMPLEX QSP(0:M1MAX, 0:N1MAX)

ii) A physical space variable $q_{j,k}$ should be declared as

   REAL QPH(K1MAX, K2MAX)

The major subroutines of module *splib* may be described as follows (where the variables in parantheses denote arguments to the subroutine, q, r denote generic scalars, $\mathbf{u} = u\mathbf{i} + v\mathbf{j}$ denotes a generic vector field, and "$\leftarrow$" denotes the assignment operator):

SPINI(M, N, a) : Initializes the spectral truncation order to (M, N) and the planetary radius to a

ZEROSP$(q_{m,n}, N)$: $q_{m,n} \leftarrow 0$

SPCOPY$(r_{m,n}, q_{m,n}, N) : r_{m,n} \leftarrow q_{m,n}$

DELSQ$(r_{m,n}, q_{m,n}, N) : r_{m,n} \leftarrow \nabla_H^2 \, q_{m,n}$

IDELSQ$(r_{m,n}, q_{m,n}, N) : r_{m,n} \leftarrow {\nabla_H}^{-2} \, q_{m,n}$

PHYSIC$(r_{j,k}, q_{m,n}) : r_{j,k} \leftarrow (q)_{j,k}$

HVELOC$(u_{j,k}, v_{j,k}, \psi_{m,n}, \chi_{m,n}) : \mathbf{u}_{j,k} \leftarrow (\mathbf{k} \times \nabla_H \, \psi + \nabla_H \, \chi)_{j,k}$

SPECTR$(r_{m,n}, q_{j,k}) : r_{m,n} \leftarrow r_{m,n} + (q)_{m,n}$

DIVERG$(r_{m,n}, u_{j,k}, v_{j,k}) : r_{m,n} \leftarrow r_{m,n} + (\mathrm{div}\, \mathbf{u})_{m,n}$

CURLZ$(r_{m,n}, u_{j,k}, v_{j,k}) : r_{m,n} \leftarrow r_{m,n} + (\mathrm{curl_z}\, \mathbf{u})_{m,n}$

In short, SPINI initializes all spectral transform operations, SPCOPY copies spectral representations, and DELSQ/IDELSQ apply the Laplacian/inverse-Laplacian operators respectively to spectral representations. PHYSIC converts spectral representations to physical representations, and HVELOC computes the physical velocity, given the spectral streamfunction and velocity potential. The last three routines (SPECTR, DIVERG, CURLZ) convert from physical to spectral representation, and *accumulate*. ZEROSP should be used to set the spectral representation to zero before accumulation, if necessary. SPECTR converts from physical to spectral representation, DIVERG converts the divergence of a physical space vector field to spectral space, and CURLZ does the same for the $\mathrm{curl_z}$ of a physical space vector field. The divergence and $\mathrm{curl_z}$ are computed through integration by parts (cf. Haltiner and Williams, p. 194).

Module *splib* also provides an error exit routine, SPERR, which takes two string arguments—the name of the calling subroutine and an error message. SPERR simply displays the two strings and then aborts the program.

(In case you are wondering as to why some of the subroutines take N as an argument, when its value has already been specified through a call to SPINI— those subroutines can also take $N + 1$ as an argument, to handle the exceptional case of the spectral representation of a vector component, which is of one meridional order higher than the spectral representation of a scalar. But for ordinary usage, N can simply be thought of as being a redundant argument)

c. Module prognos

Module *prognos* contains subroutines for initializing the planetary parameters and the vertical resolution. It also contains the time-marching and time-filtering routines. Before we describe these routines, it is useful to define the *composite level variable* $\mathbf{P}_l = (P_{1,l}, P_{2,l}, P_{3,l}, \ldots) \equiv (\xi_l, D_l, \Theta_l, \ldots)$, where $(\ldots)$ denotes the concentrations of passive traces (if any). We then use the notation $\vec{\mathbf{P}}$ to

denote the spectral representation of $\mathbf{P}$ at all the levels taken together. To use these routines, the following conventions must be used:

i) A spectral space variable $q_{m,n,l}$ should be declared as

COMPLEX QSP(0:M1MAX, 0:N1MAX, L1MAX)

ii) A physical space variable $q_{j,k,l}$ should be declared as

REAL QPH(K1MAX, K2MAX, L1MAX)

iii) A spectral composite level variable $P_{m,n,i,l}$ should be declared as

COMPLEX PSP(0:M1MAX, 0:N1MAX, NPGQ, L1MAX)

where $i = 1$ denotes ξ, $i = 2$ denotes D, $i = 3$ denotes Θ, and values of $i > 3$ denote passive tracers (if any); NPGQ $\equiv$ 3+ NTRACE, where NTRACE is the number of passive tracers.

The major subroutines of module *prognos* may be described as follows (where the variables in parantheses denote arguments to the subroutine, $\vec{\mathbf{P}}, \vec{\mathbf{Q}}, \vec{\mathbf{R}}$ denote generic composite level variables):

PLINI(Ω, R, C_p, g) : Initializes the planetary parameters

VERINI($L, p_l, p_{l+\frac{1}{2}}, \Theta_l^S, \Theta_l^R, \gamma, \Theta_l^M, \eta_{T,l}, \eta_{u,l}, \sigma^{-1}, \nu_l$) : Initializes the number of levels, level thicknesses, standard temperature profile, and forcing/damping details

DDTINI($\Delta t, \alpha$) : Initializes the semi-implicit time-stepping scheme

DDTPGQ($\vec{\mathbf{Q}}, \vec{\mathbf{P}}^{n-1}, \vec{\mathbf{P}}^{n}$) : $\vec{\mathbf{Q}} \leftarrow \left(\frac{\partial \vec{\mathbf{P}}}{\partial t}\right)^n_{ad.}$. i.e., DDTPGQ computes the "adiabatic" time-tendency (We put "" around *adiabatic* because the effects of scale-selective damping, which is the only damping term that is treated semi-implicitly, are also included.)

DDTFRC($\vec{\mathbf{R}}, \vec{\mathbf{P}}$) : $\vec{\mathbf{R}} \leftarrow$ (forcing terms). i.e. DDTFRC computes $\vec{\mathbf{R}}$, the time-tendency due to mechanistic forcing/damping effects, given the spectral representation of prognostic quantities $\vec{\mathbf{P}}$ at some approprite model time.

DDTIMP($\vec{\mathbf{Q}}$) : $\vec{\mathbf{Q}} \leftarrow$ (implicitly corrected $\vec{\mathbf{Q}}$). i.e. DDTIMP makes implicit corrections to the explicit time-tendency $\vec{\mathbf{Q}}$.

ROBFIL($q^{n-1}_{m,n}, q^{n}_{m,n}, q^{n+1}_{m,n}, \epsilon$) : $q^n_{m,n} \leftarrow (1-2\epsilon)q^n_{m,n} + \epsilon(q^{n+1}_{m,n} + q^{n-1}_{m,n})$. i.e., ROBFIL applies the Robert time-filter.

d. Module diagnos

Module *diagnos* contains subroutines for computing several standard run-time diagnostics, and for writing out history files (in direct-access format). The diagnostics are initialized by calling subroutine DIAINI at the beginning of the sampling period. After that, subroutine DIASTP should

be called once between every time-step in the sampling period (including once before the first time-step, and once after the last time step) to compute the diagnostics. Subroutine DIAEND should be called at the end of the sampling period to write out the diagnostics to a file. Since computing run-time diagnostics is a non-standard operation, the reader is referred to the source code itself for documentation.

e. Compilation and customization

Both *prognos* and *splib* contain Fortran subroutines/functions, but no MAIN programs. They may be concatenated together with a main program to form an executable model. The concatenated source code may be compiled with an *f77* command on UNIX systems (or *cf77* on UNICOS). To customize the model for different resolutions etc., many of the important C preprocessor parameters may be re-defined in the compilation command line itself. The important customization parameters are defined below:

DPRECISION: This parameter, if defined, indicates that the default floating point precision on the machine for REAL variables is equivalent to REAL*8. This parameter is needed because some of the initializations need to be carried out at double precision on 32-bit machines. This could be avoided on 64-bit machines (such as the Cray) by defining this parameter.

UDEF: The modules *splib*, *prognos*, and *diagnos* can be compiled with the *f77 -u* option on UNIX systems, because all variables and functions are declared explicitly before being used. If the MAIN program also follows this convention, the *-u* option could be used to detect bugs arising from the use of undefined variables. But a small number of "dummy" declarations had to be introduced to be able to do this on some UNIX f77 compilers, perhaps because of a bug in the f77 compiler. These dummy declarations generate errors without the *-u* option. The C preprocessor parameter UDEF is provided so that these these "dummy" declarations may be enabled only when needed. Therefore, the UDEF option should be defined if and only if the *f77 -u* option is used to compile the program.

N1MAX: This is the maximum order of triangular truncation (i.e. order of Legendre polynomials). The default value is 21.

M1MAX: This is the maximum value of zonal wavenumber resolved by the truncation. The default value is N1MAX.

K1MAX: This is the number of equally spaced longitudes in the transform grid [should be the lowest integral power of 2 $\geq$ (3*N1MAX+1)]. The default value is 64.

K2MAX: This is the number of gaussian latitudes in the transform grid [should be the lowest odd integer $\geq$ (3*N1MAX+1)/2]. The default value is 33. (Note: If K2MAX is an odd number, then K2 is always chosen to be an odd number, so that the equator is one of the gaussian latitudes)

[Other typical choices of (N1MAX, M1MAX, K1MAX, K2MAX) could be (10, 10, 32, 17) or (21, 21, 64, 33) or (42, 42, 128, 65) or (85, 85, 256, 129)]

L1MAX: This is the maximum number of pressure-levels (≥ 2). The default value is 20.

KL0MAX: No. of vector elements for FFT operations. The default value is K2MAX. Defining this parameter to be the maximum of either K2MAX or L1MAX allows transforms to be optionally vectorized over levels, rather than latitudes, using subroutines VPHYS and VSPEC of module *splib*.

NTRACE: This is the number of passive tracers (≥ 0). The default value is 0.

FIXTRUNC: This option, if defined, fixes the horizontal/vertical resolution of the model. i.e. The number of levels and the spectral truncation are fixed. By default, this option is not defined, and the resolution is allowed to vary within the limits of available storage. But by defining FIXTRUNC, one can force the counts of many DO loops to become constants, and this may help improve speed of execution (but there are no guarantees).

The next two C-preprocessor parameters, COMPLEXOP and COMPACTPOLY, affect module *splib* only. They control the details of the arithmetic operations involved in carrying out the Legendre transforms. Depending on the machine, and the compiler, defining COMPLEXOP, or COMPACTPOLY, may speed up the transforms. Defining COMPACTPOLY will certainly save a lot on table storage space.

COMPLEXOP: This option, if defined, indicates that complex arithmetic operations may be used during the Legendre transform operations. The default is to use real arithmetic operations only.

COMPACTPOLY: This option, if defined, indicates that the Legendre polynomial tables should be stored in a compact form, to minimize memory requirements. By default, the polynomial tables are stored in a "twinned" form, for efficient real arithmetic operations.

Notes:

1. Defining COMPLEXOP automatically causes COMPACTPOLY to be defined.
2. Defining COMPACTPOLY without defining COMPLEXOP causes polynomial tables to be stored with the n-index, rather than the m-index, varying most rapidly. The order of the two innermost loops in the Legendre transform is also reversed. This may be more efficient for non-triangular truncations with very few zonal wavenumbers.

e.g. One may use the commands

cat main.F prognos.F splib.F > model.F

f77 -O -DL1MAX=5 -DN1MAX=42 -DK1MAX=128 -DK2MAX=65 model.F vfftpk.f -lnag

to compile a version of the model with upto 5 pressure levels and horizontal truncation upto T42, on a Sun workstation. The first line of the main program *main.F* should be *#include "mcons.h"*. The file "vfftpk.f" contains the vectorizable NCAR-FFT routines, and the "-lnag" option invokes the NAG library. If the NAG library is not available, it may be substituted with routines from the LINPACK library. A module called *linpack.F*, containing selected LINPACK routines, is provided to simulate the NAG routines used by module *prognos*. This module may be compiled separately to form *linpack.o*, and the *-lnag* option on the above compile command line replaced with *linpack.o*.

One may use the commands

cat main.F prognos.F splib.F > model.F

cf77 -DDPRECISION -DCOMPACTPOLY -DFIXTRUNC -DM1MAX=0 -DK1MAX=1 -DL1MAX=20 ■
model.F vfftpk.f -lnag

to compile a $N = 21$ axisymmetric version of the model, with the Legendre polynomials stored in a compact form, with 20 pressure levels and fixed truncation, on a Cray running UNICOS.

f. Initialization and time-marching

First, the horizontal truncation should be initialized through a call to routine SPINI. Then the planetary parameters should be initialized by calling PLINI. This should be followed by a call to VERINI to initialize the vertical resolution, and the forcing/damping parameters.

The time-marching sequence is initialized by a call to DDTINI. If one already has available the values of prognostic variables $(\vec{\mathbf{P}}^{n-1}, \vec{\mathbf{P}}^{n})$ at two consecutive time-steps, one may use the call DDTINI$(\Delta t, \alpha)$ to initialize the semi-implicit leap-frog time-marching, with time-step Δt. Then for each time-step, one would need to execute the following sequence of instructions:

CALL DDTPGQ$(\vec{\mathbf{Q}}, \vec{\mathbf{P}}^{n-1}, \vec{\mathbf{P}}^{n})$

CALL DDTFRC$(\vec{\mathbf{R}}, \vec{\mathbf{P}}^{n-1})$

$\vec{\mathbf{Q}} \leftarrow \vec{\mathbf{Q}} + \vec{\mathbf{R}}$

$\langle$ *Other explicit forcing terms may also be added to* $\vec{\mathbf{Q}}$ *at this point* $\rangle$

CALL DDTIMP$(\vec{\mathbf{Q}})$

$\vec{\mathbf{P}}^{n+1} \leftarrow \vec{\mathbf{P}}^{n-1} + 2\Delta t\, \vec{\mathbf{Q}}$

CALL ROBFIL$(q^{n-1}_{m,n}, q^{n}_{m,n}, q^{n+1}_{m,n}, \epsilon)$ *for each constituent* q *in* $\vec{\mathbf{P}}$

But at the very beginning of model integrations, one may have only the initial condition $\vec{\mathbf{P}}^{0}$ available. In that case, one may "cheat" the time-marching routines by initializing with the call

DDTINI($\frac{1}{2}\Delta t, \alpha$). Then, for the first time-step only, one may execute the following sequence of instructions:

CALL DDTPGQ($\vec{\mathbf{Q}}, \vec{\mathbf{P}}^0, \vec{\mathbf{P}}^0$)

CALL DDTFRC($\vec{\mathbf{R}}, \vec{\mathbf{P}}^0$)

$\vec{\mathbf{Q}} \leftarrow \vec{\mathbf{Q}} + \vec{\mathbf{R}}$

$\langle$ *Other explicit forcing terms may also be added to* $\vec{\mathbf{Q}}$ *at this point* $\rangle$

CALL DDTIMP($\vec{\mathbf{Q}}$)

$\vec{\mathbf{P}}^1 \leftarrow \vec{\mathbf{P}}^0 + \Delta t\, \vec{\mathbf{Q}}$

Appendix A. Symbols and notation

a. Symbols

(Note: The word "horizontal" is used here to denote motions along isobaric surfaces, which are assumed to deviate only very little from the true horizontal)

i, **j**, **k** are the local eastward, northward, and upward unit vectors. S.I. units are used to express almost all the quantities, except for γ, and except where otherwise noted.

t	:	time (in s)
λ	:	longitude (in rad)
ϕ	:	latitude (in rad)
$\mu = \sin\phi$	:	sine-latitude
z	:	height (in m)
p	:	pressure (in Pa)
$\mathbf{u} = u\mathbf{i} + v\mathbf{j}$	:	horizontal velocity (in m/s)
u	:	zonal velocity (in m/s)
v	:	meridional velocity (in m/s)
$\omega = dp/dt$	:	"pressure velocity" (in Pa/s)
T	:	temperature (in K)
a	:	planetary radius (in m)
Ω	:	angular velocity of planetary rotation (in $1/s$)
$f = 2\Omega\sin\phi$	:	the coriolis parameter (in $1/s$)
g	:	gravitational acceleration at planetary surface (in m^2/s)
$\Phi = gz$	:	geopotential height (in m^2/s^2)
R	:	gas constant (per unit mass) for dry air (in $J/\{kgK\}$)
C_p	:	specific heat (per unit mass) at constant pressure (in $J/\{kgK\}$)
C_v	:	specific heat (per unit mass) at constant volume (in $J/\{kgK\}$)
$\kappa = C_p/C_v$	:	a ratio of specific heats
p_s	:	the (constant) reference surface pressure (in Pa)
$\zeta = (p/p_s)^\kappa$	:	an auxiliary vertical coordinate
$\Theta = T(p_s/p)^\kappa$	:	potential temperature (in K)
Θ^R	:	reference potential temperature profile (in K)
Θ^S	:	standard potential temperature profile (in K)
z^S	:	standard height values at pressure levels (in m)
Θ^M	:	"mean" potential temperature profile (in K)
$\xi = \mathrm{curl_z}\,\mathbf{u}$	:	relative vorticity (in $1/s$)
$D = \mathrm{div}\,\mathbf{u}$	:	divergence (in $1/s$)
$\mathbf{P} = (\xi, D, \Theta, \ldots)$	:	composite level variable
$\psi = \nabla_H^{-2}\xi$	:	streamfunction (in m^2/s)

$\chi = \nabla_H^{-2} D$	:	velocity potential (in m^2/s)
$E = \frac{1}{2}\mathbf{u}^2$	:	kinetic energy (per unit mass, in m^2/s^2)
α	:	"implicitness" factor ($0 \leq \alpha \leq 1$)
ϵ	:	Robert filter factor
m	:	zonal wavenumber
n	:	meridional index of spherical harmonic ($\mid m \mid \leq n$)
M	:	zonal wavenumber truncation order ($\mid m \mid \leq M$)
N	:	meridional truncation order ($n \leq N$)
$P_{m,n}(\phi)$	:	associated Legendre polynomial
$Y_{m,n}(\lambda, \phi) = P_{m,n} e^{im\lambda}$	:	spherical harmonic
K_1	:	number of longitudes in the transform grid
K_2	:	number of gaussian-latitudes in the transform grid
G_k	:	gaussian quadrature weight at $\phi = \phi_k$
γ	:	∇_H^8 damping coefficient (in units of a^8/s)
η_u	:	pressure-dependent Rayleigh friction coefficient (in $1/s$)
η_T	:	pressure-dependent Newtonian cooling coefficient (in $1/s$)
ν	:	pressure-dependent "vertical" kinematic viscosity (in m^2/s)
σ	:	Prandtl number (viscosity/thermal conductivity)

b. Operators

Vector quantities are shown in boldface. e.g. $\mathbf{V} = (V_\lambda, V_\phi)$

$$\nabla_H = \mathbf{i}\frac{1}{a\cos\phi}\frac{\partial}{\partial\lambda} + \mathbf{j}\frac{1}{a}\frac{\partial}{\partial\phi}$$

$$\nabla_H^2 = \frac{1}{a^2\cos^2\phi}\frac{\partial^2}{\partial\lambda^2} + \frac{1}{a^2\cos\phi}\frac{\partial}{\partial\phi}\cos\phi\frac{\partial}{\partial\phi}$$

$$\mathrm{div}\,\mathbf{V} = \frac{1}{a\cos\phi}\frac{\partial V_\lambda}{\partial\lambda} + \frac{1}{a\cos\phi}\frac{\partial}{\partial\phi}\cos\phi V_\phi$$

$$\mathrm{curl_z}\,\mathbf{V} = (\mathbf{k}\times\nabla_H)\cdot\mathbf{V} = \frac{1}{a\cos\phi}\frac{\partial V_\phi}{\partial\lambda} - \frac{1}{a\cos\phi}\frac{\partial}{\partial\phi}\cos\phi V_\lambda$$

Appendix B. Fortran to Symbols dictionary

A0	$\rightarrow$	a
A0INV	$\rightarrow$	a^{-1}
A0Q	$\rightarrow$	a^2
A0QINV	$\rightarrow$	a^{-2}
CIM	$\rightarrow$	$i\,m$
COSINV	$\rightarrow$	$1/\cos\phi_k$
COSPHI	$\rightarrow$	$\cos\phi_k$
CP	$\rightarrow$	C_p
D2DCAP	$\rightarrow$	$M_{D\rightarrow\widehat{D}}$
D2TT	$\rightarrow$	$-M_{D\rightarrow H} = -M_{\omega\rightarrow H}\, M_{D\rightarrow\omega}$
D2W	$\rightarrow$	$M_{D\rightarrow\omega}$
DCAP2D	$\rightarrow$	$M_{\widehat{D}\rightarrow D}$
DEL8DF	$\rightarrow$	γ
DP	$\rightarrow$	Δp_l
DPGQSP	$\rightarrow$	$\partial\vec{\mathbf{P}}/\partial t$
DT	$\rightarrow$	Δt
DTFAC	$\rightarrow$	$\alpha 2\Delta t$
F0	$\rightarrow$	2Ω
FSP01	$\rightarrow$	$2\Omega/\sqrt{3}$
G	$\rightarrow$	$\frac{1}{2}G_k$ $\quad\langle$note factor of $\frac{1}{2}\rangle$
G0	$\rightarrow$	g
IMPCOR	$\rightarrow$	implicit correction matrix
IMPFAC	$\rightarrow$	α
IMPLCT	$\rightarrow$	α
INVPNO	$\rightarrow$	σ^{-1}
J	$\rightarrow$	j
JDIV	$\rightarrow$	$i = 2$ $\quad(D)$
JPOT	$\rightarrow$	$i = 3$ $\quad(\Theta)$
JVOR	$\rightarrow$	$i = 1$ $\quad(\xi)$

K $\rightarrow$ k

K1 $\rightarrow$ K_1

K1MAX $\rightarrow$ maximum value of K_1

K2 $\rightarrow$ K_2

K2MAX $\rightarrow$ maximum value of K_2

KAPPA $\rightarrow$ κ

L $\rightarrow$ l

L1 $\rightarrow$ L

L1MAX $\rightarrow$ maximum value of L

LAMBDA $\rightarrow$ λ_j

M $\rightarrow$ m

M1 $\rightarrow$ M

M1MAX $\rightarrow$ maximum value of M

MU $\rightarrow$ μ_k

N $\rightarrow$ n

N1 $\rightarrow$ N

N1MAX $\rightarrow$ maximum value of N

N2 $\rightarrow$ $N+1$

NLEV $\rightarrow$ L

NNT2DT $\rightarrow$ $M_{\widehat{D}\rightarrow D}\, M_{\Theta\rightarrow\widehat{\Phi}}$

OMEGA0 $\rightarrow$ Ω

PGQSP0 $\rightarrow$ $\vec{\mathbf{P}}^{n-1}$

PGQSP1 $\rightarrow$ $\vec{\mathbf{P}}^{n}$

PHI $\rightarrow$ ϕ_k

PHLV $\rightarrow$ $p_{l+\frac{1}{2}}$

PKCHLV $\rightarrow$ $\widehat{\zeta}_{l+\frac{1}{2}}$

PKLV $\rightarrow$ ζ_l

PLV $\rightarrow$ p_l

PSURF $\rightarrow$ p_s

PTHICK $\rightarrow$ Δp_l

QHDAMP $\rightarrow$ γ

RGAS $\rightarrow$ R

ROBFAC $\rightarrow$ ϵ

T2GPCP $\rightarrow$ $M_{\Theta \rightarrow \widehat{\Phi}}$
TD8COR $\rightarrow$ $1/\left\{1+\gamma(\nabla_H^2)^4\right\}$
TD8FAC $\rightarrow$ $\gamma(\nabla_H^2)^4$
TDAMP $\rightarrow$ $\eta_{T,l}$
TMNLV $\rightarrow$ $\Theta_{0,n,l}^M$
TRADEQ $\rightarrow$ $\Theta_{k,l}^M$
TREFLV $\rightarrow$ Θ_l^R
TRLXLV $\rightarrow$ $\eta_{T,l}$
TSTDLV $\rightarrow$ Θ_l^S
TZSTD $\rightarrow$ Θ_l^S

UD8COR $\rightarrow$ $1/\left\{1+\gamma(\nabla_H^2+2a^{-2})^4\right\}$
UD8FAC $\rightarrow$ $\gamma(\nabla_H^2+2a^{-2})^4$
UDAMP $\rightarrow$ $\eta_{u,l}$
URLXLV $\rightarrow$ $\eta_{u,l}$

VERVIS $\rightarrow$ ν_l
VVISC $\rightarrow$ ν_l

W2TT $\rightarrow$ $-M_{\omega \rightarrow H}$

ZSTDLV $\rightarrow$ z_l^S

Appendix C. Modifications to allow forcing at bottom boundary

Instead of setting the vertically integrated divergence to zero, one could choose to specify the geopotential Φ near the bottom boundary. This boundary condition may be appropriate for, say, a model of the middle atmosphere, where the lower boundary forcing due to the troposphere is specified. We choose to set the pressure-velocity ω to zero at the upper boundary, but at the lower boundary we specify the following conditions:

$$\overline{\mathbf{u}}_{L+\frac{1}{2}} = \mathbf{u}_L, \quad \overline{\Theta}'_{L+\frac{1}{2}} = \Theta'_L, \quad \Phi_L = C_p \widehat{\zeta}_{L+\frac{1}{2}} \Theta_L + \Phi_s(\lambda, \phi, t)$$

where $\widehat{\zeta}_{L+\frac{1}{2}} \equiv (1 - \zeta_L)$, and Φ_s is some specified function.

We also assume that $\widehat{\Theta}^R_{L+\frac{1}{2}}$ has some specified value. For convenience, one may even choose to set $\widehat{\Theta}^R_{L+\frac{1}{2}} = \widehat{\Theta}^S_{L-\frac{1}{2}}$. We re-define $\vec{\omega}$ to be vector of length L, i.e., $\vec{\omega} = (\omega_{1+\frac{1}{2}}, \ldots, \omega_{L+\frac{1}{2}})$. This allows us to express $\vec{H}_{T^R}$ as

$$\vec{H}_{T^R} = M_{\omega \to H}\, \vec{\omega}$$

where $M_{\omega \to H}$ is re-defined to be an $L \times L$ matrix with the following structure:

$$M_{\omega \to H} = \begin{pmatrix} \frac{\widehat{\Theta}^R_{1+\frac{1}{2}}}{\Delta p_1} & 0 & \cdots & 0 & 0 & 0 \\ \frac{\widehat{\Theta}^R_{1+\frac{1}{2}}}{\Delta p_2} & \frac{\widehat{\Theta}^R_{2+\frac{1}{2}}}{\Delta p_2} & \cdots & 0 & 0 & 0 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ 0 & 0 & \cdots & \frac{\widehat{\Theta}^R_{L-1-\frac{1}{2}}}{\Delta p_{L-1}} & \frac{\widehat{\Theta}^R_{L-\frac{1}{2}}}{\Delta p_{L-1}} & 0 \\ 0 & 0 & \cdots & 0 & \frac{\widehat{\Theta}^R_{L-\frac{1}{2}}}{\Delta p_L} & \frac{\widehat{\Theta}^R_{L+\frac{1}{2}}}{\Delta p_L} \end{pmatrix}$$

We also re-define $M_{D \to \omega}$ to be an $L \times L$ matrix. (The extension is straightforward.) We then define the compound matrix

$$M_{D \to H} = M_{\omega \to H}\, M_{D \to \omega}$$

We also define a new quantity $\tilde{\Phi}_l \equiv \Phi_l - \Phi_s$. We then write

$$\tilde{\Phi}_l = -2\widehat{\Phi}_{l+\frac{1}{2}} + \tilde{\Phi}_{l+1} = -\sum_{l'=l}^{L-1} 2\widehat{\Phi}_{l'+\frac{1}{2}} + C_p\widehat{\zeta}_{L+\frac{1}{2}}\Theta_L$$
$$= \sum_{l'=l}^{L-1} 2C_p\widehat{\zeta}_{l'+\frac{1}{2}}\overline{\Theta}_{l'+\frac{1}{2}} + C_p\widehat{\zeta}_{L+\frac{1}{2}}\Theta_L = C_p\widehat{\zeta}_{l+\frac{1}{2}}\Theta_l + \sum_{l'=l+1}^{L} C_p(\widehat{\zeta}_{l'-\frac{1}{2}} + \widehat{\zeta}_{l'+\frac{1}{2}})\Theta_{l'}$$

Defining a column vector of length L: $\vec{\tilde{\Phi}} = (\tilde{\Phi}_1, \ldots, \tilde{\Phi}_L)$, we can write the above equation in matrix form as

$$\vec{\tilde{\Phi}} = M_{\Theta\to\tilde{\Phi}}\,\vec{\Theta}$$

where $M_{\Theta\to\tilde{\Phi}}$ is an $L \times L$ matrix defined by

$$[M_{\Theta\to\tilde{\Phi}}]_{l,l'} = \begin{cases} C_p(\widehat{\zeta}_{l'-\frac{1}{2}} + \widehat{\zeta}_{l'+\frac{1}{2}}) & \text{if } l' > l; \\ C_p\widehat{\zeta}_{l'+\frac{1}{2}} & \text{if } l' = l; \\ 0 & \text{otherwise.} \end{cases}$$

$$M_{\Theta\to\tilde{\Phi}} = \begin{pmatrix} C_p\widehat{\zeta}_{1+\frac{1}{2}} & C_p(\widehat{\zeta}_{1+\frac{1}{2}} + \widehat{\zeta}_{2+\frac{1}{2}}) & \cdots & C_p(\widehat{\zeta}_{L-\frac{1}{2}} + \widehat{\zeta}_{L+\frac{1}{2}}) \\ 0 & C_p\widehat{\zeta}_{2+\frac{1}{2}} & \cdots & C_p(\widehat{\zeta}_{L-\frac{1}{2}} + \widehat{\zeta}_{L+\frac{1}{2}}) \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & C_p\widehat{\zeta}_{L+\frac{1}{2}} \end{pmatrix}$$

We define the following "explicit time-tendencies":

$$\vec{X}_D^n = -\operatorname{div}\vec{\mathbf{H}}_u^n - \nabla_H^2\,\vec{E}^n - \nabla_H^2\,M_{\Theta\to\tilde{\Phi}}\vec{\Theta}^{n-1} - \gamma_u\,\nabla_H^8\,\vec{D}^{n-1} - \nabla_H^2\,\Phi_s^n$$
$$\vec{X}_T^n = -\vec{H}_{T'}^n - M_{D\to H}\vec{D}^{n-1} - \gamma_T\,\nabla_H^8\,\vec{\Theta}^{n-1}$$

Note that the explicit D tendency for each of the L levels has a $(-\nabla_H^2\,\Phi_s^n)$ term in it, and that is the only form in which Φ_s enters the prognostic equations. So the lower boundary condition on Φ is equivalent to a vertically uniform explicit forcing term in D tendency equation. So it may be convenient to assume that $\Phi_s \equiv 0$, and specify a vertically uniform explicit forcing to simulate the lower boundary condition on Φ.

Proceeding along similar lines as in the case with vertically integrated divergence set to zero, we obtain the following prognostic equations for D and Θ:

$$\left(\frac{\partial \vec{D}}{\partial t}\right)^n = \left[1 - (\alpha 2\Delta t)^2 \gamma_u^{corr} \gamma_T^{corr} \nabla_H^2 M_{\Theta \to \tilde{\Phi}} M_{D \to H}\right]^{-1}$$

$$\left(-\gamma_u^{corr} \gamma_T^{corr} \nabla_H^2 M_{\Theta \to \tilde{\Phi}} (\alpha 2\Delta t) \vec{X}_T^n + \gamma_u^{corr} \vec{X}_D^n\right)$$

$$\left(\frac{\partial \vec{\Theta}}{\partial t}\right)^n = -\gamma_T^{corr} M_{D \to H} (\alpha 2\Delta t) \left(\frac{\partial \vec{D}}{\partial t}\right)^n + \gamma_T^{corr} \vec{X}_T^n$$

The treatment of ξ is the same as for the case with vertically integrated divergence set to zero.

References

Haltiner, G.J., and R.T. Williams, 1980: *Numerical Prediction and Dynamic Meteorology,* second edition. John Wiley & Sons, 477pp.

Holton, J.R., 1979: *An Introduction to Dynamic Meteorology,* second edition. Academic Press, 391pp.

Lorenz, E.N., 1960: Energy and numerical weather prediction. *Tellus,* **12**, 364-373.

Press, W.H., B.P. Flannery, S.A. Teukolsky, and W.T. Vetterling, 1986: *Numerical Recipes: the art of scientific computing.* Cambridge University Press, 818pp.

Simmons, A.J., B.J.Hoskins, and D.M.Burridge, 1978: Stability of the semi-implicit method of time integration. *Mon. Wea. Rev.,* **106**, 405-412.