INTEGRATION OF THE MODEL EQUATIONS, AND EULERIAN DYNAMICS, IN THE CYCLE 46T1 OF ARPEGE/IFS.

YESSAD K. (METEO-FRANCE/CNRM/GMAP/ALGO)

December 19, 2018

Abstract:

This documentation can be seen as a long introduction to modeling. The general purpose of this documentation is to describe the set of equations used, and also the way to integrate the dynamics of the model. Two points will be examined in detail in this documentation: the Eulerian dynamics and the discretisations used. For some other aspects (semi-Lagrangian dynamics, physics, spectral transforms, horizontal diffusion, semi-implicit scheme), this documentation will not provide any detailed description, since there are other documentations describing these topics. The following points will be described: model geometry, different set of equations (non hydrostatic, primitive, shallow-water), their Eulerian formulation and their Eulerian discretisation, calculation and discretisation of some intermediate diagnosed quantities (like the geopotential height). An organigramme is provided. An introduction to tangent linear and adjoint code is provided. There is a specific chapter for the flux form of the Eulerian equation, which is the basis of the DDH diagnostics.

Résumé:

On peut voir cette documentation comme une longue introduction à la modélisation. Le but général de cette documentation est de décrire les jeux d'équations utilisés, et aussi la manière d'intégrer ces équations. On examine plus particulièrement les deux points suivants: la dynamique eulérienne et les discrétisations utilisées. Sur d'autres aspects (semi-lagrangien, physique, transformées spectrales, diffusion horizontale, schéma semi-implicite), cette documentation ne fournit aucune description détaillée, car il y a d'autres documentations décrivant ces sujets. Les points suivants sont abordés: géométrie, différents jeux d'équations (non hydrostatique, équations primitives, modèle shallow-water), leur discrétisation avec un schéma d'advection eulérien, le calcul et la discrétisation de certaines quantités diagnostiquées (comme la hauteur géopotentielle). On fournit un organigramme. Une introduction au code tangent linéaire et adjoint est également proposée. Il y a un chapitre spécifique consacré à la forme flux des équations, qui sert de base aux diagnostics DDH.

Contents

1 Introduction.

1.1 Content of this documentation.

The general purpose of this documentation is to describe the set of equations used, and also the way to integrate the dynamics of the model. Two points will be examined in detail in this documentation: the Eulerian dynamics and the discretisations used. For some other aspects (semi-Lagrangian dynamics, physics, spectral transforms, horizontal diffusion, semi-implicit scheme), this documentation will not provide any detailed description, since there are other documentations describing these topics.

This documentation can be seen as an updated version of some parts of documentations previously written (Courtier et al., 1991; Joly, 1992; Bénard, 1998; Bénard, 2004). Additional features introduced in this documentation are alternate formulations of NH model.

The following sets of equations will be described in this documentation:

- The primitive equations hydrostatic model.
- The fully compressible non-hydrostatic equations model (NHEE).
- The quasi compressible non-hydrostatic equations model (NHQE).
- The 2D shallow-water equations model (configuration 201).
- The 2D vorticity equation model (configuration 202).

1.2 Global models (ARPEGE/IFS).

ARPEGE/IFS is a spectral variable-mesh model. Geometry uses a conformal transformation defined by a high resolution pole (which can be different from the true Northern pole in a tilted geometry) and a stretching coefficient. That introduces a mapping factor M in the discretised form of the equations. M varies between "c" at the high resolution pole and "$1/c$" at the low resolution pole; "c" is the stretching coefficient. Expression of M is given by equation (7). For more details about this conformal transformation, see (Courtier and Geleyn, 1988). For more details about the spectral technique, see (Rochas and Courtier, 1992).

1.3 Limited area models (LAM).

ALADIN and AROME are limited area spectral models. The domain is obtained after a projection of a part of the sphere on a plane, according to a stereo-Lambert projection or a Mercator projection. For the Mercator projection, additionally to the conventional projection, there is now a tilted-rotated Mercator projection. That still defines a mapping factor M. For most applications, the limited area is not too large and M generally remains close to 1.

Fields are bi-periodic in a "bi-periodic" domain defined by three zones:

- an inner "conservation" zone C.
- an intermediate zone I.
- an extension zone E.

Transforms between grid-point space and spectral space are done by double-Fourier transforms. For more details about LAM models geometry, see parts 2 and 5 of (Joly, 1992). The projection on a plane has been modified and simplified so that the limited area domain is now defined by its centre and its dimensions. For more details see (Janoušek, 2001).

1.4 Eulerian advection scheme.

In Eulerian form of equations, the time dependency equation of a variable X writes:

$$\frac{\partial X}{\partial t} = -\mathbf{U}.\nabla_3 X + \dot{X} \qquad (1)$$

where $\mathbf{U}$ is the 3D wind, ∇_3 is the 3D gradient operator, $\dot{X}$ is the sum of the dynamical and physical contributions. $X(t + \Delta t)$ is computed knowing $X(t - \Delta t)$ at the same grid-point. Eulerian technique obliges to use a time-step that matches the CFL (Courant Friedrich Levy) condition everywhere.

- For the global variable-mesh spectral global model ARPEGE, the horizontal CFL condition writes:

$$M\frac{\mid \mathbf{V} \mid}{a}\frac{Dt}{2}\sqrt{N(N+1)} < 1 \tag{2}$$

  where M is the mapping factor, Dt is the time-step at the first integration step and twice the time-step otherwise (leap-frog scheme), $\mid \mathbf{V} \mid$ is the horizontal wind modulus, N is the truncation, a is the mean Earth radius.

- For spectral limited area models (LAM), the horizontal CFL condition writes:

$$M\frac{\mid \mathbf{V} \mid}{a}\frac{Dt}{2}(2\pi)\sqrt{\frac{1}{\frac{L_{\mathrm{x}}^2}{a^2 N_{\mathrm{m}}^2}+\frac{L_{\mathrm{y}}^2}{a^2 N_n^2}}} < 1 \tag{3}$$

  See above for denotations M, Dt, $\mid \mathbf{V} \mid$, a. N_{m} is the zonal truncation, N_n is the meridian truncation, L_{x} (resp. L_{y}) is the zonal (resp. meridian) length of the LAM domain taken on a surface iso $r = a$.

The vertical CFL condition writes:

$$0.5 \mid \dot{\eta} \mid Dt\Delta\eta < 1 \tag{4}$$

1.5 Semi-Lagrangian scheme.

In semi-Lagrangian form of equations, the time dependency equation of a variable X writes:

$$\frac{dX}{dt} = \dot{X} \tag{5}$$

In a three-time level semi-Lagrangian scheme $X(t+\Delta t)$ is computed at a grid-point F knowing $X(t-\Delta t)$ at the point O (not necessary a grid-point) where the same particle is at the instant $t-\Delta t$. In a two-time level semi-Lagrangian scheme $X(t+\Delta t)$ is computed at a grid-point F knowing $X(t)$ at the point O (not necessary a grid-point) where the same particle is at the instant t. The semi-Lagrangian technique is more expensive for one time-step than the Eulerian technique because it is necessary to compute the locations of the origin point O (and in some options the medium point M) along the trajectory and to interpolate some quantities at these points. But it allows to use larger time-steps: the stability condition is now the Lipschitz criterion (trajectories do not cross each other) and is less severe than the CFL condition. For more details about the semi-Lagrangian scheme and the Lipschitz criterion, see the corresponding documentation (IDSL).

1.6 Organisation of a timestep.

In equations (1) and (5), term $\dot{X}$ includes the effects of dynamics (for example the pressure gradient term and Coriolis term in the momentum equation), physics (for example convection, rainfall, radiation, vertical diffusion, soil interface, gravity wave drag), horizontal diffusion. One timestep has the different steps:

- inverse transforms from spectral to grid-point space: the horizontal derivatives necessary to compute the horizontal advection term are computed during these transforms.
- grid-point calculations to compute the explicit part of the RHS of equation (1): dynamics, physics. The temporal filter is done in the grid-point space.
- grid-point coupling for limited area models (LAM).
- direct transforms from grid-point to spectral space.
- spectral calculations: resolution of the Helmholtz equation to compute the semi-implicit correction for linear terms, horizontal diffusion in spectral space, spectral nudging for LAM models.

This paper has for aim to describe the grid-point calculations to compute the dynamics in the RHS. The other points are described in some other documentations. Description of tangent linear and adjoint codes has been introduced for a subset of options.

2 Systems of horizontal coordinates.

2.1 Systems of horizontal spherical coordinates.

In the equations, and in particular in the momentum equation, three different coordinate systems are used:

- The geographical coordinate system (λ, θ) (which appears for example in the Coriolis term).
- In a tilted geometry, the coordinate system on a tilted unstretched sphere $(\lambda_{\text{bne}}, \theta_{\text{bne}})$.
- The coordinate system on the computational (tilted and stretched) sphere (Λ, Θ).

Some relationships between these different systems of coordinates can be listed here:

- Relationship between λ_{bne} and Λ:
$$\lambda_{\text{bne}} = \Lambda \tag{6}$$
- Expression of the mapping factor M:
$$M = \frac{c^2+1}{2c} + \frac{c^2-1}{2c}\sin\Theta \tag{7}$$
At the equator of the computational sphere, $\Theta = 0$, so $M_{\Theta=0} = (c^2+1)/(2c) = 1 + ((c-1)^2)/(2c)$. One can see that, for $c > 1$, this quantity is always > 1. That means that, in a stretched geometry, the equator of the computational sphere is always in the high resolution part and is never identical to the iso-$M = 1$ which separates the high resolution domain from the low resolution domain.
M can also be computed from c and $\sin\theta_{\text{bne}}$:
$$M = 0.5\left(c + \frac{1}{c}\right) - 0.5\left(c - \frac{1}{c}\right)\left[\frac{(c^2-1) - (c^2+1)\sin\theta_{\text{bne}}}{(c^2+1) - (c^2-1)\sin\theta_{\text{bne}}}\right] \tag{8}$$
Inverting this expression yields:
$$\sin\theta_{\text{bne}} = \frac{c^2+1}{c^2-1} - \frac{2c}{(c^2-1)M} \tag{9}$$
At the iso-$M = 1$ latitude, $[\sin\theta_{\text{bne}}]_{M=1} = (c-1)/(c+1)$. For $c > 1$, $[\sin\theta_{\text{bne}}]_{M=1}$ is always > 0: that means that the geographical extension of the high resolution zone ($M > 1$) is smaller than the geographical extension of the low resolution zone. The geographical extension of the high resolution zone diminishes when c increases, and converges towards zero when c converges towards ∞.
- Relationships between θ_{bne} and Θ (the content of array **RATATH** in spherical geometry is $2\tan\theta_{\text{bne}}/a$):
$$\cos\theta_{\text{bne}} = \frac{\cos\Theta}{M} \tag{10}$$
$$\sin\theta_{\text{bne}} = \frac{(c^2-1) + (c^2+1)\sin\Theta}{(c^2+1) + (c^2-1)\sin\Theta} \tag{11}$$
$$\tan\theta_{\text{bne}} = \frac{(c^2-1) + (c^2+1)\sin\Theta}{2c\cos\Theta} \tag{12}$$
$$M\frac{\partial X}{\partial\Theta} = \frac{\partial X}{\partial\theta_{\text{bne}}} \tag{13}$$
Equation (11) can be inverted to provide $\sin\Theta$ knowing $\sin\theta_{\text{bne}}$:
$$\sin\Theta = -\frac{(c^2-1) - (c^2+1)\sin\theta_{\text{bne}}}{(c^2+1) - (c^2-1)\sin\theta_{\text{bne}}} \tag{14}$$
- Relationships giving $(\lambda_{\text{bne}}; \theta_{\text{bne}})$ knowing $(\lambda; \theta)$:
$$\cos\theta_{\text{bne}}\cos\lambda_{\text{bne}} = \cos\theta_{\text{pe}}\sin\theta - \sin\theta_{\text{pe}}\cos\theta\cos(\lambda - \lambda_{\text{pe}}) \tag{15}$$
$$\cos\theta_{\text{bne}}\sin\lambda_{\text{bne}} = -\cos\theta\sin(\lambda - \lambda_{\text{pe}}) \tag{16}$$
$$\sin\theta_{\text{bne}} = \sin\theta_{\text{pe}}\sin\theta + \cos\theta_{\text{pe}}\cos\theta\cos(\lambda - \lambda_{\text{pe}}) \tag{17}$$
- Relationships giving $(\lambda; \theta)$ knowing $(\lambda_{\text{bne}}; \theta_{\text{bne}})$:
$$\cos\theta\cos(\lambda - \lambda_{\text{pe}}) = \cos\theta_{\text{pe}}\sin\theta_{\text{bne}} - \sin\theta_{\text{pe}}\cos\theta_{\text{bne}}\cos\lambda_{\text{bne}} \tag{18}$$
$$\cos\theta\sin(\lambda - \lambda_{\text{pe}}) = -\cos\theta_{\text{bne}}\sin\lambda_{\text{bne}} \tag{19}$$
$$\sin\theta = \sin\theta_{\text{pe}}\sin\theta_{\text{bne}} + \cos\theta_{\text{pe}}\cos\theta_{\text{bne}}\cos\lambda_{\text{bne}} \tag{20}$$

The distance between two points of geographical coordinates $(\lambda_1;\theta_1)$ and $(\lambda_2;\theta_2)$ is:

$$dist = a \arccos\left[\sin(\theta_1)\sin(\theta_2) + \cos(\theta_1)\cos(\theta_2)\cos(\lambda_2 - \lambda_1)\right] \tag{21}$$

It is also useful to know the coordinates of vector $(\mathcal{G}_{\text{nordl}}, \mathcal{G}_{\text{nordm}})$ (which is the unit vector directed towards the true North pole) in a local reference system of coordinates linked to the tilted stretched sphere.

$$\mathcal{G}_{\text{nordl}} = -\frac{\cos\theta_{\text{pe}}\sin\Lambda}{\cos\theta} \tag{22}$$

$$\mathcal{G}_{\text{nordm}} = \frac{2c\sin\theta_{\text{pe}}\cos\Theta - ((c^2-1)+(c^2+1)\sin\Theta)\cos\theta_{\text{pe}}\cos\Lambda}{((c^2+1)+(c^2-1)\sin\Theta)\cos\theta} \tag{23}$$

For a horizontal vector, apparent coordinates $(X_{\text{app}}, Y_{\text{app}})$ on the computational sphere and coordinates $(X_{\text{geo}}, Y_{\text{geo}})$ on the geographical sphere match the following relationships:

$$X_{\text{app}} = \mathcal{G}_{\text{nordm}}X_{\text{geo}} + \mathcal{G}_{\text{nordl}}Y_{\text{geo}} \tag{24}$$

$$Y_{\text{app}} = -\mathcal{G}_{\text{nordl}}X_{\text{geo}} + \mathcal{G}_{\text{nordm}}Y_{\text{geo}} \tag{25}$$

2.2 Systems of horizontal coordinates in plane geometry.

In the equations, and in particular in the momentum equation, two different coordinate systems are used:

- The geographical system of coordinates $(\lambda;\theta)$.
- The coordinate systems on the plane: geographical distances $(x;y)$; apparent distances $(Mx;My)$.

Some new quantities have to be defined:

- K_{L} is the projection constant (0 if Mercator projection, between 0 and 1 excluded if Lambert projection, 1 if polar stereographic projection).
- λ_0 is the reference geographical longitude that defines the projection.
- γ is the rotation angle between the local system of coordinates on the sphere and the local system of coordinates on the plane projection. Its expression is: $\gamma = K_{\text{L}}(\lambda - \lambda_0)$
- One denotes $C = \cos\gamma$ and $S = \sin\gamma$. Vector $(-S, C)$ is the compass (unit vector directed towards the true North pole) and plays the same role as the vector $(\mathcal{G}_{\text{nordl}}, \mathcal{G}_{\text{nordm}})$ in spherical geometry. For a non tilted-rotated Mercator projection, $C = 1$ and $S = 0$.
- M is the mapping factor of the projection; it remains close to 1 if the limited area domain is not too large.
- Horizontal wind $\mathbf{V}$ has components (U, V) on the sphere and reduced components $(U^{'}, V^{'})$ on the plane. Matricial relationship between these different components writes:

$$\begin{pmatrix} U \\ V \end{pmatrix} = M \begin{pmatrix} C & S \\ -S & C \end{pmatrix} \begin{pmatrix} U^{'} \\ V^{'} \end{pmatrix} \tag{26}$$

- The geographical horizontal gradient operator ∇ has the horizontal components ∇^{u} and ∇^{v} on the sphere. For a variable X: $\nabla^{\text{u}}X = [1/a\cos\theta][\partial X/\partial\lambda]$ and $\nabla^{\text{v}}X = [1/a][\partial X/\partial\theta]$.
- $\partial^{'}_{\text{x}}X$ and $\partial^{'}_{\text{y}}X$ are the reduced horizontal derivatives on the plane; relationship between $(\nabla^{\text{u}}X;\ \nabla^{\text{v}}X)$ and $(\partial^{'}_{\text{x}}X;\ \partial^{'}_{\text{y}}X)$ is:

$$\begin{pmatrix} \nabla^{\text{u}}X \\ \nabla^{\text{v}}X \end{pmatrix} = M \begin{pmatrix} C & S \\ -S & C \end{pmatrix} \begin{pmatrix} \partial^{'}_{\text{x}}X \\ \partial^{'}_{\text{y}}X \end{pmatrix} \tag{27}$$

- Relationships giving the horizontal derivatives of the mapping factor M:

$$\frac{\partial M\cos\theta}{\partial\theta} = -K_{\text{L}}M \tag{28}$$

$$\frac{\partial M\cos\gamma}{\partial\lambda} = -K_{\text{L}}M\sin\gamma \tag{29}$$

$$\frac{\partial M\sin\gamma}{\partial\lambda} = K_{\text{L}}M\cos\gamma \tag{30}$$

For more details, see part 2.3 of (Joly, 1992).
For the tilted-rotated Mercator projection, most of what is written above remains valid (but (C, S) is generally different from (1,0)). We first define a rotation with tilting on the sphere, then we apply a Mercator projection on this transformed sphere. For more details about this type of projection, see (IDPRLAM).

2.3 Definitions of horizontal mesh-sizes.

There are several ways to define the horizontal mesh-size in spectral models. This topic was the object of a small paper of Laprise (1992), who listed several possible definitions for the horizontal mesh-size in spectral models. Other definitions have appeared, for example in documentations about the resolution of ARPEGE or in some parts of the code. See internal paper (IDMES) for more details. The most usual definition which is used is the horizontal mesh-size of the colocation grid-point.

3 The different types of horizontal derivatives used.

3.1 Global geometry.

* Horizontal gradient:

Introduction of a stretched geometry in ARPEGE, leads to define different notions of horizontal gradients.

- For a variable X, the geographical horizontal gradient operator on surfaces iso-η writes:

$$\nabla X = (\nabla_{\rm zo} X; \nabla_{\rm me} X) = \left(\frac{M}{a\cos\Theta}\frac{\partial X}{\partial \Lambda}; \frac{M}{a}\frac{\partial X}{\partial \Theta} \right) = \left(\frac{1}{a\cos\theta_{\rm bne}}\frac{\partial X}{\partial \lambda_{\rm bne}}; \frac{1}{a}\frac{\partial X}{\partial \theta_{\rm bne}} \right) \tag{31}$$

- The reduced horizontal gradient operator on surfaces iso-η (which is used in spectral space) writes:

$$\nabla^{'} X = \left(\nabla^{'}_{\rm zo} X; \nabla^{'}_{\rm me} X\right) = \left(\frac{1}{a\cos\Theta}\frac{\partial X}{\partial \Lambda}; \frac{1}{a}\frac{\partial X}{\partial \Theta} \right) \tag{32}$$

- Relationship between geographical and reduced horizontal gradient operator writes:

$$\nabla X = M \nabla^{'} X \tag{33}$$

- ∇_{Π}: the geographical horizontal gradient operator on surfaces iso-hydrostatic pressure.
- More generally when partial derivatives (horizontal or temporal ones) are provided without index, they are derivatives on surfaces iso-η.

For some non-hydrostatic applications it is interesting to provide also information about the second-order horizontal derivatives; we use the underscript "zo" for zonal derivatives and "me" for meridian derivatives.

- Double zonal reduced derivative:

$$\nabla^{'2}_{\rm zo\ zo} X = \frac{1}{a\cos\Theta}\frac{\partial\left[\frac{1}{a\cos\Theta}\frac{\partial X}{\partial\Lambda}\right]}{\partial \Lambda} \tag{34}$$

- Double meridian reduced derivative:

$$\nabla^{'2}_{\rm me\ me} X = \frac{1}{a^2}\frac{\partial^2 X}{\partial \Theta^2} \tag{35}$$

- Double mixed reduced derivative:

$$\nabla^{'2}_{\rm zo\ me} X = \frac{1}{a^2\cos\Theta}\frac{\partial^2 X}{\partial\Lambda\partial\Theta} \tag{36}$$

The relationships between geographical and reduced derivatives are $\nabla^2_{\rm zo\ zo} X = M^2 \nabla^{'2}_{\rm zo\ zo} X$, $\nabla^2_{\rm me\ me} X = M^2 \nabla^{'2}_{\rm me\ me} X$ and $\nabla^2_{\rm zo\ me} X = M^2 \nabla^{'2}_{\rm zo\ me} X$.

* Horizontal laplacian:

Relationships (2.3) of (Courtier and Geleyn, 1988), applied to the horizontal laplacian, write:

$$\nabla^2 X = M^2 \nabla^{'2} X \tag{37}$$

where $\nabla^{'2}$ is the reduced laplacian, computed in spectral space (diagonal operator in spectral space). Its expression is:

$$\nabla^{'2} X = \frac{1}{(a\cos\Theta)^2}\frac{\partial^2 X}{\partial\Lambda^2} + \frac{1}{a^2\cos\Theta}\frac{\partial\left(\cos\Theta\frac{\partial X}{\partial\Theta}\right)}{\partial\Theta} \tag{38}$$

The same considerations are also valid for the scalar product and vectorial product of two gradients. Relationships (2.3) of (Courtier and Geleyn, 1988) write:

$$(\nabla X).(\nabla Y) = M^2 (\nabla^{'} X).(\nabla^{'} Y) \tag{39}$$

$$(\nabla X) \wedge (\nabla Y) = M^2 (\nabla^{'} X) \wedge (\nabla^{'} Y) \tag{40}$$

∗ Horizontal vorticity and divergence:

Spectral computations provide the reduced divergence and vorticity $D^{'}$ and $\zeta^{'}$, from the velocity potential χ and the stream function ψ:

$$D^{'} = \nabla^{'2}\chi \quad (41)$$

$$\zeta^{'} = \nabla^{'2}\psi \quad (42)$$

Geographical divergence and vorticity D and ζ write:

$$D = M^2 D^{'} = \nabla^2\chi \quad (43)$$

$$\zeta = M^2\zeta^{'} = \nabla^2\psi \quad (44)$$

In continuity equation the notation $\nabla\mathbf{V}$ is used for D. In this equation it is convenient to isolate quantities $\nabla\mathbf{V}$ (i.e. $M^2D^{'}$) and $\nabla\wedge\mathbf{V}$ (i.e. $M^2\zeta^{'}$) which are the quantities easily available in the grid point part of the model, just after the multiplications by M^2.
Reduced divergence $D^{'}$ and reduced vorticity $\zeta^{'}$ of the wind are linked to the reduced components of the wind by the following relationships:

$$D^{'} = \frac{1}{a\cos\Theta}\frac{\partial U^{'}}{\partial\Lambda} + \frac{1}{a\cos\Theta}\frac{\partial(V^{'}\cos\Theta)}{\partial\Theta} \quad (45)$$

$$\zeta^{'} = \frac{1}{a\cos\Theta}\frac{\partial V^{'}}{\partial\Lambda} - \frac{1}{a\cos\Theta}\frac{\partial(U^{'}\cos\Theta)}{\partial\Theta} \quad (46)$$

Formulae (45) and (46) allow to compute wind meridian derivatives from wind zonal derivatives, divergence and vorticity:

$$\frac{\partial U^{'}}{\partial\Theta} = -\zeta^{'} + \frac{U^{'}}{a}\tan\Theta + \frac{1}{a\cos\Theta}\frac{\partial V^{'}}{\partial\Lambda}$$

$$\frac{\partial V^{'}}{\partial\Theta} = D^{'} + \frac{V^{'}}{a}\tan\Theta - \frac{1}{a\cos\Theta}\frac{\partial U^{'}}{\partial\Lambda}$$

∗ Wind components:

Reduced wind components are linked to the velocity potential χ and the stream function ψ by the following relationships:

$$(U^{'}a\cos\Theta) = \frac{\partial\chi}{\partial\Lambda} - \cos\Theta\frac{\partial\psi}{\partial\Theta} \quad (47)$$

$$(V^{'}a\cos\Theta) = \frac{\partial\psi}{\partial\Lambda} + \cos\Theta\frac{\partial\chi}{\partial\Theta} \quad (48)$$

3.2 Plane geometry.

∗ Horizontal gradient:

- For a variable X, the geographical horizontal gradient operator on surfaces iso-η writes:

$$\nabla X = (\nabla^{\mathrm{u}}X;\nabla^{\mathrm{v}}X) = \left(\frac{1}{a\cos\theta}\frac{\partial X}{\partial\lambda};\frac{1}{a}\frac{\partial X}{\partial\theta}\right) \quad (49)$$

- The reduced horizontal gradient operator on surfaces iso-η (which is used in spectral space) has components $\nabla^{'}X = (\partial^{'}_{\mathrm{x}}X;\partial^{'}_{\mathrm{y}}X)$ and we have the following identities: $\nabla^{'}_{\mathrm{zo}} = \partial^{'}_{\mathrm{x}}$ and $\nabla^{'}_{\mathrm{me}} = \partial^{'}_{\mathrm{y}}$.
- Relationship between geographical horizontal gradient operator and reduced horizontal gradient operator is given by equation (27); this equation can be rewritten

$$\nabla X = M\begin{pmatrix} C & S \\ -S & C \end{pmatrix}\nabla^{'}X \quad (50)$$

or:

$$\begin{pmatrix} \partial^{'}_{\mathrm{x}}X \\ \partial^{'}_{\mathrm{y}}X \end{pmatrix} = \frac{1}{M}\begin{pmatrix} C & -S \\ S & C \end{pmatrix}\begin{pmatrix} \nabla^{\mathrm{u}}X \\ \nabla^{\mathrm{v}}X \end{pmatrix} = \frac{1}{M}\begin{pmatrix} C & -S \\ S & C \end{pmatrix}\begin{pmatrix} \frac{1}{a\cos\theta}\frac{\partial X}{\partial\lambda} \\ \frac{1}{a}\frac{\partial X}{\partial\theta} \end{pmatrix} = \begin{pmatrix} \frac{1}{M}\frac{\partial X}{\partial x} \\ \frac{1}{M}\frac{\partial X}{\partial y} \end{pmatrix} \quad (51)$$

The code easily provides $\nabla^{'}X$ and $M\nabla^{'}X$.

- Like for the spherical geometry, we can give the expressions of second-order horizontal derivatives:
  Double zonal reduced derivative:
  $$\nabla'^2_{\mathrm{zo\ zo}} X = \frac{1}{M^2}\frac{\partial^2 X}{\partial x^2} \tag{52}$$
  Double meridian reduced derivative:
  $$\nabla'^2_{\mathrm{me\ me}} X = \frac{1}{M^2}\frac{\partial^2 X}{\partial y^2} \tag{53}$$
  Double mixed reduced derivative:
  $$\nabla'^2_{\mathrm{zo\ me}} X = \frac{1}{M^2}\frac{\partial^2 X}{\partial x \partial y} \tag{54}$$

$*$ Horizontal laplacian:

Equation (37) is still valid, where:

$$\nabla'^2 X = \frac{\partial^2 X}{M^2 \partial x^2} + \frac{\partial^2 X}{M^2 \partial y^2} \tag{55}$$

That yields, for the geographical laplacian operator:

$$\nabla^2 X = \frac{\partial^2 X}{\partial x^2} + \frac{\partial^2 X}{\partial y^2} = \frac{1}{(a\cos\theta)^2}\frac{\partial^2 X}{\partial \lambda^2} + \frac{1}{a^2\cos\theta}\frac{\partial\left(\cos\theta \frac{\partial X}{\partial \theta}\right)}{\partial \theta} \tag{56}$$

$*$ Horizontal vorticity and divergence:

Equations (41), (42), (43) and (44) remain valid.
Reduced divergence $D^{'}$ and reduced vorticity $\zeta^{'}$ of the wind are linked to the reduced components of the wind by the following relationships:

$$D^{'} = \frac{1}{M}\frac{\partial U^{'}}{\partial x} + \frac{1}{M}\frac{\partial V^{'}}{\partial y} \tag{57}$$

$$\zeta^{'} = \frac{1}{M}\frac{\partial V^{'}}{\partial x} - \frac{1}{M}\frac{\partial U^{'}}{\partial y} \tag{58}$$

Formulae (57) and (58) allow to compute wind meridian derivatives from wind zonal derivatives, divergence and vorticity:

$$\frac{1}{M}\frac{\partial U^{'}}{\partial y} = -\zeta^{'} + \frac{1}{M}\frac{\partial V^{'}}{\partial x}$$

$$\frac{1}{M}\frac{\partial V^{'}}{\partial y} = D^{'} - \frac{1}{M}\frac{\partial U^{'}}{\partial x}$$

$*$ Wind components:

Reduced wind components are linked to the velocity potential χ and the stream function ψ by the following relationships:

$$U^{'} = \frac{1}{M}\frac{\partial \chi}{\partial x} - \frac{1}{M}\frac{\partial \psi}{\partial y} \tag{59}$$

$$V^{'} = \frac{1}{M}\frac{\partial \psi}{\partial x} + \frac{1}{M}\frac{\partial \chi}{\partial y} \tag{60}$$

3.3 Additional remarks.

Horizontal derivatives are computed in the code during the spectral transforms from spectral space to grid-point space. When entering the grid-point space, reduced horizontal derivatives are computed. The multiplication by the mapping factor M (or a power of M) is done during the grid-point calculations.

4 The 2D equations.

4.1 Denotations for the 2D equations.

- $\mathbf{V}$ is the horizontal wind. Zonal and meridian components on computational grid are U and V.
- D is the horizontal wind divergence; ζ is the horizontal wind vorticity.
- Φ is the equivalent height. $\Phi_{\rm s}$ is the surface geopotential height (i.e. the orography). Φ^* is a reference equivalent height which is only used in the semi-implicit scheme and the linear model.
- $\mathbf{\Omega}$ is the Earth rotation angular velocity.
- ∇ is the first order horizontal gradient on η-surfaces.
- a is the Earth radius.
- $(\lambda_{\rm bne}, \theta_{\rm bne})$ are the longitude-latitude coordinates on a tilted and not stretched geometry, the tilting being the same as the one of the computational sphere.
- $\mathbf{k}$ is the unit vertical vector. One can write:

$$\mathbf{k} = \frac{\mathbf{r}}{\mid r \mid} = \frac{\mathbf{r}}{a}$$

4.2 The 2D shallow-water system of equations in spherical geometry.

Momentum equation.

Lagrangian tendency: Coriolis force can be treated explicitly ($\delta_{\mathbf{V}}$=0) or implicitly ($\delta_{\mathbf{V}}$=1).

$$\frac{d\left(\mathbf{V} + \delta_{\mathbf{V}}(2\mathbf{\Omega} \wedge \mathbf{r})\right)}{dt} = -2(1-\delta_{\mathbf{V}})(\mathbf{\Omega} \wedge \mathbf{V}) - \nabla\Phi \tag{61}$$

Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial \mathbf{V}}{\partial t} = -2(\mathbf{\Omega} \wedge \mathbf{V}) - \nabla\Phi - \mathbf{V}\nabla\mathbf{V} - \left(\frac{U}{a}\tan\theta_{\rm bne}\right)\mathbf{k} \wedge \mathbf{V} \tag{62}$$

Advection ($-\mathbf{V}\nabla\mathbf{V}$) and curvature ($-\left(\frac{U}{a}\tan\theta_{\rm bne}\right)\mathbf{k}\wedge\mathbf{V}$) terms can be rearranged, in order to show the divergence and vorticity in the equation and to eliminate the meridian derivatives of U and V:

$$\frac{\partial \mathbf{V}}{\partial t} = -2(\mathbf{\Omega} \wedge \mathbf{V}) - \nabla\Phi + \mathbf{D}_{\mathbf{V}} \tag{63}$$

where $\mathbf{D}_{\mathbf{V}}$ is the vector of coordinates:

$$\begin{pmatrix} V\zeta - \frac{1}{a\cos\theta_{\rm bne}}U\frac{\partial U}{\partial\lambda_{\rm bne}} - \frac{1}{a\cos\theta_{\rm bne}}V\frac{\partial V}{\partial\lambda_{\rm bne}} \\ -VD - \frac{1}{a\cos\theta_{\rm bne}}U\frac{\partial V}{\partial\lambda_{\rm bne}} + \frac{1}{a\cos\theta_{\rm bne}}V\frac{\partial U}{\partial\lambda_{\rm bne}} - \frac{U^2+V^2}{a}\tan\theta_{\rm bne} \end{pmatrix}$$

Continuity equation.

Lagrangian tendency:

- Conventional formulation.

$$\frac{d(\Phi - (1-\delta_{\rm TR})\Phi_{\rm s})}{dt} = -(\Phi - \Phi_{\rm s})D + \delta_{\rm TR}\mathbf{V}\nabla(\Phi_{\rm s}) \tag{64}$$

- Lagrangian formulation.

$$\frac{d((\Phi - \Phi_{\rm s})J)}{dt} = 0 \tag{65}$$

J is a "Jacobian" quantity which matches $dJ/dt = -JD$.

Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial\Phi}{\partial t} = -(\Phi - \Phi_{\rm s})D - \mathbf{V}\nabla(\Phi - \Phi_{\rm s}) \tag{66}$$

4.3 The 2D "vorticity" system of equations in spherical geometry.

The system of equations starts from the shallow-water one, but Φ and $\Phi_{\rm s}$ have to be replaced by zero. So continuity equation becomes $\Phi = 0$.
Momentum equation and definition of $\mathbf{D}_{\mathbf{V}}$ given in part 4.2 remain valid, replacing Φ by zero.

4.4 The 2D equations in plane geometry.

The equations are not significantly different from the spherical geometry ones, except for the curvature terms. They are not detailed in this documentation. The 2D model is not coded in plane geometry.

5 The 3D equations.

5.1 Denotations for the 3D equations.

- $\mathbf{V}$ is the horizontal wind. Zonal and meridian components on computational grid are U and V.
- D is the horizontal wind divergence; ζ is the horizontal wind vorticity.
- T is the temperature.
- q is the humidity, q_l the liquid water, q_i the ice and q_a the cloudiness.
- $\mathcal{O}3$ is the ozone.
- Π is the hydrostatic pressure; Π_s is the hydrostatic surface pressure.
- $\boldsymbol{\Omega}$ is the Earth rotation angular velocity.
- $(\lambda_\mathrm{bne}, \theta_\mathrm{bne})$ are the longitude-latitude coordinates on a tilted and not stretched geometry, the tilting being the same as the one of the computational sphere.
- (λ, θ) are the geographical longitude-latitude coordinates.
- (Λ, Θ) are the computational sphere longitude-latitude coordinates.
- w is the z-coordinate vertical velocity: $w = dz/dt$.
- $\omega = d\Pi/dt$ is the total temporal derivative of the hydrostatic pressure.
- gz is the geopotential height.
- Φ is the total geopotential. $\Phi = gz$ in the set of equations described.
- $\Phi_\mathrm{s} = gz_\mathrm{s}$ is the surface geopotential (i.e. the orography).
- $\mathbf{r}$ is the vector directed upwards, the length of which is the Earth radius.
- a is the average Earth radius near the surface.
- $\mathbf{i}$ (resp. $\mathbf{j}$) is the unit zonal (resp. meridian) vector on the Gaussian grid.
- $\mathbf{k}$ is the unit vertical vector. One can write $\mathbf{k} = \mathbf{r}/a$.
- g is the gravity acceleration constant, assumed to be vertically constant.
- R is the gas constant for air and R_d the gas constant for dry air.
- R_a is a generic denotation used in definition of vertical divergence d or d_hyd, which may be R_d or R.
- c_p and $c_{\mathrm{p}_\mathrm{d}}$ are respectively the specific heat at constant pressure for moist air and dry air.
- c_v and $c_{\mathrm{v}_\mathrm{d}}$ are respectively the specific heat at constant volume for moist air and dry air.
- κ and κ_d are respectively R/c_p and $R_\mathrm{d}/c_{\mathrm{p}_\mathrm{d}}$.
- ∇ is the first order horizontal gradient on η-surfaces.
- α_T is a vertical-dependent coefficient used to define a thermodynamic variable $T + \delta_\mathrm{TR}\frac{\alpha_\mathrm{T}\Phi_\mathrm{s}}{R_\mathrm{d}T_\mathrm{st}}$ less sensitive to orography than temperature T. Expression of α_T is:

$$\alpha_\mathrm{T} = B\left(-\frac{R_\mathrm{d}}{g}\left[\frac{dT}{dz}\right]_\mathrm{st}\right) T_\mathrm{st}\left(\frac{\Pi_\mathrm{st}}{\Pi_\mathrm{sst}}\right)^{\left(-\frac{R_\mathrm{d}}{g}\left[\frac{dT}{dz}\right]_\mathrm{st}-1\right)} \tag{67}$$

  where B defines the vertical hybrid coordinate (see part (8.3)). Subscript "st" stands for "standard atmosphere".
- ρ is the mass per volume unit of air.
- M is the mapping factor; $\overline{M}$ is a reference mapping factor for the semi-implicit scheme.
- $\mathbf{S}$, $\mathbf{G}$, $\mathbf{N}$ are adimensioned integral operators (see appendix 1).
- τ, γ, ν are linear operators used in the semi-implicit scheme; $\mathbf{S}^*$, $\mathbf{G}^*$, $\mathbf{N}^*$ are their adimensioned versions. For more details, see documentation (IDSI) about semi-implicit scheme.

For non-hydrostatic models:

- p is the pressure, p_s is the surface pressure.
- $\tilde{T}$ is a modified temperature used in the NHQE model. $\tilde{T} = T\exp(-\kappa\log(p/\Pi))$.
- $\mathbf{L}^*$ is a linear operator used in the NHEE semi-implicit scheme (see documentation (IDSI)).
- $\mathbf{L}$ is the non-linear counterpart of $\mathbf{L}^*$, used in the NHEE model.
- $\mathbf{L}_\kappa^*$ is a linear operator used in the NHQE semi-implicit scheme (see documentation (IDSI)).
- $\mathbf{L}_\kappa$ is the non-linear counterpart of $\mathbf{L}_\kappa^*$, used in the NHQE model.
- $\overline{\partial}$ is a first-order derivative operator defined by $\overline{\partial}Z = \partial Z/\partial\log\Pi$.

5.2 The 3D primitive equation model.

5.2.1 Momentum equation.

Lagrangian tendency: Coriolis force can be treated explicitly ($\delta_{\mathbf{V}}$=0) or implicitly ($\delta_{\mathbf{V}}$=1) in the Lagrangian equation.

$$\frac{d\left(\mathbf{V}+\delta_{\mathbf{V}}(2\mathbf{\Omega}\wedge\mathbf{r})\right)}{dt}=-2(1-\delta_{\mathbf{V}})(\mathbf{\Omega}\wedge\mathbf{V})-\nabla\Phi-RT\nabla(\log\Pi)+\mathbf{F_V} \tag{68}$$

$\mathbf{F_V}$ is the physical contribution on horizontal wind.
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial\mathbf{V}}{\partial t}=-2\mathbf{\Omega}\wedge\mathbf{V}-\nabla\Phi-RT\nabla(\log\Pi)-\mathbf{V}\nabla\mathbf{V}-\dot{\eta}\frac{\partial\mathbf{V}}{\partial\eta}-\left(\frac{U}{a}\tan\theta_{\mathrm{bne}}\right)\mathbf{k}\wedge\mathbf{V}+\mathbf{F_V} \tag{69}$$

Advection ($-\mathbf{V}\nabla\mathbf{V}$) and curvature ($-\left(\frac{U}{a}\tan\theta_{\mathrm{bne}}\right)\mathbf{k}\wedge\mathbf{V}$) terms can be rearranged, in order to show the divergence and vorticity in the equation and to eliminate the meridian derivatives of U and V:

$$\frac{\partial\mathbf{V}}{\partial t}=-2\mathbf{\Omega}\wedge\mathbf{V}-\nabla\Phi-RT\nabla(\log\Pi)+\mathbf{D_V}-\dot{\eta}\frac{\partial\mathbf{V}}{\partial\eta}+\mathbf{F_V} \tag{70}$$

where $\mathbf{D_V}$ is the vector of coordinates, in spherical geometry:

$$\begin{pmatrix} V\zeta-\frac{1}{a\cos\theta_{\mathrm{bne}}}U\frac{\partial U}{\partial\lambda_{\mathrm{bne}}}-\frac{1}{a\cos\theta_{\mathrm{bne}}}V\frac{\partial V}{\partial\lambda_{\mathrm{bne}}} \\ -VD-\frac{1}{a\cos\theta_{\mathrm{bne}}}U\frac{\partial V}{\partial\lambda_{\mathrm{bne}}}+\frac{1}{a\cos\theta_{\mathrm{bne}}}V\frac{\partial U}{\partial\lambda_{\mathrm{bne}}}-\frac{U^2+V^2}{a}\tan\theta_{\mathrm{bne}} \end{pmatrix}$$

This vector has a different expression in plane geometry (see (Joly, 1992)):

$$\begin{pmatrix} V\zeta-U\partial_{\mathrm{x}}^{'}U-V\partial_{\mathrm{x}}^{'}V+\frac{(U^{'2}+V^{'2})S}{a\cos\theta}(\sin\theta-K_{\mathrm{L}}) \\ -VD+U\partial_{\mathrm{x}}^{'}V+V\partial_{\mathrm{x}}^{'}U+\frac{(U^{'2}+V^{'2})C}{a\cos\theta}(\sin\theta-K_{\mathrm{L}}) \end{pmatrix}$$

C and S (related to compass) and K_{L} (related to plane projection) are defined in (Joly, 1992) and in section 2.

5.2.2 Thermodynamic equation.

Lagrangian tendency:

$$\frac{dT}{dt}=\frac{RT}{c_{\mathrm{p}}}\frac{\omega}{\Pi}+F_{\mathrm{T}} \tag{71}$$

F_{T} is the physical contribution on temperature.
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial T}{\partial t}=-\mathbf{V}\nabla T-\dot{\eta}\frac{\partial T}{\partial\eta}+\frac{RT}{c_{\mathrm{p}}}\frac{\omega}{\Pi}+F_{\mathrm{T}} \tag{72}$$

5.2.3 Continuity equation.

∗ 3D formulation of continuity equation: Continuity equation in the η vertical coordinate writes:

$$\frac{d\left(\frac{\partial\Pi}{\partial\eta}\right)}{dt}=-\frac{\partial\Pi}{\partial\eta}\left(D+\frac{\partial\dot{\eta}}{\partial\eta}\right)+F_{\mathrm{m}}^{'} \tag{73}$$

or:

$$\frac{\partial\left(\frac{\partial\Pi}{\partial\eta}\right)}{\partial t}=\frac{\partial\left(\frac{\partial\Pi}{\partial t}\right)}{\partial\eta}=-\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)-\frac{\partial\left(\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right)}{\partial\eta}+F_{\mathrm{m}}^{'} \tag{74}$$

$F_{\mathrm{m}}^{'}$ is the physical contribution on $\frac{\partial\Pi}{\partial\eta}$, and can be rewritten $F_{\mathrm{m}}^{'}=-g\frac{\partial F_{\mathrm{m}}}{\partial\eta}$, where F_{m} is the diabatic flux applied to continuity equation.
Equation (73) is not convenient to use directly, one rather uses an equation for the 2D variable $\log\Pi_{\mathrm{s}}$. For a 2D variable X_{2D} which does not depend on η, vertical advection is zero, so the relationship between Lagrangian temporal derivative and Eulerian temporal derivative writes:

$$\frac{dX_{\mathrm{2D}}}{dt}=\frac{\partial X_{\mathrm{2D}}}{\partial t}+\mathbf{V}\nabla X_{\mathrm{2D}} \tag{75}$$

This formula is valid in particular for the following variables:

- $X_{\mathrm{2D}}=\log\Pi_{\mathrm{s}}$.
- $X_{\mathrm{2D}}=\log\Pi_{\mathrm{s}}+\delta_{\mathrm{TR}}\frac{\Phi_{\mathrm{s}}}{R_{\mathrm{d}}T_{\mathrm{st}}}$.

T_{st} is the surface standard temperature (288.15 K), Π_{sst} is the surface standard hydrostatic pressure (101325 Pa), Φ_{s} is the surface orography. δ_{TR} is 0 or 1; when $\delta_{\mathrm{TR}}=1$ the new variable is less sensitive to the orography (new variable proposed by Ritchie and Tanguay (1996) to reduce orographic resonance).

$*$ **Eulerian formulation of continuity equation:** Calculations are not detailed, but when using the properties of the hybrid coordinate the evolution equation of $\frac{\partial\Pi}{\partial\eta}$ can be transformed into an evolution equation of $\log\Pi_{\rm s}$.

$$\frac{\partial\log(\Pi_{\rm s})}{\partial t} = -\frac{1}{\Pi_{\rm s}}\int_{\eta=0}^{\eta=1}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta - \frac{1}{\Pi_{\rm s}}\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=1} + \frac{1}{\Pi_{\rm s}}\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} - \frac{1}{\Pi_{\rm s}}g\left[F_{\rm m}\right]_{\eta=1} \tag{76}$$

$F_{\rm m}$ is the diabatic flux applied to continuity equation. $F_{\rm m}$ is assumed to be zero at the top of the atmosphere.
$\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0}$ is non-zero only when there is an upper radiative boundary condition (**LRUBC**=.TRUE.).
$\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=1}$ is non-zero only when the option "$\delta m = 1$" is activated (**NDPSFI**=1).
Using:

$$\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) = \frac{\partial\Pi}{\partial\eta}\nabla\mathbf{V} + \frac{\partial B}{\partial\eta}\mathbf{V}\nabla\Pi_{\rm s} = \frac{\partial\Pi}{\partial\eta}\left[\nabla\mathbf{V} + \frac{\partial B}{\partial\Pi}\mathbf{V}\nabla\Pi_{\rm s}\right]$$

and definition of integral operator $\mathbf{N}$, we have the following identity:

$$\frac{1}{\Pi_{\rm s}}\int_{\eta=0}^{\eta=1}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta = \mathbf{N}\left(\nabla\mathbf{V} + \frac{\partial B}{\partial\Pi}\mathbf{V}\nabla\Pi_{\rm s}\right)$$

$*$ **Lagrangian tendency of orographic term:** Term $\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}$ has no vertical variation and no local temporal variation, so:

$$\frac{d\left[\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}\right]}{dt} = \mathbf{V}\nabla\left[\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}\right] \tag{77}$$

$*$ **Lagrangian formulation of continuity equation:** Combining equations (75), (76) and (77) one obtains the following Lagrangian equation:

$$\frac{d\left[\log\Pi_{\rm s} + \delta_{\rm TR}\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}\right]}{dt} = -\frac{1}{\Pi_{\rm s}}\int_{\eta=0}^{\eta=1}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta + \mathbf{V}\nabla\left(\log\Pi_{\rm s} + \delta_{\rm TR}\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}\right) - \frac{1}{\Pi_{\rm s}}\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=1} + \frac{1}{\Pi_{\rm s}}\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} - \frac{1}{\Pi_{\rm s}}g\left[F_{\rm m}\right]_{\eta=1} \tag{78}$$

$*$ **Vertically integrated Lagrangian formulation of continuity equation:** Since equation (78) mixes 3D terms (advective terms) and 2D terms (the other terms), and the LHS is a 2D term, one actually discretizes a vertical integrated formulation of this equation, with a weight $\frac{\partial B}{\partial\eta}$; equation can be rewritten:

$$\int_{\eta=0}^{\eta=1}\frac{\partial B}{\partial\eta}\frac{d\left[\log\Pi_{\rm s}+\delta_{\rm TR}\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}\right]}{dt}d\eta =$$
$$\int_{\eta=0}^{\eta=1}\frac{\partial B}{\partial\eta}\left[-\frac{1}{\Pi_{\rm s}}\int_{\eta=0}^{\eta=1}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta + \mathbf{V}\nabla\left(\log\Pi_{\rm s} + \delta_{\rm TR}\frac{\Phi_{\rm s}}{R_{\rm d}T_{\rm st}}\right) - \frac{1}{\Pi_{\rm s}}\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=1} + \frac{1}{\Pi_{\rm s}}\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} - \frac{1}{\Pi_{\rm s}}g\left[F_{\rm m}\right]_{\eta=1}\right]d\eta \tag{79}$$

5.2.4 Advectable GFL prognostic fields.

Equation is written for moisture q, and is the same for the other advectable GFL variables.
Lagrangian tendency:

$$\frac{dq}{dt} = F_{\rm q} \tag{80}$$

$F_{\rm q}$ is the physical contribution on moisture.
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial q}{\partial t} = -\mathbf{V}\nabla q - \dot\eta\frac{\partial q}{\partial\eta} + F_{\rm q} \tag{81}$$

5.2.5 Relationship between geopotential height gz and pressure depth.

Hydrostatic relationship writes:

$$\frac{\partial\Pi}{\partial z} = -\rho g \tag{82}$$

The perfect gas formula writes:

$$\Pi = \rho RT \tag{83}$$

The combination of equations (82) and (83) provides the relationship between gz and pressure depth:

$$\frac{\partial(gz)}{\partial\Pi} = -\frac{RT}{\Pi} \tag{84}$$

The vertical integrated formulation of equation (84) writes:

$$gz = gz_{\mathrm{s}} - \int_{\Pi'=\Pi_{\mathrm{s}}}^{\Pi'=\Pi} \frac{RT}{\Pi'} d\Pi' \tag{85}$$

i.e.:

$$gz = gz_{\mathrm{s}} + \mathbf{G}(RT)$$

5.2.6 Relationship between total geopotential Φ and pressure depth.

$\Phi = gz$. See part (5.2.5) for more details.

5.2.7 Diagnostic expression of some vertical velocities.

$*$ **Term $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$:** Continuity equation is vertically integrated on $d\eta$ from $\eta = 0$ to $\eta = \eta_l$. Equation (74) becomes:

$$\left[\frac{\partial\Pi}{\partial t}\right]_{\eta_l} - \left[\frac{\partial\Pi}{\partial t}\right]_{\eta=0} + \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta + \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta_l} - \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} = -\int_{\eta=0}^{\eta=\eta_l} g\frac{\partial F_{\mathrm{m}}}{\partial\eta} d\eta \tag{86}$$

The following property is used: $\left[\frac{\partial\Pi}{\partial t}\right]_{\eta=0} = 0$. Equation (86) can be rewritten:

$$\left[\frac{\partial\Pi}{\partial t}\right]_{\eta_l} + \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta + \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta_l} - \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} = -\int_{\eta=0}^{\eta=\eta_l} g\frac{\partial F_{\mathrm{m}}}{\partial\eta} d\eta \tag{87}$$

which can be rewritten, taking in account that F_{m} at the top is zero:

$$\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta_l} = \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} - \left[\frac{\partial\Pi}{\partial t}\right]_{\eta_l} - \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta - g\left[F_{\mathrm{m}}\right]_{\eta_l} \tag{88}$$

One uses equation (167) and one replaces $\frac{\partial\Pi_{\mathrm{s}}}{\partial t}$ by its expression provided by equation (76). That yields expression of $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$:

$$\begin{aligned}\left[\dot{\eta}\tfrac{\partial\Pi}{\partial\eta}\right]_{\eta_l} &= B_{\eta_l}\int_{\eta=0}^{\eta=1} \nabla\left(\mathbf{V}\tfrac{\partial\Pi}{\partial\eta}\right) d\eta - \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\tfrac{\partial\Pi}{\partial\eta}\right) d\eta \\ &+ B_{\eta_l}\left[\dot{\eta}\tfrac{\partial\Pi}{\partial\eta}\right]_{\eta=1} - B_{\eta_l}\left[\dot{\eta}\tfrac{\partial\Pi}{\partial\eta}\right]_{\eta=0} + B_{\eta_l} g\left[F_{\mathrm{m}}\right]_{\eta=1} - g\left[F_{\mathrm{m}}\right]_{\eta_l}\end{aligned} \tag{89}$$

This equation can be rewritten in its “barycentric” formulation:

$$\begin{aligned}\left[\dot{\eta}\tfrac{\partial\Pi}{\partial\eta}\right]_{\eta_l} - B_{\eta_l} g\left[F_{\mathrm{m}}\right]_{\eta=1} + g\left[F_{\mathrm{m}}\right]_{\eta_l} &= B_{\eta_l}\int_{\eta=0}^{\eta=1} \nabla\left(\mathbf{V}\tfrac{\partial\Pi}{\partial\eta}\right) d\eta - \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\tfrac{\partial\Pi}{\partial\eta}\right) d\eta \\ &+ B_{\eta_l}\left[\dot{\eta}\tfrac{\partial\Pi}{\partial\eta}\right]_{\eta=1} - B_{\eta_l}\left[\dot{\eta}\tfrac{\partial\Pi}{\partial\eta}\right]_{\eta=0}\end{aligned} \tag{90}$$

$*$ **Pressure coordinate vertical velocity ω:** Equation (87) can be rewritten:

$$\left[\frac{d\Pi}{dt}\right]_{\eta_l} = \left[\mathbf{V}\nabla\Pi\right]_{\eta_l} - \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta + \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} - g\left[F_{\mathrm{m}}\right]_{\eta_l} \tag{91}$$

This equation can be rewritten in its “barycentric” formulation:

$$\left[\frac{d\Pi}{dt}\right]_{\eta_l} + g\left[F_{\mathrm{m}}\right]_{\eta_l} = \left[\mathbf{V}\nabla\Pi\right]_{\eta_l} - \int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta + \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} \tag{92}$$

i.e.:

$$\left[\frac{\omega}{\Pi}\right]_{\eta_l} = \left[\frac{1}{\Pi}\frac{d\Pi}{dt} + \frac{1}{\Pi}gF_{\mathrm{m}}\right]_{\eta_l} = \left[\mathbf{V}\frac{\nabla\Pi}{\Pi}\right]_{\eta_l} - \frac{1}{\Pi_{\eta_l}}\int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta + \frac{1}{\Pi_{\eta_l}}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0} \tag{93}$$

Remark:

$$\frac{1}{\Pi_{\eta_l}}\int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta = \left[\mathbf{S}\left(\nabla\mathbf{V} + \frac{\partial B}{\partial\Pi}\mathbf{V}\nabla\Pi_{\mathrm{s}}\right)\right]_{\eta_l}$$

5.3 The fully compressible non-hydrostatic model (NHEE).

These equations are well described in (IDNHPB) and we will recall here some basics. See (IDNHPB) for more details, especially for the intermediate calculations leading to the following form of the equations.
In the NHEE non-hydrostatic model there are two additional prognostic variables, one linked with the pressure departure, and the other one with the vertical divergence. For the equations already existing in the 3D primitive equation model:

- The momentum equation is slightly modified (especially the pressure gradient term).
- The temperature equation is slightly modified.
- Continuity equation computes the evolution of $\log \Pi_s$ and is not modified.
- Equations for GFL variables are not modified.

In the fully compressible NHEE model, the two additional prognostic variables are the following ones:

- in the pressure departure equation, prognostic variable is

$$\hat{Q} = \log(\frac{p}{\Pi})$$

- in the vertical divergence equation, prognostic variable is d (vertical divergence). There is a possibility to take $d_4 = d + \mathtt{X}$, where:

$$\mathtt{X} = \frac{p}{\frac{\partial \Pi}{\partial \eta} RT} \nabla \Phi \left(\frac{\partial \mathbf{V}}{\partial \eta} \right) \quad (94)$$

5.3.1 Momentum equation.

Compared to the hydrostatic model, the pressure gradient term now writes:

$$\frac{\partial p}{\partial \Pi} \nabla \Phi + RT \frac{\nabla p}{p}$$

It is highly desirable to write $\nabla p/p$ rather than $\nabla(\log p)$ because the discretisation of this term is not exactly a discretisation of $\nabla(\log p)$.
Lagrangian tendency: Coriolis force can be treated explicitly ($\delta_{\mathbf{V}}$=0) or implicitly ($\delta_{\mathbf{V}}$=1) in the Lagrangian equation.

$$\frac{d\left(\mathbf{V} + \delta_{\mathbf{V}}(2\mathbf{\Omega} \wedge \mathbf{r})\right)}{dt} = [-2(1-\delta_{\mathbf{V}})(\mathbf{\Omega} \wedge \mathbf{V})] - \frac{\partial p}{\partial \Pi} \nabla \Phi - RT \frac{\nabla(p)}{p} + \mathbf{F_V} \quad (95)$$

$\mathbf{F_V}$ is the physical contribution on horizontal wind.
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial \mathbf{V}}{\partial t} = -2\mathbf{\Omega} \wedge \mathbf{V} - \frac{\partial p}{\partial \Pi} \nabla \Phi - RT \frac{\nabla p}{p} - \mathbf{V} \nabla \mathbf{V} - \dot{\eta} \frac{\partial \mathbf{V}}{\partial \eta} - \left(\frac{U}{a} \tan \theta_{\mathrm{bne}} \right) \mathbf{k} \wedge \mathbf{V} + \mathbf{F_V} \quad (96)$$

Advection ($-\mathbf{V}\nabla\mathbf{V}$) and curvature ($-\left(\frac{U}{a}\tan\theta_{\mathrm{bne}}\right)\mathbf{k}\wedge\mathbf{V}$) terms can be rearranged, in order to show the divergence and vorticity in the equation and to eliminate the meridian derivatives of U and V:

$$\frac{\partial \mathbf{V}}{\partial t} = -2\mathbf{\Omega} \wedge \mathbf{V} - \frac{\partial p}{\partial \Pi} \nabla \Phi - RT \frac{\nabla p}{p} + \mathbf{D_V} - \dot{\eta} \frac{\partial \mathbf{V}}{\partial \eta} + \mathbf{F_V} \quad (97)$$

where $\mathbf{D_V}$ has the same expression as in the hydrostatic model.

5.3.2 Thermodynamic equation.

Compared to the hydrostatic model, there are two differences:

- The adiabatic conversion term now writes $-\frac{RT}{c_{\mathrm{v}}} D_3$, where

$$D_3 = \nabla \mathbf{V} + \frac{p}{\frac{\partial \Pi}{\partial \eta} RT} \nabla \Phi \left(\frac{\partial \mathbf{V}}{\partial \eta} \right) - \frac{gp}{\frac{\partial \Pi}{\partial \eta} RT} \left(\frac{\partial w}{\partial \eta} \right) = D + \mathtt{X} + \frac{R_{\mathrm{a}}}{R} d \quad (98)$$

- The diabatic term F_{T} must be replaced by $\left[\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}} F_{\mathrm{T}}\right]$.

Lagrangian tendency:

$$\frac{dT}{dt} = -\frac{RT}{c_{\mathrm{v}}} D_3 + \left[\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}} F_{\mathrm{T}} \right] \quad (99)$$

$\left[\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}} F_{\mathrm{T}}\right]$ is the physical contribution on temperature.
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial T}{\partial t} = -\mathbf{V} \nabla T - \dot{\eta} \frac{\partial T}{\partial \eta} - \frac{RT}{c_{\mathrm{v}}} D_3 + \left[\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}} F_{\mathrm{T}} \right] \quad (100)$$

5.3.3 Continuity equation and diagnostic expression of some vertical velocities.

Continuity equation is unchanged compared to the hydrostatic equations. The consequence is that the calculation of $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ and ω is unchanged compared to the hydrostatic equations.

5.3.4 Advectable GFL prognostic fields.

Equations of advectable GFL are unchanged compared to the hydrostatic equations.

5.3.5 Relationship for geopotential height gz and pressure depth.

Compared to the hydrostatic model, the perfect gas formula now writes:

$$p = \rho R T \tag{101}$$

The combination of equations (82) and (101) provides the relationship between gz and pressure depth:

$$\frac{\partial(gz)}{\partial\Pi} = -\frac{RT}{\Pi}\frac{\Pi}{p} \tag{102}$$

The vertical integrated formulation of equation (102) writes:

$$gz = gz_{\mathrm{s}} - \int_{\Pi'=\Pi_{\mathrm{s}}}^{\Pi'=\Pi} \frac{RT}{\Pi'}\frac{\Pi'}{p} d\Pi' \tag{103}$$

i.e.:

$$gz = gz_{\mathrm{s}} + \mathbf{G}(RT\Pi/p)$$

5.3.6 Relationship between total geopotential Φ and pressure depth.

$\Phi = gz$. See part (5.3.5) for more details.

5.3.7 Vertical velocity w and vertical divergence d equations.

$*$ Using d as prognostic variable: The relationship between d and w writes:

$$d = -\frac{gp}{R_{\mathrm{a}}T\frac{\partial\Pi}{\partial\eta}}\left(\frac{\partial w}{\partial\eta}\right) \tag{104}$$

Inverting (104) gives:

$$gw = gw_{\mathrm{surf}} - \int_{\Pi'=\Pi_{\mathrm{s}}}^{\Pi'=\Pi} \frac{R_{\mathrm{a}}T\Pi' d}{p}\frac{d\Pi'}{\Pi'} \tag{105}$$

i.e:

$$gw = gw_{\mathrm{surf}} + \mathbf{G}(R_{\mathrm{a}}T\Pi d/p)$$

Surface condition is given by:

$$gw_{\mathrm{surf}} = \mathbf{V}_{\mathrm{surf}}\nabla\Phi_{\mathrm{s}} \tag{106}$$

Lagrangian tendency:

$$\frac{dd}{dt} = -dD_3 + d\nabla\mathbf{V} - \frac{gp}{R_{\mathrm{a}}T\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta} + \frac{gp}{R_{\mathrm{a}}T\frac{\partial\Pi}{\partial\eta}}(\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right) + F_{\mathrm{d}} \tag{107}$$

The physical contribution of d is linked to F_{w} which is the physical contribution of w, and also to $F_{\mathrm{m}}^{'}$ by the following relationship:

$$F_{\mathrm{d}} = -\frac{d}{\left[\frac{\partial\Pi}{\partial\eta}\right]}F_{\mathrm{m}}^{'} - \left[\frac{gp}{R_{\mathrm{a}}T\frac{\partial\Pi}{\partial\eta}}\frac{\partial F_{\mathrm{w}}}{\partial\eta}\right] \tag{108}$$

One can notice that the RHS of equation (107) contains the term $\frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta}$ which requires the computation of $\frac{dw}{dt}$ (at least its adiabatic part).

Equation of w is:

$$\frac{dw}{dt} = g\frac{\partial(p-\Pi)}{\partial\Pi} + F_{\mathrm{w}} \tag{109}$$

At the surface we have to use another equation (which requires some assumptions about the surface wind):

$$g\frac{dw_{\mathrm{surf}}}{dt} = \frac{d\mathbf{V}_{\mathrm{surf}}}{dt}\nabla\Phi_{\mathrm{s}} + \mathbf{V}_{\mathrm{surf}}\left[\mathbf{V}_{\mathrm{surf}}\nabla(\nabla\Phi_{\mathrm{s}})\right] + F_{\mathrm{w_{surf}}} \tag{110}$$

Value of term $\frac{d\mathbf{V}_{\mathrm{surf}}}{dt}\nabla\Phi_{\mathrm{s}} + \mathbf{V}_{\mathrm{surf}}\left[\mathbf{V}_{\mathrm{surf}}\nabla(\nabla\Phi_{\mathrm{s}})\right]$ contains the horizontal second derivatives of surface orography and Coriolis term (cf. equation (32) of Geleyn and Bubnová, 1995).
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial d}{\partial t} = -\mathbf{V}\nabla d - \dot{\eta}\frac{\partial d}{\partial \eta} - dD_3 + d\nabla\mathbf{V} - \frac{gp}{R_{\mathrm{a}}T\frac{\partial \Pi}{\partial \eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial \eta} + \frac{gp}{R_{\mathrm{a}}T\frac{\partial \Pi}{\partial \eta}}(\nabla w)\left(\frac{\partial \mathbf{V}}{\partial \eta}\right) + F_{\mathrm{d}} \tag{111}$$

$*$ **Using d_4 as prognostic variable:** Eulerian equation of d_4 is:

$$\frac{\partial d_4}{\partial t} = \frac{\partial d}{\partial t} + \frac{\partial \mathtt{X}}{\partial t} \tag{112}$$

The calculation of $\frac{\partial \mathtt{X}}{\partial t}$ is not done by an Eulerian temporal advection but simply by a diagnostic evaluation and the management of this term will be explained in section (13).

5.3.8 Pressure departure equation.

The prognostic variable currently coded is (option **NPDVAR**=2):

$$\hat{Q} = \log\frac{p}{\Pi}$$

Lagrangian tendency:

$$\frac{d\hat{Q}}{dt} = -\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}}D_3 - \frac{\omega}{\Pi} + \frac{c_{\mathrm{p}}}{c_{\mathrm{v}}T}F_{\mathrm{T}} \tag{113}$$

Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial \hat{Q}}{\partial t} = -\mathbf{V}\nabla\hat{Q} - \dot{\eta}\frac{\partial \hat{Q}}{\partial \eta} - \frac{c_{\mathrm{p}}}{c_{\mathrm{v}}}D_3 - \frac{\omega}{\Pi} + \frac{c_{\mathrm{p}}}{c_{\mathrm{v}}T}F_{\mathrm{T}} \tag{114}$$

5.3.9 Geopotential equation.

This equation is not used in the NHEE model but we can mention it:

$$\frac{d\Phi}{dt} = gw + F_{\Phi} \tag{115}$$

Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial \Phi}{\partial t} = -\mathbf{V}\nabla\Phi - \dot{\eta}\frac{\partial \Phi}{\partial \eta} + gw + F_{\Phi} \tag{116}$$

5.4 The quasi compressible non-hydrostatic model (NHQE).

In the NHQE non-hydrostatic model there is one additional prognostic variable (vertical divergence variable) and one additional diagnostic variable (pressure departure variable). For the equations already existing in the 3D primitive equation model:

- The momentum equation is slightly modified (especially the pressure gradient term).
- The temperature equation is slightly modified (prognostic variable T is replaced by $\tilde{T}$).
- Continuity equation computes the evolution of $\log\Pi_{\mathrm{s}}$ and is not modified.
- Equations for GFL variables are not modified.

In the quasi compressible NHQE model, the two additional prognostic and diagnostic variables are the following ones:

- pressure departure variable $\hat{Q} = \log(\frac{p}{\Pi})$ is a diagnostic variable; diagnostic is done via the linear system.

- in the vertical divergence equation, prognostic variable is d_4. $d_4 = d + \mathtt{X} = d + \mathtt{X}_\mathrm{S} + \mathtt{X}_\mathrm{D}$, where:

$$\mathtt{X}_\mathrm{S} = \frac{\Pi}{\frac{\partial \Pi}{\partial \eta} R\tilde{T}} \nabla\Phi \left(\frac{\partial \mathbf{V}}{\partial \eta} \right) \tag{117}$$

$$\mathtt{X}_\mathrm{D} = (1-\kappa)\left(\frac{\omega}{\Pi} + \mathbf{S}D \right) \tag{118}$$

Integral operator $\mathbf{S}$ is defined by:

$$[\mathbf{S}Z]_\eta = \frac{1}{\Pi_\eta} \int_{top}^{\eta} \frac{\partial \Pi}{\partial \eta} Z d\eta'$$

5.4.1 Momentum equation.

Lagrangian tendency: Coriolis force can be treated explicitly ($\delta_\mathbf{V}$=0) or implicitly ($\delta_\mathbf{V}$=1) in the Lagrangian equation.

$$\frac{d\left(\mathbf{V} + \delta_\mathbf{V}(2\boldsymbol{\Omega} \wedge \mathbf{r})\right)}{dt} = [-2(1-\delta_\mathbf{V})(\boldsymbol{\Omega} \wedge \mathbf{V})] - \exp(\kappa\hat{Q}) \left(1 + \Pi \frac{\partial \hat{Q}}{\partial \Pi} \right) \nabla\Phi - R\tilde{T}\exp(\kappa\hat{Q}) \left(\frac{\nabla\Pi}{\Pi} + \nabla\hat{Q} \right) + \mathbf{F_V} \tag{119}$$

$\mathbf{F_V}$ is the physical contribution on horizontal wind.
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial \mathbf{V}}{\partial t} = [-2(\boldsymbol{\Omega} \wedge \mathbf{V})] - \exp(\kappa\hat{Q}) \left(1 + \Pi \frac{\partial \hat{Q}}{\partial \Pi} \right) \nabla\Phi - R\tilde{T}\exp(\kappa\hat{Q}) \left(\frac{\nabla\Pi}{\Pi} + \nabla\hat{Q} \right) - \mathbf{V}\nabla\mathbf{V} - \dot{\eta}\frac{\partial \mathbf{V}}{\partial \eta} - \left(\frac{U}{a}\tan\theta_\mathrm{bne} \right) \mathbf{k} \wedge \mathbf{V} + \mathbf{F_V} \tag{120}$$

Advection ($-\mathbf{V}\nabla\mathbf{V}$) and curvature ($-\left(\frac{U}{a}\tan\theta_\mathrm{bne}\right)\mathbf{k}\wedge\mathbf{V}$) terms can be rearranged, in order to show the divergence and vorticity in the equation and to eliminate the meridian derivatives of U and V: see hydrostatic momentum equation.

5.4.2 Thermodynamic equation.

Lagrangian tendency:

$$\frac{d\tilde{T}}{dt} = \frac{R\tilde{T}}{c_\mathrm{p}} \frac{\omega}{\Pi} + \exp(-\kappa\hat{Q}))F_\mathrm{T} \tag{121}$$

Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial \tilde{T}}{\partial t} = -\mathbf{V}\nabla T - \dot{\eta}\frac{\partial T}{\partial \eta} + \frac{R\tilde{T}}{c_\mathrm{p}} \frac{\omega}{\Pi} + \exp(-\kappa\hat{Q}))F_\mathrm{T} \tag{122}$$

5.4.3 Continuity equation and diagnostic expression of some vertical velocities.

Continuity equation is unchanged compared to the hydrostatic equations. The consequence is that the calculation of $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ and ω is unchanged compared to the hydrostatic equations.

5.4.4 Advectable GFL prognostic fields.

Equations of advectable GFL are unchanged compared to the hydrostatic equations.

5.4.5 Relationship for geopotential height gz and pressure depth.

Calculation of gz is done according to formula:

$$gz = gz_\mathrm{s} - \int_{\Pi'=\Pi_\mathrm{s}}^{\Pi'=\Pi} \frac{R\tilde{T}}{\Pi'} d\Pi' \tag{123}$$

i.e. $gz = gz_\mathrm{s} + \mathbf{G}(R\tilde{T})$.

5.4.6 Relationship between total geopotential Φ and pressure depth.

$\Phi = gz$. See part (5.4.5) for more details.

5.4.7 Vertical velocity w and vertical divergence d equations.

$*$ **Using d as prognostic variable:** In the NHQE model, d is defined with the moist R at the denominator. The relationship between d and w writes:

$$d = -\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\left(\frac{\partial w}{\partial\eta}\right) \tag{124}$$

Inverting (124) gives:

$$gw = gw_{\text{surf}} - \int_{\Pi'=\Pi_{\text{s}}}^{\Pi'=\Pi} R\tilde{T}d\frac{d\Pi'}{\Pi'} \tag{125}$$

i.e:

$$gw = gw_{\text{surf}} + \mathbf{G}(R\tilde{T}d)$$

Surface condition is given by equation (106).
Lagrangian tendency:

$$\frac{dd}{dt} = -d(d+\mathtt{X}_{\text{S}}) - \frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\text{ad}}}{\partial\eta} + \frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}(\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right) + F_{\text{d}} \tag{126}$$

The physical contribution of d is linked to F_{w} which is the physical contribution of w, and also to $F_{\text{m}}^{'}$ by the following relationship:

$$F_{\text{d}} = -\frac{d}{\left[\frac{\partial\Pi}{\partial\eta}\right]}F_{\text{m}}^{'} - \left[\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial F_{\text{w}}}{\partial\eta}\right] \tag{127}$$

One can notice that the RHS of equation (126) contains the term $\frac{\partial\left[\frac{dw}{dt}\right]_{\text{ad}}}{\partial\eta}$ which requires the computation of $\frac{dw}{dt}$ (at least its adiabatic part).

Equation of w is:

$$\frac{dw}{dt} = \frac{g}{\kappa}\left[\frac{\partial\Pi(\exp(\kappa\hat{Q})-1)}{\partial\Pi} + (\kappa-1)(\exp(\kappa\hat{Q})-1)\right] + F_{\text{w}} \tag{128}$$

This is equivalent to write:

$$\frac{dw}{dt} = g\left[\frac{\partial p^{\kappa}}{\partial\Pi^{\kappa}} - 1\right] + F_{\text{w}}$$

At the surface we have to use another equation (which requires some assumptions about the surface wind):

$$g\frac{dw_{\text{surf}}}{dt} = \frac{d\mathbf{V}_{\text{surf}}}{dt}\nabla\Phi_{\text{s}} + \mathbf{V}_{\text{surf}}\left[\mathbf{V}_{\text{surf}}\nabla(\nabla\Phi_{\text{s}})\right] + F_{\text{w}_{\text{surf}}} \tag{129}$$

Value of term $\frac{d\mathbf{V}_{\text{surf}}}{dt}\nabla\Phi_{\text{s}} + \mathbf{V}_{\text{surf}}\left[\mathbf{V}_{\text{surf}}\nabla(\nabla\Phi_{\text{s}})\right]$ contains the horizontal second derivatives of surface orography and Coriolis term (cf. equation (32) of Geleyn and Bubnová, 1995).
Eulerian tendency: the Eulerian equation writes:

$$\frac{\partial d}{\partial t} = -\mathbf{V}\nabla d - \dot{\eta}\frac{\partial d}{\partial\eta} - d(d+\mathtt{X}_{\text{S}}) - \frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\text{ad}}}{\partial\eta} + \frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}(\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right) + F_{\text{d}} \tag{130}$$

$*$ **Using d_4 as prognostic variable:**

Eulerian equation of d_4 is:

$$\frac{\partial d_4}{\partial t} = \frac{\partial d}{\partial t} + \frac{\partial\mathtt{X}}{\partial t} \tag{131}$$

The calculation of $\frac{\partial\mathtt{X}}{\partial t}$ is not done by an Eulerian temporal advection but simply by a diagnostic evaluation and the management of this term will be explained in section (13).

5.4.8 Closure equation.

The quasi-elastic assumption assumes the following constraint:

$$d_4 + \mathbf{S}_{\kappa}D = 0 \tag{132}$$

$\mathbf{S}_{\kappa}$ denotes $I - (1-\kappa_{\text{d}})\mathbf{S}$.

6 Some other diagnosed quantities.

6.1 c_p, R and κ.

For c_p (air calorific capacity at constant pressure):

$$c_\mathrm{p} = c_{\mathrm{p_d}}(1 - q - q_\mathrm{l} - q_\mathrm{i}) + c_{\mathrm{p_v}} q + c_{\mathrm{p_l}} q_\mathrm{l} + c_{\mathrm{p_i}} q_\mathrm{i} \tag{133}$$

where $c_{\mathrm{p_d}}$ is the dry air calorific capacity at constant pressure, $c_{\mathrm{p_v}}$ is the water vapour calorific capacity at constant pressure, $c_{\mathrm{p_l}}$ is the liquid water calorific capacity, and $c_{\mathrm{p_i}}$ is the ice calorific capacity.
For R (air constant):

$$R = R_\mathrm{d}(1 - q - q_\mathrm{l} - q_\mathrm{i}) + R_\mathrm{v} q \tag{134}$$

where R_d is the dry air constant and R_v is the water vapour air constant.
For κ:

$$\kappa = \frac{R}{c_\mathrm{p}} \tag{135}$$

The ratio $c_\mathrm{v}/c_\mathrm{p}$ is given by equation:

$$\frac{c_\mathrm{v}}{c_\mathrm{p}} = 1 - \kappa \tag{136}$$

6.2 $\nabla(RT)$.

Its expression writes:

$$\nabla(RT) = (R_\mathrm{v} - R_\mathrm{d}) T \nabla q + R \nabla T \tag{137}$$

6.3 Potential temperature PT and its horizontal gradient.

Expression of PT writes:

$$PT = T \left[\frac{\Pi}{\Pi_{1000}}\right]^{-\kappa} \tag{138}$$

where $\Pi_{1000} = 100000$ Pa
Its horizontal gradient writes:

$$\nabla(PT) = PT \left(\frac{\nabla T}{T} - \kappa \frac{\nabla \Pi}{\Pi}\right) \tag{139}$$

6.4 Virtual potential temperature PTV.

Its expression writes:

$$PTV = TV \left[\frac{\Pi}{\Pi_{1000}}\right]^{-\kappa_\mathrm{d}} = \frac{R}{R_\mathrm{d}} T \left[\frac{\Pi}{\Pi_{1000}}\right]^{-\kappa_\mathrm{d}} \tag{140}$$

where $\Pi_{1000} = 100000$ Pa; $\kappa_\mathrm{d} = R_\mathrm{d}/c_{\mathrm{p_d}}$.

6.5 Equivalent potential temperature PTE.

Its expression writes:

$$PTE = PT\ \exp\left[\frac{L q_\mathrm{sat}}{[c_{\mathrm{p_{sat}}}] T}\right] \tag{141}$$

where the potential temperature PT has been defined by formula (138). Here L is the vapour water latent heat per mass unit. Expression of $[c_{\mathrm{p_{sat}}}]$ writes:

$$[c_{\mathrm{p_{sat}}}] = c_{\mathrm{p_d}} + (c_{\mathrm{p_v}} - c_{\mathrm{p_d}}) q_\mathrm{sat}$$

6.6 Absolute vorticity ζ_abs.

Its definition is given by formula:

$$\zeta_\mathrm{abs} = \zeta + f \tag{142}$$

6.7 Potential vorticity PV.

Expression of PV is:

$$PV = \left[g \frac{\partial V}{\partial \Pi}\right] \left[\frac{M}{a \cos\Theta} \frac{\partial PT}{\partial \Lambda}\right] - \left[g \frac{\partial U}{\partial \Pi}\right] \left[\frac{M}{a} \frac{\partial PT}{\partial \Theta}\right] - g \zeta_\mathrm{abs} \frac{\partial PT}{\partial \Pi} \tag{143}$$

6.8 Shearing deformation SHD and stretching deformation STD.

Spherical geometry:

$$STD = \frac{1}{a\cos\Theta}\frac{\partial U}{\partial \Lambda} - 0.5D \tag{144}$$

$$SHD = \frac{1}{a\cos\Theta}\frac{\partial V}{\partial \Lambda} - 0.5\zeta \tag{145}$$

Plane geometry:

$$STD = \nabla^{\mathrm{u}}U - 0.5D \tag{146}$$

$$SHD = \nabla^{\mathrm{u}}V - 0.5\zeta \tag{147}$$

A total deformation can be defined as follows:

$$DEF = \sqrt{(2SHD)^2 + (2STD)^2} \tag{148}$$

6.9 Hydrostatic vertical divergence d_{hyd}.

Identity of the temperature equation NHEE RHS and hydrostatic RHS gives the following definition for hydrostatic vertical divergence.

$$d_{\mathrm{hyd}} = -\frac{R}{R_{\mathrm{a}}}\left[\frac{c_{\mathrm{v}}}{c_{\mathrm{p}}}\frac{\omega}{\Pi} + \nabla\mathbf{V} + \frac{\Pi}{RT}\nabla\left[gz\right]\frac{\partial\mathbf{V}}{\partial\Pi}\right] \tag{149}$$

Assuming that hydrostatic equations are in quasi compressible equilibrium yields another possible expression for d_{hyd}:

$$d_{\mathrm{hyd}} = -\frac{R}{R_{\mathrm{a}}}\left[-\mathtt{X}_{\mathrm{S}} - \mathtt{X}_{\mathrm{D}} - \mathbf{S}_{\kappa}D\right] \tag{150}$$

6.10 Hydrostatic height coordinate vertical velocity w_{hyd}.

The expression of w_{hyd} matches the following equation:

$$d_{\mathrm{hyd}} = -g\frac{\Pi}{R_{\mathrm{a}}T}\frac{\partial w_{\mathrm{hyd}}/\partial\eta}{\partial\Pi/\partial\eta} \tag{151}$$

which can be rewritten, after a vertical integration:

$$g\left[w_{\mathrm{hyd}}\right]_{\eta=\eta_l} = g\left[w_{\mathrm{hyd}}\right]_{\mathrm{surf}} + \int_{\eta=\eta_l}^{\eta=1}\frac{(\partial\Pi/\partial\eta)R_{\mathrm{a}}T}{\Pi}d_{\mathrm{hyd}}d\eta \tag{152}$$

where:

$$g\left[w_{\mathrm{hyd}}\right]_{\mathrm{surf}} = \left[\mathbf{V}\right]_{\mathrm{surf}}\nabla\Phi_{\mathrm{s}} \tag{153}$$

i.e.:

$$gw_{\mathrm{hyd}} = g\left[w_{\mathrm{hyd}}\right]_{\mathrm{surf}} + \mathbf{G}(R_{\mathrm{a}}Td_{\mathrm{hyd}})$$

6.11 Moisture convergence $CVGQ$.

This quantity is also denoted $MOCON$ and is used as input of some convection schemes. Its expression writes:

$$CVGQ = -\mathbf{V}\nabla q - \dot{\eta}\frac{\partial q}{\partial\eta} \tag{154}$$

This notion can be generalized to any intensive quantity X: for a quantity X one can define $CVGX$ as:

$$CVGX = -\mathbf{V}\nabla X - \dot{\eta}\frac{\partial X}{\partial\eta} \tag{155}$$

6.12 Montgomery potential Φ_{mg} and some other energetic quantities.

Formulae for enthalpy and kinetic energy are guaranteed at least for the hydrostatic model (must we add $0.5w^2$ in the NH model?).

- Montgomery potential: $\Phi_{\mathrm{mg}} = c_{\mathrm{p_d}}T + gz$.
- Dry static energy: $s = c_{\mathrm{p}}T + gz$.
- Moist static energy: $s_{\mathrm{h}} = c_{\mathrm{p}}T + gz + Lq$.
- Enthalpy: $h = c_{\mathrm{p}}T + gz + 0.5 * (U^2 + V^2)$.
- Kinetic energy: $KE = 0.5 * (U^2 + V^2)$.

Here L is the vapour water latent heat per mass unit.

6.13 Angular momentum of components MMA, MMB and MMC.

Axial angular momentum:

$$MMA = [U_{\mathrm{G}} + \Omega a \cos\theta]\, a \cos\theta \tag{156}$$

The two other components of the angular momentum are:

$$MMB = a\left[-\left(U_{\mathrm{G}} + \Omega a \cos\theta\right)\sin\theta \cos(\lambda + \Omega t) + V_{\mathrm{G}} \sin(\lambda + \Omega t)\right] \tag{157}$$

$$MMC = a\left[-\left(U_{\mathrm{G}} + \Omega a \cos\theta\right)\sin\theta \sin(\lambda + \Omega t) - V_{\mathrm{G}} \cos(\lambda + \Omega t)\right] \tag{158}$$

$(U_{\mathrm{G}}; V_{\mathrm{G}})$ are the geographical wind components in a geographical system of coordinates; t is the time since the departure of the model.

6.14 Entropy S.

$*$ **Hydrostatic model:** The total entropy for moist air is given by the following formula:

$$S = c_{\mathrm{p}} \log\left[\frac{T}{T_0}\right] - R \log\left[\frac{\Pi}{\Pi_{1000}}\right] + S_0 \tag{159}$$

where $\Pi_{1000} = 100000$ Pa; $T_0 = 273.15$ K; $S_0 = S(T = T_0; \Pi = \Pi_{1000})$.
It is easier to get the partial entropies S_d (for dry air), S_v (for water vapour), S_l (for liquid water) and S_i (for ice). They are respectively given by the following formulae:

$$S_d = c_{\mathrm{p_d}} \log\left[\frac{T}{T_0}\right] - R_{\mathrm{d}} \log\left[\frac{\Pi_d}{\Pi_{1000}}\right] + S_{d0} \tag{160}$$

$$S_v = c_{\mathrm{p_v}} \log\left[\frac{T}{T_0}\right] - R_{\mathrm{v}} \log\left[\frac{\Pi_v}{\Pi_{1000}}\right] + S_{v0} \tag{161}$$

$$S_l = c_{\mathrm{p_l}} \log\left[\frac{T}{T_0}\right] + S_{l0} \tag{162}$$

$$S_i = c_{\mathrm{p_i}} \log\left[\frac{T}{T_0}\right] + S_{i0} \tag{163}$$

Π_d and Π_v are respectively the partial pressures of dry air and water vapour: $\Pi_v = q\Pi$ and $\Pi_d = (1 - q)\Pi$.
Constants S_{d0}, S_{v0}, S_{l0}, S_{i0} have the following values: $S_{d0} = 6775 Jkg^{-1}K^{-1}$, $S_{v0} = 10320 Jkg^{-1}K^{-1}$, $S_{l0} = 3517 Jkg^{-1}K^{-1}$, $S_{i0} = 2296 Jkg^{-1}K^{-1}$.
The total entropy for moist air writes:

$$S = S_d(1 - q - q_{\mathrm{l}} - q_{\mathrm{i}}) + S_v q + S_l q_{\mathrm{l}} + S_i q_{\mathrm{i}} = S_d + (S_v - S_d)q + (S_l - S_d)q_{\mathrm{l}} + (S_i - S_d)q_{\mathrm{i}}$$

$*$ **Non-hydrostatic model:** The hydrostatic pressures Π, Π_d and Π_v must be replaced by the total pressures p, p_d and p_v.

7 Discretisation of the equations: general aspects.

7.1 Denotations and preliminary remarks.

$*$ Upper index:

- First integration step:

  $+$: $t + \Delta t$ quantity.

  o : t quantity.

  $-$: t quantity.

- Following integration steps:

  $+$: $t + \Delta t$ quantity.

  o : t quantity.

  $-$: $t - \Delta t$ quantity.

$*$ Particular case of the first timestep: Written discretisations are valid from the second integration step. Δt has to be replaced by $\frac{\Delta t}{2}$ for the first integration step (in this case the $t - \Delta t$ quantities are equal to the t quantities).

$*$ The different classes of prognostic variables: Prognostic variables can be split into different classes:

- 3D variables, the equation RHS of which has a non-zero adiabatic contribution and a non-zero semi-implicit correction contribution. They are called "GMV" in the code ("GMV" means "grid-point model variables"). This class of variables includes wind components, temperature (and the two additional non-hydrostatic variables in a non-hydrostatic model). The sub-class of thermodynamic variables includes T, and the two additional non-hydrostatic variables in a non-hydrostatic model. There are **NFTHER** thermodynamic variables.
- 3D "conservative" variables. The equation RHS of these variables has a zero adiabatic contribution, only the diabatic contribution (and the horizontal diffusion contribution) can be non-zero. They are called "GFL" in the code ("GFL" means "grid-point fields"). We can divide this class of variables into two sub-classes:
  - Historical variables (they are advectable). This class of variables includes for example:
    - humidity q.
    - liquid water $q_{\rm l}$.
    - ice $q_{\rm i}$.
    - cloud fraction $q_{\rm a}$.
    - rain $q_{\rm r}$.
    - snow $q_{\rm s}$.
    - convective liquid water $q_{\rm lconv}$.
    - convective ice $q_{\rm iconv}$.
    - convective rain $q_{\rm rconv}$.
    - convective snow $q_{\rm sconv}$.
    - graupels $q_{\rm g}$.
    - hail $q_{\rm h}$.
    - ozone $\mathcal{O}3$.
    - aerosols $q_{\rm AERO}$.
    - TKE ($q_{\rm TKE}$).
    - Variables for EFB turbulent parameterization ($EFB1$ to $EFB3$).
    - some extra fields.
  - Some non-advectable pseudo-historical variables which are often simple diagnostics which must be conserved from one timestep to the following one. This class of variables includes for example:
    - atmospheric total liquid water content for radiation $q_{\rm lrad}$.
    - atmospheric total ice content for radiation $q_{\rm irad}$.
    - convective precipitation flux ($q_{\rm CPF}$).
    - stratiform precipitation flux ($q_{\rm SPF}$).
    - second-order flux for AROME ($q_{\rm SRC}$).
    - forcings (1D model) ($q_{\rm FORC}$).
    - easy diagnostics for AROME physics ($q_{\rm EZDIAG}$).
    - greenhouse gases for ECMWF physics ($q_{\rm GHG}$).

* chemistry for ECMWF physics ($q_{\rm CHEM}$).
* ERA40 reanalysis fields (ECMWF) ($q_{\rm ERA40}$).
* moisture convergence for MF physics ($q_{\rm CVGQ}$).
* total humidity variation for HIRLAM physics ($q_{\rm QVA}$).
* standard deviation of the saturation depression ($q_{\rm SDSAT}$).
* convective vertical velocity ($q_{\rm CVV}$).
* downdraught mesh fraction for ALARO physics ($q_{\rm DAL}$).
* downdraught vertical velocity for ALARO physics ($q_{\rm DOM}$).
* updraught mesh fraction for ALARO physics ($q_{\rm UAL}$).
* updraught vertical velocity for ALARO physics ($q_{\rm UOM}$).
* pseudo-historic convective cloudiness for ALARO physics ($q_{\rm UNEBH}$).
* total turbulent energy ($q_{\rm TTE}$).
* prognostic mixing length ($q_{\rm MXL}$).
* shear source term for turbulence ($q_{\rm SHTUR}$).
* flux form source term for turbulence: moisture ($q_{\rm FQTUR}$).
* flux form source term for turbulence: enthalpy ($q_{\rm FSTUR}$).
* Rasch-Kristjansson enthalpy tendency for ALARO physics ($q_{\rm RKTH}$).
* Rasch-Kristjansson water vapour tendency for ALARO physics ($q_{\rm RKTQV}$).
* Rasch-Kristjansson condensates tendency for ALARO physics ($q_{\rm RKTQC}$).
* pseudo-historic entrainment for ALARO physics ($q_{\rm UEN}$).
* quantities related to methane ($q_{\rm LRCH4}$).
* output aerosols for diagnostics ($q_{\rm AEROUT}$).
* output fields from UV processor ($q_{\rm UVP}$).
* output fields from ECMWF physics ($q_{\rm PHYS}$).
* diagnostic fields for NORO GWD scheme ($q_{\rm NOGW}$).
* semi-Lagrangian dynamics diagnostic fields ($q_{\rm SLDIA}$).
* extra fields for "CRM" model (ECMWF) ($q_{\rm CRM}$).
* specific gas constant ($q_{\rm SPEC}$).
* LIMA prognostic fields ($q_{\rm LIMA}$).
* aerosol optical thicknesses ($q_{\rm AERAOT}$).
* aerosol lidar simulator ($q_{\rm AERLISI}$).

- 2D variables, the equation RHS of which mixes 3D and 2D terms, has a non-zero adiabatic contribution and a non-zero semi-implicit correction contribution. They are called "GMVS" in the code ("GMVS" means "grid-point model variables for surface"). This class of variables includes the logarithm of surface pressure (continuity equation).
- 2D surface variables used in the physics. They are purely grid-point variables and they represent data at the surface or into different layers of the soil. This class of variables includes for example the water content of the superficial reservoir of the soil, or the snow depth. Contrary to the "GMVS" variables they are never advected. They include prognostic surface fields and diagnostic surface fields.

7.2 Discretisation.

General case: 3D variable in a 3D model:

Equation

$$\frac{\partial X}{\partial t} = -\mathbf{V}\nabla X - \dot{\eta}\frac{\partial X}{\partial \eta} + \mathcal{A} + F \tag{164}$$

is discretised as follows for unlagged physics:

$$(X - \Delta t\beta\mathcal{L})^+ = X^- + [2\Delta t\mathcal{A} - 2\Delta t\beta\mathcal{L} - 2\Delta t\mathbf{V}\nabla X - 2\Delta t\dot{\eta}\frac{\partial X}{\partial \eta}]^o + [\Delta t\beta\mathcal{L} + 2\Delta tF]^- \tag{165}$$

and as follows for lagged physics:

$$(X - \Delta t\beta\mathcal{L})^+ = X^- + [2\Delta t\mathcal{A} - 2\Delta t\beta\mathcal{L} - 2\Delta t\mathbf{V}\nabla X - 2\Delta t\dot{\eta}\frac{\partial X}{\partial \eta}]^o + [\Delta t\beta\mathcal{L}]^- + [2\Delta tF]^+ \tag{166}$$

$\mathcal{A}$ is the total (non linear and linear) adiabatic contribution, $\mathcal{L}$ is the linear adiabatic contribution (semi-implicitly treated), F is the physical contribution, $\mathbf{V}\nabla X$ is the horizontal advection, $\dot{\eta}\frac{\partial X}{\partial \eta}$ is the vertical advection. All quantities are evaluated at the same grid point F. Calculation of X^+ knowing $(X - \Delta t\beta\mathcal{L})^+$ is done in spectral space. The remaining calculations are done in grid-point space. Horizontal diffusion is not taken in account in this formula and is done in spectral space.
When doing lagged physics one first computes, without physics

$$X_{\rm prov} = (X - \Delta t\beta\mathcal{L})^+ - (\Delta t\beta\mathcal{L})^- + (2\Delta t\beta\mathcal{L})^o = X^- + [2\Delta t\mathcal{A} - 2\Delta t\mathbf{V}\nabla X - 2\Delta t\dot{\eta}\frac{\partial X}{\partial \eta}]^o$$

then does the lagged physics, then computes $(X - \Delta t\beta\mathcal{L})^+$

Particuliar cases:

- GFL variables: terms $\mathcal{A}$ and $\mathcal{L}$ are zero for these variables. For non advected GFL variables, the advection terms are replaced by zero.
- 2D variable in a 3D model (continuity equation): see general case, but:
  - $\mathcal{A} + \mathbf{V}\nabla X$ is a 2D quantity which is the vertical integral of a 3D quantity.
  - $\mathcal{L}$ is a 2D quantity which is the vertical integral of a 3D quantity.
  - F is a 2D quantity which is the vertical integral of a 3D quantity.
  - $\dot{\eta}\frac{\partial X}{\partial \eta}$ is zero.
- 2D variable in a 2D model: see general case, but $\dot{\eta}\frac{\partial X}{\partial \eta}$ is zero in this case.

7.3 Vertical discretisation in a 3D model.

There are two "main" ways of managing the vertical discretisations: a conventional one which has the property of conserving some global invariants, and another discretisation using a finite elements representation.

$*$ **Finite difference vertical (VFD) discretisation (LVERTFE=.F.):** The main features of this type of discretisation are:

- Discretisations mix half level and full level quantities.
- $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ is computed at half levels.
- The prognostic 3D variables are computed at full levels.
- When computing a vertical integral from the surface (resp. the top), only quantities between the surface (resp. the top) are involved; integrals first give half level quantities, then full level quantities by adding a residual term. This is the case for example for the divergence integral term used in the continuity equation and in the diagnostic equation giving some vertical velocities.
- The half level hydrostatic pressure always matches the definition of the hybrid vertical coordinate. On the contrary, the full level hydrostatic pressure does not always do that (that depends on the way to compute it, there are different possible options).

$*$ **Finite element vertical (VFE) discretisation (LVERTFE=.T.):** The main features of this type of discretisation are:

- Discretisations generally avoid computation of half level quantities.
- $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ is computed at full levels.
- The prognostic 3D variables are computed at full levels.
- All the vertical integrals of the adiabatic part use a matricial multiplication with special coefficients computed in setup routines. When computing a vertical integral from the surface or the top, all the full levels are involved. Vertical integrals directly give full level quantities. The lines of the integral matricial operator will be denoted by $[\mathcal{R}_{\rm inte}]_{(surf,l)}$ (resp. $[\mathcal{R}_{\rm inte}]_{(top,l)}$) for an integral from the surface (resp. the top) to the layer l. The matricial operator will be denoted by $[\mathcal{R}_{\rm inte}]_{(top,surf)}$ for an integral from the top to the surface. For example, $\int_{\eta=0}^{\eta=\eta_l} X d\eta$ is discretised by the scalar product $[\mathcal{R}_{\rm inte}]_{(top,l)}\langle X\rangle$ where $[\mathcal{R}_{\rm inte}]_{(top,l)}$ is the l-th line of the matricial operator $\mathcal{R}_{\rm inte}$ and $\langle X\rangle$ is the vector of coordinates $(X_1; X_2; ...; X_l; ...; X_L)$ (values of X at full levels).
- All the vertical derivatives of the adiabatic part use a matricial multiplication with special coefficients computed in setup routines. When computing a vertical derivative, all the full levels are involved. Vertical derivatives directly give full level quantities. The lines of the matricial operator will be denoted by $[\mathcal{R}_{\rm deri}]_l$ for a derivative to the layer l. For example, $\frac{\partial X}{\partial \eta}$ is discretised by the scalar product $[\mathcal{R}_{\rm deri}]_l\langle X\rangle$ where $[\mathcal{R}_{\rm deri}]_l$ is the l-th line of the matricial operator $\mathcal{R}_{\rm deri}$ and $\langle X\rangle$ is the vector of coordinates $(X_1; X_2; ...; X_l; ...; X_L)$ (values of X at full levels).
- $\mathcal{R}_{\rm deri}$ is not the exact inverse of $\mathcal{R}_{\rm inte}$, and it is not recommended to apply $\mathcal{R}_{\rm deri}$ and $\mathcal{R}_{\rm inte}$ to the same quantities (we should avoid to make appear $\mathcal{R}_{\rm deri}(\mathcal{R}_{\rm inte}(X))$ where this is X which actually appears): this point may require incremental treatment of vertical derivatives in some parts of the code.
- The finite elements basis for $\mathcal{R}_{\rm inte}$ can be a set of linear functions or a set of Hermite cubic functions.
- Both full level and half level hydrostatic pressure always match the definition of the hybrid vertical coordinate.

More details are given in (Untch and Hortal, 2001), (Untch and Hortal, 2004) and in appendix 3.
The operator $[\mathcal{R}_{\rm inte}]$ is stored in the array **RINTE** or **RINTBF11** of **YOMVERT** and pre-computed in the setup under **SUVERTFE**. Vertical integrations are done in the routine **VERINT**.
The operator $[\mathcal{R}_{\rm deri}]$ is stored in one of the arrays **RDER..** of **YOMVERT** and pre-computed in the setup under **SUVERTFE**. Vertical derivatives are done in the routine **VERDER**. It is not exactly the inverse of the operator $[\mathcal{R}_{\rm inte}]$.
These arrays are attributes of structure **TVFE** (module **YOMVERT**).

8 The hydrostatic pressure based η vertical coordinate.

8.1 Definition of the η-coordinate and calculation of the hydrostatic pressure at half levels.

The definition of the η-coordinate yields the following relationship:

$$\Pi = A(\eta) + B(\eta)\Pi_{\mathrm{s}} \tag{167}$$

The coefficients A and B only depend on η. One has the following additional relationships:

- $\eta = 1$ at the surface.
- $\eta = 0$ at the top of the model.
- $A(\eta = 1) = 0$ and $B(\eta = 1) = 1$ at the surface.
- $A(\eta = 0) = \Pi_{\mathrm{top}}$ and $B(\eta = 1) = 0$ at the top of the model.

A and B are given at half levels, so:

$$\Pi_{\bar{l}} = A_{\bar{l}} + B_{\bar{l}}\Pi_{\mathrm{s}} \tag{168}$$

8.2 Calculation of η at half and full levels.

The explicit definition of η is used at two locations: the semi-Lagrangian vertical interpolator and the VFE integral and derivative operators.

There are two ways to compute η at half levels, according to the value of the namelist variable **LREGETA** (or **LVFE_REGETA** for the version of η used in the VFE operators, denoted by η_{vfe}):

- **LREGETA**=.TRUE. (regular spacing of η):

$$\eta_{\bar{l}} = \frac{\bar{l}}{L} \tag{169}$$

- **LREGETA**=.FALSE. (irregular spacing of η):

$$\eta_{\bar{l}} = \frac{A_{\bar{l}}}{\Pi_{\mathrm{sst}}} + B_{\bar{l}} \tag{170}$$

LREGETA is relevant only in the semi-Lagrangian scheme. **LVFE_REGETA** is relevant only if **LVERTFE**=.T. .

For η_{vfe} there is an additional option (**LVFE_REGETA**=F, **LVFE_CENTRI**=T) which gives something intermediate between the original **LVFE_REGETA**=F formulation and the original **LVFE_REGETA**=T formulation.

$$\eta_{\bar{l}} = \frac{\sum_{i=1}^{i=\bar{l}} (\Pi_{\mathrm{st}\,i} - \Pi_{\mathrm{st}\,i-1})^q}{\sum_{i=1}^{i=L} (\Pi_{\mathrm{st}\,i} - \Pi_{\mathrm{st}\,i-1})^q} \tag{171}$$

where q is an exponent between 0 and 1; $q = 0$ provides **LVFE_REGETA**=T; $q = 1$ provides the original **LVFE_REGETA**=F.

η at full levels is always computed according the following way:

$$\eta_l = 0.5(\eta_{\bar{l}} + \eta_{\bar{l}-1}) \tag{172}$$

8.3 Calculation of the hydrostatic pressure at full levels.

There are several ways to compute the hydrostatic pressure at full levels, according to the value of the namelist variables **LVERTFE**, **NDLNPR** and **LAPRXPK**.
The following denotations are used:

- **VFD0**: **LVERTFE**=.F., **NDLNPR**=0.
- **VFD1**: **LVERTFE**=.F., **NDLNPR**=1.
- **VFD2**: **LVERTFE**=.F., **NDLNPR**=2.
- **VFE0**: **LVERTFE**=.T., **NDLNPR**=0.

Discretisation of full-level hydrostatic pressure:

- **VFD0** with **LAPRXPK**=.TRUE. :
$$\Pi_l = 0.5(\Pi_{\bar{l}} + \Pi_{\bar{l}-1}) \tag{173}$$
Remark: this equation matches the definition of the hybrid, coordinate, taking $A_l = 0.5(A_{\bar{l}} + A_{\bar{l}-1})$ and $B_l = 0.5(B_{\bar{l}} + B_{\bar{l}-1})$.
For layer $l = 1$:
$$\Pi_{l=1} = 0.5\Pi_{\bar{l}=1} \tag{174}$$
- **VFD0** with **LAPRXPK**=.FALSE. :
$$\log(\Pi_l) = \frac{\Pi_{\bar{l}} \log(\Pi_{\bar{l}}) - \Pi_{\bar{l}-1} \log(\Pi_{\bar{l}-1})}{\Pi_{\bar{l}} - \Pi_{\bar{l}-1}} - 1 \tag{175}$$
Remark: this equation does not match the definition of the hybrid coordinate but it matches the identity $\log(\Pi_l) = \log(\Pi_{\bar{l}}) - \alpha_l$ (see section 8.5 for definition of α).
For layer $l = 1$ (replace $\Pi_{\bar{l}-1}$ and $\Pi_{\bar{l}-1} \log(\Pi_{\bar{l}-1})$ by zero in formula (175)):
$$\Pi_{l=1} = \Pi_{\bar{l}=1} / \exp(1) \tag{176}$$
- **VFD1** and **VFD2**:
$$\Pi_l = \sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}} \tag{177}$$
Remark: this equation does not match the definition of the hybrid coordinate but it matches the identity $\log(\Pi_l) = 0.5(\log(\Pi_{\bar{l}}) + \log(\Pi_{\bar{l}-1}))$.
For layer $l = 1$:
$$\Pi_{l=1} = \Pi_{\bar{l}=1}/\delta_{l=1} \tag{178}$$
where expression of $\delta_{l=1}$ is given in section 8.5.
- **VFE0**: for all layers (including the layer number 1):
$$\Pi_l = A_l + B_l \Pi_{\rm s} \tag{179}$$
The way of computing A_l and B_l is the following (the same method is applied to A and B, the following equations are written for A): one discretises the following equation:
$$A_\eta - A_{\eta=0} = \int_{\eta'=0}^{\eta'=\eta} \frac{dA}{d\eta'} d\eta' \tag{180}$$
This is a vertical integral and this integral is computed like any vertical integral in the case **LVERTFE**=.T., using a matricial product coded in routine **VERINT**. $\frac{dA}{d\eta}$ at full levels is discretised as follows:
$$\left[\frac{dA}{d\eta}\right]_l = \frac{A_{\bar{l}} - A_{\bar{l}-1}}{\eta_{\bar{l}} - \eta_{\bar{l}-1}} \tag{181}$$
Remarks:
  - Equation (179) matches the definition of the hybrid coordinate.
  - The result slightly depends on the definition of η (variable **LVFE_REGETA**) and gives something close to the option: **LVERTFE**=.F.; **NDLNPR**=0; **LAPRXPK**=.TRUE. .

8.4 Calculation of the horizontal gradient of the hydrostatic pressure at half and full levels.

At half levels, one simply applies the horizontal gradient operator to the half level pressure, using the formula (168).
$$(\nabla\Pi)_{\bar{l}} = B_{\bar{l}} \nabla\Pi_{\rm s} \tag{182}$$
At full levels:

- **VFD**: this gradient can be written with several different ways. The code is provided according the following formula:
$$\left[\frac{\nabla\Pi}{\Pi}\right]_l = \left[\frac{[\Delta B]_l + \delta_l \frac{A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}}{\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}}{\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\right] \nabla\Pi_{\rm s} \tag{183}$$
- **VFE**: expression is simpler and simply writes:
$$(\nabla\Pi)_l = B_l \nabla\Pi_{\rm s} \tag{184}$$
or:
$$\left[\frac{\nabla\Pi}{\Pi}\right]_l = \frac{B_l}{A_l + B_l\Pi_{\rm s}} \nabla\Pi_{\rm s} \tag{185}$$

8.5 The quantities α and δ linked to pressure depth layers.

Meaning:

δ has the following meaning: the pressure depth of a layer, divided by the “average” pressure of this layer. δ_l is close (sometimes equal) to $(\log \Pi)_{\bar{l}} - (\log \Pi)_{\bar{l}-1}$.

α has the following meaning: the pressure depth between a half level and the full level situated immediately above it, divided by the “average” pressure between this half level and this full level. α_l is close (sometimes equal) to $(\log \Pi)_{\bar{l}} - (\log \Pi)_l$.

These operators are used for discretisations of some vertical integrals.

Expressions for α and δ at full levels:

- **VFD0**:
  - For a layer l between 2 and L (and also $l = 1$ if the pressure at the top of the model is not zero), α and δ are discretised as follows at full levels:

$$\alpha_l = 1 - \frac{\Pi_{\bar{l}-1}}{\Delta\Pi_l} \log\left(\frac{\Pi_{\bar{l}}}{\Pi_{\bar{l}-1}}\right) \tag{186}$$

$$\delta_l = \log\left(\frac{\Pi_{\bar{l}}}{\Pi_{\bar{l}-1}}\right) \tag{187}$$

  - For the layer $l = 1$ if the pressure at the top of the model is zero:
    - $\alpha_{l=1} = 1$ at METEO-FRANCE.
    - $\alpha_{l=1} = \log(2)$ at ECMWF.
    - $\delta_{l=1}$ has an infinite value. In practical, in the code, $\delta_{l=1}$ is computed with a top pressure equal to 0.1 Pa in this case in order to avoid the calculation of an infinite value.
- **VFD1**:
  - For a layer l between 2 and L (and also $l = 1$ if the pressure at the top of the model is not zero), α and δ are discretised as follows at full levels:

$$\alpha_l = 1 - \sqrt{\frac{\Pi_{\bar{l}-1}}{\Pi_{\bar{l}}}} = 1 - \frac{\Pi_l}{\Pi_{\bar{l}}} \tag{188}$$

$$\delta_l = \frac{\Delta\Pi_l}{\Pi_l} = \frac{\Delta\Pi_l}{\sqrt{\Pi_{\bar{l}-1}\Pi_{\bar{l}}}} \tag{189}$$

  Equation (189) can be rewritten:

$$\left[\frac{1}{\Pi}\right]_l = \frac{\delta_l}{\Delta\Pi_l} \tag{190}$$

  - For the layer $l = 1$ if the pressure at the top of the model is zero:
    - $\alpha_{l=1} = 1$ and $\alpha_{\bar{l}=0} = 1$.
    - $\delta_{l=1} = 1 + \frac{c_{p_d}}{R_d}$.
    - $\Pi_{l=1} = \frac{\Delta\Pi_{l=1}}{\delta_{l=1}} = \frac{\Delta\Pi_{l=1}}{1+\frac{c_{p_d}}{R_d}}$.
    - $\left[\frac{1}{\Pi}\right]_{l=1} = \frac{1+\frac{c_{p_d}}{R_d}}{\Delta\Pi_{l=1}}$.
  - Remark: equation (190) is also used for case **NDLNPR**=0 to discretise $[1/\Pi]_l$.
- **VFD2**: the only difference with **NDLNPR**=1 is the expression of $\delta_{l=1}$ if the pressure at the top of the model is zero:
  - $\delta_{l=1} = 1 + \delta_{l=2}([\Delta\Pi]_{l=1}/[\Delta\Pi]_{l=2})\sqrt{\Pi_{\bar{l}=2}/\Pi_{\bar{l}=1}}$
  - $\Pi_{l=1} = \frac{\Pi_{\bar{l}=1}}{\delta_{l=1}}$
- **VFE0**: For a layer l between 1 and L, α and δ are discretised as follows at full levels:

$$\alpha_l = \frac{\Pi_{\bar{l}} - \Pi_l}{\Pi_l} \tag{191}$$

$$\delta_l = \frac{\Delta\Pi_l}{\Pi_l} = \frac{\Delta\Pi_l}{A_l + B_l\Pi_s} \tag{192}$$

  Remark: quantity α_l is not used in the adiabatic part of the model (only in the MF-physics).

Expressions for $\nabla\alpha$ and $\nabla\delta$ at full levels: For VFD, details of calculations are provided in appendix 2.

- **VFD0**:
  - For a layer l between 2 and L (and also $l=1$ if the pressure at the top of the model is not zero), $\nabla\alpha$ and $\nabla\delta$ are discretised as follows at full levels:

$$[\nabla\alpha]_l = \frac{\Pi_{\bar{l}}\Pi_{\bar{l}-1}\log\left(\frac{\Pi_{\bar{l}}}{\Pi_{\bar{l}-1}}\right) - \Pi_{\bar{l}-1}(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\left[\frac{B_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{B_{\bar{l}-1}}{\Pi_{\bar{l}-1}}\right]\nabla\Pi_{\mathrm{s}} \tag{193}$$

$$[\nabla\delta]_l = -\frac{A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}}{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\nabla\Pi_{\mathrm{s}} \tag{194}$$

  The gradient of $\alpha + \log\Pi$ has a simple expression in this case:

$$[\nabla(\alpha + \log\Pi)]_l = \frac{B_{\bar{l}}}{\Pi_{\bar{l}}}\nabla\Pi_{\mathrm{s}} \tag{195}$$

  - For the layer $l=1$ if the pressure at the top of the model is zero:
    - $[\nabla\alpha]_{l=1} = 0$.
    - $[\nabla\delta]_{l=1}$: specific calculation to ensure a finite value; $[\nabla\delta]_{l=1} = 0$ if $\Pi_{\mathrm{top}} = 0$ or if top levels are pure pressure levels.

- **VFD1** and **VFD2**:
  - For a layer l between 2 and L (and also $l=1$ if the pressure at the top of the model is not zero), $\nabla\alpha$ and $\nabla\delta$ are discretised as follows at full levels:

$$[\nabla\alpha]_l = -\frac{\alpha_l\left[A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}\right]}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\nabla\Pi_{\mathrm{s}} \tag{196}$$

$$[\nabla\delta]_l = -\frac{\delta_l\left[A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}\right]}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\nabla\Pi_{\mathrm{s}} \tag{197}$$

  Using equation (189), equation (199) can be rewritten:

$$[\nabla\delta]_l = -\frac{\left[A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}\right]}{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\nabla\Pi_{\mathrm{s}} \tag{198}$$

  - For the layer $l=1$ if the pressure at the top of the model is zero:
    - $[\nabla\alpha]_{l=1} = 0$.
    - $[\nabla\delta]_{l=1} = 0$.

- **VFE0**: For a layer l between 1 and L, $\nabla\delta$ is discretised as follows at full levels:

$$[\nabla\delta]_l = \left(\frac{[\Delta B]_l}{[\Delta\Pi]_l} - \frac{B_l}{\Pi_l}\right)\delta_l\nabla\Pi_{\mathrm{s}} = \frac{A_l[\Delta B]_l - [\Delta A]_l B_l}{\Pi_l[\Delta\Pi]_l}\delta_l\nabla\Pi_{\mathrm{s}} \tag{199}$$

  Quantity $[\nabla\alpha]_l$ is useless in this case.

8.6 Computation of a good set of A and B at half levels.

It requires a formula giving a rather regular spacing in pressure, with the following constraints:

- (C1) $\Delta\Pi$ has little variations in the free troposphere.
- (C2) $\Delta\Pi$ is smaller in the PBL than in the free troposphere, and the variations of $\Delta\Pi$ are smooth in the PBL.
- (C3) $\Delta\Pi$ regularly decreases when going up in the stratosphere.
- (C4) Π is a monotonous variable, even over high mountains.
- (C5) A is always positive, B is always between 0 and 1 and is a monotonous function.
- (C6) Near the surface, the vertical coordinate is a pure σ one. That means that $A_{\bar{l}=L}$ and $A_{\bar{l}=L-1}$ are always 0.
- (C7) Near the top, the vertical coordinate is a pure pressure one. That means that $B_{\bar{l}=1}$ and $B_{\bar{l}=0}$ are always 0.
- (C8) The set of A and B provided must ensure that the operator `B` used in the semi-implicit scheme can be always diagonalised with real positive eigenvalues in all the available vertical discretisation schemes (this is a delicate point especially when the vertical finite element scheme is used with **LVFE_REGETA**=.F.).

Until 2006, we have used a formula provided by J.F. Geleyn: A and B are given by third-degree polynomials. The freedom degrees are:

- The pressure of the first full layer.
- A ratio allowing to tune the depth of the layer number L.
- A reference pressure on the average orography of the Earth, which can be taken lower than the surface standard pressure.

It is assumed that there are only one purely σ layer and one purely pressure layer, and that $\Pi_{\text{top}} = 0$. Some tests done during autumn 2004 have shown that the operator `B` cannot always be diagonalised with real positive eigenvalues, especially when the vertical finite element scheme is used with **LVFE_REGETA**=.F. . The other shortcoming of this formula is the too stringent constraint which obliges to have exactly one purely σ layer and one purely pressure layer. We wish to be able to put several purely pressure layers in the stratosphere and to obtain sets of A and B close to the ones used at ECMWF.

At ECMWF, a different formula is used, there is no documentation about it and no available FORTRAN program giving it directly. It allows the possibility of having several purely pressure layers.

To overcome the previously mentioned shortcomings, we have regularly developed new ways of computing the A and B allowing mode freedom degrees.
An algorithm has been developed between 2004 and 2006 and is described in the internal paper (IDAB). The freedom degrees are:

- The pressure of the tropopause (generally taken to 250 hPa).
- The pressure of the top of the PBL (generally taken to 900 hPa).
- The number of layers in the stratosphere + mesosphere.
- The number of layers in the PBL.
- The number of layers in the free troposphere.
- The number of purely pressure layers.
- The number of purely σ layers.
- The pressure depth of the bottom model layer.
- The pressure of the first full layer.
- The speed of going from σ layers to pressure layers (governing spacing about mountains).

To sum-up, atmosphere is shared into three main layers and algorithm uses vertical polynomials depending on pressure.
A more recent algorithm has been developed in 2012 with a possibility to share atmosphere into more than three main layers; vertical dependency of polynomials used in this algorithm is done according to altitude instead of pressure; conversions between altitude and pressure use standard atmosphere; altitude depths are prescribed at some pre-defined altitude levels.

9 Treatment of the advection.

9.1 Treatment of the horizontal advection.

Horizontal advections are computed at full levels. For a variable X other than wind components:

$$[\mathbf{V}\nabla X]_l = \mathbf{V}_l \left[\nabla X_l\right]_l \tag{200}$$

Remarks:

- For $X = U$ or $X = V$ (horizontal wind components) the meridian derivatives of U and V are not easily available, and the horizontal gradient is rewritten in order to use the divergence, vorticity, and some curvature terms. See the part (5.2.1) concerning the momentum equation for more details.
- The value of $\mathbf{V}$ and X used in formula (200) is taken at instant t.

9.2 Treatment of the vertical advection (3D model).

VFD treatment of vertical advection.

Vertical advection is treated explicitly, and uses half level $(\dot{\eta}\frac{\partial \Pi}{\partial \eta})$. For a variable X, its vertical advection $\dot{\eta}\frac{\partial X}{\partial \eta}$ is computed at full levels and is discretised as follows:

$$\left(\dot{\eta}\frac{\partial X}{\partial \eta}\right)_l = 0.5\left(\frac{\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\bar{l}}(X_{l+1} - X_l)}{(\Delta\Pi)_l} + \frac{\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\bar{l}-1}(X_l - X_{l-1})}{(\Delta\Pi)_l}\right) \tag{201}$$

For the layer $l = 1$ $\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\eta=0}$ can be different from 0 if a radiative upper condition is applied; one uses the top value $X_{\eta=0}$ of X:

$$\left(\dot{\eta}\frac{\partial X}{\partial \eta}\right)_1 = 0.5\left(\frac{\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\bar{l}=1}(X_2 - X_1)}{(\Delta\Pi)_1} + \frac{2\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\eta=0}(X_1 - X_{\eta=0})}{(\Delta\Pi)_1}\right) \tag{202}$$

For the layer $l = L$, $\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_L$ can be different from 0 (if taking account of rainfall and evaporation in the continuity equation); one uses the bottom value $X_{\eta=1}$ of X:

$$\left(\dot{\eta}\frac{\partial X}{\partial \eta}\right)_L = 0.5\left(\frac{2\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\eta=1}(X_{\eta=1} - X_L)}{(\Delta\Pi)_L} + \frac{\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_{\bar{l}=L-1}(X_L - X_{L-1})}{(\Delta\Pi)_L}\right) \tag{203}$$

The values of X, $\left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)$ and $(\Delta\Pi)$ present in the RHS of equations (201), (202) and (203) are taken at instant t.

VFE treatment of vertical advection.

$*$ General case:
Vertical advection is treated explicitly, and uses full level $(\dot{\eta}\frac{\partial \Pi}{\partial \eta})$. For a variable X, its vertical advection $\dot{\eta}\frac{\partial X}{\partial \eta}$ is computed at full levels and is discretised as follows:

$$\left(\dot{\eta}\frac{\partial X}{\partial \eta}\right)_l = \left(\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right)_l \frac{(\Delta\eta)_l}{(\Delta\Pi)_l}\left(\frac{\partial X}{\partial \eta}\right)_l \tag{204}$$

where

$$\left(\frac{\partial X}{\partial \eta}\right)_l = [\mathcal{R}_{\mathrm{deri}}]_l\langle X\rangle$$

($\langle X\rangle$ is the vector containing all layer values of X).

$*$ Particuliar case of the moisture convergence:
The code present in routine **CPPHINP** currently uses a different vertical discretisation not described in detail.

10 Physics and physics-dynamics interface (3D model).

10.1 Time of evaluation.

In an Eulerian scheme physics can be evaluated at $t - \Delta t$ (not lagged physics) or $t + \Delta t$ (lagged physics). The lagged physics use as input the provisional $t + \Delta t$ variables available at the end of the grid-point calculations (before the inversion of the semi-implicit system and the horizontal diffusion).

10.2 Physical tendencies, flux divergences, and flux.

∗ **Physical tendencies and flux divergences:** Physical tendencies write $F = \frac{1}{\rho}\frac{\partial \mathcal{F}}{\partial z}$ in z-vertical coordinate and $F = -g\frac{\partial \mathcal{F}}{\partial \Pi}$ in pressure Π-vertical coordinate. $\mathcal{F}$ is the flux.

∗ **Relationship between fluxes and flux divergences:** Between two hydrostatic pressure layers $\Pi(1)$ and $\Pi(2)$, the fluxes difference writes:

$$\mathcal{F}(\Pi(2)) - \mathcal{F}(\Pi(1)) = \int_{\Pi=\Pi(1)}^{\Pi=\Pi(2)} \frac{\partial \mathcal{F}}{\partial \Pi} d\Pi \tag{205}$$

The discretised form of this equation writes:

$$\mathcal{F}_{\bar{l}} - \mathcal{F}_{\bar{l}-1} = \left[\frac{\partial \mathcal{F}}{\partial \Pi}\right]_l [\Delta\Pi]_l \tag{206}$$

10.3 Remark for VFE vertical discretisation.

Vertical integrals in the physics and transformations between fluxes and flux divergences keep the VFD discretisation in this case. Operator $\mathcal{R}_{\mathrm{inte}}$ is not used. Fluxes remain computed at half levels. When a flux is needed at full levels knowing its value at half levels, the following average is used:

$$\mathcal{F}_l = \mathcal{F}_{\bar{l}-1} + \frac{\alpha_l}{\delta_l}\left(\mathcal{F}_{\bar{l}} - \mathcal{F}_{\bar{l}-1}\right)$$

10.4 Levels of physics-dynamics interfaces.

To summarize, we can say that calling the different physical parameterisations is done according to the following scheme:

```
dynamics ->
 physics-dynamics interface routines ->
   physics caller routine ->
   the different physical parameterisations.
```

For non-AROME MF physics, this kind of scheme can be retrieved in the following sequence of calls:

```
CPG -> MF_PHYS -> APLPAR -> AC.. and RAD.. routines.
```

Actions done in the physics-dynamics interface routines are:

- Calculation of input quantities which are used only in the physics (for example moisture convergence, solar angle).
- Call the physics (via a physics caller routine).
- Conversion of the physics outputs (fluxes or tendencies of physics prognostic variables) into tendencies of dynamics prognostic variables.
- Store tendencies in / add tendencies to appropriate buffers.
- Perform additional diagnostics.
- Temporal advance of surface and soil variables.

Some of them may have to call **GP..** or **GNH..** routines (do intermediate dynamical calculations), for example **GPHPRE**.

In physics caller routines, we just find call to different physical parameterisation routines. These routines should not do dynamics (call to **GP..** routines like **GPHPRE** is forbidden) and should not know the kind of vertical coordinate which is used in the adiabatic part of the model.

See documentation (IDPHYE) for more details about physics, and physics-dynamics interface.

10.5 The "delta m = 1" barycentric formulation.

The "$\delta m = 0$" assumption says that, when it rains, a particle of water is not replaced by a particle of dry air, and when there is some evaporation (especially near the ground), there is no removal of dry air. This assumption can lead to bad conservation of mass, especially for high resolution simulations (below 10 km). This assumption remains acceptable for the ranges of horizontal resolutions where the hydrostatic hypothesis works well.
But a "$\delta m = 1$" formulation has been coded since several years, which takes account of the rainfall and the evaporation in the dry air budget, for example by replacing a particle of water by a particle of dry air when it rains. This "$\delta m = 1$" has evolved during the last years, going from a "non barycentric" formulation to a "barycentric" formulation (the last one treats at the same level rainfall and evaporation). Since CY29T3/AL29T3, the "barycentric" formulation of "$\delta m = 1$" formulation has been implemented in the code and we will give a short description of it.

For "$\delta m = 0$":

- when it rains, a particle of water is not replaced by a particle of dry air: the consequence is that the diabatic term linked to rainfall is replaced by zero in the continuity equation.
- when there is evaporation, a created particle of water does not pull out a particle of dry air: the consequence is that the diabatic term linked to evaporation is replaced by zero in the continuity equation.
- F_{m} becomes equal to zero.
- there is no modification of the vertical velocity, and $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ at the surface is equal to zero.

For "$\delta m = 1$":

- The precipitation flux and the evaporation flux must be taken into account, and $F_{\mathrm{m}} = F_{\mathrm{E}} + F_{\mathrm{P}}$ is non-zero. For the time being the following assumptions are done:
  - F_{E} is non-zero only at the surface, zero elsewhere.
  - F_{P} is zero at the top of the atmosphere.
- $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ is modified, excepted at the top of the atmosphere (where it remains to zero, unless **LRUBC**=.T.).

The "barycentric" formulation of "$\delta m = 1$" leads to the following modifications:

- Term ω: the RHS of equation (93) is the same for "$\delta m = 0$" and "$\delta m = 1$". The consequence is that:

$$\left[\frac{\omega}{\Pi}\right]_{\eta_l,\delta m=1} = \left[\frac{\omega}{\Pi}\right]_{\eta_l,\delta m=0}$$

- Term $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ at the surface:

$$\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\eta=1,\delta m=1} = \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\eta=1,\delta m=0} + g\left[F_{\mathrm{E}} + F_{\mathrm{P}}\right]_{\eta=1}$$

- Upper air term $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$:

$$\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\eta_l,\delta m=1} = \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\eta_l,\delta m=0} + B_{\eta_l} g\left[F_{\mathrm{E}} + F_{\mathrm{P}}\right]_{\eta=1}$$

- Diabatic part of $\frac{\partial \log(\Pi_{\mathrm{s}})}{\partial t}$:

$$\left[\frac{\partial \log(\Pi_{\mathrm{s}})}{\partial t}\right]_{\delta m=1} - \left[\frac{\partial \log(\Pi_{\mathrm{s}})}{\partial t}\right]_{\delta m=0} = -\frac{1}{\Pi_{\mathrm{s}}} g\left[F_{\mathrm{E}} + F_{\mathrm{P}}\right]_{\eta=1}$$

11 The Eulerian discretisation of the 2D shallow-water system of equations (spherical geometry).

11.1 Momentum equation.

∗ Definition of X, $\mathcal{A}$ and $\mathcal{L}$.

$$X = \mathbf{V} + \delta_{\mathbf{V}}(2\mathbf{\Omega} \wedge \mathbf{r}) \tag{207}$$

$$\mathcal{A} = -2(\mathbf{\Omega} \wedge \mathbf{V}) - \nabla\Phi \tag{208}$$

$$\mathcal{L} = -\nabla\Phi + \beta_{Co}[-2\mathbf{\Omega} \wedge \mathbf{V}] \tag{209}$$

∗ Remarks.

- If β=1, the part of the non linear term which does not contain the Coriolis term is zero.
- It is desirable to discretise the momentum equation form given by equation (63). Vector $\mathbf{D_V}$ can be computed knowing the following quantities easily available in the grid-point part of the model: U, V, D, ζ, $\frac{1}{a\cos\theta_{\rm bne}}\frac{\partial U}{\partial\lambda_{\rm bne}}$, $\frac{1}{a\cos\theta_{\rm bne}}\frac{\partial V}{\partial\lambda_{\rm bne}}$, $\frac{2}{a}\tan\theta_{\rm bne}$.

11.2 Continuity equation.

∗ Definition of X, $\mathcal{A}$ and $\mathcal{L}$.

$$X = \Phi \tag{210}$$

$$\mathcal{A} = -(\Phi - \Phi_{\rm s})D \tag{211}$$

$$\mathcal{L} = -\Phi^*(\overline{M}^2 D^{'}) \tag{212}$$

11.3 Application to the 2D "vorticity" system of equations.

- For the 2D "vorticity" system of equations, start from the shallow-water model, and replace Φ and $\Phi_{\rm s}$ by zero in it.

12 The Eulerian discretisation of the 3D primitive equation model.

12.1 Formulation of the momentum equation.

$*$ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = \mathbf{V} \tag{213}$$

$$\mathcal{A} = -2\mathbf{\Omega} \wedge \mathbf{V} - \nabla\Phi - RT\nabla(\log\Pi) \tag{214}$$

$$\mathcal{L} = -\nabla\left[\gamma T + R_{\mathrm{d}}T^{*}\log(\Pi_{\mathrm{s}})\right] + \beta_{Co}[-2(\mathbf{\Omega}\wedge\mathbf{V})] \tag{215}$$

$$F = \mathbf{F_V} \tag{216}$$

Top and bottom values are defined as follows:

$$\mathbf{V}_{\eta=0} = \mathbf{V}_{l=1} \tag{217}$$

If $\delta m = 0$:

$$\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L} \tag{218}$$

If $\delta m = 1$:

$$\mathbf{V}_{\eta=1} = 0 \tag{219}$$

$*$ Remarks.

- Coriolis term can also be put in the semi-implicit scheme by tuning β_{Co}. Values $\beta_{Co} = 0$ (**LIMPF**=.F.) and $\beta_{Co} = 1$ (**LIMPF**=.T.) are available. Caution: do not use **LIMPF**=.T. in variable resolution (formulation of spectral computations is not correct in this case for the semi-implicit scheme).
- It is desirable to discretise the momentum equation form given by equation (70). Vector $[\mathbf{D_V}]_l$ on layer l can be computed knowing the following quantities easily available in the grid-point part of the model: U_l, V_l, $M^2D_l^{'}$, $M^2\zeta_l^{'}$, $[\frac{1}{a\cos\theta_{\mathrm{bne}}}\frac{\partial U}{\partial\lambda_{\mathrm{bne}}}]_l$, $[\frac{1}{a\cos\theta_{\mathrm{bne}}}\frac{\partial V}{\partial\lambda_{\mathrm{bne}}}]_l$, $[\frac{2}{a}\tan\theta_{\mathrm{bne}}]_l$.

$*$ Discretisation of $-2\mathbf{\Omega}\wedge\mathbf{V}$ at full levels:

$$[(-2\mathbf{\Omega}\wedge\mathbf{V}).\mathbf{i}]_l = 2\Omega\sin\theta V_l \tag{220}$$

$$[(-2\mathbf{\Omega}\wedge\mathbf{V}).\mathbf{j}]_l = -2\Omega\sin\theta U_l \tag{221}$$

$*$ Discretisation of the pressure gradient term at full levels: The pressure gradient term writes:

$$-\left(\nabla\Phi + RT\nabla(\log\Pi)\right)$$

It contains the geopotential $\Phi = gz$. Expression of gz is provided by equation (85). So the pressure gradient term can be rewritten as follows:

$$-\left(\nabla\Phi_{\mathrm{s}} + \nabla\left(\int_{\Pi_{\mathrm{s}}}^{\Pi} -\frac{RT}{\Pi}d\Pi\right) + RT\nabla(\log\Pi)\right)$$

Term:

$$\nabla\Phi_{\mathrm{s}} + \nabla\left(\int_{\Pi_{\mathrm{s}}}^{\Pi} -\frac{RT}{\Pi}d\Pi\right) + RT\nabla(\log\Pi)$$

is discretised as follows if **LVERTFE**=.F.:

$$\begin{aligned}\left[\nabla\Phi_{\mathrm{s}} + \nabla\left(\int_{\Pi_{\mathrm{s}}}^{\Pi} -\tfrac{RT}{\Pi}d\Pi\right) + RT\nabla(\log\Pi)\right]_l = \\ \nabla\Phi_{\mathrm{s}} + \textstyle\sum_{k=L}^{l+1}\left[\nabla(RT)\right]_k\delta_k + \sum_{k=L}^{l+1}(RT)_k\left[\nabla\delta\right]_k \\ + \left[\nabla(RT)\right]_l\alpha_l + (RT)_l\left[\nabla(\alpha+\log\Pi)\right]_l\end{aligned} \tag{222}$$

and as follows if **LVERTFE**=.T.:

$$\begin{aligned}\left[\nabla\Phi_{\mathrm{s}} + \nabla\left(\int_{\Pi_{\mathrm{s}}}^{\Pi} -\tfrac{RT}{\Pi}d\Pi\right) + RT\nabla(\log\Pi)\right]_l = \\ \nabla\Phi_{\mathrm{s}} + [\mathcal{R}_{\mathrm{inte}}]_{(surf,l)}\left\langle -\tfrac{\nabla(RT)\delta + RT\nabla\delta}{\Delta\eta}\right\rangle + (RT)_l\left[\nabla(\log\Pi)\right]_l\end{aligned} \tag{223}$$

(where $\langle X\rangle$ denotes the vertical vector of coordinates $(X_{k=1},...,X_{k=l},...,X_{k=L})$).

∗ **Discretisation of the grid-point Rayleigh friction:** This term writes $K_{\mathrm{fric}}U\mathbf{i}$, its discretisation on layer l is $K_{\mathrm{fric}}U_l\mathbf{i}$. An optional contribution can be added to V-component: this term writes $K_{\mathrm{fric}}V\mathbf{j}$, its discretisation on layer l is $K_{\mathrm{fric}}V_l\mathbf{j}$.

12.2 Thermodynamic equation.

∗ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = T \tag{224}$$

$$\mathcal{A} = \frac{RT}{c_{\mathrm{p}}}\frac{\omega}{\Pi} \tag{225}$$

$$\mathcal{L} = -\tau(\overline{M}^2 D^{'}) \tag{226}$$

$$F = F_{\mathrm{T}} \tag{227}$$

Top:

$$T_{\eta=0} = T_{l=1} \tag{228}$$

Bottom if $\delta m = 0$:

$$T_{\eta=1} = T_{l=L} \tag{229}$$

Bottom if $\delta m = 1$ (output of physics):

$$T_{\eta=1} = T_{\mathrm{s}} \tag{230}$$

∗ Discretisation of the conversion term.

$$\left[\frac{RT}{c_{\mathrm{p}}}\frac{\omega}{\Pi}\right]_l = \frac{R_l T_l}{[c_{\mathrm{p}}]_l}\left[\frac{\omega}{\Pi}\right]_l \tag{231}$$

See part (12.7) for discretisation of $\frac{\omega}{\Pi}$ at full levels.

12.3 Formulation of the continuity equation.

∗ Definition of X, $\mathcal{A}$, $\mathcal{L}$, and F.

$$X = \log \Pi_{\mathrm{s}} \tag{232}$$

$$\mathcal{A} = -\frac{1}{\Pi_{\mathrm{s}}}\int_{\eta=0}^{\eta=1} \nabla\left(\mathbf{V}\frac{\partial \Pi}{\partial \eta}\right) d\eta + \mathbf{V}\nabla\left[\log \Pi_{\mathrm{s}}\right] - \frac{1}{\Pi_{\mathrm{s}}}\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\eta=1} + \frac{1}{\Pi_{\mathrm{s}}}\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\eta=0} \tag{233}$$

$$\mathcal{L} = -\frac{\overline{M}^2}{M^2}\nu D \tag{234}$$

$$F = \left(\frac{F_{\mathrm{m}}}{\Pi_{\mathrm{s}}}\right) \tag{235}$$

The actual form of continuity equation which is discretised replaces the sum of $\mathcal{A}$ and of the horizontal advection term by a term which is a vertical integral of a divergence term.

∗ **Discretisation of the vertical integral of the divergence term:** The term to discretise is:

$$\frac{1}{\Pi_{\mathrm{s}}}\int_{\eta=0}^{\eta=1} \nabla\left(\mathbf{V}\frac{\partial \Pi}{\partial \eta}\right) d\eta$$

In the RHS of continuity equation, term

$$\frac{1}{\Pi_{\mathrm{s}}}\nabla\left(\mathbf{V}\frac{\partial \Pi}{\partial \eta}\right)$$

is rewritten under the sum of several terms:

$$\frac{1}{\Pi_{\mathrm{s}}}\mathbf{V}\nabla\left(\frac{\partial \Pi}{\partial \eta}\right) + \frac{1}{\Pi_{\mathrm{s}}}\left(\frac{\partial \Pi}{\partial \eta}\right)\nabla\mathbf{V}$$

- Term $\frac{1}{\Pi_{\mathrm{s}}}\mathbf{V}\nabla\left(\frac{\partial \Pi}{\partial \eta}\right)$: using equation (167), the vertical integral of this term can be rewritten as:

  $$\int_{\eta=0}^{\eta=1} \mathbf{V}\frac{\partial B}{\partial \eta}(\eta)\nabla \log \Pi_{\mathrm{s}} d\eta$$

  Discretisation of this term writes:

  - VFD: $\sum_{l=1}^{L} [\Delta B]_l \mathbf{V}_l \nabla \log \Pi_\mathrm{s}$
  - VFE: $[\mathcal{R}_\mathrm{inte}]_{(top,surf)} \left\langle \frac{\Delta B}{\Delta \eta} \mathbf{V} \nabla \log \Pi_\mathrm{s} \right\rangle$

- Term $\frac{1}{\Pi_\mathrm{s}} \left(\frac{\partial \Pi}{\partial \eta} \right) \nabla \mathbf{V}$: discretisation of this term writes:
  - VFD: $\frac{1}{\Pi_\mathrm{s}} \sum_{l=1}^{L} [\Delta \Pi]_l \nabla \mathbf{V}_l$
  - VFE: $\frac{1}{\Pi_\mathrm{s}} [\mathcal{R}_\mathrm{inte}]_{(top,surf)} \left\langle \frac{\Delta \Pi}{\Delta \eta} \nabla \mathbf{V} \right\rangle$

12.4 Moisture equation.

∗ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = q \tag{236}$$

$$\mathcal{A} = 0 \tag{237}$$

$$\mathcal{L} = 0 \tag{238}$$

$$F = F_\mathrm{q} \tag{239}$$

Top:

$$q_{\eta=0} = q_{l=1} \tag{240}$$

Bottom if $\delta m = 0$:

$$q_{\eta=1} = q_{l=L} \tag{241}$$

Bottom if $\delta m = 1$ (see **CPQSOL**, relative humidity is the same for $\eta = \eta_L$ and $\eta = 1$):

$$q_{\eta=1} = q_\mathrm{s} \tag{242}$$

12.5 Other GFL variables.

∗ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values, for advectable variables. Equations are discretised as for humidity equation, the only difference is sometimes for the choice of vertical boundary conditions at the top and bottom. For example, for liquid water q_l:

$$X = q_\mathrm{l} \tag{243}$$

$$\mathcal{A} = 0 \tag{244}$$

$$\mathcal{L} = 0 \tag{245}$$

Top:

$$[q_\mathrm{l}]_{\eta=0} = [q_\mathrm{l}]_{l=1} \tag{246}$$

Bottom if $\delta m = 0$:

$$[q_\mathrm{l}]_{\eta=1} = [q_\mathrm{l}]_{l=L} \tag{247}$$

Bottom if $\delta m = 1$:

$$[q_\mathrm{l}]_{\eta=1} = 0 \tag{248}$$

Vertical boundary conditions:

- quantities are assumed constant above the middle of the upper layer and below the middle of the lower layer in case $\delta m = 0$.
- quantities are assumed constant above the middle of the upper layer in case $\delta m = 1$.
- advectable GFL variables other than humidity are assumed to be zero at the surface in case $\delta m = 1$.

∗ Particular case of non advectable pseudo-historic GFL variables: Equations given for liquid water are still valid; advections have to be replaced by zero; definition of top and bottom values is useless.

12.6 Relationship between geopotential height gz and pressure depth.

∗ Geopotential height at half levels (case LVERTFE=.F. only): Discretisation of equation (85) at half levels yields:

$$gz_{\bar{l}} = gz_\mathrm{s} + \sum_{k=L}^{k=l+1} [R_k T_k]\, \delta_k \tag{249}$$

See section (8.5) for discretisation of δ at full levels.

∗ **Geopotential height at full levels:**

- Case **LVERTFE**=.F.: It is computed from the geopotential height at half levels by the following relationship:
$$gz_l = gz_{\bar{l}} + [R_l T_l]\,\alpha_l \tag{250}$$
See section (8.5) for discretisation of α at full levels.
- Case **LVERTFE**=.T.:
$$gz_l = gz_{\rm s} + [\mathcal{R}_{\rm inte}]_{(surf,l)} \left\langle -\frac{RT\delta}{\Delta\eta} \right\rangle \tag{251}$$
See section (8.5) for discretisation of δ at full levels.

∗ **Horizontal gradient of the geopotential height at half levels (case LVERTFE=.F. only):** One applies the operator ∇ to equation (249). That yields:

$$g\,[\nabla z]_{\bar{l}} = g\nabla z_{\rm s} + \sum_{k=L}^{k=l+1} [\nabla(RT)]_k\,\delta_k + \sum_{k=L}^{k=l+1} [R_k T_k]\,[\nabla\delta]_k \tag{252}$$

See section (8.5) for discretisation of δ and $\nabla\delta$ at full levels.

∗ **Horizontal gradient of the geopotential height at full levels:** One applies the operator ∇ to equation (250). That yields:

- Case **LVERTFE**=.F.:
$$g\,[\nabla z]_l = g\,[\nabla z]_{\bar{l}} + [\nabla(RT)]_l\,\alpha_l + [R_l T_l]\,[\nabla\alpha]_l \tag{253}$$
See section (8.5) for discretisation of α and $\nabla\alpha$ at full levels.
- Case **LVERTFE**=.T.:
$$g\,[\nabla z]_l = g\nabla z_{\rm s} + [\mathcal{R}_{\rm inte}]_{(surf,l)} \left\langle -\frac{\nabla(RT)\delta + RT\nabla\delta}{\Delta\eta} \right\rangle \tag{254}$$
See section (8.5) for discretisation of δ and $\nabla\delta$ at full levels.

12.7 Diagnostic expression of some vertical velocities:

Discretisation is consistent with the one of operators **S** and **N** described in appendix 1.

∗ **Term $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ at half levels if LVERTFE=.F.:** for the half level $\bar{l}$, equation (89) is discretised as follows:

- Term:
$$B_{\eta_l} \int_{\eta=0}^{\eta=1} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta$$
is discretised as follows:
$$B_{\bar{l}} \left[\sum_{k=1}^{L} [\Delta B]_k \mathbf{V}_k \nabla\Pi_{\rm s} + \sum_{k=1}^{L} [\Delta\Pi]_k \nabla\mathbf{V}_k \right]$$
- Term:
$$-\int_{\eta=0}^{\eta=\eta_l} \nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right) d\eta$$
is discretised as follows:
$$-\left[\sum_{k=1}^{l} [\Delta B]_k \mathbf{V}_k \nabla\Pi_{\rm s} + \sum_{k=1}^{l} [\Delta\Pi]_k \nabla\mathbf{V}_k \right]$$
- Term:
$$B_{\eta_l} \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=1} + [1 - B_{\eta_l}] \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0}$$
is discretised as follows:
$$B_{\bar{l}} \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\rm surf} + \left[1 - B_{\bar{l}}\right] \left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\rm top}$$

∗ **Term $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ at full levels if LVERTFE=.T.:** for the layer l, equation (89) is discretised as follows:

- Term:
$$B_{\eta_l}\int_{\eta=0}^{\eta=1}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta$$
is discretised as follows:
$$B_l[\mathcal{R}_{\mathrm{inte}}]_{(top,surf)}\left\langle\frac{\Delta B}{\Delta\eta}\mathbf{V}\nabla\Pi_{\mathrm{s}}+\frac{\Delta\Pi}{\Delta\eta}\nabla\mathbf{V}\right\rangle$$
- Term:
$$-\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta$$
is discretised as follows:
$$-[\mathcal{R}_{\mathrm{inte}}]_{(top,l)}\left\langle\frac{\Delta B}{\Delta\eta}\mathbf{V}\nabla\Pi_{\mathrm{s}}+\frac{\Delta\Pi}{\Delta\eta}\nabla\mathbf{V}\right\rangle$$
- Term:
$$B_{\eta_l}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=1}+[1-B_{\eta_l}]\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0}$$
is discretised as follows:
$$B_l\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\mathrm{surf}}+[1-B_l]\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\mathrm{top}}$$

∗ **Pressure coordinate vertical velocity ω:** for the layer l, equation (93) is discretised as follows:

- Term $\frac{1}{\Pi_{\eta_l}}$: its discretisation at full levels is:
  – Case **LVERTFE**=.F.:
$$\left[\frac{1}{\Pi}\right]_l=\frac{\delta_l}{[\Delta\Pi]_l}\tag{255}$$
  – Case **LVERTFE**=.T.: Equation (255) is still valid but in this case it can be rewritten in a simpler way as follows:
$$\left[\frac{1}{\Pi}\right]_l=\frac{1}{[\Pi]_l}=\frac{1}{A_l+B_l\Pi_{\mathrm{s}}}\tag{256}$$
- "Vertical integral of divergence" term
$$\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta$$
  – Case **LVERTFE**=.F.: This term can be easily computed at half levels and its discretisation on the half level number $\bar{l}$ is:
$$\left[\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta\right]_{\bar{l}}=\sum_{k=1}^{k=l}[\Delta B]_k\mathbf{V}_k\nabla\Pi_{\mathrm{s}}+\sum_{k=1}^{k=l}[\Delta\Pi]_k\nabla\mathbf{V}_k\tag{257}$$
  Computation of ω/Π needs this discretisation at full levels, that needs a weighting of the following type:
$$[.]_l=[.]_{\bar{l}-1}+\frac{\alpha_l}{\delta_l}\left([.]_{\bar{l}}-[.]_{\bar{l}-1}\right)\tag{258}$$
  That yields the following discretisation at full levels:
$$\left[\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta\right]_l$$
$$=\sum_{k=1}^{k=l-1}[\Delta B]_k\mathbf{V}_k\nabla\Pi_{\mathrm{s}}+\sum_{k=1}^{k=l-1}[\Delta\Pi]_k\nabla\mathbf{V}_k+\frac{\alpha_l}{\delta_l}\left([\Delta B]_l\mathbf{V}_l\nabla\Pi_{\mathrm{s}}+[\Delta\Pi]_l\nabla\mathbf{V}_l\right)\tag{259}$$
  – Case **LVERTFE**=.T.: This term is directly computed at full levels and its discretisation on the layer number l is:
$$\left[\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta\right]_l=[\mathcal{R}_{\mathrm{inte}}]_{(top,l)}\left\langle\frac{\Delta B}{\Delta\eta}\mathbf{V}\nabla\Pi_{\mathrm{s}}+\frac{\Delta\Pi}{\Delta\eta}\nabla\mathbf{V}\right\rangle\tag{260}$$
- Product "Vertical integral of divergence" times $-\frac{1}{\Pi_{\eta_l}}$ at full levels: discretised as:

– Case **LVERTFE**=.F.:

$$-\left[\frac{1}{\Pi}\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta\right]_l$$
$$= -\frac{\delta_l}{[\Delta\Pi]_l}\left(\sum_{k=1}^{k=l-1}[\Delta B]_k\mathbf{V}_k\nabla\Pi_\mathrm{s}+\sum_{k=1}^{k=l-1}[\Delta\Pi]_k\nabla\mathbf{V}_k\right)-\frac{\alpha_l}{[\Delta\Pi]_l}[\Delta B]_l\mathbf{V}_l\nabla\Pi_\mathrm{s}-\alpha_l\nabla\mathbf{V}_l \quad (261)$$

– Case **LVERTFE**=.T.:

$$-\left[\frac{1}{\Pi}\int_{\eta=0}^{\eta=\eta_l}\nabla\left(\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right)d\eta\right]_l$$
$$= -\frac{1}{A_l+B_l\Pi_\mathrm{s}}[\mathcal{R}_\mathrm{inte}]_{(top,l)}\left\langle\frac{\Delta B}{\Delta\eta}\mathbf{V}\nabla\Pi_\mathrm{s}+\frac{\Delta\Pi}{\Delta\eta}\nabla\mathbf{V}\right\rangle \quad (262)$$

- Term $\left[\mathbf{V}\frac{\nabla\Pi}{\Pi}\right]$ on layer l is discretised as follows:

$$\left[\mathbf{V}\frac{\nabla\Pi}{\Pi}\right]_l=\mathbf{V}_l\left[\frac{\nabla\Pi}{\Pi}\right]_l$$

  Discretisation of $\frac{\nabla\Pi}{\Pi}$ is given by equation (183) or equation (185) according to the value of **LVERTFE**.

- Term $\frac{1}{\Pi_{\eta_l}}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\eta=0}$: its discretisation on the layer l is:

$$\frac{\delta_l}{[\Delta\Pi]_l}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_\mathrm{top}$$

  This term is non-zero only if a radiative upper boundary condition is applied (case **LRUBC**=.T.).

13 The Eulerian discretisation of the 3D fully compressible non-hydrostatic model (NHEE).

Some of the discretisations are well described in (IDNHPB) and we will recall here some basics. See (IDNHPB) for more details, especially for the intermediate calculations and the assumptions leading to the following discretisations.

13.1 Discretisation of some intermediate quantities.

We have to discretise at least the following quantities:

- Half level total pressure.
- Total pressure depths at full levels.
- Surface wind $\mathbf{V}_{\text{surf}}$.
- Half levels wind when required.
- $\mathtt{X} = \frac{p}{\frac{\partial \Pi}{\partial \eta} RT} \nabla \Phi \left(\frac{\partial \mathbf{V}}{\partial \eta}\right)$.
- gw and its horizontal gradient.
- d (knowing gw).
- D_3.
- Laplacian term: $\frac{\partial\left(\frac{\partial(p-\Pi)}{\partial \Pi}\right)}{\partial \eta}$.

∗ Half level total pressure:
It is required for finite difference treatment of some terms. The following assumptions are done:

- The top value of $p - \Pi$ is assumed to be zero. If the top value of Π is zero, the top value of p is also zero. The consequence is that: $\hat{Q}_{\bar{l}=0} = 0$
- $\hat{Q}$ is assumed to be constant between $l = L$ and the surface: $\hat{Q}_{\text{surf}} = \hat{Q}_{l=L}$
- For the other half levels: $\hat{Q}_{\bar{l}} = 0.5(\hat{Q}_l + \hat{Q}_{l+1})$

From $\hat{Q}_{\bar{l}}$ and $\Pi_{\bar{l}}$ we can compute $p_{\bar{l}}$.

∗ Total pressure depths at full levels:

- Finite difference discretisation (**LVFE_DELNHPRE**=F): $[\Delta p]_l = p_{\bar{l}} - p_{\overline{l-1}}$.
- Finite element discretisation if **LVFE_ECMWF**=F and **LVFE_DELNHPRE**=T: the following formula is used:
$$[\Delta p]_l = [\Delta \Pi]_l \left[\frac{\partial p}{\partial \Pi}\right]_l$$
where:
$$\left[\frac{\partial p}{\partial \Pi}\right]_l = \frac{p_l}{\Pi_l} + \left[\frac{\partial\left(\frac{p-\Pi}{\Pi}\right)}{\partial \log \Pi}\right]_l$$
Discretisation of $\frac{\partial\left(\frac{p-\Pi}{\Pi}\right)}{\partial \log \Pi}$ at full levels is:
$$\left[\frac{\partial\left(\frac{p-\Pi}{\Pi}\right)}{\partial \log \Pi}\right]_l = \left[\mathcal{R}_{\text{deri}}\left(\frac{p-\Pi}{\Pi}\right)\right]_l \frac{[\Delta \eta]_l}{\delta_l}$$
- Finite element discretisation if **LVFE_ECMWF**=T and **LVFE_DELNHPRE**=T: the following formula is used:
$$[\Delta p]_l = [\Delta \Pi]_l \frac{p_l}{\Pi_l} + p_l [\Delta \log (p/\Pi)]_l$$
where:
$$[\Delta \log (p/\Pi)]_l = [\mathcal{R}_{\text{deri}} (\log (p/\Pi))]_l [\Delta \eta]_l$$

∗ Surface wind:
$$\mathbf{V}_{\text{surf}} = \mathbf{V}_{l=L} \tag{263}$$

$*$ Half level wind:

$$\mathbf{V}_{\bar{l}} = \mathcal{W}_{\bar{l}}\mathbf{V}_l + (1 - \mathcal{W}_{\bar{l}})\mathbf{V}_{l+1} \tag{264}$$

where:

$$\mathcal{W}_{\bar{l}} = \frac{\delta_{l+1} - \alpha_{l+1}}{\delta_{l+1} - \alpha_{l+1} + \alpha_l} \tag{265}$$

Remarks:

- $\mathcal{W}_{\bar{l}=L}$ is set to zero.
- Top boundary condition: $\mathbf{V}_{\text{top}} = \mathbf{V}_{l=1}$.
- Bottom boundary condition: $\mathbf{V}_{\text{surf}} = \mathbf{V}_{l=L}$.

$*$ Quantity $\mathtt{X}$:

- Finite difference discretisation (**LVFE_X_TERM**=F): It is better to rewrite $\mathtt{X}$ as follows for the discretisation

  $$\mathtt{X} = \frac{p}{\Pi}\frac{1}{RT}\nabla[gz]\frac{\partial \mathbf{V}}{\partial \log \Pi}$$

  $\mathtt{X}$ is discretised at full levels as follows:

  $$\mathtt{X}_l = \left[\frac{p}{\Pi}\right]_l \frac{1}{[RT]_l\delta_l}\left[\nabla[gz]_{\bar{l}-1}(\mathbf{V}_l - \mathbf{V}_{\bar{l}-1}) + \nabla[gz]_{\bar{l}}(\mathbf{V}_{\bar{l}} - \mathbf{V}_l)\right] \tag{266}$$

  This discretisation uses $\nabla[gz]$ at half levels and $\mathbf{V}$ at both full and half levels.
  The discretisation of $\nabla[gz]$ at half levels is given by equation (305).
  For **NPDVAR**=2, $[p/\Pi]_l = [\exp(\hat{Q})]_l$.
- Finite element discretisation (**LVFE_X_TERM**=T): $\mathtt{X}$ is discretised at full levels as follows:

  $$\mathtt{X}_l = \left[\frac{p}{\Pi}\right]_l \frac{[\Delta\eta]_l}{[RT]_l\delta_l}\nabla[gz]_l[\mathcal{R}_{\text{deri}}\mathbf{V}]_l \tag{267}$$

$*$ Quantity D_3: D_3 is computed at full levels and requires the preliminary computation of $\mathtt{X}_l$.

$$[D_3]_l = D_l + \mathtt{X}_l + d_l\frac{R_{\text{a}l}}{R_l} \tag{268}$$

For **NVDVAR**=4 it is desirable to rewrite this equation by replacing d_l by $d_{4l} - \mathtt{X}_l$.

$*$ Retrieving full levels d from full levels $\Delta[gw]$: Such a calculation is required for example when reading initial files (where this is $\Delta[gw]$ at full levels which is stored).

It is desirable to rewrite the definition of d as follows:

$$d = -\frac{1}{R_{\text{a}}T}\frac{p}{\Pi}\frac{\partial[gw]}{\partial \log \Pi}$$

The discretisation writes:

$$d_l = -\frac{1}{R_{\text{a}l}T_l\delta_l}\left[\frac{p}{\Pi}\right]_l[\Delta[gw]]_l \tag{269}$$

$*$ Retrieving $[gw]$ from full levels d: This discretisation looks like the integration of the geopotential equation. Note that w can be required also in the hydrostatic model.

- Finite difference discretisation (**LVFE_GW**=F):
  First the surface vertical velocity must be computed, using the surface boundary condition:

  $$gw_{\text{surf}} = \mathbf{V}_{\text{surf}}\nabla[\Phi_{\text{s}}] \tag{270}$$

  Integrating equation (269) from the surface, yields:

  $$[\Delta[gw]]_l = d_lR_{\text{a}l}T_l\delta_l\left[\frac{\Pi}{p}\right]_l \tag{271}$$

  $$gw_{\bar{l}} = gw_{\text{surf}} + \sum_{k=L}^{k=l+1}\left(d_kR_{\text{a}k}T_k\delta_k\left[\frac{\Pi}{p}\right]_k\right) \tag{272}$$

  and:

  $$gw_l = gw_{\bar{l}} + d_lR_{\text{a}l}T_l\alpha_l\left[\frac{\Pi}{p}\right]_l \tag{273}$$

- Finite element discretisation (**LVFE_GW**=T):
  Use equation (270) at the surface; for upper-air full level values of w discretisation is:

$$gw_l = gw_{\mathrm{surf}} + [\mathcal{R}_{\mathrm{inte}}]_{(surf,l)} \left\langle -\frac{\Pi}{p}\frac{dR_{\mathrm{a}}T\delta}{\Delta\eta} \right\rangle \tag{274}$$

$*$ Retrieving $\nabla[gw]$ from full levels d and ∇d:

- Finite difference discretisation (**LVFE_GW**=F):
  First the horizontal gradient of the surface vertical velocity must be computed, using the surface boundary condition:

$$\nabla[gw_{\mathrm{surf}}] = \nabla[\mathbf{V}_{\mathrm{surf}}\nabla[\Phi_{\mathrm{s}}]] \tag{275}$$

  The RHS of this equation is the vector of components:

$$[\nabla_{\mathrm{zo}}U_{\mathrm{surf}}][\nabla_{\mathrm{zo}}\Phi_{\mathrm{s}}] + U_{\mathrm{surf}}[\nabla^2_{\mathrm{zo\ zo}}\Phi_{\mathrm{s}}] + [\nabla_{\mathrm{zo}}V_{\mathrm{surf}}][\nabla_{\mathrm{me}}\Phi_{\mathrm{s}}] + V_{\mathrm{surf}}[\nabla^2_{\mathrm{zo\ me}}\Phi_{\mathrm{s}}]$$

  and

$$[\nabla_{\mathrm{me}}U_{\mathrm{surf}}][\nabla_{\mathrm{zo}}\Phi_{\mathrm{s}}] + U_{\mathrm{surf}}[\nabla^2_{\mathrm{zo\ me}}\Phi_{\mathrm{s}}] + [\nabla_{\mathrm{me}}V_{\mathrm{surf}}][\nabla_{\mathrm{me}}\Phi_{\mathrm{s}}] + V_{\mathrm{surf}}[\nabla^2_{\mathrm{me\ me}}\Phi_{\mathrm{s}}]$$

  Integrating equation (269) from the surface, and applying the gradient operator, yields:

$$[\Delta\nabla[gw]]_l = \nabla[R_{\mathrm{a}l}T_l]\left[\frac{\Pi}{p}\right]_l d_l\delta_l + R_{\mathrm{a}l}T_l\nabla\left[\frac{\Pi}{p}\right]_l d_l\delta_l + R_{\mathrm{a}l}T_l\left[\frac{\Pi}{p}\right]_l \nabla d_l\delta_l + R_{\mathrm{a}l}T_l\left[\frac{\Pi}{p}\right]_l d_l\nabla\delta_l \tag{276}$$

  Note that $\nabla\left[\frac{\Pi}{p}\right]_l$ can be rewritten:

$$\nabla\left[\frac{\Pi}{p}\right]_l = -\left[\frac{\Pi}{p}\right]_l \nabla\left[\log\frac{p}{\Pi}\right]_l$$

  The half level values are given by:

$$\nabla[gw_{\bar{l}}] = \nabla[gw_{\mathrm{surf}}] + \sum_{k=L}^{k=l+1}[\Delta\nabla[gw]]_k \tag{277}$$

  and:

$$\nabla[gw_l] = \nabla[gw_{\bar{l}}] + \nabla[R_{\mathrm{a}l}T_l]\left[\frac{\Pi}{p}\right]_l d_l\alpha_l + R_{\mathrm{a}l}T_l\nabla\left[\frac{\Pi}{p}\right]_l d_l\alpha_l + R_{\mathrm{a}l}T_l\left[\frac{\Pi}{p}\right]_l \nabla d_l\alpha_l + R_{\mathrm{a}l}T_l\left[\frac{\Pi}{p}\right]_l d_l\nabla\alpha_l \tag{278}$$

- Finite element discretisation (**LVFE_GW**=T): Use equation (275) at the surface; for upper-air full level values of $\nabla[gw]$ discretisation is:

$$\nabla[gw_l] = \nabla[gw_{\mathrm{surf}}] + [\mathcal{R}_{\mathrm{inte}}]_{(surf,l)} \left\langle -\frac{\nabla[\frac{\Pi}{p}dR_{\mathrm{a}}T\delta]}{\Delta\eta} \right\rangle \tag{279}$$

  with:

$$\left[\nabla\left(\frac{\Pi}{p}dR_{\mathrm{a}}T\delta\right)\right]_l = \nabla[R_{\mathrm{a}l}T_l]\left[\frac{\Pi}{p}\right]_l d_l\delta_l + R_{\mathrm{a}l}T_l\nabla\left[\frac{\Pi}{p}\right]_l d_l\delta_l + R_{\mathrm{a}l}T_l\left[\frac{\Pi}{p}\right]_l \nabla d_l\delta_l + R_{\mathrm{a}l}T_l\left[\frac{\Pi}{p}\right]_l d_l\nabla\delta_l$$

Remarks: there are two difficulties which can appear for some options:

- Case **NVDVAR**=4: calculation of ∇d is required, but in this case this is ∇d_4 which is available, and $\nabla\mathtt{X}$ is not convenient to compute. This is why there is an additional thermodynamic variable “NHX” which contains $\mathtt{X}$ and $\nabla\mathtt{X}$.
- Case **LSPRT**=.T. and $R_{\mathrm{a}} = R_{\mathrm{d}}$: in this case RT is stored instead of T, and $\nabla[RT]$ is available in the code, but not $\nabla[R_{\mathrm{d}}T]$. We can see in formula (276) that the computation of $\nabla[R_{\mathrm{d}}T]$ is required, and currently we have approximated it by $\nabla[RT]$. This issue disappears when using a definition of d with R.

∗ Laplacian term $\frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta}$.

- VFD discretisation (**LVFE_LAPL**=F): both derivatives are computed separately. $\frac{\partial(p-\Pi)}{\partial\Pi}$ (at half levels) appears in the RHS of the w-equation. $\frac{\partial...}{\partial\eta}$ (at full levels) is applied when transforming w into d or in term $\frac{\partial[\frac{dw}{dt}]_{\mathrm{ad}}}{\partial\eta}$ in the RHS of the d or d_4 equation.
- VFE discretisation (**LVFE_LAPL**=T): this term requires some attention and its study is still an issue. It seems that writing it into a combination of two first-order derivatives does not give a stable scheme. For the time being the following formula is applied:

$$\frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta} = \frac{1}{\Pi}\frac{\partial\left(\frac{p-\Pi}{\Pi}\right)}{\partial\eta}\frac{\partial\left[\frac{\partial\eta}{\partial\Pi}\Pi^2\right]}{\partial\eta} + \frac{\partial\eta}{\partial\log\Pi}\frac{\partial^2\left(\frac{p-\Pi}{\Pi}\right)}{\partial\eta^2} \tag{280}$$

Discretisation is:

$$\left[\frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta}\right]_l = \frac{1}{\Pi_l}\left[\mathcal{R}_{\mathrm{deri}}\left\langle\frac{p-\Pi}{\Pi}\right\rangle\right]_l\left[\mathcal{R}_{\mathrm{deri}}\left\langle\frac{\partial\eta}{\partial\Pi}\Pi^2\right\rangle\right]_l + \frac{[\Delta\eta]_l}{\delta_l}\left[\mathcal{R}_{\mathrm{dderi}}\left\langle\frac{p-\Pi}{\Pi}\right\rangle\right]_l \tag{281}$$

where $\mathcal{R}_{\mathrm{dderi}}$ is the second-order derivative VFE operator.

13.2 Momentum equation.

∗ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = \mathbf{V} \tag{282}$$

$$\mathcal{A} = -2\mathbf{\Omega}\wedge\mathbf{V} - \frac{\partial p}{\partial\Pi}\nabla\Phi - RT\frac{\nabla p}{p} \tag{283}$$

$$\mathcal{L} = -\nabla\left[\gamma T - T^*(\gamma\hat{Q}) + R_{\mathrm{d}}T^*\log(\Pi_{\mathrm{s}}) + R_{\mathrm{d}}T^*\hat{Q}\right] + \beta_{Co}[-2(\mathbf{\Omega}\wedge\mathbf{V})] \tag{284}$$

$$F = \mathbf{F_V} \tag{285}$$

Top and bottom values are defined as follows:

$$\mathbf{V}_{\eta=0} = \mathbf{V}_{l=1} \tag{286}$$

If $\delta m = 0$:

$$\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L} \tag{287}$$

If $\delta m = 1$:

$$\mathbf{V}_{\eta=1} = 0 \tag{288}$$

Note that this bottom condition is not fully consistent with the one done in the remaining part of the NH code ($\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L}$).

∗ Remarks. the remarks done for the hydrostatic model remain valid here. We must add an additional remark: in some cases we need this equation also at the surface, with some assumptions (explicit treatment of the Coriolis term, $\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L}$).

∗ Discretisation of $-2\mathbf{\Omega}\wedge\mathbf{V}$ at full levels: It is the same as for the hydrostatic model.

∗ Discretisation of the pressure gradient term at full levels: It is modified compared to the hydrostatic case.
The pressure gradient term writes:

$$-\left(\frac{\partial p}{\partial\Pi}\nabla\Phi + RT\frac{\nabla p}{p}\right)$$

It contains the geopotential $\Phi = gz$.

- **LVERTFE**=.F.: the pressure gradient term is rewritten in order to isolate a main part which looks like the hydrostatic pressure gradient term, and to isolate some purely anhydrostatic contributions. Its discretisation at full levels is written under a sum of 4 terms. The first one looks like the hydrostatic pressure gradient term; the following ones become zero in a hydrostatic model.

$$-\left[\nabla\Phi + \frac{\Pi}{p}RT\nabla\log\Pi\right]_l - \left[\left(\frac{\Delta p}{\Delta\Pi} - 1\right)\nabla\Phi\right]_l - \left[RT\nabla\left[\log\frac{p}{\Pi}\right]\right]_l - \left[RT\left(1 - \frac{\Pi}{p}\right)\nabla\log\Pi\right]_l$$

- Term $\left[\nabla\Phi + \frac{\Pi}{p} RT\nabla\log\Pi\right]_l$: it looks like the pressure gradient term in the hydrostatic model, but with an additional factor $\frac{\Pi}{p}$; its treatment is similar to what is done in the hydrostatic model:

$$\left[\nabla\Phi + \tfrac{\Pi}{p} RT\nabla\log\Pi\right]_l =$$
$$\nabla\Phi_{\rm s} + \sum_{k=L}^{l+1}\left[\tfrac{\Pi}{p}\right]_k [\nabla(RT)]_k\,\delta_k - \sum_{k=L}^{l+1}\left[\tfrac{\Pi}{p}\right]_k \nabla\log\left[\tfrac{p}{\Pi}\right]_k (RT)_k\delta_k + \sum_{k=L}^{l+1}\left[\tfrac{\Pi}{p}\right]_k (RT)_k\,[\nabla\delta]_k$$
$$+\left[\tfrac{\Pi}{p}\right]_l [\nabla(RT)]_l\,\alpha_l - \left[\tfrac{\Pi}{p}\right]_l \nabla\log\left[\tfrac{p}{\Pi}\right]_l (RT)_l\alpha_l + \left[\tfrac{\Pi}{p}\right]_l (RT)_l\,[\nabla(\alpha+\log\Pi)]_l \quad (289)$$

- Term $\left[\left(\frac{\Delta p}{\Delta\Pi} - 1\right)\nabla\Phi\right]_l$: its discretisation is:

$$\left[\left(\frac{\Delta p}{\Delta\Pi} - 1\right)\nabla\Phi\right]_l = \left(\frac{[\Delta p]_l}{[\Delta\Pi]_l} - 1\right)[\nabla\Phi]_l \quad (290)$$

  For $[\nabla\Phi]_l$ see equation (303). Discretisation of $[\Delta p]_l$ has already been studied.

- Term $\left[RT\nabla\left[\log\frac{p}{\Pi}\right]\right]_l$: its discretisation is:

$$\left[RT\nabla\left[\log\frac{p}{\Pi}\right]\right]_l = [RT]_l\nabla\hat{Q}_l \quad (291)$$

- Term $\left[RT\left(1-\frac{\Pi}{p}\right)\nabla\log\Pi\right]_l$: its discretisation is:

$$\left[RT\left(1-\frac{\Pi}{p}\right)\nabla\log\Pi\right]_l = [RT]_l\left[1-\frac{\Pi_l}{p_l}\right]\left[\frac{\nabla\Pi}{\Pi}\right]_l \quad (292)$$

  See part (8.4) for the discretisation of $\left[\frac{\nabla\Pi}{\Pi}\right]_l$.

- **LVERTFE**=.T.:

$$-\left[\frac{\partial p}{\partial\Pi}\nabla\Phi + RT\frac{\nabla p}{p}\right]_l = -\frac{[\Delta p]_l}{[\Delta\Pi]_l}[\nabla\Phi]_l - [RT]_l\left[\frac{\nabla p}{p}\right]_l \quad (293)$$

  where:

  - $[\nabla\Phi]_l$ is discretised with applying the operator $\mathcal{R}_{\rm inte}$ (see equation (304)).
  - $[\Delta p]_l$ has been already studied above.
  - $\left[\frac{\nabla p}{p}\right]_l$: for **NPDVAR**=2 it is convenient to rewrite this term as follows:

$$\left[\frac{\nabla p}{p}\right]_l = [\nabla\hat{Q}]_l + \left[\frac{\nabla\Pi}{\Pi}\right]_l$$

∗ **Discretisation of the pressure gradient term at the surface:** It is computed in the RHS of equation giving $\frac{d\mathbf{V}_{\rm surf}}{dt}$, in the case **LVERTFE**=.F. . Currently the pressure gradient term at the surface is set to the pressure gradient term at the full level $l = L$.

∗ **Discretisation of the grid-point Rayleigh friction:** It is the same as for the hydrostatic model.

13.3 Thermodynamic equation.

∗ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = T \quad (294)$$

$$\mathcal{A} = -\frac{RT}{c_{\rm v}}D_3 \quad (295)$$

$$\mathcal{L} = -\frac{R_{\rm d}T^*}{c_{\rm v\,d}}\left[\overline{M}^2 D' + d\right] \quad (296)$$

$$F = \left[\frac{c_{\rm p}}{c_{\rm v}}F_{\rm T}\right] \quad (297)$$

Top:

$$T_{\eta=0} = T_{l=1} \quad (298)$$

Bottom if $\delta m = 0$:

$$T_{\eta=1} = T_{l=L} \quad (299)$$

Bottom if $\delta m = 1$ (output of physics):

$$T_{\eta=1} = T_{\rm s} \quad (300)$$

$*$ **Discretisation of the conversion term.**

$$\left[\frac{RT}{c_{\rm v}}D_3\right]_l = \frac{R_lT_l}{[c_{\rm v}]_l}\,[D_3]_l \tag{301}$$

See part (13.1) for discretisation of D_3 at full levels.

13.4 Continuity equation.

Discretisation is identical to the hydrostatic model one.

13.5 Moisture and other GFL equations.

Formulation is identical to the hydrostatic model.

13.6 Relationship between geopotential height gz and pressure depth.

$*$ **Geopotential height at half levels (case LVERTFE=.F. only):** Discretisation of equation (103) at half levels yields:

$$gz_{\bar{l}} = gz_{\rm s} + \sum_{k=L}^{k=l+1} \frac{\Pi_k}{p_k}\,[R_kT_k]\,\delta_k \tag{302}$$

See section (8.5) for discretisation of δ at full levels.

$*$ **Geopotential height at full levels:**

- Case **LVERTFE**=.F.: It is computed from the geopotential height at half levels by the following relationship:

$$gz_l = gz_{\bar{l}} + \frac{\Pi_l}{p_l}\,[R_lT_l]\,\alpha_l \tag{303}$$

  See section (8.5) for discretisation of α at full levels.

- Case **LVERTFE**=.T.:

$$gz_l = gz_{\rm s} + [\mathcal{R}_{\rm inte}]_{(surf,l)}\left\langle -\frac{\Pi}{p}\frac{RT\delta}{\Delta\eta}\right\rangle \tag{304}$$

  See section (8.5) for discretisation of δ at full levels.

$*$ **Horizontal gradient of the geopotential height at half levels (case LVERTFE=.F. only):** One applies the operator ∇ to equation (302). That yields:

$$g\,[\nabla z]_{\bar{l}} = g\nabla z_{\rm s} + \textstyle\sum_{k=L}^{k=l+1}\frac{\Pi_k}{p_k}\,[\nabla(RT)]_k\,\delta_k - \sum_{k=L}^{k=l+1}\frac{\Pi_k}{p_k}\nabla\log\frac{p_k}{\Pi_k}\,[R_kT_k]\,\delta_k$$
$$+\textstyle\sum_{k=L}^{k=l+1}\frac{\Pi_k}{p_k}\,[R_kT_k]\,[\nabla\delta]_k \tag{305}$$

See section (8.5) for discretisation of δ and $\nabla\delta$ at full levels.

$*$ **Horizontal gradient of the geopotential height at full levels:** One applies the operator ∇ to equation (303). That yields:

- Case **LVERTFE**=.F.:

$$g\,[\nabla z]_l = g\,[\nabla z]_{\bar{l}} + \frac{\Pi_l}{p_l}\,[\nabla(RT)]_l\,\alpha_l - \frac{\Pi_l}{p_l}\nabla\log\frac{p_l}{\Pi_l}\,[R_lT_l]\,\alpha_l + \frac{\Pi_l}{p_l}\,[R_lT_l]\,[\nabla\alpha]_l \tag{306}$$

  See section (8.5) for discretisation of α and $\nabla\alpha$ at full levels.

- Case **LVERTFE**=.T.:

$$g\,[\nabla z]_l = g\nabla z_{\rm s} + [\mathcal{R}_{\rm inte}]_{(surf,l)}\left\langle -\frac{\frac{\Pi}{p}\nabla(RT)\delta - \frac{\Pi}{p}\nabla(\log\frac{p}{\Pi})RT\delta + \frac{\Pi}{p}RT\nabla\delta}{\Delta\eta}\right\rangle \tag{307}$$

  See section (8.5) for discretisation of δ and $\nabla\delta$ at full levels.

13.7 Diagnostic expression of some vertical velocities.

Formulation is identical to the hydrostatic model one.

13.8 Pressure departure equation.

$*$ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = \hat{Q} \tag{308}$$

$$\mathcal{A} = -\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}} D_3 - \frac{\omega}{\Pi} \tag{309}$$

$$\mathcal{L} = -\left[\frac{c_{\mathrm{p\,d}}}{c_{\mathrm{v\,d}}}(\overline{M}^2 D' + d) - \frac{c_{\mathrm{p\,d}}}{R_{\mathrm{d}} T^*}\tau(\overline{M}^2 D')\right] \tag{310}$$

$$F = \frac{c_{\mathrm{p}}}{c_{\mathrm{v}} T} F_{\mathrm{T}} \tag{311}$$

Top:

$$\hat{Q}_{\eta=0} = 0 \tag{312}$$

Bottom if $\delta m = 0$:

$$\hat{Q}_{\eta=1} = \hat{Q}_{l=L} \tag{313}$$

Bottom if $\delta m = 1$ (output of physics):

$$\hat{Q}_{\eta=1} = \hat{Q}_{l=L} \tag{314}$$

$*$ Discretisation of the RHS term.

$$\left[-\frac{c_{\mathrm{p}}}{c_{\mathrm{v}}} D_3 - \frac{\omega}{\Pi}\right]_l = -\frac{[c_{\mathrm{p}}]_l}{[c_{\mathrm{v}}]_l}[D_3]_l - \left[\frac{\omega}{\Pi}\right]_l \tag{315}$$

See part (13.1) for discretisation of D_3 at full levels. Discretisation of $\frac{\omega}{\Pi}$ has already been studied.

13.9 Vertical divergence d and gw equations.

13.9.1 Definitions.

$*$ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values, for the vertical divergence.

$$X = d \tag{316}$$

$$\mathcal{A} = -dD_3 + d\nabla\mathbf{V} - \frac{gp}{R_{\mathrm{a}} T \frac{\partial\Pi}{\partial\eta}} \frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta} + \frac{gp}{R_{\mathrm{a}} T \frac{\partial\Pi}{\partial\eta}} (\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right) \tag{317}$$

$$\mathcal{L} = -\frac{g^2}{R_{\mathrm{d}} T_{\mathrm{a}}^*}(\mathbf{L}^* \hat{Q}) \tag{318}$$

$$F = -\frac{d}{\left[\frac{\partial\Pi}{\partial\eta}\right]} F_{\mathrm{m}}' - \left[\frac{gp}{R_{\mathrm{a}} T \frac{\partial\Pi}{\partial\eta}} \frac{\partial F_{\mathrm{w}}}{\partial\eta}\right] \tag{319}$$

Top:

$$d_{\eta=0} = d_{l=1} \tag{320}$$

Bottom if $\delta m = 0$:

$$d_{\eta=1} = d_{l=L} \tag{321}$$

Bottom if $\delta m = 1$: currently cf. $\delta m = 0$. Note that this bottom condition is not fully consistent with the one done on the horizontal wind and the vertical velocity w.

$*$ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values, for the vertical velocity.

$$X = w \tag{322}$$

$$\mathcal{A} = g\frac{\partial(p - \Pi)}{\partial\Pi} \tag{323}$$

$$F = F_{\mathrm{w}} \tag{324}$$

Top:

- **LVFE_GW**=.F.: $w_{\eta=0}$ is computed by the general formula giving w at half levels.
- **LVFE_GW**=.T.: top value of w is useless.

Bottom if $\delta m = 0$:

$$w_{\eta=1} = \mathbf{V}_{\mathrm{surf}} \nabla z_{\mathrm{surf}} \tag{325}$$

Bottom if $\delta m = 1$: currently the same as for $\delta m = 0$. Remark: we also need the equation of the surface vertical velocity.

∗ Meaning of variable LGWADV:

- **LGWADV**=F: d or d_4 is the prognostic variable for both explicit model and linear terms. The RHS of gw equation may be used for intermediate calculations.
- **LGWADV**=T: gw is the prognostic variable for explicit model; linear terms are computed separately for the d or d_4 equation. Order of calculations follows:
  - Compute $w(t)$, and $w(t - \Delta t)$ too for leap-frog advection.
  - Compute the RHS of gw equation.
  - Provide an explicit value of $w(t + \Delta t)$.
  - Convert it into an explicit value of $d(t + \Delta t)$.
  - Add $\mathtt{X}$ contribution and linear terms to provide the $d_4(t + \Delta t)$ entering spectral computations.

LGWADV=T is currently coded for the semi-Lagrangian advection, not for the Eulerian advection.

13.9.2 Discretisation of the RHS of $[gw_{\rm surf}]$ equation.

This calculation is required in most cases; this RHS is the sum of the product:

$$\frac{d\mathbf{V}_{\rm surf}}{dt}\nabla\Phi_{\rm s}$$

and the "Jacobian" term:

$$\mathbf{V}_{\rm surf}\left(\mathbf{V}_{\rm surf}\nabla[\nabla\Phi_{\rm s}]\right)$$

- $\frac{d\mathbf{V}_{\rm surf}}{dt}$ has already been studied above.
- The "Jacobian" term can be developed as follows:

$$U_{\rm surf}^2[\nabla^2_{\rm zo\ zo}\Phi_{\rm s}] + 2U_{\rm surf}V_{\rm surf}[\nabla^2_{\rm zo\ me}\Phi_{\rm s}] + V_{\rm surf}^2[\nabla^2_{\rm me\ me}\Phi_{\rm s}]$$

13.9.3 Use of d_4 as prognostic variable: discretisation of $\frac{\partial \mathtt{X}}{\partial t}$.

In equation (112), the calculation of $\frac{\partial \mathtt{X}}{\partial t}$ is not done by an Eulerian temporal advection but simply by a diagnostic evaluation. In the Eulerian scheme without any predictor-corrector scheme, the discretisation of $\frac{\partial \mathtt{X}}{\partial t}$ follows:

$$\frac{\partial \mathtt{X}}{\partial t} = \frac{\mathtt{X}^o - \mathtt{X}^-}{\Delta t} \tag{326}$$

Some remarks:

- Advection terms are for the time being ignored in $\frac{\partial \mathtt{X}}{\partial t}$ (to be added in the future?).
- A better evaluation would be probably

$$\frac{\partial \mathtt{X}}{\partial t} = \frac{\mathtt{X}^+ - \mathtt{X}^-}{2\Delta t}$$

  where $\mathtt{X}^+$ is a sort of provisional evaluation of $\mathtt{X}$ at $t + \Delta t$, mixing t and $t + \Delta t$ quantities (like we do in the semi-Lagrangian scheme for **ND4SYS**=2).

13.9.4 Discretisation of the RHS of the vertical divergence d equation: terms other than the "Laplacian term".

Variable **LNHEE_SVDLAPL_FIRST** has no influence on calculation of these terms, which is done only if **LGWADV**=F.

- Term $-dD_3 + d\nabla\mathbf{V}$ at full levels:

$$[-dD_3 + d\nabla\mathbf{V}]_l = -d_l[D_3]_l + d_l[\nabla\mathbf{V}]_l \tag{327}$$

  This discretisation can be applied for both finite difference and finite element vertical discretisations.
- Term $\mathtt{Z} = -\frac{gp}{R_{\rm a}T\frac{\partial\Pi}{\partial\eta}}(\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right)$: its discretisation looks like the one of $\mathtt{X}$.

– **LVFE_Z_TERM**=.F.: This term is rewritten:

$$\mathtt{Z} = -\frac{gp}{R_\mathrm{a}T\Pi}(\nabla w)\left(\frac{\partial \mathbf{V}}{\partial \log \Pi}\right)$$

Its discretisation at full levels is:

$$\mathtt{Z}_l = -\frac{p_l}{\Pi_l}\frac{g}{R_{\mathrm{a}l}T_l\delta_l}\left[\nabla[w]_{\bar{l}-1}(\mathbf{V}_l - \mathbf{V}_{\bar{l}-1}) + \nabla[w]_{\bar{l}}(\mathbf{V}_{\bar{l}} - \mathbf{V}_l)\right] \quad (328)$$

This discretisation uses $\nabla[w]$ at half levels and $\mathbf{V}$ at both full and half levels. The discretisation of $\nabla[gw]$ at half levels is given by equation (277).

– **LVFE_Z_TERM**=.T.: $\mathtt{Z}$ is discretised at full levels as follows:

$$\mathtt{Z}_l = -\frac{p_l}{\Pi_l}\frac{g[\Delta\eta]_l}{R_{\mathrm{a}l}T_l\delta_l}\nabla[w]_l[\mathcal{R}_\mathrm{deri}\mathbf{V}]_l \quad (329)$$

13.9.5 Discretisation of term $g\frac{\partial(p-\Pi)}{\partial\Pi}$, and of the "Laplacian term" in the RHS of the vertical divergence d equation: case LNHEE_SVDLAPL_FIRST=F approach.

Option **LNHEE_SVDLAPL_FIRST**=F:

- uses the decomposition of second-order derivative **L** into the product of two first-order derivatives ($\overline{\partial}$ and $(\overline{\partial} + \mathtt{I})$) in both the d and gw equations RHS.
- first computes $g\frac{\partial(p-\Pi)}{\partial\Pi}$, (equivalent to use the product by $(\overline{\partial} + \mathtt{I})$).
- then apply $\overline{\partial}$ to compute the Laplacian term in the RHS of d equation.
- that means that $g\frac{\partial(p-\Pi)}{\partial\Pi}$ is computed even if **LGWADV**=F.
- this approach is easy to code in the NHEE model, and very difficult to code in the NHQE model. **LNHEE_SVDLAPL_FIRST**=F is indeed not convenient to keep for future research options expected in the future (not yet coded nor documented here).
- we also notice use of redundant key **LVFE_DELNHPRE**, not always combined with **LVFE_GW** and **LVFE_LAPL** in a consistent manner.

∗ Discretisation of term $g\frac{\partial(p-\Pi)}{\partial\Pi}$:

- **LVFE_GW**=.F., if **LVERTFE**=.T., **LVFE_ECMWF**=.F. and **LVFE_DELNHPRE**=.T.:

$$\left[g\frac{\partial(p-\Pi)}{\partial\Pi}\right]_{\bar{l}} = g[\mathcal{R}_\mathrm{deri}\,(p-\Pi)]_{\bar{l}}\frac{\eta_{l+1}-\eta_l}{\Pi_{l+1}-\Pi_l} \quad (330)$$

- **LVFE_GW**=.F., other cases:

$$\left[g\frac{\partial(p-\Pi)}{\partial\Pi}\right]_{\bar{l}} = g\frac{[p-\Pi]_{l+1}-[p-\Pi]_l}{\Pi_{l+1}-\Pi_l} \quad (331)$$

At the top of the model, this formula becomes:

$$\left[g\frac{\partial(p-\Pi)}{\partial\Pi}\right]_\mathrm{top} = g\frac{[p-\Pi]_{l=1}}{\Pi_{l=1}-\Pi_\mathrm{top}} \quad (332)$$

- **LVFE_GW**=.T.: This quantity must be evaluated at full levels: use the full level $[\Delta p]_l$ (see its discretisation in part 13.1).

  Additional remark for **LGWADV**=T: once updated temporally gw, using the gw equation, it is recommended to apply vertical derivatives on temporal increments (and not directly on the temporally updated gw) to restore the temporally updated d. This precaution is to ensure that we discretise $d(t+\Delta t) = d(t-\Delta t) + 2\Delta t\frac{\partial d}{\partial t}$ and not something looking like: $d(t+\Delta t) = \mathcal{R}_\mathrm{deri}(\mathcal{R}_\mathrm{inte}(d(t-\Delta t))) + 2\Delta t\frac{\partial d}{\partial t}$.

$*$ Discretisation of the "Laplacian" term in the RHS of the vertical divergence d equation: The term denoted "Laplacian" term is the term $-\frac{gp}{R_a T \frac{\partial\Pi}{\partial\eta}} \frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta}$:

- **LVFE_GW**=.F.: This term is rewritten:

$$-\frac{gp}{R_a T\Pi} \frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\log\Pi}$$

  Its discretisation at full levels is:

$$\left[-\frac{gp}{R_a T\Pi} \frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\log\Pi}\right]_l = -\frac{p_l}{\Pi_l} \frac{g}{R_{al} T_l \delta_l} \left[\left[\frac{dw}{dt}\right]_{\mathrm{ad},\bar{l}} - \left[\frac{dw}{dt}\right]_{\mathrm{ad},\bar{l}-1}\right] \quad (333)$$

- **LVFE_GW**=.T.: It is desirable to isolate the Laplacian term:

$$\left[\frac{dw}{dt}\right]_{\mathrm{ad}} = \left[\frac{dw}{dt}\right]_{\mathrm{lap}} + \left[\frac{dw}{dt}\right]_{\mathrm{oth}}$$

  The Laplacian contribution provides the term:

$$-\frac{g^2 p}{R_a T \frac{\partial\Pi}{\partial\eta}} \frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta}$$

  Its discretisation at full levels is:

$$\left[-\frac{g^2 p}{R_a T \frac{\partial\Pi}{\partial\eta}} \frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta}\right]_l = -\frac{p_l}{\Pi_l} \frac{g^2}{R_{al} T_l \delta_l} [\Delta\eta]_l \left[\frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta}\right]_l$$

  The discretisation of $\frac{\partial\left(\frac{\partial(p-\Pi)}{\partial\Pi}\right)}{\partial\eta}$ has been studied in part 13.1.

Remark: the discretisation of this term, combined with the discretisation of the RHS of the w equation, must respect the constraint "C2" of (IDVNH2.1).

13.9.6 Discretisation of term $g\frac{\partial(p-\Pi)}{\partial\Pi}$, and of the "Laplacian term" in the RHS of the vertical divergence d equation: case LNHEE_SVDLAPL_FIRST=T approach.

Option **LNHEE_SVDLAPL_FIRST**=T:

- directly uses the discretisation of **L** in the RHS of d equation.
- to discretize $g\frac{\partial(p-\Pi)}{\partial\Pi}$, we use the property, for a quantity Z:

$$\mathbf{L}Z = \overline{\partial}(\overline{\partial} + \mathtt{I})Z$$

  which can be rewritten:

$$(\overline{\partial} + \mathtt{I})Z = (\overline{\partial}^{-1}\mathbf{L})Z$$

- uses the discretisation of the product $\overline{\partial}^{-1}\mathbf{L}$ in the RHS of gw equation; note that $\overline{\partial}^{-1}$ is a **G**-type vertical integral (see Appendix 1) which is similar to the one allowing to compute gw from d; for **LVFE_GW=F** this **G**-type vertical integral is discretised by a VFD method and this is the only case coded and documented.
- the study and the implementation is for the time being limited to **LVFE_LAPL**=F, **LVFE_GW**=F, **LVFE_DELNHPRE**=F.
- organisation of the code is simpler than with **LNHEE_SVDLAPL_FIRST**=F, and key **LVFE_DELNHPRE** becomes irrelevant.

$*$ Discretisation of the "Laplacian" term in the RHS of the vertical divergence d equation: Term $-\frac{gp}{R_a T \frac{\partial\Pi}{\partial\eta}} \frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta}$: its expression can be rewritten, using operators **L** and $\overline{\partial}$:

$$-\frac{gp}{R_a T \frac{\partial\Pi}{\partial\eta}} \frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta} = -\frac{g^2}{R_a T}[\mathbf{L}(p/\Pi - 1)]$$

- **LVFE_LAPL**=.F.: **L** is a tridiagonal operator giving a full level quantity from full level values of $(p/\Pi-1)$. Discretisation writes, for $Z = p/\Pi - 1$:

$$(\mathbf{L}Z)_l = \mathbf{A}_l(Z_{l-1} - Z_l) + \mathbf{C}_l(Z_{l+1} - Z_l)$$

  with

$$\mathbf{A}_l = \frac{\Pi_{l-1}}{\delta_l(\Pi_l - \Pi_{l-1})}$$
$$\mathbf{C}_l = \frac{\Pi_{l+1}}{\delta_l(\Pi_{l+1} - \Pi_l)}$$

  There are particular discretisations at the top and bottom. For example we replace Z_{L+1} and Z_0 by 0. To take account of bottom condition (w_{surf} different from 0) we add $(1/g)\frac{dw_{surf}/dt}{\delta_L}$ to $(\mathbf{L}(p/\Pi - 1))_L$.

  Full level quantity $\frac{g^2}{R_\mathrm{a}T}$ in factor is discretised:

$$\left[\frac{g^2}{R_\mathrm{a}T}\right]_l = \frac{g^2}{R_{\mathrm{a}l}T_l}$$

- **LVFE_LAPL**=.T.: this case has not been yet studied, but the code under (**LNHEE_SVDLAPL_FIRST**=.F.,**LVFE_LAPL**=.T.) already provides a VFE discretisation for **L**.

$*$ **Discretisation of term** $g\frac{\partial(p-\Pi)}{\partial\Pi}$**:** We can notice the identity:

$$g\frac{\partial(p-\Pi)}{\partial\Pi} = g(\overline{\partial} + \mathtt{I})(p/\Pi - 1)$$

and we prefer to use the equivalent formulation:

$$g\frac{\partial(p-\Pi)}{\partial\Pi} = g(\overline{\partial}^{-1}\mathbf{L})(p/\Pi - 1)$$

- The product $\mathbf{L}(p/\Pi - 1)$, also done in the Laplacian term of the vertical divergence equation, provides $[\mathbf{L}(p/\Pi - 1)]$ at full levels from $(p/\Pi - 1)$ at full levels, for both VFD and VFE discretisation of **L**.
- We then apply the multiplication by operator $\overline{\partial}^{-1}$ to full levels of $[\mathbf{L}(p/\Pi - 1)]$.
  - for **LVFE_GW**=F this is a VFD **G**-type vertical integral, the same integral which is used to retrieve half-level $[gw]$ from full levels d (studied above). It provides $g\frac{\partial(p-\Pi)}{\partial\Pi}$, i.e. $\left[\frac{dw}{dt}\right]_{\mathrm{ad}}$, at half levels.

$$\left[\frac{dw}{dt}\right]_{\bar{l}} = \frac{dw_{\mathrm{surf}}}{dt} - g\sum_{k=L}^{k=l+1}([\mathbf{L}(p/\Pi - 1)]_k\delta_k)$$

  - for **LVFE_GW**=T (not implemented) this is a VFE **G**-type vertical integral, the same integral which is used to retrieve full-level $[gw]$ from full levels d.
  - the surface condition giving the RHS of $[gw_{\mathrm{surf}}]$ equation is used.
  - note that "VFE vs VFD" involves keys **LVFE_GW** and **LVFE_LAPL**, but never **LVFE_DELNHPRE** (which is redundant).

14 The Eulerian discretisation of the 3D quasi-compressible non-hydrostatic model (NHQE).

14.1 Discretisation of some intermediate quantities.

We have to discretise at least the following quantities:

- Surface wind $\mathbf{V}_{\text{surf}}$.
- Half levels wind when required.
- $\mathtt{X}_{\text{S}}$ and $\mathtt{X}_{\text{D}}$.
- κ at half levels.
- $\exp(\kappa\hat{Q})$ and $\exp(-\kappa\hat{Q})$ (levels and half levels).
- gw and its horizontal gradient.
- d (knowing gw).
- D_3.

Apart from some vertical integrals (the discretisation of which is provided for both VFE and VFD), the discretisations provided are generally valid for VFD only (especially for vertical derivatives or expressions requiring half level quantities). VFE discretisation is still under study.

$*$ **$[\Delta\Pi]$ at half levels 1 to $L-1$:** This quantity appears at several locations, and there is a specificity for $\bar{l}=1$.

- Half levels 2 to $L-1$: $[\Delta\Pi]_{\bar{l}} = \Pi_{l+1} - \Pi_l$
- Half level 1: $[\Delta\Pi]_{\bar{l}=1} = \Pi_2/\delta_1 - \Pi_1(\alpha_2 - \delta_2)$

$*$ **κ at half levels:** This quantity appears in the horizontal pressure gradient term:

- Half levels 1 to $L-1$: $\kappa_{\bar{l}} = 0.5(\kappa_l + \kappa_{l+1})$
- Top: $\kappa_{\text{top}} = \kappa_{l=1}$
- Bottom: $\kappa_{\text{surf}} = \kappa_{l=L}$

$*$ **$\hat{Q}$ at half levels:** Discretisation is similar to the NHEE one, and is used for the horizontal pressure gradient term:

- $\hat{Q}_{\text{top}} = 0$
- $\hat{Q}_{\text{surf}} = \hat{Q}_{l=L}$
- For the other half levels: $\hat{Q}_{\bar{l}} = 0.5(\hat{Q}_l + \hat{Q}_{l+1})$

Calculation of half level p and of full level Δp is not necessary.

$*$ **$\exp(\kappa\hat{Q})$ at full levels:** $\exp(\kappa\hat{Q})$ at full levels is simply:

$$[\exp(\kappa\hat{Q})]_l = \exp(\kappa_l\hat{Q}_l)$$

$*$ **$\exp(\kappa\hat{Q})$ at half levels for horizontal pressure gradient term:**

- Half levels 1 to $L-1$ for horizontal pressure gradient term: $[\exp(\kappa\hat{Q})]_{\bar{l}} = \exp(\kappa_{\bar{l}}\hat{Q}_{\bar{l}})$
- Surface: $\exp(\kappa\hat{Q})_{\text{surf}} = \exp(\kappa\hat{Q})_{l=L}$
- Top: $\exp(\kappa\hat{Q})_{\text{top}} = 1$

$*$ **Surface wind and half level wind:** same discretisation as in the NHEE model.

$*$ **Quantity $\mathtt{X}_{\text{S}}$ at full levels:** It is close to the NHEE discretisation of $\mathtt{X}$. For VFD discretisation (**LVFE_XTERM**=F) that yields:

$$[\mathtt{X}_{\text{S}}]_l = \frac{1}{R_l\tilde{T}_l\delta_l}\left[[\nabla\Phi]_{\overline{l-1}}(\mathbf{V}_l - \mathbf{V}_{\overline{l-1}}) + [\nabla\Phi]_{\bar{l}}(\mathbf{V}_{\bar{l}} - \mathbf{V}_l)\right]$$

$*$ **Quantity $\mathtt{X}_{\rm D}$ at full levels:** It is better to use the following alternate expression to avoid $\mathbf{S}D - \mathbf{S}D$ kind cancellation; this is more accurate:

$$\mathtt{X}_{\rm D} = (1-\kappa)\left(\vec{V}\frac{\nabla\Pi}{\Pi} - \frac{1}{\Pi}\int_{top}^{\eta}[\vec{V}\nabla\Pi_{\rm s}]\frac{\partial B}{\partial\eta}d\eta'\right) \tag{334}$$

which also writes:

$$\mathtt{X}_{\rm D} = (1-\kappa)\left(\vec{V}\frac{\nabla\Pi}{\Pi} - \mathbf{S}([\partial B/\partial\Pi][\vec{V}\nabla\Pi_{\rm s}])\right)$$

Discretisation at full levels writes:

$$[\mathtt{X}_{\rm D}]_l = (1-\kappa_l)\left([\vec{V}]_l[\nabla\Pi/\Pi]_l - [\mathbf{S}([\partial B/\partial\Pi][\vec{V}\nabla\Pi_{\rm s}])]_l\right)$$

See appendix 1, part b, for VFD and VFE discretisation of $[\mathbf{S}([\partial B/\partial\Pi][\vec{V}\nabla\Pi_{\rm s}])]_l$.

$*$ **Quantity D_3:** Discretisation at full levels writes:

$$[D_3]_l = D_l + d_l + [\mathtt{X}_{\rm S}]_l = D_l + d_{4l} - [\mathtt{X}_{\rm D}]_l$$

But D_3 does not appear explicitly in the code.

$*$ **Retrieving full levels d from full levels $\Delta[gw]$:** The discretisation writes:

$$d_l = -\frac{1}{R\tilde{T}_l\delta_l}[\Delta[gw]]_l$$

$*$ **Retrieving $[gw]$ from full levels d:** This discretisation looks like the integration of the geopotential equation. We can start from the NHEE equations, replacing $R_{\rm a}T$ by $R\tilde{T}$ and Π/p by 1; the surface condition (equation (270)) is still valid.

- Finite difference discretisation (**LVFE_GW**=F):

$$[\Delta[gw]]_l = d_l R_l \tilde{T}_l \delta_l$$

$$gw_{\tilde{l}} = gw_{\rm surf} + \sum_{k=L}^{k=l+1}(d_k R_k \tilde{T}_k \delta_k)$$

$$gw_l = gw_{\tilde{l}} + d_l R_l \tilde{T}_l \alpha_l$$

- Finite element discretisation (**LVFE_GW**=T):

$$gw_l = gw_{\rm surf} + [\mathcal{R}_{\rm inte}]_{(surf,l)}\left\langle -\frac{dR\tilde{T}\delta}{\Delta\eta}\right\rangle$$

$*$ **Retrieving $\nabla[gw]$ from full levels d and ∇d:** We can start from the NHEE equations, replacing $R_{\rm a}T$ by $R\tilde{T}$ and Π/p by 1; the surface condition (equation (275) is still valid.

- Finite difference discretisation (**LVFE_GW**=F):

$$[\Delta\nabla[gw]]_l = \nabla[R_l\tilde{T}_l]d_l\delta_l + R_l\tilde{T}_l\nabla d_l\delta_l + R_l\tilde{T}_l d_l\nabla\delta_l$$

$$\nabla[gw_{\tilde{l}}] = \nabla[gw_{\rm surf}] + \sum_{k=L}^{k=l+1}[\Delta\nabla[gw]]_k$$

$$\nabla[gw_l] = \nabla[gw_{\tilde{l}}] + \nabla[R_l\tilde{T}_l]d_l\alpha_l + R_l\tilde{T}_l\nabla d_l\alpha_l + R_l\tilde{T}_l d_l\nabla\alpha_l$$

- Finite element discretisation (**LVFE_GW**=T):

$$\nabla[gw_l] = \nabla[gw_{\rm surf}] + [\mathcal{R}_{\rm inte}]_{(surf,l)}\left\langle -\frac{\nabla[dR\tilde{T}\delta]}{\Delta\eta}\right\rangle$$

with:

$$\left[\nabla\left(dR\tilde{T}\delta\right)\right]_l = \nabla[R_l\tilde{T}_l]d_l\delta_l + R_l\tilde{T}_l\nabla d_l\delta_l + R_l\tilde{T}_l d_l\nabla\delta_l$$

14.2 Momentum equation.

$*$ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = \mathbf{V} \tag{335}$$

$$\mathcal{A} = [-2(\mathbf{\Omega} \wedge \mathbf{V})] - \exp(\kappa\hat{Q}) \left(1 + \Pi \frac{\partial \hat{Q}}{\partial \Pi}\right) \nabla\Phi - R\tilde{T}\exp(\kappa\hat{Q}) \left(\frac{\nabla\Pi}{\Pi} + \nabla\hat{Q}\right) \tag{336}$$

$$\mathcal{L} = -\nabla[\gamma\tilde{T} + R_{\mathrm{d}}T^* \log(\Pi_{\mathrm{s}}) + R_{\mathrm{d}}T^*\hat{Q}] + \beta_{\mathrm{Co}}[-2(\mathbf{\Omega} \wedge \mathbf{V})] \tag{337}$$

$$F = \mathbf{F_V} \tag{338}$$

Top and bottom values are defined as follows:

$$\mathbf{V}_{\eta=0} = \mathbf{V}_{l=1} \tag{339}$$

If $\delta m = 0$:

$$\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L} \tag{340}$$

If $\delta m = 1$:

$$\mathbf{V}_{\eta=1} = 0 \tag{341}$$

Note that this bottom condition is not fully consistent with the one done in the remaining part of the NH code ($\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L}$).

$*$ Remarks. the remarks done for the hydrostatic model remain valid here. We must add an additional remark: in some cases we need this equation also at the surface, with some assumptions (explicit treatment of the Coriolis term, $\mathbf{V}_{\eta=1} = \mathbf{V}_{l=L}$).

$*$ Discretisation of $-2\mathbf{\Omega} \wedge \mathbf{V}$ at full levels: It is the same as for the hydrostatic model.

$*$ Discretisation of the pressure gradient term at full levels: It is significantly different from the NHEE one. The horizontal pressure gradient term is written as follows, to isolate the pseudo-hydrostatic part $-\nabla\Phi - R\tilde{T}\frac{\nabla\Pi}{\Pi}$:

$$A1[-\nabla\Phi - R\tilde{T}\frac{\nabla\Pi}{\Pi} + A2]$$

The choice currently retained is to set $A1 = 1$ and to write $A2$ (which contain anhydrostatic effects) as the sum of five contributions denoted $A21$ to $A25$:

$$A21 = -R\tilde{T}(\exp(\kappa\hat{Q}) - 1)\frac{\nabla\Pi}{\Pi}$$

$$A22 = -R\tilde{T}\exp(\kappa\hat{Q})\nabla\hat{Q}$$

$$A23 = -\frac{1}{\kappa}\frac{\partial \Pi(\exp(\kappa\hat{Q}) - 1)}{\partial \Pi}\nabla\Phi$$

$$A24 = \left(\frac{1}{\kappa} - 1\right)(\exp(\kappa\hat{Q}) - 1)\nabla\Phi$$

$$A25 = \frac{1}{\kappa}\hat{Q}\exp(\kappa\hat{Q})\left(\Pi\frac{\partial\kappa}{\partial\Pi}\right)\nabla\Phi$$

Discretisation of $-\nabla\Phi - R\tilde{T}\frac{\nabla\Pi}{\Pi}$ is as close as possible to the hydrostatic model one:

$$\left[\nabla\Phi + R\tilde{T}\frac{\nabla\Pi}{\Pi}\right]_l = [\nabla\Phi]_{\bar{l}} + \left[\nabla(R\tilde{T})\right]_l \alpha_l + \left[R_l\tilde{T}_l\right][\nabla(\alpha + \log\Pi)]_l$$

Discretisation of $A21$ and $A22$ is valid for VFD and VFE and follows:

$$[A21]_l = -[R\tilde{T}]_l([\exp(\kappa\hat{Q})]_l - 1)\left[\frac{\nabla\Pi}{\Pi}\right]_l$$

$$[A22]_l = -[R\tilde{T}]_l[\exp(\kappa\hat{Q})]_l[\nabla\hat{Q}]_l$$

Discretisation of $A23$ to $A25$ is given at least for VFD discretisation and follows:

$$[A23]_l = -\frac{1}{\kappa_l}\frac{\Pi_{\bar{l}}([\exp(\kappa\hat{Q})]_{\bar{l}} - 1) - \Pi_{\overline{l-1}}([\exp(\kappa\hat{Q})]_{\overline{l-1}} - 1)}{[\Delta\Pi]_l}[\nabla\Phi]_l$$

$$[A24]_l = \left(\frac{1}{\kappa_l} - 1\right)([\exp(\kappa\hat{Q})]_{\bar{l}} - 1)[\nabla\Phi]_l$$

$$[A25]_l = \frac{1}{\kappa_l}\hat{Q}_l[\exp(\kappa\hat{Q})]_l\frac{\kappa_{\bar{l}} - \kappa_{\overline{l-1}}}{\delta_l}[\nabla\Phi]_l$$

Term $[A25]_l$ is provisionally neglected.

$*$ **Discretisation of the pressure gradient term at the surface:** It is computed in the RHS of equation giving $\frac{d\mathbf{V}_{\rm surf}}{dt}$, in the case **LVERTFE**=.F. . Currently the pressure gradient term at the surface is set to the pressure gradient term at the full level $l = L$.

$*$ **Discretisation of the grid-point Rayleigh friction:** It is the same as for the hydrostatic model.

14.3 Thermodynamic equation.

$*$ Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values.

$$X = T \tag{342}$$

$$\mathcal{A} = \frac{R\tilde{T}}{c_{\rm p}}\frac{\omega}{\Pi} \tag{343}$$

$$\mathcal{L} = -\tau(\overline{M}^2 D^{'}) \tag{344}$$

$$F = \exp(-\kappa\hat{Q}))F_{\rm T} \tag{345}$$

Top:

$$T_{\eta=0} = T_{l=1} \tag{346}$$

Bottom if $\delta m = 0$:

$$T_{\eta=1} = T_{l=L} \tag{347}$$

Bottom if $\delta m = 1$ (output of physics):

$$T_{\eta=1} = T_{\rm s} \tag{348}$$

$*$ Discretisation of the conversion term.

$$\left[\frac{R\tilde{T}}{c_{\rm p}}\frac{\omega}{\Pi}\right]_l = \frac{R_l\tilde{T}_l}{[c_{\rm p}]_l}\left[\frac{\omega}{\Pi}\right]_l = \kappa_l\tilde{T}_l\left[\frac{\omega}{\Pi}\right]_l \tag{349}$$

It is very similar to the hydrostatic conversion term.

14.4 Continuity equation.

Discretisation is identical to the hydrostatic model one.

14.5 Moisture and other GFL equations.

Formulation is identical to the hydrostatic model.

14.6 Relationship between geopotential height gz and pressure depth.

We can start from the NHEE equations, replacing RT by $R\tilde{T}$ and Π/p by 1;

$*$ **Geopotential height at half levels (case LVERTFE=.F. only):** Discretisation of equation (123) at half levels yields:

$$gz_{\bar{l}} = gz_{\rm s} + \sum_{k=L}^{k=l+1}\left[R_k\tilde{T}_k\right]\delta_k \tag{350}$$

$*$ Geopotential height at full levels:

- Case **LVERTFE**=.F.:

$$gz_l = gz_{\bar{l}} + \left[R_l\tilde{T}_l\right]\alpha_l \tag{351}$$

- Case **LVERTFE**=.T.:

$$gz_l = gz_{\rm s} + [\mathcal{R}_{\rm inte}]_{(surf,l)}\left\langle -\frac{R\tilde{T}\delta}{\Delta\eta}\right\rangle \tag{352}$$

$*$ **Horizontal gradient of the geopotential height at half levels (case LVERTFE=.F. only):** One applies the operator ∇ to equation (350). That yields:

$$g\left[\nabla z\right]_{\bar{l}} = g\nabla z_{\rm s} + \sum_{k=L}^{k=l+1}\left[\nabla(R\tilde{T})\right]_k\delta_k + \sum_{k=L}^{k=l+1}\left[R_k\tilde{T}_k\right]\left[\nabla\delta\right]_k \tag{353}$$

$*$ **Horizontal gradient of the geopotential height at full levels:** One applies the operator ∇ to equation (351). That yields:

- Case **LVERTFE**=.F.:

$$g\left[\nabla z\right]_l = g\left[\nabla z\right]_{\bar{l}} + \left[\nabla(R\tilde{T})\right]_l \alpha_l + \left[R_l\tilde{T}_l\right]\left[\nabla\alpha\right]_l \tag{354}$$

- Case **LVERTFE**=.T.:

$$g\left[\nabla z\right]_l = g\nabla z_{\rm s} + [\mathcal{R}_{\rm inte}]_{(surf,l)}\left\langle -\frac{\nabla(R\tilde{T})\delta + R\tilde{T}\nabla\delta}{\Delta\eta}\right\rangle \tag{355}$$

14.7 Diagnostic expression of some vertical velocities.

Formulation is identical to the hydrostatic model one.

14.8 Vertical divergence d and gw equations.

$*$ **Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values, for the vertical divergence.**

$$X = d \tag{356}$$

$$\mathcal{A} = -d(d+\mathtt{X}_{\rm S}) - \frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\rm ad}}{\partial\eta} + \frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}(\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right) \tag{357}$$

$$\mathcal{L} = -\frac{g^2}{R_{\rm d}T^*_{\rm a}}(\mathbf{L}^*_\kappa\hat{Q}) \tag{358}$$

$$F = -\frac{d}{\left[\frac{\partial\Pi}{\partial\eta}\right]}F'_{\rm m} - \left[\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial F_{\rm w}}{\partial\eta}\right] \tag{359}$$

Top:

$$d_{\eta=0} = d_{l=1} \tag{360}$$

Bottom if $\delta m = 0$:

$$d_{\eta=1} = d_{l=L} \tag{361}$$

Bottom if $\delta m = 1$: currently cf. $\delta m = 0$. Note that this bottom condition is not fully consistent with the one done on the horizontal wind and the vertical velocity w.

$*$ **Definition of X, $\mathcal{A}$, $\mathcal{L}$ and F, top and bottom values, for the vertical velocity w.**

$$X = w \tag{362}$$

$$\mathcal{A} = \frac{g}{\kappa}\left[\frac{\partial\Pi(\exp(\kappa\hat{Q})-1)}{\partial\Pi} + (\kappa-1)(\exp(\kappa\hat{Q})-1)\right] \tag{363}$$

$$F = F_{\rm w} \tag{364}$$

Top:

- **LVFE_GW**=.F.: $w_{\eta=0}$ is computed by the general formula giving w at half levels.
- **LVFE_GW**=.T.: top value of w is useless.

Bottom if $\delta m = 0$:

$$w_{\eta=1} = \mathbf{V}_{\rm surf}\nabla z_{\rm surf} \tag{365}$$

Bottom if $\delta m = 1$: currently the same as for $\delta m = 0$. Remark: we also need the equation of the surface vertical velocity.

$*$ **Discretisation of the RHS of the vertical divergence d equation (LGWADV=F):**

- Term $-d(d+\mathtt{X}_{\rm S})$ at full levels:

$$[-d(d+\mathtt{X}_{\rm S})]_l = -d_l(d_l + [\mathtt{X}_{\rm S}]_l)$$

  This discretisation can be applied for both VFD and VFE.

- Term $\mathtt{Z} = -\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}(\nabla w)\left(\frac{\partial\mathbf{V}}{\partial\eta}\right)$: its discretisation looks like the one of $\mathtt{X}_{\rm S}$.

– **LVFE_Z_TERM**=.F.: This term is rewritten:

$$\mathtt{Z} = -\frac{g}{R\tilde{T}}(\nabla w)\left(\frac{\partial \mathbf{V}}{\partial \log \Pi}\right)$$

Its discretisation at full levels is:

$$\mathtt{Z}_l = -\frac{g}{R_l\tilde{T}_l\delta_l}\left[\nabla[w]_{\bar{l}-1}(\mathbf{V}_l - \mathbf{V}_{\bar{l}-1}) + \nabla[w]_{\bar{l}}(\mathbf{V}_{\bar{l}} - \mathbf{V}_l)\right] \qquad (366)$$

This discretisation uses $\nabla[w]$ at half levels and $\mathbf{V}$ at both full and half levels. The discretisation of $\nabla[gw]$ at half levels is given by equation (277).

– **LVFE_Z_TERM**=.T.: this case has not been yet studied.

- Term $-\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta}$: its expression can be rewritten, using operators $\mathbf{L}_\kappa$ and $\bar{\partial}$:

$$-\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta} = -\frac{g^2}{\kappa R\tilde{T}}[\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1)] - \frac{g^2}{R\tilde{T}}\left[\frac{\partial(1/\kappa)}{\partial\log\Pi}\right][(\bar{\partial}+\kappa)(\exp(\kappa\hat{Q})-1)]$$

First we consider that term containing vertical variations of κ (only due to moisture and hydrometeors) is very small and negligible; it would be (at least provisionally) omitted in the code. Formula retained is:

$$-\frac{g\Pi}{R\tilde{T}\frac{\partial\Pi}{\partial\eta}}\frac{\partial\left[\frac{dw}{dt}\right]_{\mathrm{ad}}}{\partial\eta} \simeq -\frac{g^2}{\kappa R\tilde{T}}[\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1)]$$

– **LVFE_LAPL**=.F.: $\mathbf{L}_\kappa$ is a tridiagonal operator giving a full level quantity from full level values of $(\exp(\kappa\hat{Q})-1)$. This is the non-linear counterpart of $\mathbf{L}^*_\kappa$, discretisation writes, for $Z = \exp(\kappa\hat{Q})-1$:

$$(\mathbf{L}_\kappa Z)_l = [1+(\kappa_l-1)(\alpha_l-\delta_l)]\mathbf{A}_l(Z_{l-1}-Z_l) + [1+(\kappa_l-1)\alpha_l]\mathbf{C}_l(Z_{l+1}-Z_l)$$

with

$$\mathbf{A}_l = \frac{\Pi_{l-1}}{\delta_l(\Pi_l - \Pi_{l-1})}$$

$$\mathbf{C}_l = \frac{\Pi_{l+1}}{\delta_l(\Pi_{l+1} - \Pi_l)}$$

There are particular discretisations at the top and bottom. For example we replace Z_{L+1} and Z_0 by 0. To take account of bottom condition (w_{surf} different from 0) we add $(\kappa_l/g)\frac{dw_{surf}/dt}{\delta_L}$ to $(\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1))_L$.

Full level quantity $\frac{g^2}{\kappa R\tilde{T}}$ in factor is discretised:

$$\left[\frac{g^2}{\kappa R\tilde{T}}\right]_l = \frac{g^2}{\kappa_l R_l\tilde{T}_l}$$

– **LVFE_LAPL**=.T.: this case has not been yet studied.

$*$ **Discretisation of the RHS of $[gw_{\mathrm{surf}}]$ equation:** calculation is identical to the NHEE model, and is done for both cases **LGWADV**=T and **LGWADV**=F.

$*$ **Discretisation of term $\frac{g}{\kappa}\left[\frac{\partial\Pi(\exp(\kappa\hat{Q})-1)}{\partial\Pi} + (\kappa-1)(\exp(\kappa\hat{Q})-1)\right]$ (LGWADV=T):**
Neglecting term containing $\frac{\partial(1/\kappa)}{\partial\log\Pi}$, formula (363) can be rewritten:

$$\mathcal{A} \simeq \frac{g}{\kappa}\bar{\partial}^{-1}[\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1)]$$

- **LVFE_GW**=.F.: discretisation uses $(\exp(\kappa\hat{Q})-1)$ at full levels and gives $\mathcal{A}$ at half levels. We first compute $\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1)$ at full levels. Then we do a vertical integration, starting from the bottom:

$$\left[\frac{d(gw)}{dt}\right]_{\bar{l}=L-1} = \left[\frac{d(gw_{surf})}{dt}\right] - \delta_{l=L}\frac{g^2}{\kappa_{l=L}}[\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1)]_{l=L}$$

$$\left[\frac{d(gw)}{dt}\right]_{\bar{l}-1} = \left[\frac{d(gw)}{dt}\right]_{\bar{l}} - \delta_l\frac{g^2}{\kappa_l}[\mathbf{L}_\kappa(\exp(\kappa\hat{Q})-1)]_l$$

- **LVFE_GW**=.T.: this case has not been yet studied.

$*$ **Use of d_4 as prognostic variable:** In equation (131), the calculation of $\frac{\partial \mathtt{X}}{\partial t}$ is not done by an Eulerian temporal advection but simply by a diagnostic evaluation (cf. what is done in the NHEE model).

15 Discretisation of some other diagnosed quantities.

15.1 c_{p}, R and κ.

For c_{p} (air calorific capacity at constant pressure) at full levels:

$$[c_{\mathrm{p}}]_l = c_{\mathrm{p_d}}\left(1 - [q]_l - [q_{\mathrm{l}}]_l - [q_{\mathrm{i}}]_l\right) + c_{\mathrm{p_v}}[q]_l + c_{\mathrm{p_l}}[q_{\mathrm{l}}]_l + c_{\mathrm{p_i}}[q_{\mathrm{i}}]_l \tag{367}$$

where $c_{\mathrm{p_d}}$ is the dry air calorific capacity at constant pressure, $c_{\mathrm{p_v}}$ is the water vapour calorific capacity at constant pressure, $c_{\mathrm{p_l}}$ is the liquid water calorific capacity, and $c_{\mathrm{p_i}}$ is the ice calorific capacity.
For R (air constant) at full levels:

$$[R]_l = R_{\mathrm{d}}\left(1 - [q]_l - [q_{\mathrm{l}}]_l - [q_{\mathrm{i}}]_l\right) + R_{\mathrm{v}}[q]_l \tag{368}$$

where R_{d} is the dry air constant and R_{v} is the water vapour air constant.
For κ at full levels:

$$[\kappa]_l = \frac{[R]_l}{[c_{\mathrm{p}}]_l} \tag{369}$$

The ratio $c_{\mathrm{v}}/c_{\mathrm{p}}$ at full levels is given by equation:

$$\left[\frac{c_{\mathrm{v}}}{c_{\mathrm{p}}}\right]_l = 1 - [\kappa]_l \tag{370}$$

15.2 $\nabla(RT)$.

Its discretisation at full levels writes:

$$[\nabla(RT)]_l = (R_{\mathrm{v}} - R_{\mathrm{d}})[T]_l[\nabla q]_l + [R]_l[\nabla T]_l \tag{371}$$

15.3 Potential temperature PT and its horizontal gradient.

Discretisation of PT at full levels writes:

$$[PT]_l = [T]_l\left[\frac{[\Pi]_l}{\Pi_{1000}}\right]^{-[\kappa]_l} \tag{372}$$

where $\Pi_{1000} = 100000$ Pa
Discretisation of its horizontal gradient at full levels writes:

$$[\nabla(PT)]_l = [PT]_l\left(\frac{[\nabla T]_l}{[T]_l} - [\kappa]_l\left[\frac{\nabla\Pi}{\Pi}\right]_l\right) \tag{373}$$

The expression of $\frac{\nabla\Pi}{\Pi}$ at full levels is given in part (8.4).
Remark: in the cycle 46t1 the code does not match exactly formula (373); this has to be changed in the future.

15.4 Virtual potential temperature PTV.

Its discretisation at full levels writes:

$$[PTV]_l = [TV]_l\left[\frac{[\Pi]_l}{\Pi_{1000}}\right]^{-\kappa_{\mathrm{d}}} = \frac{R_l}{R_{\mathrm{d}}}T_l\left[\frac{[\Pi]_l}{\Pi_{1000}}\right]^{-\kappa_{\mathrm{d}}} \tag{374}$$

where $\Pi_{1000} = 100000$ Pa; $\kappa_{\mathrm{d}} = R_{\mathrm{d}}/c_{\mathrm{p_d}}$.

15.5 Equivalent potential temperature PTE.

Its discretisation at full levels writes:

$$[PTE]_l = [PT]_l \ \exp\left[\frac{L_l[q_{\mathrm{sat}}]_l}{[c_{\mathrm{p_{sat}}}]_l T_l}\right] \tag{375}$$

where the discretisation of the potential temperature PT has been provided by formula (372). Discretisation of $[c_{\mathrm{p_{sat}}}]$ at full levels writes:

$$[c_{\mathrm{p_{sat}}}]_l = c_{\mathrm{p_d}} + (c_{\mathrm{p_v}} - c_{\mathrm{p_d}})[q_{\mathrm{sat}}]_l$$

15.6 Absolute vorticity $\zeta_{\rm abs}$.

Its discretisation at full levels writes:

$$[\zeta_{\rm abs}]_l = [\zeta]_l + f \tag{376}$$

15.7 Potential vorticity PV.

$*$ Vertical finite difference discretisation.

Discretisation of PV at full levels is:

$$[PV]_l = g\left[\frac{\partial V}{\partial \Pi}\right]_l \left[\frac{M}{a\cos\Theta}\frac{\partial PT}{\partial \Lambda}\right]_l - g\left[\frac{\partial U}{\partial \Pi}\right]_l \left[\frac{M}{a}\frac{\partial PT}{\partial \Theta}\right]_l - g\left[\zeta_{\rm abs}\right]_l \left[\frac{\partial PT}{\partial \Pi}\right]_l \tag{377}$$

Discretisation of vertical derivatives relative to hydrostatic pressure: for a variable X:

$$\left[\frac{\partial X}{\partial \Pi}\right]_l = 0.5\left(\frac{X_{l+1}-X_{l-1}}{[\Delta\Pi]_l}\right) \tag{378}$$

$$\left[\frac{\partial X}{\partial \Pi}\right]_{l=1} = \left(\frac{X_{l=2}-X_{l=1}}{[\Delta\Pi]_{l=1}}\right) \tag{379}$$

$$\left[\frac{\partial X}{\partial \Pi}\right]_{l=L} = \left(\frac{X_{l=L}-X_{l=L-1}}{[\Delta\Pi]_{l=L}}\right) \tag{380}$$

This discretisation is applied to $X = U$, $X = V$ and $X = PT$.
Discretisation of the components of $\nabla(PT)$: see equation (373).

$*$ Vertical finite element discretisation.

The same discretisations are currently used. VFE discretisation of vertical derivatives has not been implemented for this quantity which is currently used only in the post-processing.

15.8 Shearing deformation SHD and stretching deformation STD.

Spherical geometry:

$$[STD]_l = M\left[\frac{1}{a\cos\Theta}\frac{\partial U'}{\partial \Lambda}\right]_l - 0.5D_l \tag{381}$$

$$[SHD]_l = M\left[\frac{1}{a\cos\Theta}\frac{\partial V'}{\partial \Lambda}\right]_l - 0.5\zeta_l \tag{382}$$

Plane geometry:

$$[STD]_l = M^2[\partial'_{\rm x}U']_l - 0.5D_l \tag{383}$$

$$[SHD]_l = M^2[\partial'_{\rm x}V']_l - 0.5\zeta_l \tag{384}$$

15.9 Hydrostatic vertical divergence $d_{\rm hyd}$.

Discretisation of $d_{\rm hyd}$ at full levels is:

$$[d_{\rm hyd}]_l = -\frac{R_l}{R_{\rm a}l}\left[\frac{c_{\rm v}}{c_{\rm p}}\right]_l \left[\frac{\omega}{\Pi}\right]_l - \frac{R_l}{R_{\rm a}l}[\nabla\mathbf{V}]_l - \frac{1}{R_{\rm a}lT_l}\left[\nabla[gz]\,\Pi\frac{\partial\mathbf{V}}{\partial\Pi}\right]_l \tag{385}$$

with:

- if finite differences discretisation applied to vertical derivatives:

$$\left[\nabla[gz]\,\Pi\frac{\partial\mathbf{V}}{\partial\Pi}\right]_l = \frac{1}{\delta_l}\left[\left([\mathbf{V}]_{\bar{l}} - [\mathbf{V}]_l\right)[\nabla[gz]]_{\bar{l}} + \left([\mathbf{V}]_l - [\mathbf{V}]_{\overline{l-1}}\right)[\nabla[gz]]_{\overline{l-1}}\right] \tag{386}$$

- if VFE discretisation applied to vertical derivatives:

$$\left[\nabla[gz]\,\Pi\frac{\partial\mathbf{V}}{\partial\Pi}\right]_l = \frac{[\Delta\eta]_l}{\delta_l}[\mathcal{R}_{\rm deri}(\mathbf{V})]_l\,[\nabla[gz]]_l \tag{387}$$

One needs the discretisation of the horizontal wind at half levels: this discretisation has been given in part (13.1). Discretisation of each other RHS terms has been previously studied in this documentation.

15.10 Hydrostatic height coordinate vertical velocity w_{hyd}.

∗ Vertical finite difference discretisation.

Discretisation of w_{hyd} matches the following equation:

$$g\,[w_{\text{hyd}}]_{\bar{l}} = g\,[w_{\text{hyd}}]_{\text{surf}} + \sum_{k=l+1}^{k=L} R_{\text{a}k} T_k\,[d_{\text{hyd}}]_k\,\delta_k \tag{388}$$

Discretisation of w_{hyd} at full levels is given by the following equation:

$$g\,[w_{\text{hyd}}]_l = g\,[w_{\text{hyd}}]_{\bar{l}} + R_{\text{a}l} T_l\,[d_{\text{hyd}}]_l\,\alpha_l \tag{389}$$

At the surface:

$$g\,[w_{\text{hyd}}]_{\text{surf}} = [\mathbf{V}]_{l=L}\,\nabla\Phi_{\text{s}} \tag{390}$$

∗ Vertical finite element discretisation.

One directly computes w_{hyd} at full levels.
Discretisation of w_{hyd} at full levels is given by the following equation:

$$g\,[w_{\text{hyd}}]_l = g\,[w_{\text{hyd}}]_{\text{surf}} + [\mathcal{R}_{\text{inte}}]_{(surf,l)} \left\langle \frac{R_{\text{a}} T d_{\text{hyd}} \delta}{\Delta\eta} \right\rangle \tag{391}$$

At the surface, $g\,[w_{\text{hyd}}]_{\text{surf}}$ is still given by equation (390).

15.11 Moisture convergence $CVGQ$.

This quantity is evaluated at full levels. For **NCOMP_CVGQ**=0 or 1, calculation is done as follows:

$$[CVGQ]_l = -\,[\mathbf{V}\nabla q]_l - \left[\dot{\eta}\frac{\partial q}{\partial \eta}\right]_l \tag{392}$$

For **NCOMP_CVGQ**=2, calculation is done as follows:

$$[CVGQ]_l = \left[\frac{\partial q}{\partial t}\right]_l - \left[\frac{dq}{dt}\right]_l \tag{393}$$

NCOMP_CVGQ=2 can be used only in the semi-Lagrangian scheme.
For the way of discretizing $-\,[\mathbf{V}\nabla q]_l$ (which is an horizontal advection term) report to part 9.1. For the way of discretizing $-\left[\dot{\eta}\frac{\partial q}{\partial \eta}\right]_l$ (which is a vertical advection term) report to part 9.2.

15.12 Montgomery potential Φ_{mg} and some other energetic quantities.

All these quantities are discretised at full levels. Formulae for enthalpy and kinetic energy are guaranteed at least for the hydrostatic model.

- Montgomery potential: $[\Phi_{\text{mg}}]_l = c_{\text{p}_\text{d}} T_l + [gz]_l$.
- Dry static energy: $s_l = [c_{\text{p}}]_l\,T_l + [gz]_l$.
- Moist static energy: $[s_{\text{h}}]_l = [c_{\text{p}}]_l\,T_l + [gz]_l + L_l q_{\text{l}}$.
- Enthalpy: $h_l = [c_{\text{p}}]_l\,T_l + [gz]_l + 0.5 * (U_l^2 + V_l^2)$.
- Kinetic energy: $[KE]_l = 0.5 * (U_l^2 + V_l^2)$.

15.13 Angular momentum of components MMA, MMB and MMC.

All these quantities are discretised on layer using the full level values of $(U_{\text{G}}; V_{\text{G}})$.

15.14 Entropy S.

Hydrostatic model: the entropies are discretised at full levels using the layer values of q, q_{l}, q_{i}, T and Π; the layer values of Π_v and Π_d have to be taken consistently with the one of Π: $[\Pi_v]_l = [q]_l\,[\Pi]_l$ and $[\Pi_d]_l = (1 - [q]_l)\,[\Pi]_l$.

Non hydrostatic model: the hydrostatic pressure Π must be replaced by the total pressure p in the previous formulae.

16 Miscellaneous (linear terms, Asselin filter, LBC).

16.1 Treatment of the linear terms.

The grid-point calculations of the model provide the quantity $(X - \Delta t \beta \mathcal{L})^+$ for a subset of equations. Retrieving X^+ is done in the spectral computations after the direct spectral transforms and just before the horizontal diffusion. The algorithm of solving the semi-implicit scheme is described in a specific documentation (IDSI) about semi-implicit scheme.

16.2 The Asselin filter.

In a leap-frog scheme, a weak time filter is used after Asselin (1972). If X_{fil} denotes the filtered value of X:

$$X_{\text{fil}}^o = X^o + \epsilon_{\text{ass1}}(X_{\text{fil}}^- - X^o) + \epsilon_{\text{ass2}}(X^+ - X^o) \tag{394}$$

16.3 Lateral boundary coupling and upper boundary coupling.

A limited area model (LAM) needs to get information from the lateral boundaries. It can be done:

- By lateral boundary coupling, in grid-point space.
  The LAM domain is divided into 3 zones (C for inner zone, I for intermediate zone, E for extension zone). Davies (1976) relaxation is applied to fields:

  $$X_{\text{coupled}} = (1 - \alpha_{\text{rel}})X_{\text{coupled}} + \alpha_{\text{rel}} X_{\text{lbc}} \tag{395}$$

  α_{rel} is equal to 1 in the extension zone, 0 in the inner zone, and varies between 0 and 1 in the intermediate zone. α_{rel} is vertically constant and can depend on the coupled variable.

- By upper boundary coupling.
  The same kind of formula:

  $$X_{\text{coupled}} = (1 - \alpha_{\text{nud}})X_{\text{coupled}} + \alpha_{\text{nud}} X_{\text{lbc}} \tag{396}$$

  is still applied, but now α_{nud} is vertically-dependent (generally non-zero near the model top only) and may depend on the wavenumber too.

  α_{nud} writes:

  $$\alpha_{\text{nud}} = \frac{\alpha_{\text{v}} \alpha_{\text{h}} K_{\text{nud}}}{1 + \alpha_{\text{v}} \alpha_{\text{h}} K_{\text{nud}}}$$

  α_{v} is a vertically-dependent coefficient: 1 for l below a threshold l_{nen1}; 0 for l above a threshold l_{nen2}; continuous and monotonic function between 0 and 1 otherwise. α_{v} is identical for all variables.

  α_{h} is a wavenumber-dependent coefficient: 1 for low wavenumbers (below a threshold n_{nek0}); 0 for high wavenumbers (above a threshold n_{nek1}); continuous and monotonic function between 0 and 1 otherwise. α_{h} is identical for all variables.

  K_{nud} is a tunable coefficient which may be different for each variable.

  Such scheme is activated in practical to force the LAM large-scale patterns by the large scale model in the high atmosphere.

  There are two ways to activate such a scheme:

  - Spectral nudging, in spectral space: both vertical and wavenumber dependencies are possible. Could be applied in spectral space to spectral fields only.
  - Grid-point nudging, in grid-point space: only vertical dependency is possible. Could be applied in grid-point space to grid-point fields.

X_{lbc} is needed at each timestep, but coupling files are not read at each timestep (frequency of reading such files is generally between 1h and 3h). A temporal linear, quadratic or cubic interpolation is applied to retrieve X_{lbc} at the current instant from values read on files.

A call-tree for lateral boundary coupling and upper boundary coupling is given in part 19.6 .

17 The flux form of an equation and its discretisation.

17.1 The flux form of the equation of a variable X.

17.1.1 Introduction.

The aim is to evaluate the Eulerian tendency of an additive quantity $Q_{\rm X}$ corresponding to some content on an additive quantity in a particle. The additive quantity $Q_{\rm X}$ is linked to an intensive quantity X. For example:

- For $X = 1$ the additive quantity Q_1 is the total mass in a particle.
- For $X = q$ (specific humidity), the additive quantity $Q_{\rm q}$ is the water vapour mass in the same particle.
- For $X = \mathbf{V}$, the additive quantity $Q_{\mathbf{V}}$ is the momentum in the same particle.

X and $Q_{\rm X}$ are linked by the following relationship:

$$Q_{\rm X} = Q_1 X$$

Given a particle of dimensions δx, δy, δz, of density ρ: the total mass contained in this particle is

$$Q_1 = \rho\delta x\delta y\delta z$$

Using the hydrostatic relationship $\delta\Pi = -\rho g\delta z$, expression of Q_1 can be rewritten as follows:

$$Q_1 = -\frac{1}{g}\delta x\delta y\delta\Pi$$

Product $\delta x\delta y$ involves a surface δS_a which does not depend on the vertical nor on the hydrostatic pressure. Q_1 writes:

$$Q_1 = -\frac{1}{g}\delta S_a\delta\Pi = -\frac{1}{g}\frac{\partial\Pi}{\partial\eta}\delta S_a\delta\eta$$

All what does not depend on the time t has to be put in factor (for example g, δS_a and $\delta\eta$); the Eulerian tendency of $\frac{\partial\Pi}{\partial\eta}X$ has to be taken to write the flux form of the evolution equation of X.

17.1.2 The flux form of the equation of X.

One does a combination between the Eulerian form of the evolution equation of X and the Eulerian form of continuity equation to obtain it.
Equation for X writes:

$$\frac{\partial X}{\partial t} = -\mathbf{V}\nabla X - \dot\eta\frac{\partial X}{\partial\eta} + \left[\frac{dX}{dt}\right]_{\rm ad} + F_{\rm X} \tag{397}$$

where $\left[\frac{dX}{dt}\right]_{\rm ad}$ is the adiabatic Lagrangian tendency of X. One here omits the horizontal diffusion terms and semi-implicit correction terms.

Continuity equation can be rewritten under its flux form:

$$\frac{\partial\left[\frac{\partial\Pi}{\partial\eta}\right]}{\partial t} = -\nabla\left[\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right] - \frac{\partial\left[\dot\eta\frac{\partial\Pi}{\partial\eta}\right]}{\partial\eta} - g\frac{\partial F_{\rm P}}{\partial\eta} \tag{398}$$

Combining equations (397) and (398), and after calculations not detailed, one obtains the flux form of the Eulerian equation of X:

$$\frac{\partial\left[\frac{\partial\Pi}{\partial\eta}X\right]}{\partial t} = -\nabla\left[\mathbf{V}\frac{\partial\Pi}{\partial\eta}X\right] - \frac{\partial\left[X\dot\eta\frac{\partial\Pi}{\partial\eta}\right]}{\partial\eta} + \left[\frac{\partial\Pi}{\partial\eta}\right]\left[\frac{dX}{dt}\right]_{\rm ad} + \left[\frac{\partial\Pi}{\partial\eta}\right]F_{\rm X} - gX\frac{\partial F_{\rm P}}{\partial\eta} \tag{399}$$

17.2 Use of the discretisation of the flux form of equations.

The flux form of equations is used in some diagnostics, like the DDH package (horizontal diagnostics in boxes). There is a specific documentation for the DDH package (Piriou, 2006), so no additional detail will be given here. Only the list of variables diagnosed (to which is applied the flux form of equations) is given below:

- $X = 1$ (continuity equation).
- $X = q$ (moisture equation).
- $X = \mathbf{V}$ (horizontal wind).

- $X = 0.5(U^2 + V^2) = 0.5\mathbf{V}.\mathbf{V}$ (kinetic energy).
- $X = c_\mathrm{p}T$.
- $X = c_\mathrm{p}T + gz + 0.5(U^2 + V^2)$ (total energy).
- $X = q_\mathrm{l}$ (liquid water).
- $X = q_\mathrm{i}$ (ice).
- $X = \mathbf{M} = \mathbf{r} \wedge (\mathbf{\Omega} \wedge \mathbf{r} + \mathbf{V})$ (kinetic momentum).
- $X = S$ (entropy).

For some of these variables, $\left[\frac{dX}{dt}\right]_\mathrm{ad}$ is assumed to be zero ($X = 1$; q; $c_\mathrm{p}T + gz + 0.5(U^2 + V^2)$; q_l; q_i; S); for some other ones which are not prognostic variables, computation of $\left[\frac{dX}{dt}\right]_\mathrm{ad}$ need some additional calculations (for example for the kinetic energy).

17.3 The discretisation of the flux form of the equation of a variable X.

17.3.1 Left hand side.

In the LHS, $\left[\frac{\partial\Pi}{\partial\eta}X\right]$ is computed at full levels, and is discretised as follows:

$$\left[\frac{\partial\Pi}{\partial\eta}X\right]_l = \frac{[\Delta\Pi]_l}{[\Delta\eta]_l}X_l$$

17.3.2 Term containing the horizontal divergence in the RHS.

This term is rewritten under a sum of 3 terms, in order to show the quantities $\nabla\mathbf{V}$, $\nabla\Pi$ and ∇X, and writes:

$$-\left[\frac{\partial\Pi}{\partial\eta}X\right]\nabla\mathbf{V} - [\mathbf{V}X]\,\nabla\frac{\partial\Pi}{\partial\eta} - \left[\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right]\nabla X$$

Discretisation shows the quantities $[\Delta\Pi]_l$, $[\Delta\eta]_l$, $[\nabla\mathbf{V}]_l$, $[X]_l$, $[\Delta B]_l$. For the second term one uses the definition of the hybrid vertical coordinate, showing $\frac{\partial B}{\partial\eta}$ and Π_s. So the discretised form of the sum of 3 terms writes:

$$-\left[\frac{[\Delta\Pi]_l}{[\Delta\eta]_l}X_l\right][\nabla\mathbf{V}]_l - [\mathbf{V}_l X_l]\left[\frac{[\Delta B]_l}{[\Delta\eta]_l}\nabla\Pi_\mathrm{s}\right] - \left[\mathbf{V}_l\frac{[\Delta\Pi]_l}{[\Delta\eta]_l}\right]_l[\nabla X]_l$$

Remark: term $-\nabla\left[\mathbf{V}\frac{\partial\Pi}{\partial\eta}X\right]$ can be rewritten as $-X\nabla\left[\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right] - \mathbf{V}\frac{\partial\Pi}{\partial\eta}\nabla X$ to show the quantity $\nabla\left[\mathbf{V}\frac{\partial\Pi}{\partial\eta}\right]$ easily available in the model.

17.3.3 Term containing the vertical divergence in the RHS.

$*$ Vertical finite difference discretisation: Its discretisation writes:

$$-\frac{\left[X\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\bar{l}} - \left[X\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\bar{l}-1}}{[\Delta\eta]_l}$$

- Vertical integration of continuity provides $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ at half levels, so this is $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ at half levels which has to be used in the discretisation.
- X is known at full levels; its discretisation at half levels is given by formula:

$$X_{\bar{l}} = 0.5\,(X_l + X_{l+1})$$

The former expression can be rewritten as a difference of two fluxes (more exactly $-\frac{g}{[\Delta\eta]_l}(\mathrm{flux}_{\bar{l}} - \mathrm{flux}_{\bar{l}-1})$) where:

$$\mathrm{flux}_{\bar{l}} = \frac{0.5(X_l + X_{l+1})}{g}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_{\bar{l}}$$

$$\mathrm{flux}_\mathrm{top} = \frac{X_\mathrm{top}}{g}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_\mathrm{top}$$

$$\mathrm{flux}_\mathrm{surf} = \frac{X_\mathrm{surf}}{g}\left[\dot{\eta}\frac{\partial\Pi}{\partial\eta}\right]_\mathrm{surf}$$

The DDH package (routine **CPDYDDH**) stores these half level fluxes.

$*$ **Vertical finite element discretisation:** What is currently coded is a mixed FD-VFE discretisation where the operator $\mathcal{R}_{\mathrm{deri}}$ is not used:

- Vertical integration of continuity provides $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ at full levels; its discretisation at half levels is given by formula:

$$\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\bar{l}} = 0.5\left(\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l} + \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l+1}\right)$$

- X is known at full levels; its discretisation at half levels is given by formula:

$$X_{\bar{l}} = 0.5\,(X_l + X_{l+1})$$

The discretisation currently proposed is:

- Layers 2 to $L-1$:

$$-\frac{[0.5\,(X_l + X_{l+1})]\left[0.5\left(\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l} + \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l+1}\right)\right] - [0.5\,(X_{l-1} + X_l)]\left[0.5\left(\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l-1} + \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l}\right)\right]}{[\Delta\eta]_l}$$

- Layer 1:

$$-\frac{[0.5\,(X_1 + X_2)]\left[0.5\left(\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{1} + \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{2}\right)\right] - X_{\mathrm{top}}\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\mathrm{top}}}{[\Delta\eta]_1}$$

- Layer L:

$$-\frac{X_{\mathrm{surf}}\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\mathrm{surf}} - [0.5\,(X_{L-1} + X_L)]\left[0.5\left(\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{L-1} + \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{L}\right)\right]}{[\Delta\eta]_L}$$

This discretisation does not match completely the way of discretising a vertical derivative (which uses the operator $\mathcal{R}_{\mathrm{deri}}$), but this is the discretisation which leads to the minimum amount of modifications in the DDH package (routine **CPDYDDH**) compared to the case **LVERTFE**=.F., since the fluxes remain defined and stored at half levels. The former expressions can be rewritten as a difference of two fluxes (more exactly $-\frac{g}{[\Delta\eta]_l}(\mathrm{flux}_{\bar{l}} - \mathrm{flux}_{\overline{l-1}})$) where:

$$\mathrm{flux}_{\bar{l}} = \frac{0.5(X_l + X_{l+1})}{g}\left[0.5\left(\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l} + \left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{l+1}\right)\right]$$

$$\mathrm{flux}_{\mathrm{top}} = \frac{X_{\mathrm{top}}}{g}\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\mathrm{top}}$$

$$\mathrm{flux}_{\mathrm{surf}} = \frac{X_{\mathrm{surf}}}{g}\left[\dot{\eta}\frac{\partial \Pi}{\partial \eta}\right]_{\mathrm{surf}}$$

The DDH package (routine **CPDYDDH**) stores these half level fluxes.

17.3.4 Other terms in the RHS.

- Discretisation of term containing $\left[\frac{dX}{dt}\right]_{\mathrm{ad}}$ writes:

$$\frac{[\Delta\Pi]_l}{[\Delta\eta]_l}\left[\left[\frac{dX}{dt}\right]_{\mathrm{ad}}\right]_l$$

- Discretisation of term containing F_{X} writes:

$$\frac{[\Delta\Pi]_l}{[\Delta\eta]_l}\,[F_{\mathrm{X}}]_l$$

- Discretisation of term containing $\frac{\partial F_{\mathrm{p}}}{\partial \eta}$ writes:

$$-gX_l\frac{[\Delta F_{\mathrm{p}}]_l}{[\Delta\eta]_l}$$

18 Organigramme.

18.1 Organigramme for the setup and control routines (direct code).

∗ **General architecture under CNT0:** The organigramme will not be detailed (that will be too long), but it is possible to give some basics of its organisation:

```
CNT0 ->
* IFS_INIT -> setup of ‘‘INIT" object (call tree not detailed)
* SU0YOMA ->
  - SUGEOMETRY (call tree not detailed)
  - setup after geometry (call tree not detailed)
* SU0YOMB -> (call tree not detailed)
* some other routines not detailed here
* CNT1 ->
  - SU1YOM -> (call tree not detailed)
  - CNT2 -> CNT3 ->
    * some routines not detailed here
    * CNT4 ->
      - some routines not detailed here
      - STEPO -> (see below for more details)
```

- **CNT0**, **CNT1**, **CNT2**, **CNT3**, **CNT4** control respectively the zero, first, second, third and fourth level of the set-up.
- **STEPO** controls one time-step of the model integration.

∗ General architecture under STEPO:

```
STEPO ->
* Management of read/write: IOPACK
* Inverse spectral transforms: TRANSINVH -> TRANSINV_MDL
* Grid point computations: SCAN2M
* Grid-point coupling in LAM models: ECOUPL1 and ECOUPL2 (see below)
* Direct spectral transforms: TRANSDIRH -> TRANSDIR_MDL
* Spectral computations: (E)SPCM (see below), SPC2M.
```

The sequences of call to **STEPO** are controlled by a variable (often called **CDCONF** or **CLCONF**) containing 9 letters or zeros [L1][L2][L3][L4][L5][L6][L7][L8][L9]

- L1 controls the file write/read.
- L2+L3 controls the inverse transforms.
- L4 controls the grid-point computations for dynamics and physics.
- L5 controls the grid-point computations for some diagnostics.
- L6 controls the grid-point computations for assimilation.
- L7 controls the coupling in LAM models.
- L8 controls the direct transforms.
- L9 controls the spectral computations.

For example a model integration time-step is defined by the sequence [L1]AAA00AAA.

∗ The different objects and their set-up:

Data are put into big structures, for example for the following items:

- Init object (variables are not gathered in a type).
- Geometry: type GEOMETRY, variable **YRGEOMETRY** in **CNT0**.
- Model object: type MODEL, variable **YRMODEL** in **CNT0**.
- Fields: type FIELDS, variable **YRFIELDS** in **CNT0**.
- Trajectory: type MTRAJ, variable **YRMTRAJ** in **CNT0**.

And also:

- VARBC: type CLASS_VARBC, variable **YVARBC** in **CNT0**.
- TOVS control variable: type TOVSCV, variable **YLTCV** in **CNT0**.
- TOVS control variable for background: type TOVSCV_BGC, variable **YLTCV_BGC** in **CNT0**.
- Jo-table: type JO_TABLE, variable **YLJOT** in **CNT0**.
- ODB: class DBASE, variable **YLODB** in **CNT0**.
- FULL-POS: type TFPOS, variable **YLFPOS** in **CNT0**.

The set-up code is done according to the following order:

- Set-up code starts to set-up the **INIT** object, under **IFS_INIT** and before calling **SUGEOMETRY**.
- Once the **INIT** object set-up is done (entering **SUGEOMETRY**), it is forbidden to modify the content of **INIT** object.
- Set-up code does the set-up of the **GEOMETRY** object (variable **YRGEOMETRY**), via the call to **SUGEOMETRY**.
- Once exited from **SUGEOMETRY**, it is forbidden to modify the content of **YRGEOMETRY**.
- **YRGEOMETRY** is a dummy argument of **SUGEOMETRY**; **YRGEOMETRY** or some of its components is passed as dummy argument (intent IN) to routines called after **SUGEOMETRY**.
- **YRGEOMETRY** never appears in the set-up of object **INIT**.
- Once exited from **SUGEOMETRY**, the set-up fills the content of some other objects, like **MODEL** (variable **YRMODEL**).
- Variables like **YRMODEL**, **YRFIELDS**, **YRMTRAJ** are always passed via a dummy argument to routines using them.
- Variables like **YRMODEL**, **YRFIELDS**, **YRMTRAJ** never appear in the set-up of objects **INIT** and **GEOMETRY**.

18.2 Organigramme under STEPO for the direct Eulerian 2D model.

```
STEPO -> SCAN2M -> GP_MODEL ->
  * CPG2 -> GPTF2
  * CPG2LAG -> GPTF1
```

- **CPG2**: unlagged grid-point computations.
- **CPG2LAG**: lagged grid-point computations.
- **GPTF1**: first part of the Asselin temporal filter.
- **GPTF2**: second part of the Asselin temporal filter.

Communications between unlagged grid-point computations and lagged grid-point computations need a buffer using pointers of module **PTRSLB2**.

∗ Basic description of the data flow under routine STEPO: No detail will be given, one can read the part 19 with the following changes:

- There is no GFL field.
- The distinction between GMV and GMVS does not exist any longer, all fields can be considered as GMV or GMVS ones. The current way of coding considers the wind components as GMV fields, and the equivalent height as a GMVS field, but this way of coding is not satisfactory and has to be changed in the future.
- In the GMV fields, there are no thermodynamic variables.
- The model is purely adiabatic, so there is no surface field for physics.
- The structure of the code has not been completely updated relatively to the equivalent one of the 3D model. There are only two blocks in the grid-point calculations: a non-lagged block (**CPG2**) which is the equivalent of **CPG** in the 3D model; a lagged block (**CPG2LAG**) which is the equivalent of **CALL_SL+CPGLAG** in the 3D model.

18.3 Organigramme under STEPO for the direct Eulerian 3D model.

∗ Organigramme:

```
STEPO -> SCAN2M -> GP_MODEL_HEAP or GP_MODEL_STACK -> GP_MODEL ->
  * CPG_DRV -> CPG ->
    - CPG_GP ->
      * GPTF2
      * GPMPFC
      * GP_SPV
      * surface_fields_mix.F90/GPPOPER
      * CPG_GP_HYD -> some GP.. routines computing intermediate grid-point quantities.
      * CPG_GP_NHEE -> some GP.. and GNH.. routines computing intermediate grid-point quantities.
      * CPG_GP_NHQE -> some GP.. and GNH.. routines computing intermediate grid-point quantities.
      * GPINISLB
      * CP_FORCING (RHS of equations for 1D model with LSFORC=T)
    - GPINIDDH
    - EC_PHYS or EC_PHYS_LSLPHY (organigramme not detailed)
    - MF_PHYS_PREP
    - MF_PHYS (organigramme not detailed)
    - CPG_DIA -> (routines for some diagnostics, organigramme not detailed)
    - CPG_DYN ->
      * CPEULDYN ->
        - some GP.. routines computing intermediate g.p. quantities.
        - some SI.. routines computing some linear terms used in SI scheme.
        - VERDISINT -> VERDER (vertical derivatives for vertical finite elements scheme)
      * VDIFLCZ (organigramme not detailed)
    - CPG_END ->
      * surface_fields_mix.F90/GPPOPER
      * WRPHTRSF (organigramme not detailed)
      * GPMPFC
  * RADDRV (radiation scheme used at ECMWF, organigramme not detailed)
  * EC_PHYS_DRV (organigramme not detailed)
  * CPGLAG ->
    - GPTF1
    - GPRCP
```

∗ List of routines:

- **CPG_DRV**: driver for unlagged grid-point computations.
- **CPG**: unlagged grid-point computations.
- **CPG_GP**: beginning of unlagged grid-point computations; reads $t - \Delta t$ data in buffers, computes some diagnostic grid-point quantities (call to some **GP...** and **GNH...** routines via **CPG_GP_HYD**, **CPG_GP_NHEE**, **CPG_GP_NHQE**), does multiplications by the mapping factor, applies the second part of the temporal filter.
- **CPG_DIA**: interface for diagnostics (DDH,CFU,XFU) in the unlagged grid-point computations.
- **CPG_DYN**: interface routine for unlagged part of the Eulerian dynamics and simplified Buizza physics.
- **CPG_END**: end of unlagged grid-point computations; writes data in buffer, first part of the temporal filter, divisions by the mapping factor.
- **CPGLAG**: lagged grid-point computations and final filling of $t + \Delta t$ arrays.
- **GPTF1**: first part of the Asselin temporal filter.
- **GPTF2**: second part of the Asselin temporal filter.
- **GPPOPER**: Asselin temporal filter and some memory transfers for prognostic surface fields.
- **GPMPFC**: multiplications or divisions by the mapping factor or a power of the mapping factor to convert reduced quantities into geographical quantities or the inverse.
- **GPINIDDH**: some initialisation for DDH diagnostics; a subset of data computed by this routine may be modified by the unlagged physics.
- **GPINISLB**: initialisation of the buffer **SLB2...**.
- **EC_PHYS**: calls unlagged physics used at ECMWF.
- **EC_PHYS_LSLPHY**: calls split physics used at ECMWF.
- **MF_PHYS**: calls unlagged physics used at METEO-FRANCE.
- **MF_PHYS_PREP**: preparation of input data for **MF_PHYS**.
- **VDIFLCZ**: simplified Buizza physics.
- **CPEULDYN**: computes the RHS of Eulerian equations.
- **CP_FORCING**: computes the RHS of Eulerian equations for 1D model with **LSFORC**=T.
- **RADDRV**: radiation scheme used at ECMWF.
- **WRPHTRSF**: write out the trajectory for surface arrays to work file.

∗ CPG_GP_HYD: calculations done at instant t: The following quantities are computed in the following order:

- Hydrostatic pressure and quantities derivated from hydrostatic pressure:
  - **GPHPRE**:
    - ∗ computes half level hydrostatic pressure Π.
    - ∗ computes $\Delta\Pi$, $\frac{1}{\Delta\Pi}$, δ and α at full levels; the ratio between $\nabla\Pi/\Pi$ at full levels and $\nabla\Pi_{\mathrm{s}}$; $\frac{1}{\Pi}$ at half levels and $\frac{1}{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}$.
    - ∗ computes full level hydrostatic pressure Π.
  - **GPGRXYB**: $\nabla\delta$, $\nabla\alpha$ and $\nabla\alpha + \log\Pi$ at full levels.
- Dynamical quantities for hydrostatic model:
  - **GPRCP**: R, c_{p} and $\kappa = R/c_{\mathrm{p}}$ at full levels.
  - **GPRT**: RT and its horizontal derivatives at full levels.

  - **GPCTY**: vertical velocities $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ (at half levels) and ω/Π (at full levels); the divergence integral term.
  - **GPCTY_FORC**: the same vertical velocities but for 1D model with **LSFORC**=.T. .
  - **GPGEO**: gz at full levels and half levels.
  - **GPGRGEO**: horizontal gradient of gz at full levels and half levels.
  - Kinetic energy at full levels.
  - **GPHLWI**, **GPHLUV** and **GPUVS** if diagnostic of w is required.
  - **GPGRP**: pressure gradient force term used in the RHS of the horizontal wind equation.
  - **GPXX**, calculation of vertical divergence and **GPGW** if diagnostic of w is required.
- Adiabatic tendencies:
  - **GP_TNDLAGADIAB_UV**: explicit adiabatic Lagrangian tendency of the horizontal wind.

Some of these calculations are required also at $t - \Delta t$ for leap-frog advections.

$*$ CPG_GP_NHEE: calculations done at instant t: The following quantities are computed in the following order:

- Hydrostatic pressure and quantities derivated from hydrostatic pressure: cf. hydrostatic model (**GPHPRE** and **GPGRXYB**).
- Dynamical quantities:
  - **GPRCP**: R, $c_{\rm p}$ and $\kappa = R/c_{\rm p}$ at full levels.
  - **GPRT**: RT and its horizontal derivatives at full levels.
  - **GNHPRE**: p, $p - \Pi$, $\frac{p}{\Pi}$ and $\frac{\Pi}{p}$ at full levels.
  - **GNHPREH**: p at half levels and Δp at full levels.
  - **GNHGRPRE**: ∇p, $\frac{\nabla p}{p}$ and $\nabla\hat{Q}$ at full levels.
  - **GPCTY**: vertical velocities $\dot{\eta}\frac{\partial\Pi}{\partial\eta}$ (at half levels) and ω/Π (at full levels); the divergence integral term.
  - **GPCTY_FORC**: the same vertical velocities but for 1D model with **LSFORC**=.T. .
  - **GPGEO**: gz at full levels and half levels.
  - **GPGRGEO**: horizontal gradient of gz at full levels and half levels.
  - Kinetic energy at full levels.
  - **GPHLWI**: weights for interpolation of winds to model half levels.
  - **GPHLUV**: horizontal wind components at half levels (needs some weights computed in **GPHLWI**).
  - **GPUVS**: $U_{\rm surf}$ and $V_{\rm surf}$.
  - **GNHEE_GRP**: pressure gradient force term used in the RHS of the horizontal wind equation.
  - **GPXX**: $\mathtt{X}$ at full levels.
  - d and its horizontal gradient at full levels.
  - **GPGW**: gw at full levels and half levels.
  - **GNHGRGW**: horizontal gradient of gw at full levels and half levels.
  - **GNHD3**: D_3 at full levels.
- Adiabatic tendencies:
  - **GP_TNDLAGADIAB_UV**: explicit adiabatic Lagrangian tendency of the horizontal wind.
  - **GNH_TNDLAGADIAB_UVS**: explicit adiabatic Lagrangian tendency of the surface horizontal wind.
  - **GNH_TNDLAGADIAB_GW** or **GNHEE_TNDLAGADIAB_GW**: explicit adiabatic Lagrangian tendency of gw.
  - **GNH_TNDLAGADIAB_SVD** or **GNHEE_TNDLAGADIAB_SVD**: explicit adiabatic Lagrangian tendency of the vertical divergence variable.
  - **GNH_TNDLAGADIAB_SPD**: explicit adiabatic Lagrangian tendency of the pressure departure variable.

Some of these calculations are required also at $t - \Delta t$ for leap-frog advections.

∗ **CPG_GP_NHQE: calculations done at instant t:** The following quantities are computed in the following order:

- Hydrostatic pressure and quantities derivated from hydrostatic pressure: cf. hydrostatic model (**GPHPRE** and **GPGRXYB**).
- Dynamical quantities:
  - **GPRCP**: R, $c_{\rm p}$ and $\kappa = R/c_{\rm p}$ at full levels.
  - **GPRCPH**: $\kappa = R/c_{\rm p}$ at half levels.
  - **GPRT**: RT and its horizontal derivatives at full levels.
  - **GNHPRE**: p, $\exp(\kappa \hat{Q})$ and $\exp(-\kappa \hat{Q})$ at full levels.
  - **GNHQE_PREH**: $\exp(\kappa \hat{Q})$ at half levels.
  - **GPCTY**: vertical velocities $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ (at half levels) and ω/Π (at full levels); the divergence integral term.
  - **GPCTY_FORC**: the same vertical velocities but for 1D model with **LSFORC**=.T. .
  - **GPGEO**: gz at full levels and half levels.
  - **GPGRGEO**: horizontal gradient of gz at full levels and half levels.
  - Kinetic energy at full levels.
  - **GPHLWI**: weights for interpolation of winds to model half levels.
  - **GPHLUV**: horizontal wind components at half levels (needs some weights computed in **GPHLWI**).
  - **GPUVS**: $U_{\rm surf}$ and $V_{\rm surf}$.
  - **GNHQE_GRP**: pressure gradient force term used in the RHS of the horizontal wind equation.
  - **GPXX**: $\mathtt{X}_{\rm S}$ at full levels.
  - **GNHQE_XXD**: $\mathtt{X}_{\rm D}$ at full levels.
  - d and its horizontal gradient at full levels.
  - **GPGW**: gw at full levels and half levels.
  - **GNHGRGW**: horizontal gradient of gw at full levels and half levels.
- Adiabatic tendencies:
  - **GP_TNDLAGADIAB_UV**: explicit adiabatic Lagrangian tendency of the horizontal wind.
  - **GNH_TNDLAGADIAB_UVS**: explicit adiabatic Lagrangian tendency of the surface horizontal wind.
  - **GNHQE_TNDLAGADIAB_GW**: explicit adiabatic Lagrangian tendency of gw.
  - **GNHQE_TNDLAGADIAB_SVD**: explicit adiabatic Lagrangian tendency of the vertical divergence variable.

Some of these calculations are required also at $t - \Delta t$ for leap-frog advections.

18.4 Inventory of GP... and GNH... routines computing intermediate dynamical quantities (direct code).

∗ **GP... routines: the most important ones called in the hydrostatic model:**

- **GPCTY**: computes the vertical velocities $\dot{\eta}\frac{\partial \Pi}{\partial \eta}$ (at half levels) and ω/Π (at full levels).
- **GPCTY_FORC**: the same vertical velocities but for 1D model with **LSFORC**=.T. .
- **GPGEO**: computes geopotential height gz at full levels and half levels.
- **GPGRGEO**: computes the horizontal gradient of the geopotential height gz at full levels and half levels.
- **GPGRP**: computes the pressure gradient force term used in the RHS of the horizontal wind equation.
- **GPGRXYB**: computes the horizontal gradient of α and δ.
- **GP_KAPPA**: computes the quantity $\kappa_{\rm slhd}$ depending on deformation, which is used for example in the SLHD diffusion (see documentation (IDSL)).
- **GP_KAPPAT**: alternate way to compute $\kappa_{\rm slhd}$ (SLHD applied to "thermic" variables).
- **GP_STDDIS**: computes the quantity $STDDIS$ used to compute COMAD interpolation weights in the semi-Lagrangian scheme (see documentation (IDSL)).
- **GPHPRE**: computes hydrostatic pressure at full levels and half levels; computes different quantities linked to hydrostatic pressure, like α, δ, $1/\Pi$.
- **GPRCP**: computes $c_{\rm p}$, R and $\kappa = R/c_{\rm p}$ at full levels.
- **GPRH**: computes relative humidity at full levels.
- **GPRT**: computes RT and its horizontal derivatives at full levels.
- **GP_SPV**: computes $\Pi_{\rm s}$ and its horizontal derivatives.
- **GP_TNDLAGADIAB_UV**: explicit adiabatic Lagrangian tendency of the horizontal wind.

Most of these routines have tangent linear and adjoint versions.

∗ Some other GP... routines:

- **GPRCPH**: computes $\kappa = R/c_\mathrm{p}$ at half levels (NHQE).
- **GPEPT**: computes the equivalent potential temperature at full levels.
- **GPGW**: computes gw at full levels and half levels.
- **GPHLUV**: computes the horizontal wind components at half levels (needs some weights computed in **GPHLWI**).
- **GPHLWI**: computes weights for interpolation of winds to model half levels.
- **GPIET**: computes the isobaric equivalent temperature at full levels.
- **GPNOX**: computes NO2 concentration at full levels.
- **GPPRS0D**: computes simulated reflectivities at full levels.
- **GPPVO**: computes the potential temperature and potential vorticity at full levels.
- **GPPWC**: computes the total content of specific humidity in columns bounded by the top and half levels.
- **GPRH_2D**: cf. **GPRH** but for specific applications.
- **GPTCO3**: computes total column ozone mass per surface unit.
- **GPTET**: computes the potential temperature at full levels.
- **GPUVS**: computes U_surf and V_surf.
- **GPXX**: computes $\mathtt{X}$ (NHEE) or $\mathtt{X}_\mathrm{S}$ (NHQE) at full levels.

Most of these routines have tangent linear and adjoint versions.

∗ Some GNH... routines:

- **GNHD3**: computes D_3 at full levels.
- **GNHGRGW**: computes the horizontal gradient of gw at full levels and half levels.
- **GNHGRPRE**: computes ∇p, $\frac{\nabla p}{p}$ and $\nabla\hat{Q}$ at full levels.
- **GNHGW2SVD**: retrieves the vertical divergence d from gw.
- **GNHGW2SVDAROME**: special version of **GNHGW2SVD** called after the AROME physics.
- **GNHPRE**: computes p, $p - \Pi$, $\frac{p}{\Pi}$ and $\frac{\Pi}{p}$ at full levels.
- **GNHPREH**: computes p at half levels and Δp at full levels (NHEE).
- **GNHQE_PREH**: $\exp(\kappa\hat{Q})$ at half levels.
- **GNHQE_XXD**: computes $\mathtt{X}_\mathrm{D}$ (NHQE) at full levels.
- **GNHEE_LAPL**: multiplication by $\mathbf{L}$ (NHEE) at full levels.
- **GNHQE_LAPK**: multiplication by $\mathbf{L}_\kappa$ (NHQE) at full levels.
- **GNHSVD2GW**: cf. **GPGW** but with additional intermediate calculations.
- **GNHX**: computes $\mathtt{X}$, with additional intermediate calculations.
- **GNHEE_GRP**: computes the pressure gradient force term used in the RHS of the horizontal wind equation (NHEE).
- **GNHQE_GRP**: computes the pressure gradient force term used in the RHS of the horizontal wind equation (NHQE).
- **GNH_TNDLAGADIAB_SPD**: computes the explicit adiabatic Lagrangian tendency of the pressure departure variable (NHEE).
- **GNH_TNDLAGADIAB_UVS**: computes the explicit adiabatic Lagrangian tendency of the surface horizontal wind.
- **GNH_TNDLAGADIAB_GW**: computes the explicit adiabatic Lagrangian tendency of gw (NHEE) if **LNHEE_SVDLAPL_FIRST**=F.
- **GNH_TNDLAGADIAB_SVD**: computes the explicit adiabatic Lagrangian tendency of the vertical divergence variable (NHEE) if **LNHEE_SVDLAPL_FIRST**=F.
- **GNHEE_TNDLAGADIAB_GW**: computes the explicit adiabatic Lagrangian tendency of gw (NHEE) if **LNHEE_SVDLAPL_FIRST**=T.
- **GNHEE_TNDLAGADIAB_SVD**: computes the explicit adiabatic Lagrangian tendency of the vertical divergence variable (NHEE) if **LNHEE_SVDLAPL_FIRST**=T.
- **GNHQE_TNDLAGADIAB_GW**: computes the explicit adiabatic Lagrangian tendency of gw (NHQE).
- **GNHQE_TNDLAGADIAB_SVD**: computes the explicit adiabatic Lagrangian tendency of the vertical divergence variable (NHQE).

19 Basic description of the data flow under routine STEPO.

19.1 Introduction about OOPS-oriented structures.

Data are put into big structures, for example for the following items:

- Geometry: type GEOMETRY, variable **YRGEOMETRY** in **CNT0**.
- Model object: type MODEL, variable **YRMODEL** in **CNT0**.
- Fields: type FIELDS, variable **YRFIELDS** in **CNT0**.
- Trajectory: type MTRAJ, variable **YRMTRAJ** in **CNT0**.

And also:

- VARBC: type CLASS_VARBC, variable **YVARBC** in **CNT0**.
- TOVS control variable: type TOVSCV, variable **YLTCV** in **CNT0**.
- TOVS control variable for background: type TOVSCV_BGC, variable **YLTCV_BGC** in **CNT0**.
- Jo-table: type JO_TABLE, variable **YLJOT** in **CNT0**.
- ODB: class DBASE, variable **YLODB** in **CNT0**.
- FULL-POS: type TFPOS, variable **YLFPOS** in **CNT0**.

Variables referenced below are generally attributes of such structures. For simplification:

- One writes for example **NPROMA** (stands for **YRGEOMETRY%YRDIM%NPROMA**).
- One writes for example **NFLEVG** (stands for **YRGEOMETRY%YRDIMV%NFLEVG**).
- One writes for example **NGPBLKS** (stands for **YRGEOMETRY%YRDIM%NGPBLKS**).
- One writes for example **GMV** (stands for **YRFIELDS%YRGMV%GMV**).
- One writes for example **NDIMGMV** (stands for **YRFIELDS%YRGMV%NDIMGMV**).
- One writes for example **YT0%MT** (stands for **YRFIELDS%YRGMV%YT0%MT**).
- One writes for example **GFL** (stands for **YRFIELDS%YRGFL%GFL**).
- One writes for example **SP_SB** (stands for **YRFIELDS%YRSURF%SP_SB**).

Prognostic variables can be split into different classes:

- GMV variables.
- GFL variables.
- GMVS variables.
- 2D surface variables used in the physics.

A more detailed description of these classes has already been given in part 7.1. The GMV, GMVS, and a subset of GFL variables, require spectral and grid-point arrays. The 2D surface variables used in the physics require only grid-point arrays. Additional buffers are required to transmit data between the different parts of the grid-point calculations.

19.2 The GMV structure data.

Most grid-point space "GMV" variables referenced below are attributes of **YRFIELDS%YRGMV**.

$*$ **Treatment of $t - \Delta t$ and t data in grid-point space:** In grid-point space, the $t - \Delta t$ and t GMV data are stored in the array **GMV** which is an allocatable array allocated with (**NPROMA**,**NFLEVG**,**NDIMGMV**,**NGPBLKS**). Attribute **NDIMGMV** is the number of $t - \Delta t$ and t GMV fields. It is always above or equal to the number of GMV prognostic variables; the number of GMV prognostic variables is **NFTHER**+2, where **NFTHER** is the number of thermodynamic variables. Calculation of **NDIMGMV** and allocation of **GMV** are done in routine **SETUP_GMV** (encapsulated in **gmv_subs_mod.F90**). To identify each variable (for example T at instant t), one needs a pointer (in our example it is **YT0%MT**). Note that **GMV** contains not only prognostic variables (such as U, V, T) but also horizontal derivatives, divergence, vorticity, and also some additional diagnostic quantities.

For quantities at instant t, pointers are **YT0%M[X]** for a quantity [**X**], and **YT0%M[X]L**, **YT0%M[X]M** for zonal and meridian derivatives. Attribute **YT0** is a variable of type **TYPE_T0** (this type is defined in module **TYPE_GMVS**). Pointers are computed in routine **SETUP_T0** (encapsulated in **gmv_subs_mod.F90**) and these calculations determine the order of storing the t GMV fields in the array **GMV**.

For quantities at instant $t-\Delta t$, pointers are **YT9%M[X]** for a quantity [**X**], and **YT9%M[X]L**, **YT9%M[X]M** for zonal and meridian derivatives. Attribute **YT9** is a variable of type **TYPE_T9** (this type is defined in module **TYPE_GMVS**). Pointers are computed in routine **SETUP_T9** (encapsulated in **gmv_subs_mod.F90**) and these calculations determine the order of storing the $t - \Delta t$ GMV fields in the array **GMV**.

∗ **Treatment of $t + \Delta t$ data in grid-point space:** In grid-point space, the $t + \Delta t$ GMV data are stored in the array **GMVT1** which is an allocatable array allocated with (**NPROMA**,**NFLEVG**,**YT1%NDIM**,**NGPBLKS**). **YT1%NDIM** is the number of $t + \Delta t$ GMV fields. It is always above or equal to the number of GMV prognostic variables. Calculation of **YT1%NDIM** is done in routine **SETUP_T1** (encapsulated in **gmv_subs_mod.F90**). To identify each variable (for example T at instant $t + \Delta t$), one needs a pointer (in our example it is **YT1%MT**).

For quantities at instant $t + \Delta t$, pointers are **YT1%M[X]** for a quantity **[X]**. Attribute **YT1** is a variable of type **TYPE_T1** (this type is defined in module **TYPE_GMVS**). Pointers are computed in routine **SETUP_T1** (encapsulated in **gmv_subs_mod.F90**) and these calculations determine the order of storing the $t + \Delta t$ GMV fields in the array **GMVT1**.

∗ **Spectral GMV data:** Individual GMV fields are attributes of variable **YRFIELDS%YRSPEC**.

- Attributes **VOR**, **DIV**, **T**, **SPD**, **SVD**, **NHX**: contain spectral data respectively for vorticity, divergence, temperature, pressure departure variable, vertical divergence variable, `X`.
- Attribute **HV**: collective array for thermodynamic variables. The "standard order" in this array is temperature, pressure departure variable, vertical divergence variable.
- Attribute **SP3D**: collective array for all 3D variables.

Features about the **SPECTRAL_FIELD** structure can be found in some "algor/module" modules, the name starts by "spectral_fields".

∗ **Number of variables: sum-up.**

- Number of GMV prognostic variables: **NFTHER**+2.
- Number of thermodynamic GMV prognostic variables: **NFTHER**.
- Number of fields in the array **GMV**: **NDIMGMV**.
- Number of fields in the array **GMVT1**: **YT1%NDIM**.
- Position of a field X at the instant t in the array **GMV**: **YT0%M[X]**.
- Position of the zonal derivative of a field X at the instant t in the array **GMV**: **YT0%M[X]L**.
- Position of the meridian derivative of a field X at the instant t in the array **GMV**: **YT0%M[X]M**.
- Position of a field X at the instant $t - \Delta t$ in the array **GMV**: **YT9%M[X]**.
- Position of the zonal derivative of a field X at the instant $t - \Delta t$ in the array **GMV**: **YT9%M[X]L**.
- Position of the meridian derivative of a field X at the instant $t - \Delta t$ in the array **GMV**: **YT9%M[X]M**.
- Position of a field X at the instant $t + \Delta t$ in the array **GMVT1**: **YT1%M[X]**.

19.3 The GMVS structure data.

Most grid-point space "GMVS" variables referenced below are attributes of **YRFIELDS%YRGMV**.

∗ **Treatment of $t - \Delta t$ and t data in grid-point space:** In grid-point space, the $t - \Delta t$ and t GMVS data are stored in the array **GMVS** which is an allocatable array allocated with (**NPROMA**,**NDIMGMVS**,**NGPBLKS**). Attribute **NDIMGMVS** is the number of $t - \Delta t$ and t GMVS fields. It is always above or equal to the number of GMVS prognostic variables. Calculation of **NDIMGMVS** and allocation of **GMVS** are done in routine **SETUP_GMV** (encapsulated in **gmv_subs_mod.F90**). To identify each variable (for example $\log \Pi_s$ at instant t), one needs a pointer (in our example it is **YT0%MSP**). Note that **GMVS** contains not only prognostic variables (such as $\log \Pi_s$) but also horizontal derivatives.

Pointers **YT0%M[X]**, **YT0%M[X]L**, **YT0%M[X]M**, **YT9%M[X]**, **YT9%M[X]L**, **YT9%M[X]M** (see part 19.2) are also used to determine the order of storing the GMVS fields in the array **GMVS**.

∗ **Treatment of $t + \Delta t$ data in grid-point space:** In grid-point space, the $t + \Delta t$ GMVS data are stored in the array **GMVT1S** which is an allocatable array allocated with (**NPROMA**,**YT1%NDIMS**,**NGPBLKS**). **YT1%NDIMS** is the number of $t + \Delta t$ GMVS fields. It is always above or equal to the number of GMVS prognostic variables. Calculation of **YT1%NDIMS** is done in routine **SETUP_T1** (encapsulated in **gmv_subs_mod.F90**). To identify each variable (for example $\log \Pi_s$ at instant $t + \Delta t$), one needs a pointer (in our example it is **YT1%MSP**).

Pointers **YT1%M[X]** (see part 19.3) are also used to determine the order of storing the GMVT1S fields in the array **GMVT1S**.

∗ **Spectral GMVS data:** Individual GMVS fields are attributes of array **YRFIELDS%YRSPEC**.

- Attribute **SP**: contains spectral data for $\log \Pi_s$.
- Attribute **SP2D**: collective array for all GMVS variables; also contains additional spectral data, such as the surface orography.

∗ **Number of variables: sum-up.**

- Number of GMVS prognostic variables: 1.
- Number of fields in the array **GMVS**: **NDIMGMVS**.
- Number of fields in the array **GMVT1S**: **YT1%NDIMS**.
- Position of a field X at the instant t in the array **GMVS**: **YT0%M[X]**.
- Position of the zonal derivative of a field X at the instant t in the array **GMVS**: **YT0%M[X]L**.
- Position of the meridian derivative of a field X at the instant t in the array **GMVS**: **YT0%M[X]M**.
- Position of a field X at the instant $t - \Delta t$ in the array **GMVS**: **YT9%M[X]**.
- Position of the zonal derivative of a field X at the instant $t - \Delta t$ in the array **GMVS**: **YT9%M[X]L**.
- Position of the meridian derivative of a field X at the instant $t - \Delta t$ in the array **GMVS**: **YT9%M[X]M**.
- Position of a field X at the instant $t + \Delta t$ in the array **GMVT1S**: **YT1%M[X]**.

19.4 The GFL structure data.

Most grid-point space "GFL" variables referenced below are attributes of **YRFIELDS%YRGFL**.

∗ **Treatment of $t - \Delta t$ and t data in grid-point space:** In grid-point space, the $t - \Delta t$ and t GFL data are stored in the array **GFL** which is an allocatable array allocated with (**NPROMA**,**NFLEVG**,**YGFL%NDIM**,**NGPBLKS**). Not that the array **GFL** contains not only prognostic variables (such as q), but also horizontal derivatives, and non advectable pseudo-historic grid-point GFL variables (such as $q_{\rm CPF}$).

YGFL%NDIM is the number of $t - \Delta t$ and t GFL fields in grid-point space. It is always above or equal to the number of GFL prognostic variables; the number of GFL prognostic variables is **YGFL%NUMFLDS**. Calculation of **YGFL%NDIM** is done in routines **DEFINE_GFL_COMP** and **SET_GFL_ATTR** (these routines are encapsulated in **gfl_subs_mod.F90**). Allocation of **GFL** is done in routine **SUSC2C**.

To identify each variable (for example q at instant t), one needs a pointer (in our example it is **YGFL%YQ%MP**; variable **YGFL** is in **YOM_YGFL**). Note that **GFL** contains not only prognostic variables (such as q, $q_{\rm l}$, $q_{\rm i}$, $q_{\rm a}$, $\mathcal{O}3$) but also horizontal derivatives.

For quantities at instant t, pointers are **YGFL%Y[X]%MP** for a quantity **[X]**, and **YGFL%Y[X]%MPL**, **YGFL%Y[X]%MPM** for zonal and meridian derivatives. Attribute **Y[X]** is a variable of type **TYPE_GFL_COMP** declared in module **yom_ygfl.F90**; the attributes of this type are defined in the part **TYPE_GFL_COMP** of module **YOM_YGFL**. Pointers **YGFL%Y[X]%MP**, **YGFL%Y[X]%MPL** and **YGFL%Y[X]%MPM** are computed in routine **DEFINE_GFL_COMP** (encapsulated in **gfl_subs_mod.F90**) called by **SUGFL2**, and these calculations determine the order of storing the t GFL fields in the array **GFL**. The "**SUGFL2**-standard order" of storing the different variables [X] is defined in routine **SUGFL2**. The purely grid-point GFL variables (including the non advected pseudo-historic GFL variables) are stored first (in the "**SUGFL2**-standard order"), then the GFL variables with a spectral representation are stored (in the "**SUGFL2**-standard order"). Note that no horizontal derivative is available for purely grid-point GFL variables.

For quantities at instant $t - \Delta t$, pointers are **YGFL%Y[X]%MP9** for a quantity **[X]** (no zonal nor meridian derivative is required for GFL variables at $t - \Delta t$). Pointers **YGFL%Y[X]%MP9** are computed in routine **SET_GFL_ATTR** (encapsulated in **gfl_subs_mod.F90**) called by **SUGFL3**. The GFL variables are stored in the "**SUGFL3**-standard order" (no distinction between spectral and purely grid-point variables) and this "**SUGFL3**-standard order" defined in routine **SUGFL3** can be different from the "**SUGFL2**-standard order". Note that no $t - \Delta t$ variable is available for the non advected pseudo-historic GFL variables (because in this case the discretisation always writes $X(t + \Delta t) - X(t) = \Delta t F_{\rm X}$, even in a leap-frog scheme).

∗ **Treatment of $t + \Delta t$ data in grid-point space:** In grid-point space, the $t + \Delta t$ GFL data are stored in the array **GFLT1** which is an allocatable array allocated with (**NPROMA**,**NFLEVG**,**YGFL%NDIM1**,**NGPBLKS**). **YGFL%NDIM1** is the number of $t + \Delta t$ GFL fields in grid-point space. It is always above or equal to the number of GFL prognostic variables. Allocation of **GFLT1** is done in routine **GP_MODEL** in the direct code. Calculation of **YGFL%NDIM1** is done in routines **DEFINE_GFL_COMP** and **SET_GFL_ATTR** (encapsulated in **gfl_subs_mod.F90**). To identify each variable (for example q at instant $t + \Delta t$), one needs a pointer (in our example it is **YGFL%YQ%MP1**; variable **YGFL** is in **YOM_YGFL**). Note that **GFLT1** contains prognostic variables at $t + \Delta t$.

For quantities at instant $t + \Delta t$, Pointers are **YGFL%Y[X]%MP1** for a quantity [X]. **YGFL%Y[X]%MP1** is computed in **DEFINE_GFL_COMP** (encapsulated in **gfl_subs_mod.F90**) called by **SUGFL2**. The purely grid-point GFL variables (including the non advected pseudo-historic GFL variables) are stored first (in the "**SUGFL2**-standard order"), then the GFL variables with a spectral representation are stored (in the "**SUGFL2**-standard order").

∗ **Spectral GFL data:** There are individual arrays, and also a collective array for all GFL variables (attributes of variable **YRFIELDS%YRSPEC**).

- Attribute **GFL**: collective array for GFL variables. In **GFL**, the spectral GFL variables are stored in the "**SUGFL2**-standard order".
- Attributes **Q**, **L**, **I**, **O3**: contain spectral data respectively for humidity, liquid water, ice and ozone.

Dimension of **YRFIELDS%YRSPEC%GFL** is (**NFLSUR**,**NSPEC2**,**YGFL%NUMSPFLDS**). The number of spectral GFL prognostic variables is **YGFL%NUMSPFLDS**. The total number of GFL prognostic variables is **YGFL%NUMFLDS**. The following relationship is always true:
YGFL%NUMSPFLDS $\leq$ **YGFL%NUMFLDS** $\leq$ **YGFL%NDIM**.
For example, if the GFL variables are q and q_l and if we assume additionally:

- q is treated in spectral space and is advected, its horizontal derivatives are required.
- q_l is treated as a purely grid-point data and is not advected, its horizontal derivatives are not required.
- the advection scheme is an Eulerian one.

In this case, the three above integer numbers have the following value: **YGFL%NUMSPFLDS**=1, **YGFL%NUMFLDS**=2, **YGFL%NDIM**=6 (the last one corresponding to $q(t)$, zonal and meridian derivatives of $q(t)$, $q(t - \Delta t)$, $q_l(t)$, $q_l(t - \Delta t)$).
YGFL%NUMSPFLDS and **YGFL%NUMFLDS** are computed in the sequence **SU0YOMA** − > **SUGFL2** − > **DEFINE_GFL_COMP** (encapsulated in **gfl_subs_mod.F90**).
In spectral space, GFL variables can be accessed by each individual **YRFIELDS%YRSPEC%[X]** array, but also via the collective array **YRFIELDS%YRSPEC%GFL**. In this case one needs to know the location of the variable [X] which is searched for.

- If *jnum* is the numbering of the GFL field among the **YGFL%NUMFLDS** fields, the index *jnumsp* among the subset of GFL fields which have a spectral representation is given by **YGFL%COMP**(*jnum*)**%MPSP** (variable **YGFL%COMP** is in module **YOM_YGFL**); *jnumsp* always matches *jnumsp* $\leq$ *jnum*. In our above example, for q, *jnum* and *jnumsp* are equal to 1.
- If we want to know, for a specific variable [X], its location in **YRFIELDS%YRSPEC%GFL**, the answer is given by **YGFL%Y[X]%MPSP**. In our above example, for q, **YGFL%Y[X]%MPSP** is equal to 1.

∗ **Ordering of the GFL data:** As seen previously, there are two standard orders, the first one defined by **SUGFL3** and the other one defined by **SUGFL2**. See code sources of **SUGFL2** and **SUGFL3** to know about GFL orderings. The purpose of having two different standard orders is only to limit the chance of hidden bugs. The important rule to be kept in mind is the following: **No reference to the ordering of the GFL variables must appear elsewhere than in:**

- **the sequence of calls to SET_GFL_ATTR in SUGFL3 .**
- **the sequence of calls to DEFINE_GFL_COMP in SUGFL2 .**

No hard coded explicit reference to this order must appear elsewhere in the code, the other parts must work for an indifferent order of the GFL variables and must use the pointers defined by the attributes of the variables YGFL%COMP and YGFL%Y[X] .

∗ **Groups of GFL variables:** Some generic names (like **AERO**) contain several variables: for example extra-GFL variables, $q_{\rm FORC}$, $q_{\rm EZDIAG}$, $q_{\rm GHG}$, $q_{\rm CHEM}$, $q_{\rm AERO}$, $q_{\rm ERA40}$, $q_{\rm AEROUT}$, $q_{\rm UVP}$, $q_{\rm PHYS}$, $q_{\rm NOGW}$, $q_{\rm SLDIA}$, $q_{\rm CRM}$. It is possible to add some new GFL variables in these groups without deeply modifying the code, but with some minor modifications in the namelist (and also in the files to be read).

- The number of variables in each group is respectively **NGFL_EXT**, **NGFL_FORC**, **NGFL_EZDIAG**, **NGHG**, **NCHEM**, **NAERO**, **NERA40**, **NAEROUT**, **NUVP**, **NGFL_PHYS**, **NNOGW**, **NSLDIA**, **NCRM** (namelist **NAMGFL**). In **YOM_YGFL**, these variables are attributes of **YGFL**.
- Some information for **TFP_GFL**(*jgfl*)**%CLNAME** (extra-GFL), **TFP_GHG**(*jgfl*)**%CLNAME** (greenhouse gases), **TFP_CHEM**(*jgfl*)**%CLNAME** (chemistry), **TFP_AERO**(*jgfl*)**%CLNAME** (aerosols), can be required in **NAMAFN** to give the name of the fields which appear in the files.
- Some modifications of the attributes of **YEXT_NL**, **YFORC_NL**, **YEZDIAG_NL**, **YGHG_NL**, **YCHEM_NL**, **YAERO_NL**, **YERA40_NL**, **YAEROUT_NL**, **YUVP_NL**, **YPHYS_NL**, **YNOGW_NL**, **YSLDIA_NL**, **YCRM_NL**, can be required in **NAMGFL**.
- The numbering of these GFL is included in **YGFL%NUMSPFLDS** and **YGFL%NUMFLDS**.
- About extra-GFL, some variables in the code have kept the letters **SV** or **SCVA** (for scalar variables).

∗ Number of variables: sum-up.

- Number of GFL variables (spectral and grid-point, including the non advected pseudo-historic ones): **YGFL%NUMFLDS**.
- Number of spectral GFL variables: **YGFL%NUMSPFLDS**.
- Number of purely grid-point GFL variables: **YGFL%NUMGPFLDS**.
- Number of advectable GFL variables (spectral and grid-point): **YGFL%NUMFLDS_SL1**.
- Number of advectable GFL variables (spectral and grid-point) which require a spline cubic vertical interpolation in the semi-Lagrangian scheme: **YGFL%NDIM_SPL**.
- Number of GFL variables having a time $t - \Delta t$ representation: **YGFL%NUMFLDS9**.
- Number of GFL variables having a time $t + \Delta t$ representation: **YGFL%NUMFLDS1**.
- Number of spectral GFL variables having a time $t + \Delta t$ representation: **YGFL%NUMSPFLDS1**.
- Number of extra-GFL variables: **YGFL%NGFL_EXT**.
- Number of forcing variables: **YGFL%NGFL_FORC**.
- Number of easy diagnostics variables: **YGFL%NGFL_EZDIAG**.
- Number of greenhouse gases: **YGFL%NGHG**.
- Number of chemistry components: **YGFL%NCHEM**.
- Number of aerosols: **YGFL%NAERO**.
- Number of ERA40 fields: **YGFL%NERA40**.
- Number of output aerosol fields: **YGFL%NAEROUT**.
- Number of output fields from UV processor: **YGFL%NUVP**.
- Number of GFL variables for physics diagnostics: **YGFL%NGFL_PHYS**.
- Number of diagnostic fields for NORO GWD scheme: **YGFL%NNOGW**.
- Number of SL dynamics diagnostic fields: **YGFL%NSLDIA**.
- Number of CRM extra fields: **YGFL%NCRM**.
- Number of fields in the array **GFL**: **YGFL%NDIM**.
- Number of fields at time t in the array **GFL**: **YGFL%NDIM0**.
- Number of fields at time $t - \Delta t$ in the array **GFL**: **YGFL%NDIM9**.
- Number of fields in the array **GFLT1**: **YGFL%NDIM1**.
- Number of fields in the array **GFL**, associated to variables which require a spline cubic vertical interpolation in the semi-Lagrangian scheme: **YGFL%NDIM_SPL**.
- Position of a field X at the instant t in the array **GFL**: **YGFL%Y[X]%MP**.
- Position of the zonal derivative of a field X at the instant t in the array **GFL**: **YGFL%Y[X]%MPL**.
- Position of the meridian derivative of a field X at the instant t in the array **GFL**: **YGFL%Y[X]%MPM**.
- Position of a field X at the instant $t - \Delta t$ in the array **GFL**: **YGFL%Y[X]%MP9**.
- Position of a field X at the instant $t + \Delta t$ in the array **GFLT1**: **YGFL%Y[X]%MP1**.
- Position of an advectable field X in an array of dimension **YGFL%NUMFLDS_SL1**: **YGFL%Y[X]%MP_SL1**.
- Position of a spectral field X in the array **SPGFL**: **YGFL%Y[X]%MPSP**.

Note that some other dimensioning and logical variables are available in yom_ygfl.F90.

∗ How to add new GFL variables or new GFL attributes. A specific user's guide (IDNGFL) has been written.

19.5 The 2D surface variables used in the physics structure data.

∗ **Introduction:** These fields are never transformed in spectral space and they are never advected (they concern the inner soil or the surface where the wind is assumed to be zero). They describe mainly surface or soil 2D data, but some of them can describe an upper-air 2D data (for example 2 meter temperature, total convective cloudiness), an upper-air 3D data, or a total atmospheric content (for example total column water vapour). Even if these data are not always 2D surface data, we keep the generic name "2D surface data" for simplification.

The 2D surface variables are divided into two main classes: the prognostic surface variables and the diagnostic surface variables. The class of diagnostic surface variables includes a couple of pseudo-prognostic variables which need some care. The main code features describing these variables are in **SURFACE_FIELDS_MIX** and in **SU_SURF_FLDS**.

All the variables **SP_..**, **YSP_..**, **SD_..**, **YSD_..** referenced below are attributes of variable **YRFIELDS%YRSURF** (type **TSURF**). Denotations **SP_..**, **YSP_..**, **SD_..**, **YSD_..** respectively stand for **YRFIELDS%YRSURF%SP_..**, **YRFIELDS%YRSURF%YSP_..**, **YRFIELDS%YRSURF%SD_..**, **YRFIELDS%YRSURF%YSD_..**.

∗ **Prognostic surface variables:** This class of variables can be divided into several groups, each group has a short two-letter code and a long 3 to 5-letter code.

- SB=SOILB: soil quantities for the different reservoirs.
- SG=SNOWG: quantities linked to surface snow.
- SL=LAKEB: quantities linked to lake (FLAKE model).
- RR=RESVR: soil quantities for the surface and sometimes the first (upper) reservoir.
- CL=CLS : surface boundary layer prognostic quantities.
- OM=OML : prognostic quantities for ocean mixed layer model (KPP).
- EP=EXTRP: extra 3D prognostic fields.
- X2=XTRP2: extra 2D prognostic fields.
- CI=CANRI: 2D fields used in CANARI.

The comprehensive list of variables is not detailed here and one can find it in **SURFACE_FIELDS_MIX**. Ordering the individual variables inside each group is given by the content of **SU_SURF_FLDS**. It is desirable that the ordering of the declarations in **SURFACE_FIELDS_MIX** match the ordering of **SU_SURF_FLDS**. **No reference to variable ordering should appear elsewhere than in SU_SURF_FLDS and SURFACE_FIELDS_MIX.**

Buffers, pointers and number of variables in each group: sum-up:

- Group variables containing the data are **SP_[group two-letter code]**.
- Variable **YSP_[group two-letter code]D** contains some dimensions. For example, there are **YSP_[group two-letter code]D%NDIM** fields in the above group variable.
- Variable **YSP_[group two-letter code]** contains pointers and some other attributes. For example, if [X] is the generic name of an individual variable in the current group, the pointer allowing to retrieve the t-value (resp. $t - \Delta t$-value, $t + \Delta t$-value) of [X] is **YSP_[group two-letter code]%Y[X]%MP0** (resp. **YSP_[group two-letter code]%Y[X]%MP9**, **YSP_[group two-letter code]%Y[X]%MP1**).
- Some other dimensioning variables and some other attributes can be found in routine **SURFACE_FIELDS_MIX**.

∗ **Diagnostic surface variables:** This class of variables can be divided into several groups, each group has a short two-letter code and a long 5-letter code or 6-letter code.

- VF=VARSF : climatological/geographical parameters.
- VP=VCLIP : deep soil parameters.
- VV=VCLIV : vegetation parameters.
- VN=VCLIN : cloudiness predictors.
- VH=VCLIH : convective cloud parameters.
- VK=VCLIK : convective cloud pseudo-historic fields.
- VA=VCLIA : aerosols parameters.
- VG=VCLIG : ice-coupler parameters.

- VC=VO3ABC : climatological ozone profiles.
- V2=VDIAGO2: 2-D climatological/diagnostic fields for an ocean mixed layer model (KPP).
- V3=VDIAGO3: 3-D climatological/diagnostic fields for an ocean mixed layer model (KPP).
- VD=VDIAG : diagnostic fields (generally used only at ECMWF).
- SM=SATSIM : (ECMWF) simulated satellite images.
- WS=WAVES : surface quantities over sea (used by IFS).
- WW=WAM : surface quantities over sea (used by WAM).
- VX=VCLIX : auxiliary climatological parameters.
- XA=VEXTRA : extra 3D fields.
- DI=VEXTRDI: targeted 3D fields.
- XR=VEXTRR : extra 3D fields for radiation.
- X2=VEXTR2 : extra 2D fields.
- SFL=SFLUX : surface flux for EDKF.
- SFO=SFORC : surface forcing for 1D model (MUSC).
- OC=OCE : ocean model fields.
- PF : precipitation fraction.

The comprehensive list of variables is not detailed here and one can find it in **SURFACE_FIELDS_MIX**. Ordering the individual variables inside each group is given by the content of **SU_SURF_FLDS**. It is desirable that the ordering of the declarations in **SURFACE_FIELDS_MIX** match the ordering of **SU_SURF_FLDS**. **No reference to variable ordering should appear elsewhere than in SU_SURF_FLDS and SURFACE_FIELDS_MIX.**

Note that we can also divide these variables into 3 sub-classes:

- Constants: they have no time-evolution and they are always input data in the model dynamics and physics: for example the land-sea mask.
- Diagnostics: they are generally diagnosed by the physics and kept at the current time-step only: they are computed for diagnostics (XFU, DDH for example). The content is not conserved at the following timestep. Example: the total convective cloudiness.
- Pseudo-prognostic variables: they are generally diagnosed by the physics and conserved until the following timestep where they are used as input data by the physics. For MF purpose there are currently 7 pseudo-prognostic variables:
  - VF=VARSF: variable **Z0F** (gravity $*$ surface roughness length).
  - VV=VCLIV: variable **HV** (resistance to evapotranspiration).
  - VV=VCLIV: variable **Z0H** (gravity $*$ surface roughness length for heat).
  - VH=VCLIH: variable **PBLH** (PBL height).
  - VH=VCLIH: variable **SPSH** (variable for ALARO prognostic convection scheme).
  - VH=VCLIH: variable **QSH** (surface moisture historic variable, used by TOUCANS).
  - VK=VCLIK: variable **UDGRO** (convective cloud updraught top position).

Buffers, pointers and number of variables in each group: sum-up:

- Group variables containing the data are **SD_[group two-letter code]**.
- Variable **YSD_[group two-letter code]D** contains some dimensions. For example, there are **YSD_[group two-letter code]D%NDIM** fields in the above group variable.
- Variable **YSP_[group two-letter code]** contains pointers and some other attributes. For example, if [X] is the generic name of an individual variable in the current group, the pointer allowing to retrieve its value is **YSD_[group two-letter code]%Y[X]%MP**.
- Some other dimensioning variables and some other attributes can be found in routine **SURFACE_FIELDS_MIX**.

$*$ Set-up:

- Pointer calculations, allocations are done in **SU_SURF_FLDS**.
- Some of the buffers (initial value of the prognostic variables, and input constants) are read on files via **SUGRIDA** (ARPEGE files) or **SUGRIDG** (GRIB files). **SUGRIDA** and **SUGRIDG** directly fill the surface buffers.

∗ **Data flow under CPG:** The buffers **SP...** and **SD...** are filled (they currently appear as dummy arguments **PSP...** and **PSD...**, although **YDSURF** is passed too), without any intermediate local variable.

- In **CPG_GP**, when leap-frog scheme, t data of **SP...** are copied in the $t - \Delta t$ part of **SP...** via routine **GPPOPER**.
- In **CPG_GP**, when required, $t - \Delta t$ part of **SP...** is copied in the $t + \Delta t$ part of **SP...**.
- In **MF_PHYS** the t data (SL2TL) or $t - \Delta t$ data (leap-frog) of **SP...** is used as input data of the physics. Constants of **SD...** are used as input data. Diagnostics of **SD...** are diagnosed via the physics output. Pseudo-historic variables of **SD...** enter the physics (input value is used) and are updated by the physics. In some cases (**MF_PHYS** called for only diagnostic purpose, without temporal evolution), it is necessary to restore the input data at the end of **MF_PHYS**. The $t + \Delta t$ part of **SP...** is updated via routine **CPTENDS**.
- In **CPG_END** the $t + \Delta t$ part of **SP...** is copied into the t part, with a temporal filter for leap-frog schemes (call to **GPPOPER**).

∗ **Other groups of surface variables:** There are groups which have not been reported in the new surface data scheme:

- VU=VCLIU: reference variables for nudging: stored in arrays **XVU[X]** of **YOMNUD**.
- VR=VRADF: radiation fields: stored in some arrays of **YOMRADF**.
- VT=VTILE: tile surface scheme fields (ECMWF only): stored in arrays **R[X]TI** of **YOE_TILE_PROP**.

∗ **Additional remarks:**

- ECMWF clearly makes a difference between the soil reservoirs (4 reservoirs) and the surface. The soil reservoirs prognostic variables are always in the group SOILB, and the surface prognostic variables are always in the group RESVR (this is not a very logical choice!).
- METEO-FRANCE does not always a distinction between the surface ("skin" variables) and the superficial reservoir (there are 2 reservoirs: a superficial reservoir and a deep reservoir): the distinction is done for the water content but not for temperature. The deep reservoir prognostic variables are always in the group SOILB, and the surface ("skin") and superficial reservoir prognostic variables are always in the group RESVR.
- The surface buffer **GPARBUF** which is used in the externalised surface (SURFEX) still exists.
- The generic term "surface" is applied to several topics and the reader has to keep in mind the following definitions in order to avoid confusions between the different topics called "surface". The different topics called "surface" are:
  - The surface fields previously described, mainly stored in the buffers **SP_...** and **SD_...**.
  - The externalised surface projects SURFEX and MSE, using the array **GPARBUF** to communicate surface data between the ARP/ALD routines and the SURFEX+MSE routines: it is currently used in the AROME physics.
  - The externalised ECMWF surface (project SUR) for ECMWF physics.

  For example, to deallocate some buffers used in the ECMWF surface package one calls the routine **SURFDEALLO** and to deallocate the **SP_...** and **SD_...** buffers one calls the routine **surface_fields_mix.F90/DEALLO_SURF**.

∗ **How to add new surface variables or new surface attributes.** A specific user's guide (IDNSUR) has been written.

19.6 Treatment of the data flow under routine STEPO.

∗ **Introduction:** The purpose is now to describe all the steps done under **STEPO** (one timestep) and what are the different arrays involved. We give an overview of the data flow for **GMV**, **GMVS**, **GFL** and grid-point surface fields under **STEPO**. One focusses on the Eulerian direct model but a short reference of the semi-Lagrangian scheme will be also done.

The time where arrays were frequently reallocated and deallocated is now behind us, computers provide enough memory to keep them allocated during all the model integration. Most of the variables among **YRGFL**, **YRGMVS**, **YRGMV**, **YRSURF**, **YRSPEC**, remain allocated during all the model integration.

These arrays, generally encapsulated, are now passed via dummy arguments to **STEPO** and callees.

We recall that a timestep integration under **STEPO** has the following steps:

- I/O in spectral space.

- Inverse transforms.
- Grid-point calculations.
- Grid-point coupling for LAM models.
- Direct transforms.
- Spectral calculations.

$\ast$ **Spectral space:** **YRFIELDS%YRSPEC** is passed as dummy argument to **STEPO** and its callees, in particuliar to **(E)SPCM**. Dummy argument name is generally **YDSPEC** or **YDSP**. Individual quantities **YDSPEC%[X]** or **YDSP%[X]** may appear in deeper pieces of the call-tree.

$\ast$ **Inverse spectral transformation:** It is done under the routine **TRANSINVH** (**ETRANSINVH** in LAM models) for the variable at instant t. The input spectral data is **YDSPEC**. The output grid-point data (also containing derivatives) are **YDGMV%GMV**, **YDGMV%GMVS** and **YDGFL%GFL**.

$\ast$ **Grid-point space:** Dummy argument of **STEPO** is **YDFIELDS**, with attributes **YRGMV**, **YRGMVS**, **YRGFL**, **YRSURF**.

$\ast$ **Grid-point computations:** It is done under **GP_MODEL** (called under **SCAN2M**). The code is organized into several loops on **NPROMA**-blocks (each unit of the loop does calculations for one **NPROMA**-block). These different loops are:

- **CPG_DRV** $->$ **CPG**: unlagged grid-point calculations.
- The ECMWF radiation scheme.
- The semi-Lagrangian interpolations (under **CALL_SL..** routines).
- The lagged physics.
- Some final lagged grid-point calculations (**CPGLAG**) (including the phase "1" of the Asselin temporal filter).

Additional arrays are necessary to communicate data between the different parts of the grid-point calculations.

- For quantities to be interpolated in the semi-Lagrangian scheme: **ZSLBUF1** in **GP_MODEL** (more details will be given later about how it is used under **CPG** and **CALL_SL**).
- For other quantities to be transmitted: **ZSLBUF2**.
- For specific applications (PC schemes in particular) some specific arrays can be used: buffer **GPPCBUF** and local array **ZGPPC**.

For more details:

- The array **ZSLBUF1** (the dimensions of which are (**YRSL%NASLB1**;**NFLDSLB1**)) is allocated or declared in **GP_MODEL_HEAP** or **GP_MODEL_STACK**. It is needed to store all data which will be used in the semi-Lagrangian interpolations. This array has the name **PB1** under **CPG**.
- The array **ZSLBUF2** is allocated or declared in **GP_MODEL_HEAP** or **GP_MODEL_STACK** (name **PB2** under **CPG**).
- Under **CPG** some local arrays are used. The collective arrays are **ZGPPC** (used in the predictor-corrector scheme), **ZSLBUF1AU** (intermediate array, the data of which will be transferred in **PB1** for semi-Lagrangian interpolations).
- In **CPG**, input dummy or global arrays are **PGMV** (GMV fields at instants $t - \Delta t$ and t), **PGMVS** (GMVS fields at instants $t - \Delta t$ and t), **PGFL** (GFL fields at instants $t - \Delta t$ and t), and the surface buffers **PSP_..** and **PSD_..** (surface fields for unlagged MF physics at instant t or $t - \Delta t$).
- **CPG_GP**:
  - does the part two of the Asselin temporal filter (**GPTF2**).
  - converts the reduced variables into geographical variables via a multiplication by a power of the mapping factor (**GPMPFC**).
  - computes some intermediate variables linked with the hydrostatic surface pressure (routine **GP_SPV**).
  - does memory transfers in the surface buffer **PSP_..**.
  - computes some intermediate grid point variables like δ, α, RT, c_p, gz, Φ, etc..., which are used as input of the physics, the dynamics and some diagnostics.
  - computes adiabatic RHS of equations.

  - calls **GPINISLB** which puts some results in **PGMVT1**, **PGMVT1S**, **PGFLT1**, **PB2**.

  When exiting **CPG_GP**: in the semi-Lagrangian scheme **PGMVT1**, **PGMVT1S** and **PGFLT1** contain 0; in the Eulerian scheme these arrays contains $X(t-\Delta t)$ after modification by the phase two of the Asselin filter.

- Unlagged physics: it is the MF physics (managed by **MF_PHYS**). It uses as input the t variables for a SL2TL scheme and the $t-\Delta t$ variables for a leap-frog scheme. After computation of the physical fluxes and divergences of fluxes the physical tendencies are computed and added to the RHS of the equations; the physical tendencies are put in the interpolation buffer in a SL scheme and in **PGMVT1**, **PGMVT1S** and **PGFLT1** in an Eulerian scheme. At the end of this stage: for an Eulerian scheme, these T1-arrays contain $X(t-\Delta t)+F_X(t-\Delta t)$; for a SL3TL scheme, **ZSLBUF1AU** then **PB1** contains $X(t-\Delta t)+F_X(t-\Delta t)$; for a SL2TL scheme, **ZSLBUF1AU** then **PB1** contains $X(t)+F_X(t)$. The current physics has a vertical dependency but no horizontal dependency (the only horizontal dependency is the computation of the moisture convergence as input to the MF convection scheme, the moisture convergence needs the horizontal derivatives of the moisture). Some diagnostics can be done after the physics (CFU, XFU, DDH).
- Adiabatic dynamics: done under **CPG_DYN**: it includes the calculation of advections, linear terms.
  - Eulerian scheme: adds $2\Delta t\,[\mathcal{A}-\beta\mathcal{L}]$ at time t to the arrays **PGMVT1**, **PGMVT1S** and **PGFLT1**; stores $\Delta t\beta\mathcal{L}$ time $t-\Delta t$ in the array **P[X]SI**.
  - SL3TL scheme: increments the buffer (**ZSLBUF1AU** in **CPG**) with data which will be interpolated at the origin point O, and **PGMVT1**, **PGMVT1S** and **PGFLT1** with data which will be evaluated at the final point. For more details, see the documentation about the semi-Lagrangian scheme.
  - SL2TL scheme: does roughly the same thing as in the SL3TL scheme, and also updates the part of **PGMV** and **PGMVS** corresponding to the pointers **YDGMV%YT9%M[X]NL**.
- **CPG_END**: end of the unlagged grid-point calculations.
  - calls **GPPOPER**: updates the prognostic surface buffers with the $t+\Delta t$ surface variables.
  - redivides some quantities by a power of the mapping factor in order to retrieve reduced variables (**GPMPFC**).
- **CALL_SL**: called in a semi-Lagrangian scheme only; does the trajectory research, the interpolations, and adds the interpolated quantities. At the end of this stage, the arrays **PGMVT1**, **PGMVT1S** and **PGFLT1** contain a "provisional" value of $X(t+\Delta t)$ equal to the true $X(t+\Delta t)$ (to be computed after resolution of the semi-implicit scheme) plus an increment equal to $\beta\Delta t\mathcal{L}(t+\Delta t)-\beta\Delta t\mathcal{L}(t)$; the "provisional" value of $X(t+\Delta t)$ will be used as input data for the lagged physics.
- Lagged physics (currently ECMWF physics only): It uses as input the "provisional" value of $X(t+\Delta t)$. After computation of the physical fluxes and divergences of fluxes the physical tendencies are computed and added to the RHS of the equations (arrays **PGMVT1**, **PGMVT1S** and **PGFLT1**).
- **CPGLAG**:
  - does the part "one" of the Asselin temporal filter via **GPTF1**.
  - adds $\beta\Delta t\mathcal{L}(t)$ in order to get $X(t+\Delta t)+\beta\Delta t\mathcal{L}(t+\Delta t)$.
  - divides the wind components by M in order to retrieve the reduced wind components.

  at the end of **CPGLAG**:
  - the t data have been moved in the $t-\Delta t$ part of **PGMV**, **PGMVS** and **PGFL**.
  - the $t+\Delta t$ data are in **PGMVT1**, **PGMVT1S** and **PGFLT1**.
- At the end of **GP_MODEL**, **ZSLBUF2** is deallocated. **YRGMV%GMVT1** and **YRGMV%GMVT1S** contain $X(t+\Delta t)+\beta\Delta t\mathcal{L}(t+\Delta t)$ for GMV and GMVS variables. **YRGFL%GFLT1** contains $X(t+\Delta t)$ for GFL variables. Note that $X(t+\Delta t)$ is a provisional value on which the horizontal diffusion has not still been applied.

∗ **Direct spectral transformation:** It is done under the routine **TRANSDIRH** (**ETRANSDIRH** in LAM models) for the variable $X(t+\Delta t)+\beta\Delta t\mathcal{L}(t+\Delta t)$ (GMV and GMVS variables) or $X(t+\Delta t)$ (GFL variables). The grid-point input array are **YDGMV%GMVT1**, **YDGMV%GMVT1S** and **YDGFL%GFLT1**, the output spectral data is **YDSPEC**.

∗ **Spectral computations:** They are done under routine **SPCM** (**ESPCM** for LAM models) on attributes of **YRFIELDS%YRSPEC**. Some array copies are necessary for some applications, especially in distributed mode on several processors.

∗ **Lateral boundary coupling in LAM models:** It is done just before the direct Fourier transforms by the routine **ECOUPL1** called by **STEPO** on variables **YDGMV%GMVT1**, **YDGMV%GMVT1S** and **YDGFL%GFLT1**.

```
STEPO -> ECOUPL1 ->
  * ESC2R (temporal interpolation to get coupler at current timestep)
  * ESEIMPLS (add linear terms to coupler)
  * ESRLXT1 (grid-point relaxation)
```

Upper boundary coupling is done in grid-point calculations:

```
STEPO -> ECOUPL2 ->
  * ESC2R (temporal interpolation to get coupler at current timestep)
  * ESEIMPLS (add linear terms to coupler)
  * ESURLXT1 (grid-point relaxation)
```

Spectral nudging is done in spectral calculations:

```
STEPO -> ESPCM ->
  * ESPSC2R (temporal interpolation to get coupler at current timestep)
  * ESPCPL (spectral relaxation)
```

Reading coupling files, and transferring couplers in specific buffers is done in **ELSRW**.

```
CNT4 or DFI3 -> ELSRW -> ERLBC ->
  * SUEINIF (read coupler in file)
  * ELSWA3 ->
    - inverse transforms to get grid-point coupler.
    - intermediate memory transfers.
    - EPAK3W and EPAK3WSP (memory transfers in specific buffers).
```

19.7 Restriction to a dry model.

The implementation of the GFL structure data would allow, in theory, to run a dry adiabatic model (with physics it is more problematic, since the physical package uses the moisture in several parts; at least the radiation scheme, the gravity wave drag scheme and the vertical diffusion scheme could be used in a dry model). This configuration has indeed never been validated, and there probably remains some parts of the code where the use of moisture is hard coded (in some **GP...** routines especially).

It is possible to run an adiabatic dry model as follows:

- activate only q for GFL variables.
- set initial value $q = 0$ everywhere.

20 Tangent linear code.

20.1 Basics about TL code.

The coded linearised formulae will not be detailed in this documentation: the tangent linear code is taken line by line, thus it depend on the way the direct code is written. The main features of the tangent linear code are roughly the following ones:

- the tangent linear code looks like the direct code but with a differentiated shape. For example if the direct code contains an instruction like
$$Z = XY$$
the tangent linear code will contain at the same place the instruction
$$Z = XY_5 + X_5Y$$
using some trajectory variables subscripted by index "5". In the tangent linear code, X is a perturbation.
- There is additionally a "trajectory" code which is a copy of the direct code, but applied to the trajectory variables subscripted by index "5".

The tangent linear code performs the evolution of a "linear" perturbation; for example if the direct model discretizes the scalar 2D equation:
$$\frac{dX}{dt} = KX^2$$
where K is a constant, the tangent linear model discretizes the equation:
$$\frac{d[\delta X]}{dt} = 2KX[\delta X]$$
One can see that the evolving variable is now $[\delta X]$ and that the "trajectory" information provided by the direct model (here X) is also required. In our case $[\delta X]$ is assumed to be small enough to match the identity:
$$K[X + \Delta X]^2 - KX^2 \simeq 2KX[\delta X]$$
which means that $[\delta X]$ has to remain negligible compared to $2X$. At each time step the matrix defining the linear transformation is in our case $[2KX]$.

20.2 Tangent linear code for the 2D model: organigramme under STEPOTL for the Eulerian 2D model.

```
STEPOTL -> SCAN2MTL -> GP_MODEL_TL ->
  * CPG2TL ->
    - GPTF2
  * CPG2LAGTL ->
    - GPTF1
```

- **CPG2TL**: TL of unlagged grid-point computations.
- **CPG2LAGTL**: TL of lagged grid-point computations.
- **GPTF1**: first part of the Asselin temporal filter.
- **GPTF2**: second part of the Asselin temporal filter.

Communications between unlagged grid-point computations and lagged grid-point computations need a buffer using pointers of module **PTRSLB2**.

20.3 Tangent linear code for the 3D model: organigramme under STEPOTL for the Eulerian 3D model.

```
STEPOTL -> SCAN2MTL -> GP_MODEL_TL ->
  * CPG_DRV_TL -> CPGTL ->
    - Allocations
    - CPG5_GP (for trajectory) ->
      * some routines reading trajectory (RDPHTRAJM)
      * GPMPFC5 (for trajectory)
      * GP_SPV (for trajectory)
      * some GP.. routines computing intermediate grid-point quantities for trajectories.
    - CPG_GP_TL ->
      * GPTF2
      * GPMPFC (for model variables)
      * some GP..TL routines computing intermediate grid-point quantities for model variables.
      * GPINISLB
    - MF_PHYSTL (organigramme not detailed)
    - CPG_DYN_TL ->
      * CPEULDYNTL ->
        - some SI.. routines computing some linear terms used in SI scheme.
      * VDIFLCZTL (organigramme not detailed)
    - CPG_END_TL ->
      * GPMPFC (for model variables)
    - GPMPFC5 (for trajectory)
    - Deallocations
  * EC_PHYS_TL (organigramme not detailed)
  * CPGLAGTL ->
    - GPTF1
```

- **CPG_DRV_TL**: driver for TL of unlagged grid-point computations.
- **CPGTL**: TL of unlagged grid-point computations.
- **CPG_GP_TL**: TL of beginning of unlagged grid-point computations; reads $t-\Delta t$ data in buffers, computes some diagnostic grid-point quantities (call to some **GP...** routines), does multiplications by the mapping factor, applies the second part of the temporal filter.
- **CPG_DYN_TL**: TL of interface routine for unlagged part of the Eulerian dynamics and simplified Buizza physics.
- **CPG_END_TL**: TL of end of unlagged grid-point computations; writes data in buffer, first part of the temporal filter, divisions by the mapping factor.
- **CPGLAGTL**: TL of lagged grid-point computations.
- **GPMPFC**: multiplications/divisions by the mapping factor.
- **GPMPFC5**: trajectory version of **GPMPFC**.
- **GPTF1**: first part of the Asselin temporal filter.
- **GPTF2**: second part of the Asselin temporal filter.
- **VDIFLCZTL**: TL of simplified Buizza physics.
- **EC_PHYS_TL**: calls TL of lagged physics used at ECMWF.
- **MF_PHYSTL**: calls TL of unlagged physics used at METEO-FRANCE.
- **CPEULDYNTL**: computes the RHS of TL of Eulerian equations.

Communications between unlagged grid-point computations and lagged grid-point computations need a buffer using pointers of module **PTRSLB2**.

20.4 Testing the tangent linear code: configurations 501, 521 and 522.

∗ **Aim, and configurations:** The aim is to test that the tangent linear code provided in the code, is the genuine tangent linear code of the direct code and that there is no bug in it. Configuration 501 performs this test for the 3D primitive equation model or the 3D non-hydrostatic model. Configurations 521 and 522 perform this test respectively for the 2D shallow water model and for the 2D "vorticity" equation model.

∗ **Algorithm:** The algorithm will be described for a "generic" 2D variable X (that can be for example $\log(\Pi_{\rm s})$) but it can extend to a vector of variables containing all layers of different 3D variables and some additional 2D variables. The number of timesteps run is denoted by $N_{\rm stop}$.

- The test configuration reads two files and computes a perturbation. The two files can be identical. One of these files contains the initial state; for our generic 2D variable X the initial state is denoted by X_0. An initial perturbation $[\delta X]_0$ is also computed.
- First the model performs a direct integration of $N_{\rm stop}$ timesteps, like in the configurations 1, 201 or 202. That provides the different forecast steps of X: X_1, X_2, ..., $X_{N_{\rm stop}}$. This trajectory is stored in a buffer; it will be used to run the tangent linear model.
- The model then performs a tangent linear integration to compute the evolution of the perturbation $[\delta X]$; this integration uses the trajectory X_0 to $X_{N_{\rm stop}}$ and provides $[\delta X]_1$ to $[\delta X]_{N_{\rm stop}}$.
- The model then does a loop (index j_λ from 0 to 10); for each value of j_λ the model runs a direct integration with an initial state of $Y_0 = [X + 10^{-j_\lambda}\delta X]_0$ and forecasts Y_1 to $Y_{N_{\rm stop}}$. In spectral space, for each couple of wavenumbers (m, n), the following ratio is computed:

$$\mathtt{R}_{N_{\rm stop}}(m,n) = \frac{Y_{N_{\rm stop}}(m,n) - X_{N_{\rm stop}}(m,n)}{10^{-j_\lambda}[\delta X]_{N_{\rm stop}}(m,n)}$$

  (what is printed on the listing is a subset of $\mathtt{R}_{N_{\rm stop}}(m,n)$ taken at random). The test must show that $\mathtt{R}_{N_{\rm stop}}(m,n)$ converges towards 1 when j_λ increases.

∗ **Organigramme:** Only the basic features of the organigramme of configurations 501, 521 and 522 is given here (for example the canonical injections and the printing routines are omitted); for details about **STEPO** and **STEPOTL** see above.

- From **CNT0** to **TESTLI**: **CNT0** − >
  - **IFS_INIT**, **SU0YOMA** and **SU0YOMB** − > (0-level setup; organigramme not detailed)
  - **CTL1** − > **SU1YOM** − >
    - ∗ **SUVAZX** − > **SUSPEC** − > (reads the second file (name ICMSH[.]IMIN))
    - ∗ **SUSPEC** − > (reads the initial conditions file (name ICMSH[.]INIT))
  - **TESTLI** − > (see below; controls all the algorithm of testing the tangent linear code)
- Under **TESTLI**:
  - **SUPERT** − > (computes the perturbation $[\delta X]_0$).
  - **CNT3** − > **CNT4** − > **STEPO** − > (direct model integration which computes X_1, X_2, ..., $X_{N_{\rm stop}}$).
  - **CNT3TL** − > **CNT4TL** − >
    - ∗ **STEPO** − > **TRANSINVH** − > (converts the trajectory X_1 to $X_{N_{\rm stop}}$ into grid-point space).
    - ∗ **STEPOTL** − > (runs the tangent linear model to provide $[\delta X]_1$ to $[\delta X]_{N_{\rm stop}}$).
  - in a loop on j_λ from 0 to 10:
    - ∗ **CNT3** − > **CNT4** − > **STEPO** − > (direct model integration which computes Y_1, Y_2, ..., $Y_{N_{\rm stop}}$ from the perturbed initial state $Y_0 = [X + 10^{-j_\lambda}\delta X]_0$)
    - ∗ computes the ratio $\mathtt{R}_{N_{\rm stop}}$ and prints it for a subset of wavenumbers (m, n).

21 Adjoint code.

21.1 Basics about AD code.

One starts from the linear tangent code and “adjoints” the code: for example a matricial expression of the following type

$$Y = MX$$

leads to the expression

$$X = X + {}^{t}MY$$

in the adjoint code.

21.2 Adjoint code for the 2D model: organigramme under STEPOAD for the Eulerian 2D model.

```
STEPOAD -> SCAN2MAD -> GP_MODEL_AD ->
  * CPG2LAGAD ->
    - GPTF1AD
  * CPG2AD ->
    - GPTF2AD
```

- **CPG2AD**: AD of unlagged grid-point computations.
- **CPG2LAGAD**: AD of lagged grid-point computations.
- **GPTF1AD**: AD of first part of the Asselin temporal filter.
- **GPTF2AD**: AD of second part of the Asselin temporal filter.

Communications between unlagged grid-point computations and lagged grid-point computations need a buffer using pointers of module **PTRSLB2**.

21.3 Adjoint code for the 3D model: organigramme under STEPOAD for the Eulerian 3D model.

```
STEPOAD -> SCAN2MAD -> GP_MODEL_AD ->
  * CPGLAGAD ->
    - GPTF1AD
  * EC_PHYS_AD (organigramme not detailed)
  * CPG_DRV_AD -> CPGAD ->
    - Allocations
    - CPG5_GP (for trajectory) ->
      * some routines reading trajectory (RDPHTRAJM).
      * GPMPFC5 (for trajectory)
      * GP_SPV (for trajectory)
      * some GP.. routines computing intermediate grid-point quantities for trajectories.
    - CPG_END_AD ->
      * GPMPFC
    - CPG_ZERO_AD
    - CPG_DYN_AD ->
      * VDIFLCZAD (organigramme not detailed)
      * CPEULDYNAD ->
        - some SI.. routines computing some linear terms used in SI scheme.
    - MF_PHYSAD (organigramme not detailed)
    - CPG_GP_AD ->
      * some GP..AD routines computing intermediate grid-point quantities for model variables.
      * GPMPFC (for model variables)
      * GPTF2AD
    - GPMPFC5 (for trajectory)
```

- **CPG_DRV_AD**: driver of AD of unlagged grid-point computations.
- **CPGAD**: AD of unlagged grid-point computations.
- **CPG_GP_AD**: AD of beginning of unlagged grid-point computations; reads $t - \Delta t$ data in buffers, computes some diagnostic grid-point quantities (call to some **GP...** routines), does multiplications by the mapping factor, applies the second part of the temporal filter.
- **CPG_DYN_AD**: AD of interface routine for unlagged part of the Eulerian dynamics and simplified Buizza physics.
- **CPG_END_AD**: AD of end of unlagged grid-point computations; writes data in buffer, first part of the temporal filter, divisions by the mapping factor.
- **CPG_ZERO_AD**: zeroing some arrays before adjointing.
- **CPGLAGAD**: AD of lagged grid-point computations.
- **GPMPFC**: multiplications/divisions by the mapping factor.
- **GPMPFC5**: trajectory version of **GPMPFC**.
- **GPTF1AD**: AD of first part of the Asselin temporal filter.
- **GPTF2AD**: AD of second part of the Asselin temporal filter.
- **VDIFLCZAD**: AD of simplified Buizza physics.
- **EC_PHYS_AD**: calls AD of lagged physics used at ECMWF.
- **MF_PHYSAD**: calls AD of unlagged physics used at METEO-FRANCE.
- **CPEULDYNAD**: computes the RHS of AD of Eulerian equations.

Communications between unlagged grid-point computations and lagged grid-point computations need a buffer using pointers of module **PTRSLB2**.

21.4 Testing the adjoint code: configurations 401, 421 and 422.

$*$ **Aim, and configurations:** The aim is to test that the adjoint code provided in the code, is the genuine adjoint code of the tangent linear code and that there is no bug in it. Configuration 401 performs this test for the 3D primitive equation model or the 3D non-hydrostatic model. Configurations 421 and 422 perform this test respectively for the 2D shallow water model and for the 2D "vorticity" equation model.

$*$ **Algorithm:** The algorithm will be described for a "generic" 2D variable X (that can be for example $\log(\Pi_s)$) but it can extend to a vector of variables containing all layers of different 3D variables and some additional 2D variables. The number of timesteps run is denoted by N_{stop}.

- The test configuration reads two files and computes two perturbations. The two files can be identical. One of these files contains the initial state; for our generic 2D variable X the initial state is denoted by X_0. An initial perturbation $[\delta X]_0$ and a "final" perturbation $[\delta Y]_{N_{\text{stop}}}$ are also computed.
- First the model performs a direct integration of N_{stop} timesteps, like in the configurations 1, 201 or 202. That provides the different forecast steps of X: X_1, X_2, ..., $X_{N_{\text{stop}}}$. This trajectory is stored in a buffer; it will be used to run the tangent linear and the adjoint models.
- The model then performs a tangent linear integration to compute the evolution of the perturbation $[\delta X]$; this integration uses the trajectory X_0 to $X_{N_{\text{stop}}}$ and provides $[\delta X]_1$ to $[\delta X]_{N_{\text{stop}}}$.
- The model then performs an adjoint integration to compute the "adjoint" evolution of the perturbation $[\delta Y]$; this integration uses the trajectory X_0 to $X_{N_{\text{stop}}}$, starts from $[\delta Y]_{N_{\text{stop}}}$ and provides $[\delta Y]_{N_{\text{stop}}-1}$ to $[\delta Y]_0$.
- If one denotes by `M` the matricial operator verifying:

  $$[\delta X]_{N_{\text{stop}}} = \mathtt{M}[\delta X]_0$$

  (this is a product of N_{stop} matrices), the two following scalar products are computed:

  $$S1 = \left\langle \mathtt{M}[\delta X]_0, [\delta Y]_{N_{\text{stop}}} \right\rangle$$

  $$S2 = \left\langle [\delta X]_0, \mathtt{M}^T[\delta Y]_{N_{\text{stop}}} \right\rangle$$

  These scalar products are defined in spectral space. These scalar products are compared; they must be identical if the provided adjoint code is correct relative to the tangent linear code.

$*$ **Organigramme:** Only the basic features of the organigramme of configurations 401, 421 and 422 is given here (for example the canonical injections and the printing routines are omitted); for details about **STEPO**, **STEPOTL** and **STEPOAD** see above.

- From **CNT0** to **TESADJ**: **CNT0** $- >$
  - **IFS_INIT**, **SU0YOMA** and **SU0YOMB** $- >$ (0-level setup; organigramme not detailed)
  - **CAD1** $- >$ **SU1YOM** $- >$
    - **SUVAZX** $- >$ **SUSPEC** $- >$ (reads the second file (name ICMSH[.]IMIN))
    - **SUSPEC** $- >$ (reads the initial conditions file (name ICMSH[.]INIT))
  - **TESADJ** $- >$ (see below; controls all the algorithm of testing the adjoint code)
- Under **TESADJ**:
  - **SUPERT** $- >$ (computes the perturbations $[\delta X]_0$ and $[\delta Y]_{N_{\rm stop}}$).
  - **CNT3** $- >$ **CNT4** $- >$ **STEPO** $- >$ (direct model integration which computes X_1, X_2, ..., $X_{N_{\rm stop}}$).
  - **CNT3TL** $- >$ **CNT4TL** $- >$
    - **STEPO** $- >$ **TRANSINVH** $- >$ (converts the trajectory X_1 to $X_{N_{\rm stop}}$ into grid-point space).
    - **STEPOTL** $- >$ (runs the tangent linear model to provide $[\delta X]_1$ to $[\delta X]_{N_{\rm stop}}$).
  - **CNT3AD** $- >$ **CNT4AD** $- >$
    - **STEPO** $- >$ **TRANSINVH** $- >$ (converts the trajectory X_1 to $X_{N_{\rm stop}}$ into grid-point space).
    - **STEPOAD** $- >$ (runs the adjoint model from the perturbation $[\delta Y]_{N_{\rm stop}}$ in order to provide $[\delta Y]_{N_{\rm stop}-1}$ to $[\delta Y]_0$).
  - Checks comparing the two scalar products $S1$ and $S2$ that the adjoint code is really the adjoint of the tangent linear code.

22 Some distributed memory features.

Note that some variables referenced below may be attributes of structures.

∗ Message passing division into processors, structure of grid-point data.

The total number of processors is **NPROC**. There are two levels of parallelisation. Distribution on these two levels is different in the different spaces of calculations.

NPROC=NPRGPNS∗NPRGPEW. The total number of processors involved in the A-level parallelisation is **NPRGPNS**. The total number of processors involved in the B-level parallelisation is **NPRGPEW**.

One 2D model field has **NGPTOTG** points divided into **NPRGPNS*NPRGPEW** sets of **NGPTOT** points treated by each processor. **NGPTOT** may be processor-dependent (with very small variations): the maximum value of **NGPTOT** is **NGPTOTMX**. A processor treats complete columns in grid-point space.

For one given processor, the **NGPTOT** points are divided into packets of length **NPROMA** (the useful number of values in each packet is lower or equal than **NPROMA**). **NPROMA** is identical for all processors. There are **NGPBLKS** blocks of **NPROMA** packets:

$$\mathbf{NGPBLKS} = int[(\mathbf{NGPTOT} + \mathbf{NPROMA} - 1)/\mathbf{NPROMA}]$$

A **NPROMA**-packet does not always contain a set of complete latitudes.

There are necessary transpositions (reorganisation of data) between grid point computations and Fourier transforms because Fourier transforms need complete latitudes.

∗ Transmission of data used in several "JKGLO" blocks in the grid-point computations.

There are several "JKGLO" blocks in the grid-point calculations. Calculations start for the first block, for the **NGPBLKS** packets of **NPROMA** points. Calculations for the second block follow then, and so on. Some data may be needed in more than one block.

Some of these data can be stored in the arrays **GMVT1**, **GMVT1S**, **GFLT1**, **GMV**, **GMVS**, **GFL**, but some other more specific data require a special buffer, allocated and deallocated in **GP_MODEL** (local name is **ZSLBUF2**, dummy name is generally **PB2** under **CPG**, **CALL_SL** and **CPGLAG**). This array is used both in the Eulerian and semi-Lagrangian schemes. Each individual quantity stored in **ZSLBUF2** can be retrieved by a pointer with a name looking like **MSLB2...** (in **PTRSLB2**). Pointers starting by letters **MSLB2...** are pre-computed in the set-up routine **SUSLB**.

For example the "t" linear contribution which is used in the semi-implicit scheme and which appears in the RHS of temperature equation, can be stored in **ZSLBUF2**(.,**MSLB2TSI**:**MSLB2TSI**+**NFLEVG**-1,.); it is currently computed under **CPG_DYN**, and needed for example under **CPGLAG** which is in a different **JKGLO** block. No halo computation is necessary for this buffer.

∗ Case LEQ_REGIONS=T.

This case is relevant only when **NPRGPEW**>1 (B-level parallelisation at least in the grid-point calculations), and use a global model with a reduced Gaussian grid. This is an optimised version of the LEQ_REGIONS=F case which is well designed for reduced Gaussian grid and it improves the load balance in this case. A comprehensive description can be found in (Mozdzynski, 2006). To sum-up, we can say that:

- the A-level grid-point distribution splits the Earth into **N_REGIONS_NS** bands. **N_REGIONS_NS** can be slightly different from **NPRGPNS**.
- for each band *jroca*, the B-level grid-point distribution splits the band into **N_REGIONS**(*jroca*) zones: the minimum value of **N_REGIONS** is at the poles of the computational sphere (equal to 1 in the examples provided by Mozdzynski); the maximum value of **N_REGIONS** is at the equator of the computational sphere and this maximum is equal to **N_REGIONS_EW**. The meridian variations of **N_REGIONS** are highly correlated to those of **NLOENG**.
- In the examples provided by Mozdzynski, **NPRGPNS=NPRGPEW=NPRTRW=NPRTRV** and we notice that **N_REGIONS_NS** is slightly below **NPRGPNS**, and that **N_REGIONS_EW** is slightly below 2∗**NPRGPEW**.

When **LEQ_REGIONS**=F, variables **N_REGIONS_NS**, **N_REGIONS** and **N_REGIONS_EW** are still used but in this case:

- **N_REGIONS_NS=NPRGPNS**.
- **N_REGIONS=NPRGPEW** everywhere.
- **N_REGIONS_EW=NPRGPEW**.

23 Specific Eulerian model variables in modules and namelists.

These modules are auto-documented so description of each variable is provided in the code source. We can recall here the most important variables to know for each module:

- **PARDIM** (parameter dimensions).
- **PTRSLB2** (pointers for arrays to communicate information between the different blocks in the grid-point calculations).
- **PTRGPPC** (pointers for grid point array (GPPC) for PC schemes).
- **YEMLBC..** modules for lateral or vertical coupling and spectral nudging: all variables. Some of these variables are in namelists **NEMELBC0A** and **NEMELBC0B**.
- **YEMDYN** (LAM model dynamics): LESIDG, RTHRESIDG. RTHRESIDG is in namelist **NEMDYN**.
- **YOMARG** (0-level control, former command line) and **YOMCT0** (0-level control):
  - All NAMARG variables: for ex. NCONF, LELAM, LECMWF, LSLAG, NSUPERSEDE.
  - LR3D, LR2D, LRSHW, LRVEQ.
  - LTWOTL.
  - LREGETA, LVFE_REGETA.
  - LNHDYN, LNHEE, LNHQE.
  - LRPLANE.
  - LAROME.
  - LCALLSFX, LSFXLSM.
  - LSFORC, LSFORCS.

  Some of these variables are in namelist **NAMCT0**.
- **YOMCVER** (vertical finite element discretisation keys). Some of these variables are in namelist **NAMCVER**.
- **YOMDIM**, **YOMDIMV** and **YOMDIMF** (dimensioning): most of variables. Some of these variables are in namelist **NAMDIM**.
- **YOMDYNA** (adiabatic dynamics: first part):
  - LAPRXPK, NDLNPR, RHYDR0 (vertical discretisation).
  - L3DTURB (3D turbulence).
  - LSLDIA, LRPRSLTRJ (diagnostics).
  - LRUBC.
  - NPDVAR, NVDVAR, ND4SYS, LNHX, LNHXDER, L_RDRY_VD, LNHEE_SVDLAPL_FIRST (NH model).
  - LGWADV, NGWADVSI, LRDBBC (treatment of vertical divergence equation in NH model).

  Some of these variables are in namelist **NAMDYNA**.
- **YOMDYN** (adiabatic dynamics: second part):
  - REPS1, REPS2, REPSM1, REPSM2, REPSP1 (Asselin filter).
  - RKRF, NMAXLEVRF, RRFZ1, RRFPLM (Rayleigh friction).
  - RTEMRB, NRUBC (upper radiative boundary condition).
  - LSIDG, BETADT, RBT, RBTS2, NITERHELM, LIMPF (semi-implicit scheme).
  - REFGEO, SIPR, SITR, SITRA, SITRUB, SIPRUB, SITIME, SIRPRG, SIRPRN (reference values used in the semi-implicit scheme).
  - NSITER, NCURRENT_ITER, LRHDI_LASTITERPC (predictor-corrector scheme).
  - NCOMP_CVGQ, LDRY_ECMWF.
  - LRFRIC, LRFRICISOTR (Rayleigh friction).

  Some of these variables are in namelist **NAMDYN**.
- **YOMGPPCB** (contains buffer for gridpoint workfile needed for PC schemes).
- **YOMGPPB** (contains buffer for the externalised SURFEX surface scheme).
- **YOMMP0** and **YOMMP** (distributed memory environment, see documentation (IDDM) for more details).
- **YOMRIP0** (model date). Some of these variables are in namelist **NAMRIP0**.
- **YOMRIP** (timestep dependent variables), in particuliar TSTEP, TDT, CSTOP and NSTOP. Some of these variables are in namelist **NAMRIP**.

- **YOMSTA** (constants for standard atmosphere).
- **TYPE_FADS** (to define the Field Arpege Descriptors).
- Modules for trajectory stored for TL and AD models:
  - **TRAJECTORY_MOD** (manages trajectory for TL and AD runs in a unified way).
  - **TRAJ_MAIN_MOD** (manages main trajectory).
  - **TRAJ_PHYSICS_MOD** (manages physics trajectory).
  - **TRAJ_SEMILAG_MOD** (manages semi-Lagrangian trajectory).
  - **TRAJ_SURFACE_MOD** (manages surface trajectory).
- Modules for geometry:
  - **YOMGSGEOM** (geographic space grid-point geometry)
  - **YOMCSGEOM** (computational space grid-point geometry)
  - **YOMOROG** (orography)
  - **YOMGEM** (horizontal geometry): some variables are in namelist **NAMGEM**.
  - **SPGEOM_MOD** (spectral geometry).
  - **YOMVERT** (vertical geometry and VFE operators).
  - **YOMVV1** (namelist variables for vertical geometry): variables are in namelist **NAMVV1**.
  - **YEMGEO** (LAM model geometry): some variables are in namelist **NEMGEO**.
  - **YEMLBC_GEO** (LAM model geometry for LBC).
  - **TYPE_GEOMETRY**, **TYPE_SPGEOM**.
- Modules for GMV dataflow:
  - **YOMGMV** (grid-point arrays for GMV fields).
  - **TYPE_GMVS** (derived types for GMV).
  - **GMV_SUBS_MOD** (contains encapsulated routines for GMV management).
- Modules for GFL dataflow:
  - **YOMGFL** (grid-point arrays for GFL fields).
  - **YOM_YGFL** (descriptors of GFL arrays). Some of these variables are in namelist **NAMGFL**.
  - **GFL_SUBS_MOD** (contains encapsulated routines for GFL management).
- Modules for surface dataflow:
  - **SURFACE_FIELDS_MIX** (surface dataflow).
- **YOMSP** and **YOMSP5** (spectral arrays).

We give there a sum-up of the main modules and structures belonging to ifs_init part, geometry part, and model structures. This is important to understand what is OOPS-compliant.

- Part ifs_init: the set-up is done before calling **SUGEOMETRY**; content of these modules should not be modified by **SUGEOMETRY** or after calling **SUGEOMETRY**. List contains at least:
  - **YOMMP0**
  - **YOMGSTATS**
  - **YOMLUN**
  - **YOMMSC**
  - **YOMCST**
  - **YOMARG**
  - **YOMCT0** and **YEMCT0**
  - **ENKF_MIX**
  - **YOMJFH**
  - **YOMDYNA**
  - **YOMCVER**
  - **YOMDYNCORE**
  - **YEMLBC_INIT**
  - **YOMVAR**
  - **YOMRIP0**

  - **YOMINI**
  - **YOMIOP**
  - **YOMTIM**
  - **YOMVWRK**
  - **YOMTRANS**
  - **INTDYN_MOD**
- Geometry: sets-up the following structures and modules; some of these variables are attributes of variable **YRGEOMETRY**; content of these variables should not be modified after the call to **SUGEOMETRY**.
  - **YOMVV1**: not encapsulated, vertical geometry only
  - **YOMDIM**
  - **YOMDIMV**
  - **YEMDIM**
  - **YOMVERT**
  - **YOMGEM**
  - **YEMGEO**
  - **YEMLBC_GEO**
  - **YOMMP**
  - **YEMMP**
  - **YOMLEG**
  - **YOMLAP**
  - **YEMLAP**
  - **YOMSTA**
  - **SPGEOM_MOD**
  - **YOMCSGEOM**
  - **YOMGSGEOM**
  - **YOMVSPLIP**
  - **YOMVSLETA**
  - **YOMHSLMER**
- Model structures and fields structures: all what is computed after **SUGEOMETRY**. For example, content of the following modules:
  - **YOMGMV**
  - **YOMGFL**
  - **SURFACE_FIELDS_MIX**
  - **YOMSP**
  - **YOMSP5**
  - **YOMTRAJ**
  - **YEMLBC_MODEL**
  - **YEMLBC_FIELDS**
  - **FIELDS_MOD**

  To be OOPS-compliant, the code should pass these variables via dummy arguments (and never via use of modules) in routines called under STEPO/AD/TL.

24 Bibliography.

24.1 Publications.

- Bénard, P., 2003: Stability of semi-implicit and iterative centred implicit time discretization for various equation systems used in NWP. *Mon. Wea. Rev.*, **131**, 2479-2491.
- Bénard, P., J. Vivoda, J. Mašek, P. Smolíková, K. Yessad, C. Smith, R. Brozkova, and J.-F. Geleyn, 2010: Dynamical kernel of the ALADIN-NH spectral limited-area model: revised formulation and sensitivity experiments. *Quart. J. Roy. Meteor. Soc.*, **136**, 155-169.
- Brousseau, P., Y. Seity, D. Ricard and J. Léger, 2016: Improvement of the forecast of convective activity from the AROME-France system. *Quart. J. Roy. Meteor. Soc.*, **142**, 2231-2243.
- Bubnová, R., G. Hello, P. Bénard, and J.F. Geleyn, 1995: Integration of the fully elastic equations cast in the hydrostatic pressure terrain-following coordinate in the framework of the ARPEGE/Aladin NWP system. *Mon. Wea. Rev.*, **123**, 515-535.
- Courtier, Ph., and J. F. Geleyn, 1988: A global model with variable resolution. Application to the shallow-water equations. *Q. J. R. Meteorol. Soc.*, **114**, 1321-1346.
- Courtier, Ph., C. Freydier, J. F. Geleyn, F. Rabier and M. Rochas, 1991: The ARPEGE project at METEO-FRANCE. ECMWF Seminar Proceedings 9-13 September 1991, Volume II, 193-231.
- Davies, H.C., 1976: A lateral boundary formulation for multi-level prediction models. *Quart. J. Roy. Meteor. Soc.*, **102**, 405-418.
- Geleyn, J.F., and R. Bubnová, 1995: The fully elastic equations cast in hydrostatic pressure coordinate: accuracy and stability aspects of the scheme as implemented in ARPEGE/Aladin. *Atm. Ocean, special issue memorial André Robert.*
- Girard, C., and Y. Delage, 1990: Stable schemes for nonlinear vertical diffusion in atmospheric circulation models. *Mon. Wea. Rev.*, **118**, 137-146.
- Gravel, S., and A. Staniforth, 1992: Variable resolution and robustness. *Mon. Wea. Rev.*, **120**, 2633-2640.
- Laprise, R., 1992: The Euler equations of motion with hydrostatic pressure as an independent variable. *Mon. Wea. Rev.*, **120**, 197-207.
- Laprise, R., 1992: The resolution of global spectral models. *Bulletin American Meteor. Society*, **73**, 1453-1454.
- Pailleux, J., J.-F. Geleyn, M. Hamrud, Ph. Courtier, J.-N. Thépaut, F. Rabier, E. Andersson, D. Burridge, A. Simmons, D. Salmond, R. El Khatib, C. Fischer, 2014: Twenty-five years of IFS/ARPEGE. *ECMWF Newsletter nr* **141**, 22-30.
- Ritchie, H., and M. Tanguay, 1996: A comparison of spatially-averaged Eulerian and semi-Lagrangian treatments of mountains. *Mon. Wea. Rev.*, **124**, 167-181.
- Robert, A., 1981: A stable numerical integration scheme for the primitive meteorological equations. Atmos. Ocean, **19**, 35-46.
- Untch, A., and M. Hortal, 2004: A finite-element scheme for the vertical discretization of the semi-Lagrangian version of the ECMWF forecast model. *Q. J. R. Meteorol. Soc.*, **130**, 1505-1530.
- White, A. A., 2000: A view of the equations of meteorological dynamics and various approximations. *Forecasting Research Scientific Paper nr*, **58**, 106pp.
- White, A. A., B. J. Hoskins, I. Roulstone and A. Staniforth, 2005: Consistent approximate models of the global atmosphere: shallow, deep, hydrostatic, quasi-hydrostatic and non-hydrostatic. *Q. J. R. Meteorol. Soc.*, **131**, 2081-2107.
- Williamson, D. L., and J. G. Olson, 1994: Climate simulations with a semi-Lagrangian version of the NCAR community climate model. *Mon. Wea. Rev.*, **122**, 1594-1610.

24.2 Some internal notes and other ARPEGE notes.

- (TDECDYN) 2017: IFS technical documentation (CY43R3). Part III: dynamics and numerical procedures. Available at "https://software.ecmwf.int/wiki/display/IFS/Official+IFS+Documentation".
- (TDECTEC) 2017: IFS technical documentation (CY43R3). Part VI: technical and computational procedures. Available at "https://software.ecmwf.int/wiki/display/IFS/Official+IFS+Documentation".
- (IDAB) Bénard, P., 2008: On the design of the vertical coordinate η. Internal note (7pp).
- (IDNHPB) Bénard, P., 2013: Scientific documentation for ALADIN NH model (version 3.1). Internal note (94pp), available on "http://www.umr-cnrm.fr/gmapdoc/".
- (IDPRLAM) Bénard, 2011: "Rotated/Tilted Mercator" geometry in ALADIN. Internal note, 26pp, available on "http://www.umr-cnrm.fr/gmapdoc/".

- Bénard, P., 2004: Study of the VFE discretisation in view of NH modelling. Internal note, 37pp.
- De Boor, C., 2001: A practical guide to splines. *Applied mathematical sciences 27, Springer-Verlag: New York.*
- (IDPHYE) Gérard, L., 2001: Physical parameterizations in ARPEGE/ALADIN (CY23T1/AL14 ?). Internal note, available on "http://www.umr-cnrm.fr/gmapdoc/". (this documentation has been written in 2000 for CY23T1/AL14 and some parts seem have been updated for more recent cycles).
- (IDRDFL) Hamrud, M., 2003: Revised data flow in IFS/ARPEGE (internal note).
- (IDPRAL) Janoušek, M., 2001: New port of the new geographical routines to ALADIN/ARPEGE (internal note).
- (IDMOAL) Joly, A., 1992: ARPEGE/ALADIN: adiabatic model equations and algorithm (internal note, 56 pp).
- (IDEQR) Mozdzynski, G., 2006: A new partitioning approach for IFS. Internal note, 6pp.
- (IDDDH) Piriou, J.-M., 2014: Diagnostics in Horizontal Domains (DDH). Variables and budget equations, in horizontal mean. ARPEGE, ALADIN and AROME models. Guide for users and developpers. Internal note, 75pp, available on "http://www.umr-cnrm.fr/gmapdoc/".
- (NTA30) Rochas, M., et Ph. Courtier, 1992: La méthode spectrale en météorologie. Note de travail ARPEGE numéro **30**, 58pp.
- (IDVNH1) Smolíková, P., 2001: New strategies for non-hydrostatic temporal scheme. Internal note.
- (IDVNH2) Smolíková, P., 2002: New NH variables: d_4 in three dimensions (in the cycle CY25T1). Internal note.
- Smolíková, P., and Jozef Vivoda, 2013: Finite elements used in the vertical discretization of the fully compressible forecast model ALADIN-NH. *ALADIN - HIRLAM Newsletter no* **1**, 31-47.
- Untch, A., and M. Hortal, 2001: A finite element scheme for the vertical discretization in the semi-Lagrangian version of the ECMWF forecast model. (internal note, 9 pp).
- (IDMES) Yessad, K., 2012: Definition of mesh-size in spectral models: application to ARPEGE/IFS, ALADIN, AROME. Internal note, 5pp.
- (IDBAS) Yessad, K., 2018: Basics about ARPEGE/IFS, ALADIN and AROME in the cycle 46t1 of ARPEGE/IFS (internal note).
- (IDSL) Yessad, K., 2018: Semi-Lagrangian computations in the cycle 46t1 of ARPEGE/IFS (internal note).
- (IDSI) Yessad, K., 2018: Semi-implicit spectral computations in the cycle 46t1 of ARPEGE/IFS (internal note).
- (IDDH) Yessad, K., 2018: Horizontal diffusion in the cycle 46t1 of ARPEGE/IFS (internal note).
- (IDTS) Yessad, K., 2018: Spectral transforms in the cycle 46t1 of ARPEGE/IFS (internal note).
- (IDDM) Yessad, K., 2018: Distributed memory features in the cycle 46t1 of ARPEGE/IFS (internal note).
- (IDNGFL) Yessad, K., 2018: User's guide to add new GFL variables or new GFL attributes in ARPEGE/IFS, ALADIN, AROME: cycle 46t1. Internal note, 10pp, available on the internet server "http://www.umr-cnrm.fr/gmapdoc/".
- (IDNSUR) Yessad, K., 2018: User's guide to add new surface variables or new surface attributes in ARPEGE/IFS, ALADIN, AROME: cycle 46t1. Internal note, 8pp, available on the internet server "http://www.umr-cnrm.fr/gmapdoc/".
- (IDLAM) Zagar, M., and C. Fischer, 2007: The ARPEGE/ALADIN Tech'Book: Implications of LAM aspects on the global model code for CY33/AL33. Internal note, 31pp, available on the internet server "http://www.umr-cnrm.fr/gmapdoc/".

Appendix 1: vertical integrals and their discretisation.

Model dynamics uses different type of vertical integrals: VFD (vertical finite difference) and VFE (vertical finite element) discretisations will be given.

a) S type vertical integrals (from the top).

Expression of integral (used for example in **GPSKAP**, **SITNU**, **GPCTY**) is:

$$[\mathbf{S}Z]_\eta = \frac{1}{\Pi_\eta} \int_{\eta'=0}^{\eta'=\eta} \frac{\partial \Pi}{\partial \eta} Z d\eta'$$

Discretisation:

- VFD discretisation at full level l: $[\mathbf{S}Z]_l = \alpha_l Z_l + \frac{\delta_l}{[\Delta\Pi]_l} \sum_{k=1}^{k=l-1} [\Delta\Pi]_k Z_k$
- VFE discretisation at full level l: $[\mathbf{S}Z]_l = \frac{\delta_l}{[\Delta\Pi]_l} [\mathcal{R}_{\mathrm{inte}}]_{(top,l)} \left\langle \frac{[\Delta\Pi]Z}{[\Delta\eta]} \right\rangle = \frac{1}{\Pi_l} [\mathcal{R}_{\mathrm{inte}}]_{(top,l)} \left\langle \frac{[\Delta\Pi]Z}{[\Delta\eta]} \right\rangle$
- Discretisation at half level $\bar{l}$, for both VFD and VFE: $[\mathbf{S}Z]_{\bar{l}} = \frac{1}{\Pi_{\bar{l}}} \sum_{k=1}^{k=l} [\Delta\Pi]_k Z_k$

VFD discretisation matches the following identity:

$$[\mathbf{S}Z]_l = \frac{\alpha_l \Pi_{\bar{l}}}{[\Delta\Pi]_l} [\mathbf{S}Z]_{\bar{l}} + \frac{(\delta_l - \alpha_l)\Pi_{\overline{l-1}}}{[\Delta\Pi]_l} [\mathbf{S}Z]_{\overline{l-1}}$$

which can be approximated as follows:

$$[\mathbf{S}Z]_l \simeq \frac{\alpha_l}{\delta_l} [\mathbf{S}Z]_{\bar{l}} + \frac{(\delta_l - \alpha_l)}{\delta_l} [\mathbf{S}Z]_{\overline{l-1}}$$

b) Extension to $[\mathbf{S}([\partial B/\partial\Pi]Z)]$.

Expression of integral (used for example in **GPCTY**, **GNHQE_XXD**) is:

$$\left[\mathbf{S}\left(\frac{\partial B}{\partial \Pi} Z\right)\right]_\eta = \frac{1}{\Pi_\eta} \int_{\eta'=0}^{\eta'=\eta} \frac{dB}{d\eta} Z d\eta'$$

Discretisation:

- VFD discretisation at full level l: $\left[\mathbf{S}\left(\frac{\partial B}{\partial \Pi} Z\right)\right]_l = \alpha_l \frac{[\Delta B]_l}{[\Delta\Pi]_l} Z_l + \frac{\delta_l}{[\Delta\Pi]_l} \sum_{k=1}^{k=l-1} [\Delta B]_k Z_k$
- VFE discretisation at full level l: $\left[\mathbf{S}\left(\frac{\partial B}{\partial \Pi} Z\right)\right]_l = \frac{\delta_l}{[\Delta\Pi]_l} [\mathcal{R}_{\mathrm{inte}}]_{(top,l)} \left\langle \frac{[\Delta B]Z}{[\Delta\eta]} \right\rangle$
- Discretisation at half level $\bar{l}$, for both VFD and VFE: $\left[\mathbf{S}\left(\frac{\partial B}{\partial \Pi} Z\right)\right]_{\bar{l}} = \frac{1}{\Pi_{\bar{l}}} \sum_{k=1}^{k=l} [\Delta B]_k Z_k$

c) N type vertical integrals on the whole column.

This is an extension of $[\mathbf{S}Z]_\eta$ to $\eta = 1$. Expression of integral (used for example in **GPCTY**, **SITNU**) is:

$$[\mathbf{N}Z] = \frac{1}{\Pi_{\mathrm{s}}} \int_{\eta'=0}^{\eta'=1} \frac{\partial \Pi}{\partial \eta} Z d\eta'$$

Discretisation:

- VFD discretisation: $[\mathbf{N}Z] = \frac{1}{\Pi_{\mathrm{s}}} \sum_{k=1}^{k=L} [\Delta\Pi]_k Z_k$
- VFE discretisation: $[\mathbf{N}Z] = \frac{1}{\Pi_{\mathrm{s}}} [\mathcal{R}_{\mathrm{inte}}]_{(top,surf)} \left\langle \frac{[\Delta\Pi]Z}{[\Delta\eta]} \right\rangle$

d) Extension to $[\mathbf{N}([\partial B/\partial\Pi]Z)]$.

This is an extension of $[\mathbf{S}([\partial B/\partial\Pi]Z)]_\eta$ to $\eta = 1$. Expression of integral (used for example in **GPCTY**) is:

$$\left[\mathbf{N}\left(\frac{\partial B}{\partial \Pi} Z\right)\right] = \frac{1}{\Pi_{\mathrm{s}}} \int_{\eta'=0}^{\eta'=1} \frac{dB}{d\eta} Z d\eta'$$

Discretisation:

- VFD discretisation: $\left[\mathbf{N}\left(\frac{\partial B}{\partial \Pi} Z\right)\right] = \frac{1}{\Pi_{\mathrm{s}}} \sum_{k=1}^{k=L} [\Delta B]_k Z_k$
- VFE discretisation: $\left[\mathbf{N}\left(\frac{\partial B}{\partial \Pi} Z\right)\right] = \frac{1}{\Pi_{\mathrm{s}}} [\mathcal{R}_{\mathrm{inte}}]_{(top,surf)} \left\langle \frac{[\Delta B]Z}{[\Delta\eta]} \right\rangle$

e) G type vertical integrals (from the bottom).

Expression of integral (used for example in **GPGEO** to compute gz, **GPGW** to compute gw, **SIGAM**) is:

$$[\mathbf{G}Z]_\eta = \int_{\eta'=\eta}^{\eta'=1} \frac{1}{\Pi}\frac{\partial \Pi}{\partial \eta} Z d\eta' = -\int_{\Pi'=\Pi_s}^{\Pi'=\Pi} Z\frac{d\Pi'}{\Pi'}$$

Discretisation:

- VFD discretisation at full level l: $[\mathbf{G}Z]_l = \alpha_l Z_l + \sum_{k=l+1}^{k=L} \delta_k Z_k$
- VFE discretisation at full level l: $[\mathbf{G}Z]_l = [\mathcal{R}_{\text{inte}}]_{(l,surf)} \left\langle \frac{Z\delta}{[\Delta\eta]} \right\rangle = -[\mathcal{R}_{\text{inte}}]_{(surf,l)} \left\langle \frac{Z\delta}{[\Delta\eta]} \right\rangle$
- Discretisation at half level $\bar{l}$, for both VFD and VFE: $[\mathbf{G}Z]_{\bar{l}} = \sum_{k=l+1}^{k=L} \delta_k Z_k$

VFD discretisation matches the following identity:

$$[\mathbf{G}Z]_l = \left(1 - \frac{\alpha_l}{\delta_l}\right)[\mathbf{G}Z]_{\bar{l}} + \frac{\alpha_l}{\delta_l}[\mathbf{G}Z]_{\overline{l-1}}$$

Neglecting the vertical variations of Z compared to those of α and δ:

$$[\mathbf{G}Z]_{\bar{l}} \simeq \frac{\alpha_l}{\delta_{l+1} - \alpha_{l+1} + \alpha_l}[\mathbf{G}Z]_{l+1} + \frac{\delta_{l+1} - \alpha_{l+1}}{\delta_{l+1} - \alpha_{l+1} + \alpha_l}[\mathbf{G}Z]_l$$

f) G type vertical integrals on the whole column.

Expression of integral (used for example to compute top atmosphere value of gz or gw) is:

$$[\mathbf{G}Z]_{top} = \int_{\eta'=0}^{\eta'=1} \frac{1}{\Pi}\frac{\partial \Pi}{\partial \eta} Z d\eta' = -\int_{\Pi'=\Pi_s}^{\Pi'=\Pi_{\text{top}}} Z\frac{d\Pi'}{\Pi'}$$

Discretisation:

- VFD discretisation: $[\mathbf{G}Z]_{top} = \sum_{k=1}^{k=L} \delta_k Z_k$
- VFE discretisation: $[\mathbf{G}Z]_{top} = [\mathcal{R}_{\text{inte}}]_{(top,surf)} \left\langle \frac{Z\delta}{[\Delta\eta]} \right\rangle$

g) ∇G type vertical integrals (from the bottom).

Expression of integral (used for example in **GPGRGEO** to compute $\nabla(gz)$, **GNHGRGW** to compute $\nabla(gw)$) is:

$$\nabla[\mathbf{G}Z]_\eta = \nabla\left(\int_{\eta'=\eta}^{\eta'=1} \frac{1}{\Pi}\frac{\partial \Pi}{\partial \eta} Z d\eta'\right) = -\nabla\left(\int_{\Pi'=\Pi_s}^{\Pi'=\Pi} Z\frac{d\Pi'}{\Pi'}\right)$$

Discretisation:

- VFD discretisation at full level l: $[\nabla\mathbf{G}Z]_l = \alpha_l[\nabla Z]_l + Z_l[\nabla\alpha]_l + \sum_{k=l+1}^{k=L}(\delta_k[\nabla Z]_k + Z_k[\nabla\delta]_k)$
- VFE discretisation at full level l: $[\nabla\mathbf{G}Z]_l = [\mathcal{R}_{\text{inte}}]_{(l,surf)} \left\langle \frac{\delta(\nabla Z)+Z(\nabla\delta)}{[\Delta\eta]} \right\rangle$
- Discretisation at half level $\bar{l}$, for both VFD and VFE: $[\nabla\mathbf{G}Z]_{\bar{l}} = \sum_{k=l+1}^{k=L}(\delta_k[\nabla Z]_k + Z_k[\nabla\delta]_k)$

h) $\nabla\mathbf{G} + \nabla(\log\Pi)$ type vertical integrals (from the bottom).

Such a quantity appears at least to compute the horizontal pressure gradient term. We give its VFD discretisation at full levels:

$$[\nabla\mathbf{G}Z + (\nabla(\log\Pi))Z]_l = \alpha_l[\nabla Z]_l + Z_l[\nabla(\alpha + \log\Pi)]_l + \sum_{k=l+1}^{k=L}(\delta_k[\nabla Z]_k + Z_k[\nabla\delta]_k)$$

$[\nabla(\alpha + \log\Pi)]_l$ has sometimes a simpler expression than separate quantities $[\nabla\alpha]_l$ and $[\nabla(\log\Pi)]_l$. For accuracy, discretisation must use the simpler expression of $[\nabla(\alpha + \log\Pi)]_l$ which is pre-computed in **GPGRXYB**.

Appendix 2: expressions for $\nabla\alpha$ and $\nabla\delta$ at full levels.

a) Expressions for $\nabla\alpha$ and $\nabla\delta$ at full levels if LVERTFE=.F. and NDLNPR=0.

$*$ **For δ:**

$$[\nabla\delta]_l = \nabla \log\left(\frac{\Pi_{\bar{l}}}{\Pi_{\bar{l}-1}}\right) = \nabla\log\left(\Pi_{\bar{l}}\right) - \nabla\log\left(\Pi_{\bar{l}-1}\right) = \frac{[\nabla\Pi]_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{[\nabla\Pi]_{\bar{l}-1}}{\Pi_{\bar{l}-1}} = \left[\frac{B_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{B_{\bar{l}-1}}{\Pi_{\bar{l}-1}}\right]\nabla\Pi_{\rm s}$$

Thus:

$$[\nabla\delta]_l = \left[\frac{B_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{B_{\bar{l}-1}}{\Pi_{\bar{l}-1}}\right]\nabla\Pi_{\rm s} \tag{400}$$

This formula can be rewritten as follows (the code uses this expression):

$$[\nabla\delta]_l = -\frac{A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}}{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\nabla\Pi_{\rm s} \tag{401}$$

$*$ **For α:**

$$[\nabla\alpha]_l = -\frac{\Pi_{\bar{l}-1}}{\Pi_{\bar{l}} - \Pi_{\bar{l}-1}}[\nabla\delta]_l - \nabla\left[\frac{\Pi_{\bar{l}-1}}{\Pi_{\bar{l}} - \Pi_{\bar{l}-1}}\right]\delta_l = -\frac{\Pi_{\bar{l}-1}}{\Pi_{\bar{l}} - \Pi_{\bar{l}-1}}[\nabla\delta]_l - \frac{[\nabla\Pi]_{\bar{l}-1}}{\Pi_{\bar{l}} - \Pi_{\bar{l}-1}}\delta_l - \nabla\left[\frac{1}{\Pi_{\bar{l}} - \Pi_{\bar{l}-1}}\right]\Pi_{\bar{l}-1}\delta_l$$

Expressions $\frac{1}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}$ and $\nabla\Pi_{\rm s}$ are put in factor. The term containing $[\nabla\delta]_l$ yields:

$$\frac{-\Pi_{\bar{l}-1}(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})\left[\frac{B_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{B_{\bar{l}-1}}{\Pi_{\bar{l}-1}}\right]}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\nabla\Pi_{\rm s}$$

The sum of terms containing δ_l writes:

$$\frac{-(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})B_{\bar{l}-1}\delta_l + \Pi_{\bar{l}-1}B_{\bar{l}}\delta_l - \Pi_{\bar{l}-1}B_{\bar{l}-1}\delta_l}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\nabla\Pi_{\rm s}$$

$$= \frac{(-\Pi_{\bar{l}}B_{\bar{l}-1} + \Pi_{\bar{l}-1}B_{\bar{l}-1} + \Pi_{\bar{l}-1}B_{\bar{l}} - \Pi_{\bar{l}-1}B_{\bar{l}-1})\delta_l}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\nabla\Pi_{\rm s}$$

$$= \frac{(\Pi_{\bar{l}-1}B_{\bar{l}} - \Pi_{\bar{l}}B_{\bar{l}-1})\delta_l}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\nabla\Pi_{\rm s} = \frac{\Pi_{\bar{l}}\Pi_{\bar{l}-1}\left[\frac{B_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{B_{\bar{l}-1}}{\Pi_{\bar{l}-1}}\right]\delta_l}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\nabla\Pi_{\rm s}$$

From the previous equations, $[\nabla\alpha]_l$ can be rewritten:

$$[\nabla\alpha]_l = \frac{\Pi_{\bar{l}}\Pi_{\bar{l}-1}\log\left(\frac{\Pi_{\bar{l}}}{\Pi_{\bar{l}-1}}\right) - \Pi_{\bar{l}-1}(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})}{(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2}\left[\frac{B_{\bar{l}}}{\Pi_{\bar{l}}} - \frac{B_{\bar{l}-1}}{\Pi_{\bar{l}-1}}\right]\nabla\Pi_{\rm s} \tag{402}$$

$*$ **For $\alpha + \log\Pi$:** Horizontal gradient of this term is used in the discretization of the pressure gradient term in the RHS of momentum equation: it expression is in fact significantly simpler than the ones of the horizontal gradients of α and δ taken separately. Starting from formulae giving $\frac{\nabla\Pi}{\Pi}$ and $\nabla\alpha$ (formula (402)) at full levels one can see that:

- the sum of terms containing $B_{\bar{l}}\delta_l$ is zero.
- the sum of terms containing $B_{\bar{l}-1}\delta_l$ is zero.
- the sum of the other terms containing $B_{\bar{l}-1}$ is zero.
- $(\Pi_{\bar{l}} - \Pi_{\bar{l}-1})^2$ can be eliminated at the numerator and denominator of the other terms containing $B_{\bar{l}-1}$.

The consequence is that $[\nabla(\alpha + \log\Pi)]_l$ can be rewritten:

$$[\nabla(\alpha + \log\Pi)]_l = \frac{B_{\bar{l}}}{\Pi_{\bar{l}}}\nabla\Pi_{\rm s} \tag{403}$$

b) Expressions for $\nabla\alpha$ and $\nabla\delta$ at full levels if LVERTFE=.F. and NDLNPR=1.

$*$ **For δ:**

$$\delta_l = \frac{\Delta\Pi_l}{\Pi_l} = \frac{\Delta\Pi_l}{\sqrt{\Pi_{\bar{l}-1}\Pi_{\bar{l}}}} \tag{404}$$

One applies operator ∇ to equation (404):

$$[\nabla\delta]_l = \left[\frac{1}{\Pi}\right]_l \left([\nabla\Pi]_{\bar{l}} - [\nabla\Pi]_{\bar{l}-1}\right) - \left[\frac{1}{\Pi}\right]_l \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)\left[\frac{\nabla\Pi}{\Pi}\right]_l$$

which can be rewritten:

$$[\nabla\delta]_l = \left[\frac{1}{\Pi}\right]_l \left([\Delta B]_l \nabla\Pi_{\rm s} - \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)\left[\frac{\nabla\Pi}{\Pi}\right]_l\right) \tag{405}$$

Expression of $[\nabla\delta]_l$ contains the two quantities $\left[\frac{1}{\Pi}\right]_l$ and $\left[\frac{\nabla\Pi}{\Pi}\right]_l$, the equations of which (equations (190) and (183)) have to be used. That yields:

$$[\nabla\delta]_l = \frac{1}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}}\left([\Delta B]_l - \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)\frac{[\Delta B]_l + \delta_l\frac{A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}}{\left(\Pi_{\bar{l}}-\Pi_{\bar{l}-1}\right)}}{\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\right)\nabla\Pi_{\rm s}$$

The sum of terms containing $[\Delta B]_l$ is zero, so the previous equation can be rewritten as:

$$[\nabla\delta]_l = -\frac{\delta_l\left[A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}\right]}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\nabla\Pi_{\rm s} \tag{406}$$

If the hydrostatic pressure at the top of the model is zero:

$$[\nabla\delta]_{l=1} = 0$$

$*$ **For α:**

$$\alpha_l = 1 - \sqrt{\frac{\Pi_{\bar{l}-1}}{\Pi_{\bar{l}}}} = 1 - \frac{\Pi_l}{\Pi_{\bar{l}}} \tag{407}$$

One applies operator ∇ to equation (407):

$$[\nabla\alpha]_l = -\left[\frac{1}{\Pi}\right]_l [\nabla\Pi]_{\bar{l}-1} - \Pi_{\bar{l}-1}\left[-\left[\frac{1}{\Pi}\right]_l\left[\frac{\nabla\Pi}{\Pi}\right]_l\right]$$

which can be rewritten:

$$[\nabla\alpha]_l = \left[\frac{1}{\Pi}\right]_l \left(\Pi_{\bar{l}-1}\left[\frac{\nabla\Pi}{\Pi}\right]_l - [\nabla\Pi]_{\bar{l}-1}\right) \tag{408}$$

Expression of $[\nabla\alpha]_l$ contains the two quantities $\left[\frac{1}{\Pi}\right]_l$ and $\left[\frac{\nabla\Pi}{\Pi}\right]_l$, the equations of which (equations (190) and (183)) have to be used. That yields:

$$[\nabla\alpha]_l = \frac{1}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}}\left(\Pi_{\bar{l}-1}\frac{[\Delta B]_l + \delta_l\frac{A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}}{\left(\Pi_{\bar{l}}-\Pi_{\bar{l}-1}\right)}}{\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)} - B_{\bar{l}-1}\right)\nabla\Pi_{\rm s}$$

This expression is rewritten in order to factorise the two terms $A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}$ and $1/\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)$.

$$\Pi_{\bar{l}-1}[\Delta B]_l / \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right) - B_{\bar{l}-1}$$

$$= \left(\Pi_{\bar{l}-1}B_{\bar{l}} - \Pi_{\bar{l}-1}B_{\bar{l}-1} - \Pi_{\bar{l}}B_{\bar{l}-1} + \Pi_{\bar{l}-1}B_{\bar{l}-1}\right) / \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right) = \left(\Pi_{\bar{l}-1}B_{\bar{l}} - \Pi_{\bar{l}}B_{\bar{l}-1}\right) / \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)$$

$$= \left(\left(A_{\bar{l}-1} + B_{\bar{l}-1}\Pi_{\rm s}\right)B_{\bar{l}} - \left(A_{\bar{l}} + B_{\bar{l}}\Pi_{\rm s}\right)B_{\bar{l}-1}\right) / \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right) = -\left(A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}\right) / \left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)$$

Thus:

$$[\nabla\alpha]_l = \frac{1}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}}\left[\frac{A_{\bar{l}}B_{\bar{l}-1} - A_{\bar{l}-1}B_{\bar{l}}}{\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\right]\left[-1 + \frac{\delta_l\Pi_{\bar{l}-1}}{\left(\Pi_{\bar{l}} - \Pi_{\bar{l}-1}\right)}\right]\nabla\Pi_{\rm s}$$

Term $-1+\delta_l \Pi_{\bar{l}-1}/\left(\Pi_{\bar{l}}-\Pi_{\bar{l}-1}\right)$ in factor can be still rewritten:

$$\frac{\frac{\Pi_{\bar{l}}-\Pi_{\bar{l}-1}}{\Pi_l}\Pi_{\bar{l}-1}}{\left(\Pi_{\bar{l}}-\Pi_{\bar{l}-1}\right)}-1=\frac{\Pi_{\bar{l}-1}}{\Pi_l}-1=-\alpha_l$$

Thus:

$$[\nabla\alpha]_l=-\frac{\alpha_l\left[A_{\bar{l}}B_{\bar{l}-1}-A_{\bar{l}-1}B_{\bar{l}}\right]}{\sqrt{\Pi_{\bar{l}}\Pi_{\bar{l}-1}}\left(\Pi_{\bar{l}}-\Pi_{\bar{l}-1}\right)}\nabla\Pi_{\mathrm{s}} \qquad (409)$$

Top value: $[\nabla\alpha]_{l=1}=0$.

c) Expression for $\nabla\delta$ at full levels for VFE.

Equation $\delta_l=\frac{[\Delta\Pi]_l}{\Pi_l}$ can be rewritten $\log(\delta_l)=\log([\Delta\Pi]_l)-\log(\Pi_l)$. Applying ∇ yields:

$$\frac{[\nabla\delta]_l}{\delta_l}=\frac{[\nabla[\Delta\Pi]]_l}{[\Delta\Pi]_l}-\frac{[\nabla\Pi]_l}{\Pi_l}$$

But $[\nabla[\Delta\Pi]]_l=[\Delta B]_l\nabla\Pi_{\mathrm{s}}$ and $[\nabla\Pi]_l=B_l\nabla\Pi_{\mathrm{s}}$. So:

$$[\nabla\delta]_l=\left[\frac{[\Delta B]_l}{[\Delta\Pi]_l}-\frac{B_l}{\Pi_l}\right]\delta_l\nabla\Pi_{\mathrm{s}}$$

Factor $\left[\frac{[\Delta B]_l}{[\Delta\Pi]_l}-\frac{B_l}{\Pi_l}\right]$ can also be rewritten $\frac{[\Delta B]_l\Pi_l-[\Delta\Pi]_lB_l}{[\Delta\Pi]_l\Pi_l}$, i.e. $\frac{A_l[\Delta B]_l+B_l[\Delta B]_l\Pi_{\mathrm{s}}-[\Delta A]_lB_l-[\Delta B]_lB_l\Pi_{\mathrm{s}}}{[\Delta\Pi]_l\Pi_l}$, i.e. $\frac{A_l[\Delta B]_l-[\Delta A]_lB_l}{[\Delta\Pi]_l\Pi_l}$.

Thus:

$$[\nabla\delta]_l=\frac{A_l[\Delta B]_l-[\Delta A]_lB_l}{[\Delta\Pi]_l\Pi_l}\delta_l\nabla\Pi_{\mathrm{s}} \qquad (410)$$

Appendix 3: expression of the vertical integration and of the vertical derivative matricial operators (vertical finite element scheme).

The content of this section is valid for the code provided by ECMWF in 2004 (**LVFE_ECMWF**=T). New formulations used if (**LVFE_ECMWF**=F) are not described in detail (see (Smolíková and Vivoda, 2013), (de Boor, 2001)); in this case set-up is done by routine **SUVERTFEB**.

a) Introduction.

The algorithm is described in (Untch and Hortal, 2001), (Untch and Hortal, 2004). A vertically-dependent field X can be projected on a finite element basis e_i, i being the layer numbering.

$$X(\eta) = \sum_{i=0}^{i=L} X_i e_i(\eta) \tag{411}$$

Its vertical integral

$$S(\eta) = \int_{\text{top}}^{\eta} X(\eta^{'})d\eta^{'}$$

has the following finite element representation:

$$S(\eta) = \sum_{i=1}^{i=L+1} S_i d_i(\eta) \tag{412}$$

The basis d_i is the same as e_i excepted for some boundary values (top and bottom) where some differences can occur. One can show that the discretisation of vertical integration is equivalent to write the matricial product:

$$\mathcal{A}\langle S\rangle = \mathcal{B}\langle X\rangle \tag{413}$$

which is equivalent to write:

$$\langle S\rangle = \mathcal{A}^{-1}\mathcal{B}\langle X\rangle = [\mathcal{R}_{\text{inte}}]_{(top)}\langle X\rangle \tag{414}$$

where $\langle S\rangle$ is the vector of components $(S_1, S_2, ..., S_L, S_{L+1})$ and $\langle X\rangle$ is the vector of components $(X_1, X_2, ..., X_L)$. For $i = 1$ to $i = L$ quantities X_i and S_i are defined and computed at full levels. S_{L+1} is the vertical integral on the whole atmosphere. The setup code of the model successively computes $\mathcal{A}$, $\mathcal{A}^{-1}$, $\mathcal{B}$ and the product $\mathcal{A}^{-1}\mathcal{B}$. $[\mathcal{R}_{\text{inte}}]_{(top)}$ is a $(L+1)*L$ matrix and $[\mathcal{R}_{\text{inte}}]_{(top,l)}$ is the vector containing the l-th line of $[\mathcal{R}_{\text{inte}}]_{(top)}$. S_l is given by the scalar product:

$$S_l = [\mathcal{R}_{\text{inte}}]_{(top,l)}\langle X\rangle$$

The $L+1$-th line of $[\mathcal{R}_{\text{inte}}]_{(top)}$ contains $[\mathcal{R}_{\text{inte}}]_{(top,surf)}$ (discretisation of the integral on the whole atmosphere):

$$S_{L+1} = S_{\text{surf}} = [\mathcal{R}_{\text{inte}}]_{(top,surf)}\langle X\rangle$$

One can also discretise an integral from the surface (for example to compute the geopotential height).

$$\int_{\text{surf}}^{\eta} X(\eta^{'})d\eta^{'}$$

can be rewritten:

$$\int_{\text{top}}^{\eta} X(\eta^{'})d\eta^{'} - \int_{\text{top}}^{surf} X(\eta^{'})d\eta^{'} = S(\eta) - S(\eta = 1)$$

discretised by operator $[\mathcal{R}_{\text{inte}}]_{(surf,l)} = [\mathcal{R}_{\text{inte}}]_{(top,l)} - [\mathcal{R}_{\text{inte}}]_{(top,surf)}$.

b) Basis used in the vertical finite element scheme.

$*$ Linear finite elements.
For $0 \leq i \leq L+1$: $d_i(\eta) = e_i(\eta)$. Definition of e_i is:

- $e_i(\eta) = 0$ for $\eta < \eta_{i-1}$.
- $e_i(\eta) = (\eta - \eta_{i-1})/(\eta_i - \eta_{i-1})$ for $\eta_{i-1} \leq \eta \leq \eta_i$.
- $e_i(\eta) = (\eta_{i+1} - \eta)/(\eta_{i+1} - \eta_i)$ for $\eta_i \leq \eta \leq \eta_{i+1}$.
- $e_i(\eta) = 0$ for $\eta > \eta_{i+1}$.

$*$ **Hermite cubic finite elements.** $d_i(\eta)$ is generally equal to $e_i(\eta)$ excepted for $i \leq 2$ and $\eta < \eta_1$. Definition of e_i is:

- $e_i(\eta) = 0$ for $\eta < \eta_{i-2}$.
- for $\eta_{i-2} \leq \eta \leq \eta_i$, $e_i(\eta)$ is a third-order polynomial which matches with $e_i(\eta_{i-2}) = 0$, $e_i(\eta_i) = 1$, $\left[\frac{de}{d\eta}\right]_i(\eta_{i-2}) = 0$, $\left[\frac{de}{d\eta}\right]_i(\eta_i) = 0$. After some long calculations which are not detailed in this documentation that yields:

$$e_i(\eta) = \frac{1}{[\eta_i - \eta_{i-2}]^3}\left[-2\eta^3 + 3(\eta_{i-2} + \eta_i)\eta^2 - 6(\eta_i\eta_{i-2})\eta + (\eta_{i-2}^3 - 3\eta_{i-2}^2\eta_i)\right]$$

- for $\eta_i \leq \eta \leq \eta_{i+2}$, $e_i(\eta)$ is a third-order polynomial which matches with $e_i(\eta_{i+2}) = 0$, $e_i(\eta_i) = 1$, $\left[\frac{de}{d\eta}\right]_i(\eta_{i+2}) = 0$, $\left[\frac{de}{d\eta}\right]_i(\eta_i) = 0$. After some long calculations which are not detailed in this documentation that yields:

$$e_i(\eta) = \frac{1}{[\eta_i - \eta_{i+2}]^3}\left[-2\eta^3 + 3(\eta_{i+2} + \eta_i)\eta^2 - 6(\eta_i\eta_{i+2})\eta + (\eta_{i+2}^3 - 3\eta_{i+2}^2\eta_i)\right]$$

- $e_i(\eta) = 0$ for $\eta > \eta_{i+2}$.

c) Vertical integration matricial operator.

$*$ **General expression of matrices $\mathcal{A}$ and $\mathcal{B}$.** After equation (4.3) of (Untch and Hortal, 2001), (Untch and Hortal, 2004) and taking $t = d$ (what is done in the code) one obtains:

$$\mathcal{A}_{(i,j)} = \int_{\eta=0}^{\eta=1} d_i(\eta)d_j(\eta)d\eta \tag{415}$$

$$\mathcal{B}_{(i,j)} = \int_{\eta=0}^{\eta=1}\left[d_j(\eta)\int_{\eta'=0}^{\eta'=\eta} d_i(\eta')d\eta'\right]d\eta \tag{416}$$

$\mathcal{A}$ and $\mathcal{B}$ are $(L+1)*(L+1)$ matrices. $\mathcal{A}$ is a symmetric one.

$*$ **More details about matrices $\mathcal{A}$ and $\mathcal{B}$ for linear finite elements.** The vertical integrals of equations (415) and (416) can be analytically computed using formulae giving e_i and d_i. For $\mathcal{A}$ that yields:

- $\mathcal{A}$ is a symmetric tridiagonal matrix.
- $\mathcal{A}_{(1,1)} = \frac{\eta_2}{3}$.
- $\mathcal{A}_{(i,i)} = \frac{\eta_{i+1}-\eta_{i-1}}{3}$ for $2 \leq i \leq L-1$.
- $\mathcal{A}_{(L,L)} = \frac{1-\eta_{L-1}}{3}$.
- $\mathcal{A}_{(L+1,L+1)} = \frac{1-\eta_L}{3}$.
- $\mathcal{A}_{(i,i-1)} = \mathcal{A}_{(i-1,i)} = \frac{\eta_i-\eta_{i-1}}{6}$ for $2 \leq i \leq L$.
- $\mathcal{A}_{(L+1,L)} = \mathcal{A}_{(L,L+1)} = \frac{1-\eta_L}{6}$
- The other coefficients are equal to zero.

For $\mathcal{B}$ that yields:

- $\mathcal{B}_{(1,1)} = \frac{5}{24}{\eta_1}^2 + \frac{1}{4}\eta_1(\eta_2 - \eta_1)$
- $\mathcal{B}_{(j,1)} = \frac{1}{4}\eta_1(\eta_{j+1} - \eta_{j-1})$ for $2 \leq j \leq L$.
- $\mathcal{B}_{(L+1,j)} = \frac{1}{4}(1 - \eta_L)(\eta_j - \eta_{j-2})$ for $2 \leq j \leq L$.
- $\mathcal{B}_{(L+1,L+1)} = \frac{1}{3}(1 - \eta_L)^2 + \frac{1}{4}(\eta_L - \eta_{L-1})(1 - \eta_L)$
- $\mathcal{B}_{(L,L+1)} = \frac{1}{8}(\eta_L - \eta_{L-1})^2 + \frac{1}{4}(\eta_L - \eta_{L-1})(1 - \eta_L) + \frac{1}{6}(1 - \eta_L)^2$
- $\mathcal{B}_{(j,i)} = \frac{1}{4}(\eta_i - \eta_{i-2})(\eta_{j+1} - \eta_{j-1})$ for $i < j$.
- $\mathcal{B}_{(j,j)} = \frac{1}{4}(\eta_{j-1} - \eta_{j-2})(\eta_j - \eta_{j-1}) + \frac{5}{24}(\eta_j - \eta_{j-1})^2 + \frac{1}{4}(\eta_{j+1} - \eta_j)(\eta_j - \eta_{j-2})$
- $\mathcal{B}_{(j,j+1)} = \frac{1}{8}(\eta_j - \eta_{j-1})^2 + \frac{1}{4}(\eta_j - \eta_{j-1})(\eta_{j+1} - \eta_j) + \frac{1}{8}(\eta_{j+1} - \eta_j)^2$
- $\mathcal{B}_{(j,j+2)} = \frac{1}{24}(\eta_{j+1} - \eta_j)^2$
- $\mathcal{B}_{(j,i)} = 0$ for $i > j + 2$.

∗ **More details about matrices $\mathcal{A}$ and $\mathcal{B}$ for Hermite cubic finite elements.** The vertical integrals of equations (415) and (416) can be analytically computed using formulae giving e_i and d_i. Expression of the coefficients of $\mathcal{B}$ and $\mathcal{A}$ is rather tricky and is not detailed in this documentation; Matrix $\mathcal{A}$ is symmetric with a non-zero main diagonal and three non-zero upper and lower side diagonals.

d) Vertical derivative matricial operator.

Not yet described.

e) Where to find the code?

∗ **Vertical integral.** The setup routines computing the matrix $\mathcal{A}^{-1}\mathcal{B}$ are **SUVERTFE1** for the linear finite elements and **SUVERTFE3** for Hermite cubic finite elements. The product $\mathcal{A}^{-1}\mathcal{B}$ is stored after one line shift in the array **RINTE** (module **YOMVERT**) which actually contains a matrix $(L+1)*L$. The intermediate quantities $\mathcal{A}$, $\mathcal{A}^{-1}$ and $\mathcal{B}$ are stored in the local arrays **ZAMAT**, **ZAMATI** and **ZBMAT**. The vertical integration, giving $\langle S\rangle$ for layers 1 to L and the additional layer L matching with the surface, knowing $\langle X\rangle$ for layers 1 to L, is done by routine **VERINT**. One can notice that the matrix product $\mathcal{A}^{-1}\mathcal{B}$ is a full matrix, contrary to the one which is used in the case **LVERTFE**=.F. which is triangular with some additional good properties (currently the routine **VERINT** is used only in the case **LVERTFE**=.T.). For most applications and uses of ARPEGE/IFS and ALADIN, the content of routines **SUVERTFE1** and **SUVERTFE3** can be seen as a "black box".

∗ **Vertical derivative.** The vertical derivative operator $\mathcal{R}_{\mathrm{deri}}$ is stored in the array **RDERI**, or **RDERB** if top and bottom boundary conditions are taken into account (module **YOMVERT**). **RDERI** is computed in the setup routine **SUVERTFE3D** or **SUNH_VERTFE3D**. **RDERB** is computed in the setup routine **SUNH_VERTFE3DBC**. The vertical derivation is done by routine **VERDER**. More details will be given in a future version of this documentation.

f) Additional remark: operator $\mathcal{R}_{\mathrm{inte}}$ for case LVERTFE=.F. .

If **LVERTFE**=.F. the vertical integration is also a matricial product but in this case the operator $\mathcal{R}_{\mathrm{inte}}$ is considerably simpler and tridiagonal. One can compute an operator $\mathcal{R}_{\mathrm{inte}}$ giving integrals at half levels (from the top), the matrix is $L*L$ and has the following content:

$$\begin{pmatrix} [\Delta\eta]_1 & 0 & ... & 0 & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & 0 & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & 0 & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & [\Delta\eta]_l & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & [\Delta\eta]_l & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & [\Delta\eta]_l & ... & [\Delta\eta]_L \end{pmatrix}$$

Since coefficients are constant on a column, but not temporally constant, the use of routine **VERINT** is not very interesting in this case (mode multiplications) and the code can be let as it is currently, but **VERINT** and a pre-computation of **RINTE** could be possible in the model set-up. The operator $\mathcal{R}_{\mathrm{inte}}$ which provides integrals at full levels (from the top) has the following shape:

$$\begin{pmatrix} [\Delta\eta]_1\left(1-\frac{\alpha_1}{\delta_1}\right) & 0 & ... & 0 & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2\left(1-\frac{\alpha_2}{\delta_2}\right) & ... & 0 & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & 0 & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & [\Delta\eta]_l\left(1-\frac{\alpha_l}{\delta_l}\right) & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & [\Delta\eta]_l & ... & 0 \\ [\Delta\eta]_1 & [\Delta\eta]_2 & ... & [\Delta\eta]_l & ... & [\Delta\eta]_L\left(1-\frac{\alpha_L}{\delta_L}\right) \end{pmatrix}$$

- In the linear model, replace $\frac{\alpha_l}{\delta_l}$ by $\frac{\alpha_l^*}{\delta_l^*}$.
- $\frac{\alpha_l}{\delta_l}=\frac{\Pi_{\bar{l}}-\Pi_l}{\Pi_{\bar{l}}-\Pi_{\overline{l-1}}}$ remains close to 1/2.

We can also provide the vertical integration matrix for integrations from the bottom:

$$\begin{pmatrix} [\Delta\eta]_1\frac{\alpha_1}{\delta_1} & [\Delta\eta]_2 & ... & [\Delta\eta]_l & ... & [\Delta\eta]_L \\ 0 & [\Delta\eta]_2\frac{\alpha_2}{\delta_2} & ... & [\Delta\eta]_l & ... & [\Delta\eta]_L \\ 0 & 0 & ... & [\Delta\eta]_l & ... & [\Delta\eta]_L \\ 0 & 0 & ... & [\Delta\eta]_l\frac{\alpha_l}{\delta_l} & ... & [\Delta\eta]_L \\ 0 & 0 & ... & 0 & ... & [\Delta\eta]_L \\ 0 & 0 & ... & 0 & ... & [\Delta\eta]_L\frac{\alpha_L}{\delta_L} \end{pmatrix}$$