ARPEGE-Climate Version 6.2
Algorithmic Documentation

UNDER CONSTRUCTION — June 2016
Contents

1 Basic hypotheses and related constants 11
   1 Astronomical constants . . . . . . . . . . . . . . . . . . . . . 12
      1.1 Calendar . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
      1.2 Time . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 12
      1.3 Astronomical elements . . . . . . . . . . . . . . . . . . . 13
      1.4 Sun trajectory relative to Earth . . . . . . . . . . . . . . 13
   2 Geometry, geoid . . . . . . . . . . . . . . . . . . . . . . . . . . 14
      2.1 Geometry . . . . . . . . . . . . . . . . . . . . . . . . . . . 14
      2.2 Coordinate system . . . . . . . . . . . . . . . . . . . . . . 15
      2.3 Geoid . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 15
   3 Fundamental constants . . . . . . . . . . . . . . . . . . . . . . . 15
   4 Radiation . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 16
   5 Thermodynamics, gas phase . . . . . . . . . . . . . . . . . . . . 16
   6 Thermodynamics, liquid phase . . . . . . . . . . . . . . . . . . 17
   7 Thermodynamics, solid phase . . . . . . . . . . . . . . . . . . . 17
   8 Thermodynamics, phase transition . . . . . . . . . . . . . . . . 17
      8.1 Vaporization . . . . . . . . . . . . . . . . . . . . . . . . . . 17
      8.2 Sublimation . . . . . . . . . . . . . . . . . . . . . . . . . . 18
      8.3 Melting . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 18
   9 Consequences on saturation . . . . . . . . . . . . . . . . . . . . 18
  10 Thermodynamic functions . . . . . . . . . . . . . . . . . . . . . 19
     10.1 Absolute Functions . . . . . . . . . . . . . . . . . . . . . . . 19
     10.2 Functions in the model parametrizations . . . . . . . . . . . 20
1.7 Modification since cycle 37

2 Definition of Eulerian and semi-Lagrangian schemes
   2.1 Eulerian scheme
   2.2 Semi-Lagrangian scheme

3 The 2D equations
   3.1 Notations for the 2D equations
   3.2 The 2D shallow-water system of equations in spherical geometry

4 The 3D equations in spherical geometry (ARPEGE/IFS)
   4.1 Notations for the 3D equations
   4.2 The thin layer 3D primitive equation model: Lagrangian formulation
   4.3 The deep layer 3D primitive equation model according to White and Bromley, 1995: Lagrangian formulation
   4.4 The thin layer 3D NH-PDVD non-hydrostatic model: Lagrangian formulation
   4.5 The deep layer NH-PDVD non-hydrostatic model according to Wood and Staniforth, 2003
   4.6 The thin layer NH-GEOGW non-hydrostatic model: Lagrangian formulation

5 Discretisation of the equations: general aspects
   5.1 Notations
   5.2 Discretisation for a 3D variable in a 3D model: general case where the RHS has non-zero linear terms
   5.3 Discretisation for a 3D variable in a 3D model: particular case where the RHS has zero linear terms
   5.4 Discretisation for a 2D variable in a 3D model (GMVS variables, for example continuity equation)
   5.5 Discretisation for a 2D variable in a 2D model
   5.6 Additional vertical derivatives
   5.7 Implicit treatment of some non-linear terms in the case of NH-PDVD non-hydrostatic model
   5.8 Case when some variables are evaluated at half levels
   5.9 Remarks for spline cubic vertical interpolations

6 Computation of medium and origin points
   6.1 Medium point \( M \) (subroutines LARMES and LARMES2)
   6.2 Origin point \( O \) (subroutines LARMES and LAINOR2)
   6.3 Refined recomputation of point \( O \)
   6.4 Remarks

7 The SL discretisation of the 2D shallow-water system of equations (spherical geometry)
   7.1 Momentum equation

7.2 Vorticity equation
   7.3 Pressure equation
   7.4 Coriolis force
CONTENTS

7.2 Continuity equation ........................................ 90
7.3 Quantities to be interpolated ............................... 91

8 The SL discretisation of the 3D primitive equation model ..... 92
8.1 Thin layer formulation of the momentum equation ....... 92
8.2 White and Bromley deep layer formulation of the momentum equation 93
8.3 Thermodynamic equation .................................. 94
8.4 Thin layer formulation of the continuity equation ....... 94
8.5 White and Bromley deep layer formulation of the continuity equation 95
8.6 Moisture equation ............................................ 95
8.7 Other advectable GFL variables ............................ 96
8.8 Case of lagged physics ....................................... 96
8.9 Quantities to be interpolated (computation under subroutine LACDYN) 96

9 The SL discretisation of the 3D non hydrostatic model .... 98
9.1 Thin layer NH-PDVD model .................................. 98
9.2 Wood and Staniforth deep layer NH-PDVD model ....... 101
9.3 Thin layer NH-GEOGW model .............................. 103

10 R operator ....................................................... 104
10.1 No tilting ....................................................... 104
10.2 Tilting .......................................................... 104
10.3 Plane geometry (ALADIN) .................................. 105

11 Computation of longitudes and latitudes on the computational sphere 106

12 Interpolations and weights computations .................. 107
12.1 Interpolation grid and weights (subroutine LASCAW) .... 107
12.2 Interpolations ................................................ 114
12.3 Code structures to store weights ......................... 117

13 Computation of $\dot{\eta}$ at full levels ....................... 126

14 Lateral boundary conditions ................................ 127
14.1 Extra longitudes ............................................ 127
14.2 Extra latitudes ............................................. 127
14.3 Vertical boundary conditions in the 3D model .......... 127

15 2D shallow water and 3D models organigrammes ........... 129
<table>
<thead>
<tr>
<th>Contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>15.1 Interpolation routines</td>
</tr>
<tr>
<td>15.2 Routines common to 3D and 2D organigrams</td>
</tr>
<tr>
<td>15.3 Routines specific to 2D organigram</td>
</tr>
<tr>
<td>15.4 Routines specific to 3D organigram</td>
</tr>
<tr>
<td>15.5 Set-up routines</td>
</tr>
<tr>
<td>15.6 Organigramme for 2D model</td>
</tr>
<tr>
<td>15.7 Organigramme for 3D model</td>
</tr>
<tr>
<td>16 Tangent linear and adjoint codes</td>
</tr>
<tr>
<td>16.1 3D hydrostatic model</td>
</tr>
<tr>
<td>16.2 3D non hydrostatic model</td>
</tr>
<tr>
<td>16.3 2D shallow-water model</td>
</tr>
<tr>
<td>17 Some distributed memory features</td>
</tr>
<tr>
<td>17.1 Case LEQ_REGIONS=F</td>
</tr>
<tr>
<td>17.2 Case LEQ_REGIONS=T</td>
</tr>
<tr>
<td>18 Specific SI variables in pointer modules, modules and namelists</td>
</tr>
<tr>
<td>18.1 Modules for declarations</td>
</tr>
<tr>
<td>18.2 Modules for type definition</td>
</tr>
<tr>
<td>18.3 Modules containing subroutines</td>
</tr>
<tr>
<td>19 An example of operational SL2TL namelist for ARPEGE</td>
</tr>
<tr>
<td>20 Bibliography</td>
</tr>
<tr>
<td>20.1 Publications</td>
</tr>
<tr>
<td>20.2 Some internal notes and other ARPEGE notes</td>
</tr>
<tr>
<td>20.3 Workshop proceedings</td>
</tr>
<tr>
<td>5 Semi-implicit spectral computations and predictor-corrector schemes</td>
</tr>
<tr>
<td>1 Introduction</td>
</tr>
<tr>
<td>1.1 Interest of semi-implicit and iterative centred-implicit schemes</td>
</tr>
<tr>
<td>1.2 Distributed memory</td>
</tr>
<tr>
<td>1.3 The different models described</td>
</tr>
<tr>
<td>1.4 Other restrictions of this documentation</td>
</tr>
<tr>
<td>1.5 Modifications since cycle 37</td>
</tr>
<tr>
<td>Section</td>
</tr>
<tr>
<td>---------</td>
</tr>
<tr>
<td>2</td>
</tr>
<tr>
<td>3</td>
</tr>
<tr>
<td>3.1</td>
</tr>
<tr>
<td>3.2</td>
</tr>
<tr>
<td>3.3</td>
</tr>
<tr>
<td>3.4</td>
</tr>
<tr>
<td>3.5</td>
</tr>
<tr>
<td>3.6</td>
</tr>
<tr>
<td>4</td>
</tr>
<tr>
<td>4.1</td>
</tr>
<tr>
<td>4.2</td>
</tr>
<tr>
<td>4.3</td>
</tr>
<tr>
<td>5</td>
</tr>
<tr>
<td>5.1</td>
</tr>
<tr>
<td>5.2</td>
</tr>
<tr>
<td>5.3</td>
</tr>
<tr>
<td>5.4</td>
</tr>
<tr>
<td>5.5</td>
</tr>
<tr>
<td>5.6</td>
</tr>
<tr>
<td>6</td>
</tr>
<tr>
<td>6.1</td>
</tr>
<tr>
<td>6.2</td>
</tr>
<tr>
<td>6.3</td>
</tr>
<tr>
<td>6.4</td>
</tr>
<tr>
<td>6.5</td>
</tr>
<tr>
<td>7</td>
</tr>
<tr>
<td>8</td>
</tr>
<tr>
<td>9</td>
</tr>
<tr>
<td>9.1</td>
</tr>
<tr>
<td>9.2</td>
</tr>
<tr>
<td>9.3</td>
</tr>
</tbody>
</table>
CONTENTS

9.4 Action done by these routines .................................. 218
10 Other remarks ...................................................... 221
11 Precomputed module and namelist quantities ...................... 223
  11.1 YEMDYN ..................................................... 223
  11.2 YOEPHY ..................................................... 223
  11.3 YOMCT0 ..................................................... 223
  11.4 YOMCVY ..................................................... 224
  11.5 YOMDIM ..................................................... 224
  11.6 YOMDYN ..................................................... 225
  11.7 YOMDYNA .................................................... 225
  11.8 YOMMP ...................................................... 227
  11.9 YOMMPG .................................................... 227
  11.10 YOMSP ...................................................... 228
12 References .......................................................... 229
  12.1 Publications ................................................... 229
  12.2 Internal notes and documentation ............................ 229

6 Horizontal diffusion .............................................. 231
  1 Introduction ..................................................... 231
  2 Formulation of the horizontal diffusion schemes ............... 232
    2.1 Main horizontal diffusion scheme ......................... 232
    2.2 Rayleigh friction ........................................... 234
    2.3 Nudging ...................................................... 234
  3 Discretisation of the horizontal diffusion schemes ........... 234
    3.1 Main horizontal diffusion scheme ......................... 234
    3.2 Modification of the horizontal diffusion scheme if “SLHD” interpolations are done in the semi-Lagrangian model .................................................. 234
    3.3 Particular use of a second reference truncation \(N_2\) for standard pressure levels \(\Pi_{ST}(l)\) below the surface ........................................ 235
    3.4 Application of diffusion to each variable ................. 239
    3.5 Other spectral calculations ............................... 240
  4 Organigramme of the horizontal diffusion computations ....... 240
    4.1 Order of spectral computations ........................... 240
4.2 Set-up and control routines until STEPO . . . . . . . . 241
4.3 Direct code under STEPO or tangent linear code under STEPOTL for spectral calculations. 242
4.4 Adjoint code under STEPOAD for spectral calculations. 242
4.5 Rayleigh friction done in grid-point space (ARPEGE only). 243
4.6 Action done by set-up routines. . . . . . . . . . . . . . 243
4.7 Control routines. . . . . . . . . . . . . . . . . . . . . . 243
4.8 Action done by routines under STEPO, STEPOAD or STEPOTL 243

5 Some distributed memory features . . . . . . . . . . . . . . . 244
6 Module and namelist variables . . . . . . . . . . . . . . . . . . 244
6.1 YEMDYN . . . . . . . . . . . . . . . . . . . . . . . . 245
6.2 YEMGEO . . . . . . . . . . . . . . . . . . . . . . . . 245
6.3 YOMCT0 . . . . . . . . . . . . . . . . . . . . . . . . 245
6.4 YOMDIM . . . . . . . . . . . . . . . . . . . . . . . . 245
6.5 YOMDYNA . . . . . . . . . . . . . . . . . . . . . . . . 246
6.6 YOMDYN . . . . . . . . . . . . . . . . . . . . . . . . 246
6.7 YOMGEM and namelist NAMGEM . . . . . . . . . . . . . 248
6.8 YOMGFL . . . . . . . . . . . . . . . . . . . . . . . . 248
6.9 YOMMP . . . . . . . . . . . . . . . . . . . . . . . . . 248
6.10 YOMMPG . . . . . . . . . . . . . . . . . . . . . . . . 248
6.11 YOMNUD and namelist NAMNUD . . . . . . . . . . . . . 248
6.12 YOMSP . . . . . . . . . . . . . . . . . . . . . . . . . 248
6.13 Functions of SUHDF and FCHDIF . . . . . . . . . . . . . 249

7 References . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 249
7.1 Publications: . . . . . . . . . . . . . . . . . . . . . . . 249
7.2 Internal notes: . . . . . . . . . . . . . . . . . . . . . . 249

7 Gravity wave drag . . . . . . . . . . . . . . . . . . . . . . . . . . 251
1 Parametrization of orographic gravity wave drag . . . . . . . 251
1.1 Calculation at the surface . . . . . . . . . . . . . . . . . . 251
1.2 Calculation at a given level . . . . . . . . . . . . . . . . . 251
2 Refinements for orographic gravity wave drag . . . . . . . . . 252
CONTENTS

2.1 Effective surface wind ........................................... 252
2.2 Mountain anisotropy ............................................. 252
2.3 Resonance ............................................................. 253
2.4 Trapping ............................................................... 254
2.5 Sub-grid orography .................................................. 254
2.6 The lift effect ......................................................... 255

3 Numerical securities in orography waves ......................... 256
  3.1 Implicit formulation versus surface wind ................. 256
  3.2 Implicit advective formulation .............................. 256

4 Parametrization of convective gravity wave drag .......... 257
  4.1 Calculations at the top of the cloud ....................... 257
  4.2 Calculations above the cloud ................................. 258
  4.3 Calculations inside the cloud ............................... 258

5 Numerical securities for convective waves .................. 258
6 Parametrization of non-orographic gravity wave drag .... 259
  6.1 Formalism ........................................................... 259
  6.2 Launch spectrum .................................................. 260
  6.3 Vertical propagation ............................................. 261
  6.4 Computation aspects ............................................. 261

References .................................................................. 263
1

Basic hypotheses and related constants

ARPEGE-IFS is a complex code designed not only for weather forecast or climate simulation, but also for data assimilation, forecast pre- and post-processing. It has been extended, diversified and complexified since 1986 jointly by Météo-France and ECMWF. The present documentation restricts to the description of the French climate version of ARPEGE-IFS. Some features are not compatible with the version used by ECMWF. In this case, we will use the term ARPEGE. Some features are specific to the French climate version, and we will use the term ARPEGE-climat. The core of the model is cycle 37T1 of ARPEGE-IFS.

ARPEGE-climat is now an atmosphere-only model. So the calculations concerning the surface boundary layer, vegetation, snow cover and soil are done in SURFEX (SURface EXternalisée), which is another model. SURFEX simulates the exchanges of momentum, heat, water, carbon dioxide concentration or chemical species between the surface and the atmosphere. It uses the concept of 'tile' to describe the surface (nature, town, sea, water) and can perform different parametrizations. Each surface grid box is made of the four adjacent tiles. The coverage of each of these surfaces is known through the global ECOCLIMAP database. SURFEX receives atmospheric forcing terms, runs the surface schemes, computes the average surface fluxes over the nature, town, sea and water weighted by their respective fraction and sends them back to the atmosphere in addition to radiatives terms. All this information is then used as lower boundary conditions for the atmospheric radiation and turbulent schemes. The present document gives some information about SURFEX and how SURFEX interacts with the atmosphere. The complete documentation on SURFEX (algorithmic and user’s guide) is available on the site.
1. Basic hypotheses and related constants

The model relies upon a geometrical assumption: the thin layer approximation, and a certain number of phenomenological assumptions such as the law of perfect gases or the hydrostatic approximation. With these assumptions a set of basic constants is presented here.

1 Astronomical constants

This section follows the last recommendations of the International Astronomical Union. It should be noted that the formulas are not valid for dates too far away from the 1st January 2000 (more than one century).

1.1 Calendar

The calendar used is the Gregorian calendar. The dates are given in the form:

\[ AAAAMMJ, sssss \]

with:

- \( AAAA \) year,
- \( MM \) month,
- \( JJ \) day,
- \( sssss \) seconds in the day.

1.2 Time

The length of the day is:

\[ d = 86400 \, s \]

Time \( t \) is expressed in seconds, the date of reference being 20000101.43200 (2451545.0 in Julian calendar). It is negative before this date and positive afterward. \( t \) is deduced from the calendar date by:

\[ t = (JD - 2451545)86400 + sssss \]

with \( JD \), date in the Julian calendar (E indicating the integer part):

\[ JD = 1720994.5 + K + E(365.25a) + E(30.601(m + 1)) + JJ \]
and:

\[
\begin{align*}
a &= \begin{cases} 
    \text{AAAA,} & \text{if } MM > 2 \\
    \text{AAAA} - 1, & \text{if } MM \leq 2
\end{cases} \\
\end{align*}
\]

\[
\begin{align*}
m &= \begin{cases} 
    MM, & \text{if } MM > 2 \\
    MM + 12, & \text{if } MM \leq 2
\end{cases} \\
\end{align*}
\]

\[
K = 2 - E(a/100) + E(E(a/100)/4)
\]

### 1.3 Astronomical elements

In the following we set:

\[
\theta = t/(dy_j)
\]

with:

\[
y_j = 365.25 \text{ days}
\]

The constants between square brackets are not used in the model; however, we provide them because they form a consistent set with those needed by the model.

- **Half great axis** \( e_a = 149597870000 \ m \pm 5 \times 10^{-5} \)
- **Excentricity** \( e_a = 0.016704 \pm 10^{-4} \)
- **Inclination** \( \theta = 0 \pm 2 \times 10^{-4} \)
- **Mean longitude** \( e_l = 1.7535 + 6.283076 \theta \)
- **Longitude of perihelion** \( e_l = 1.79661 + 0.0000563 \theta \)
- **Longitude of ascending node** \( e_M = 6.1937 \text{ if } t < 0, 3.0521 \text{ if } t > 0 \)
- **Mean anomaly** \( e_M = 6.240075 + 6.283020 \theta \\
\]
- **Sun-Earth distance** \( R_s = e_a(1.0001 - 0.0163 \sin(e_l) \\
+ 0.0037 \cos(e_l)) \)

### 1.4 Sun trajectory relative to Earth

- **Mean longitude** \( l_s = 4.8951 + 6.283076 \theta \)
- **True longitude** \( L_s = 4.8952 + 6.283320 \theta \\
-0.0075 \sin(e_l) - 0.0326 \cos(e_l) \\
-0.0003 \sin(2e_l) + 0.0002 \cos(2e_l) \)

| true latitude | 0 |
obliquity \[ \epsilon_s = 0.409093 \]

deciliation \[ \delta_s = \text{Asin}(\sin(\epsilon_s) \sin L_s) \]

right ascension \[
\begin{align*}
0 & \leq \alpha_s \leq 2\pi \\
\cos(\alpha_s) \cos(\delta_s) &= \cos(L_s) \\
\sin(\alpha_s) \cos(\delta_s) &= \sin(L_s) \cos(\epsilon_s)
\end{align*}
\]

equation of time
(true solar time − mean solar time) \[
Et = 591.8 \sin(2l_s) - 459.4 \sin(e_M) + 39.5 \sin(e_M) \cos(2l_s) - 12.7 \sin(4l_s) - 4.8 \sin(2e_M)
\]

Over the period 1980–2020, the relative accuracy on \( R_s \) is of \( 5 \times 10^{-4} \), the accuracy on the various angles is of \( 5 \times 10^{-4} \) \( \text{rd} \), and that on equation of time is of 10 \( \text{s} \). These constants implicitly define the length of the sidereal year:

\[
y_s = \frac{2\pi d y_j}{6.283076}
\]

and therefore the length of sidereal day:

\[
d_s = \frac{d}{1 + d/y_s}
\]

and earth rotation:

\[
\Omega = \frac{2\pi}{d_s}
\]

2 Geometry, geoid

2.1 Geometry

We mentioned in the introduction that the thin layer assumption is the base of Arpege-IFS equations. To make sense, it requires the choice of one surface on which the equations are written.

As we write the momentum equation in vorticity-divergence, the Laplacian operator must have his kernel reduced to constant functions; we suppose moreover than the surface is of revolution around the axis of rotation of the planet.
2.2 Coordinate system

On the horizontal, we use longitude $\lambda$ varying from 0 to $2\pi$ to parametrize the circles of revolution. The East is directed towards increasing longitudes. In the orthogonal direction, we use $\mu$ variable from $-1$ at South pole to $+1$ at North pole (by definition) to parametrize the surface generator.

On the vertical, we use a coordinate $\eta$ varying from 0 at the top of the fluid to 1 at the bottom. This vertical coordinate has no geometrical significance with the ordinary metrics. The 3d metrics is obtained as the product of horizontal metrics by vertical one.

2.3 Geoid

The preceding assumptions imply that the vertical coordinate does not have any geometrical significance and that gravity is not explicitly used. In place we need two infinitely close equipotential surfaces between which the equations are written. We make the additional assumption that for the description of the Earth, equipotential surfaces are spheres of radius $a$ (average value of the reference ellipsoid):

$$a = 6371229 \text{ m}$$

To transform an elevation value into geopotential in $J kg^{-1}$, it should be multiplied by the conventional value:

$$g = 9.80665 \text{ m s}^{-2}$$

If we used an ellipsoid instead of a sphere, gravity would vary with the latitude according to a formula close to that of Clairault, but it would not appear explicitly in the equations and the preceding remarks would remain valid.

Because of the sphericity assumption, the notions of “North, East, longitude . . . ” used above should not be taken in their geographical meaning since, as we will see further, the pole of the coordinate system is not necessarily in the Arctic.

3 Fundamental constants

<table>
<thead>
<tr>
<th>Constant</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>light speed</td>
<td>$c = 299792458 \text{ m s}^{-1}$</td>
</tr>
<tr>
<td>Planck’s constant</td>
<td>$h = 6.6260755 \times 10^{-34} \text{ J s}$</td>
</tr>
<tr>
<td>Boltzmann’s constant</td>
<td>$k = 1.380658 \times 10^{-23} \text{ J K}^{-1}$</td>
</tr>
<tr>
<td>Avogadro’s number</td>
<td>$N = 6.0221367 \times 10^{23} \text{ mol}^{-1}$</td>
</tr>
</tbody>
</table>
4 Radiation

Stefan-Boltzmann’s constant

$$\sigma = \frac{2\pi^5 k^4}{15c^2h^3}$$

solar constant

$$I_0 = 1370 \text{ W m}^{-2}$$

5 Thermodynamics, gas phase

The fluid is a mixture of dry air, of water in gas, liquid and solid phases.

- gas constant
  $$\mathcal{R} = Nk$$
- dry air molar mass
  $$M_a = 28.9644 \times 10^{-3} \text{ kg mol}^{-1}$$
- water vapor molar mass
  $$M_v = 18.0153 \times 10^{-3} \text{ kg mol}^{-1}$$

$$R_a = \frac{\mathcal{R}}{M_a} \text{ J kg}^{-1} \text{ K}^{-1}$$

$$R_v = \frac{\mathcal{R}}{M_v} \text{ J kg}^{-1} \text{ K}^{-1}$$

It is supposed here that dry air and water vapor are perfect gases. The maximum error is 0.1 \%

$$c_{pa} = \frac{7}{2} R_a$$

$$c_{va} = \frac{5}{2} R_a$$

These quantities are not constant in the atmosphere. But this assumption is coherent with the approximation of perfect gases, and the error introduced is less than 1 \%.

$$c_{pv} = 4 R_v$$

$$c_{vv} = 3 R_v$$

In the case of water vapor, supposing them constant leads to an error less than 5 \%.

If one activates the optional “prognostic physical parametrizations”, atmosphere contains also four species: cloud water, cloud ice, rain water and rain ice.
6 Thermodynamics, liquid phase

water molar mass \( M_l = M_v \)

massic volume \( v_l = 0 \)

\( c_{pl} = c_{vl} = c_l = 4.218 \times 10^3 \text{ J kg}^{-1} \text{K}^{-1} \) (value at triple point \( T_l \))

The identity between \( c_{pl} \) and \( c_{vl} \) is very well satisfied and is coherent with the constancy of the massic volume. The fact that \( c_l \) is constant is satisfied with less than 1\% error in the temperature range \([0 ^\circ C, 30 ^\circ C]\), but the error grows for the negative temperatures and reaches 12.5\% at \(-40 ^\circ C\).

7 Thermodynamics, solid phase

\( M_g = M_l \)

\( v_g = v_l \)

\( c_{pg} = c_{vg} = c_g = 2.106 \times 10^3 \text{ J kg}^{-1} \text{K}^{-1} \) (value at \( T_l \))

Actually, \( c_g \) decreasing linearly with the temperature, the error introduced is 13\% at \(-40 ^\circ C\).

8 Thermodynamics, phase transition

\( T_l = 273.16 \text{ K} \)

8.1 Vaporization

\( L_v(T) = L_v(T_l) + (c_{pv} - c_l)(T - T_l) \)

\( L_v(T_l) = 2.5008 \times 10^6 \text{ J kg}^{-1} \)

It is supposed that \( L_v \) is independent of the pressure, which is accurate at 0.5\%, and is coherent with \( v_l = 0 \).
8.2 Sublimation

\[ L_s(T) = L_s(T_t) + (c_{pv} - c_g)(T - T_t) \]

\[ L_s(T_t) = 2.8345 \times 10^6 \text{ J kg}^{-1} \]

c\textit{pv} - c\textit{g} is an order of magnitude weaker than c\textit{pv} - c\textit{l}; however, to neglect the variation of \( L_s \) with temperature, it would be necessary to write: \( c_g = c_{pv} \).

8.3 Melting

\[ L_f = L_s - L_v \]

9 Consequences on saturation

With \( \nu_l = 0 \), Clapeyron’s equation becomes:

\[ \frac{d \ln(e_s)}{dT} = \frac{L_v}{R_v T^2} \]

Using the expression of \( L_v \), and integrating from:

\((T_t, e_s(T_t) = 611.14 \text{ Pa})\)

yields:

\[ \ln(e_s) = \alpha_l - \frac{\beta_l}{T} - \gamma_l \ln T \]

with:

\[ \alpha_l = \ln(e_s(T_t)) + \frac{\beta_l}{T_t} + \gamma_l \ln T_t \]
\[ \beta_l = \frac{L_v(T_t)}{R_v} + \gamma_l T_t \]
\[ \gamma_l = \frac{c_l - c_{pv}}{R_v} \]

In the presence of ice, the formula remains valid if the \( l \) are replaced by \( g \).
10 Thermodynamic functions

The thermodynamic functions are gathered in the block of declarations FCT-TRM. They are divided into two groups: the first corresponds to the absolute functions and to the second to the approximate functions whose approximations are consistent with the above statements. This module is inserted in all the subroutines which require thermodynamic calculations. Thus coherence between the various parts of the code is ensured.

Notations:

- $T_t$ temperature of water triple point
- $\gamma_l = (c_l - c_{pv})/R_v$
- $\gamma_i = (c_i - c_{pv})/R_v$
- $\beta_l = L_v(T_t)/R_v + \gamma_lT_t$
- $\beta_i = L_s(T_t)/R_v + \gamma_iT_t$
- $\alpha_l = \ln e_s(T_l) + \beta_l/T_t + \gamma_l\ln T_l$
- $\alpha_i = \ln e_s(T_i) + \beta_i/T_t + \gamma_i\ln T_l$
- $\delta$ index for liquid/solid calculation: $\delta = \begin{cases} 1 & \text{if liquid whatever } T \\ 0 & \text{if solid whatever } T \end{cases}$
- $\delta_T$ index of temperature positivity: $\delta_T = \begin{cases} 1 & \text{if } T \geq T_t \\ 0 & \text{otherwise} \end{cases}$

10.1 Absolute Functions

Latent heat of vaporization:

$$L_v(T) = RLV(T) = L_v(T_t) + (c_{pv} - c_l)(T - T_t)$$

Latent heat of sublimation:

$$L_s(T) = RLS(T) = L_s(T_t) + (c_{pv} - c_i)(T - T_t)$$

Latent heat of fusion:

$$L_f(T) = RLF(T) = L_s(T) - L_v(T)$$
Saturation pressure above liquid water:

\[ e_{sl}(T) = ESW(T) = \exp\left[\alpha_l - \frac{\beta_l}{T} - \gamma_l \ln T\right] \]

Saturation pressure above solid water:

\[ e_{si}(T) = ESS(T) = \exp\left[\alpha_i - \frac{\beta_i}{T} - \gamma_i \ln T\right] \]

Saturation pressure above liquid/solid water:

\[ e_s(T) = ES(T) = \exp\left[\alpha_l + (\alpha_i - \alpha_l)\delta - \frac{\beta_l + (\beta_i - \beta_l)\delta}{T} - (\gamma_l + (\gamma_i - \gamma_l)\delta) \ln T\right] \]

### 10.2 Functions in the model parametrizations

Saturation pressure:

\[ e_s(T, \delta) = \text{FOEW}(T, \delta) \]

\[ = \exp\left[\alpha_l + (\alpha_i - \alpha_l)\delta - \frac{\beta_l + (\beta_i - \beta_l)\delta}{T} - (\gamma_l + (\gamma_i - \gamma_l)\delta) \ln T\right] \]

Derivative of the logarithm of the saturation pressure:

\[ \frac{\partial \ln e_s(T, \delta)}{\partial T} = \text{FODLEW}(T, \delta) = \frac{\beta_l + (\beta_i - \beta_l)\delta - (\gamma_l + (\gamma_i - \gamma_l)\delta) T}{T^2} \]

Saturation specific moisture:

\[ q_s = \text{FOQS}\left(\frac{e_s}{p}\right) = \frac{e_s/p}{1 + (R_a/R_v - 1) \max(0, 1 - e_s/p)} \]

This formulation makes it possible to have:

\[ q_s = \begin{cases} \frac{e_s/R_v}{(p - e_s)/R_a + e_s/R_v} & \text{if } e_s(T) \leq p \\ \frac{e_s}{p} & \text{if } e_s(T) \geq p \end{cases} \]
1. Basic hypotheses and related constants

Derivative saturation of specific moisture:

\[
\frac{\partial q_s}{\partial T} = FODQS(q_s, \frac{e_s}{p}, \frac{\partial \ln e_s}{\partial T}) = \frac{q_s - q_s^2}{1 - e_s/p} \frac{\partial \ln e_s}{\partial T}
\]

Latent heat:

\[
L(T, \delta) = FOLH(T, \delta) = R_v \left[ \beta_l + (\beta_i - \beta_l)\delta - (\gamma_l + (\gamma_i - \gamma_l)\delta)T \right] = L_v(T_l) + \left[ L_s(T_l) - L_v(T_l) \right] \delta + \left[ c_{pv} - c_l + (c_l - c_i)\delta \right] (T - T_l)
\]
1. Basic hypotheses and related constants
Dynamics equations

1 Introduction

Each equation of the model can generally write as:

\[
\frac{dX}{dt} = \mathcal{A} + \mathcal{F}
\]

where \( X \) is a prognostic variable, the evolution of which one wants to know. \( \mathcal{A} \) represents all the effects which can be explicitly represented for the current resolution (often named “adiabatic effects”). They are:

- Coriolis force (momentum equation)
- pressure-gradient force term (momentum equation)
- conversion term (temperature equation)
- divergence term (continuity equation)

\( \mathcal{F} \) represents all the sub-scale effects (often named “diabatic effects”) which are calculated by physical parametrization routines. They are:

- radiation
- clouds and turbulence
- large-scale precipitations
- vertical diffusion
2. Dynamics equations

- convection
- orographic gravity wave drag
- soil, snow and vegetation

The time derivative of $X$ means the total temporal derivative, including advection, also known as Lagrangian derivative.

2 Primitive equations in Eulerian form

Making the hydrostatic assumption, we use for vertical coordinate a hybrid coordinate $\eta(p, p_s)$ derived from the pressure coordinate $p$ and terrain following. It must satisfy:

$$\begin{cases}
\eta(0, p_s) = 0 \\
\eta(p_s, p_s) = 1 \\
\frac{\partial \eta}{\partial p}(p, p_s) > 0
\end{cases}$$

This vertical coordinate $\eta$ is defined by two functions $A(\eta)$ and $B(\eta)$, in such a way that the pressure at a given point is:

$$p = A(\eta) + B(\eta)p_s$$

where $p_s$ is surface pressure. We have:

$$A(0) = 0 \quad A(1) = 0 \\
B(0) = 0 \quad B(1) = 1$$

ensuring for $\eta$-surfaces to follow orography at the bottom and to be pressure surfaces at the top. The model does not need to explicitly know the functional form of $A$ and $B$, only their values at the interface of the layers are necessary.

The hydrostatic assumption leads to the equation:

$$\frac{\partial \Phi}{\partial \eta} = -\frac{RT}{p} \frac{\partial p}{\partial \eta}$$

which is used as a diagnostic equation to calculate geopotential $\Phi$ on level $p$ by an integral starting at the lower boundary condition $\Phi(p_s) = \Phi_s$. 
The evolution of the parameters which define the state of the atmosphere, horizontal wind \( \vec{v} \), temperature \( T \) and mass moisture ratio \( q_v \) is controlled by the following equations, where the total temporal derivative is written as:

\[
\frac{dX}{dt} = \frac{\partial X}{\partial t} + \vec{v} \nabla X + \dot{\eta} \frac{\partial X}{\partial \eta} \quad (1)
\]

**Momentum equation**

\[
\frac{d\vec{v}}{dt} + 2\Omega \times \vec{v} + RT \nabla \ln p + \nabla \Phi = -g \frac{\partial \eta}{\partial p} \frac{\partial F_{\vec{v}}}{\partial \eta} + S_{\vec{v}} + K_{\vec{v}} \quad (2)
\]

To conserve momentum in the vertical discretization, the acceleration term due to the pressure force is transformed into:

\[
\frac{\partial \eta}{\partial p} \left( \Phi \nabla \frac{\partial p}{\partial \eta} - \frac{\partial \Phi}{\partial \eta} \nabla p \right) + \nabla \Phi
\]

**Thermodynamics equation**

\[
\frac{dT}{dt} + \kappa T \frac{\omega}{p} = -g \frac{\partial \eta}{\partial p} \frac{\partial F_h}{\partial \eta} + S_h + K_h \quad (3)
\]

**Moisture equation**

\[
\frac{dq_v}{dt} = -g \frac{\partial \eta}{\partial p} \frac{\partial F_{q_v}}{\partial \eta} + S_{q_v} + K_{q_v} \quad (4)
\]

In the above equations one takes:

\[
R = q_a R_a + q_v R_v
\]
\[
c_p = q_a c_{p_a} + q_v c_{p_v}
\]
\[
\kappa = \frac{R}{c_p}
\]

The terms in the right-hand members of Equations (2), (3) and (4) respectively represent vertical fluxes (noted \( F \)), sources (noted \( S \)) and horizontal diffusion (noted \( K \)) of momentum, enthalpy, and specific moisture.

The continuity equation is written as:

\[
\frac{\partial}{\partial \eta} \left( \frac{\partial p}{\partial t} \right) + \nabla \cdot \left( \vec{v} \frac{\partial p}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( \dot{\eta} \frac{\partial p}{\partial \eta} \right) = -g \frac{\partial F_{\vec{v}}}{\partial \eta} \quad (5)
\]
2. Dynamics equations

$F_p$ is the mass flux, no source term being considered.

By integrating, one obtains the evolution equation of surface pressure:

$$\frac{\partial p_s}{\partial t} = - \int_0^1 \nabla \cdot \left( \vec{v} \frac{\partial p}{\partial \eta} \right) d\eta - g F_p(1)$$

vertical velocity in pressure coordinate:

$$\omega = - \int_0^\eta \nabla \cdot \left( \vec{v} \frac{\partial p}{\partial \eta} \right) d\eta + \vec{v} \cdot \nabla p - g F_p(\eta)$$

and vertical velocity:

$$\frac{\partial \eta}{\partial \eta} = \frac{\partial p}{\partial t} - \int_0^\eta \nabla \cdot \left( \vec{v} \frac{\partial p}{\partial \eta} \right) d\eta - g F_p(\eta)$$

The momentum equations are integrated divergence and rotational form:

$$\frac{\partial \zeta}{\partial t} = \nabla \times \left( H_u \vec{v} - g \frac{\partial \eta}{\partial p} \frac{\partial F_v}{\partial \eta} + S_v \right) + K \zeta$$

$$\frac{\partial D}{\partial t} = \nabla \cdot \left( H_v \vec{v} - g \frac{\partial \eta}{\partial p} \frac{\partial F_v}{\partial \eta} + S_v \right) - \Delta (\Phi + E_c) + K D$$

with:

$$H_u = (\zeta + f) v - \eta \frac{\partial u}{\partial \eta} + \frac{\partial \eta}{\partial p} \frac{\partial \Phi}{\partial \eta} \frac{1}{a} \frac{\partial p}{\partial \lambda}$$

$$H_v = -(\zeta + f) u - \eta \frac{\partial v}{\partial \eta} + \frac{\partial \eta}{\partial p} \frac{\partial \Phi}{\partial \eta} \frac{1 - \mu^2}{a} \frac{\partial p}{\partial \mu}$$

$$E_c = \frac{1}{2} (u^2 + v^2)$$

The wind is calculated from velocity potential $\chi$ and stream function $\psi$ by:

$$\vec{v} = \nabla \chi + \nabla \times \psi$$

Velocity potential and stream function are obtained from divergence and vorticity by solving Poisson equations:
\[ \chi = \Delta^{-1} D \]
\[ \psi = \Delta^{-1} \zeta \]

It is in these last three relations, the kernel of the Laplacian \( \Delta \) is implicitly supposed to reduce to constant functions. It is then equivalent to know the wind or divergence and vorticity pair. This property is true on the sphere as well as on the torus.

3 Variable mesh

3.1 Stretched and tilted grid

ARPEGE makes it possible to increase the horizontal resolution on part of the sphere, while preserving locally the isotropy (Courtier and Geleyn, 1988). For that one uses a new set of coordinates \((\lambda', \mu')\). First, North Pole is shifted at the point of coordinates \((\lambda_0, \mu_0)\) which becomes the new pole (or tilted pole). One defines new coordinates \((\lambda_b, \mu_b)\) by:

\[
\mu_b = \mu_0 \mu + \sqrt{1 - \mu_0^2} \sqrt{1 - \mu^2} \cos(\lambda - \lambda_0)
\]
\[
\cos \lambda_b = (1 - \mu_b^2)^{-\frac{1}{2}} (\mu \sqrt{1 - \mu_0^2} - \mu_0 \sqrt{1 - \mu^2} \cos(\lambda - \lambda_0))
\]
\[
\sin \lambda_b = (1 - \mu_b^2)^{-\frac{1}{2}} \sqrt{1 - \mu^2} \sin(\lambda_0 - \lambda)
\]

The origin longitude is the one which contains the geographical North Pole. Reciprocally:

\[
\mu = \mu_0 \mu_b + \sqrt{1 - \mu_0^2} \sqrt{1 - \mu_b^2} \cos \lambda_b
\]
\[
\cos(\lambda - \lambda_0) = (1 - \mu^2)^{-\frac{1}{2}} (\mu_b \sqrt{1 - \mu_0^2} - \mu_0 \sqrt{1 - \mu^2} \cos \lambda_b)
\]
\[
\sin(\lambda_0 - \lambda) = (1 - \mu^2)^{-\frac{1}{2}} \sqrt{1 - \mu_b^2} \sin \lambda_b
\]

Then, one carries out a stretching of the latitudes (without modifying longitudes, \(\lambda' = \lambda_b\)) obtained by homothety of a factor \(c\) on stereographic projection at the pole of stretching. It comes:

\[
\mu' = \frac{(1 - c^2) + (1 + c^2) \mu_b}{(1 + c^2) + (1 - c^2) \mu_b}
\]
and reciprocally:

\[
\mu_b = \frac{(c^2 - 1) + (c^2 + 1) \mu'}{(c^2 + 1) + (c^2 - 1) \mu'}
\]
3.2 Impact on the equations

The conformal transform of $\eta$-surfaces described above changes the model equations. The fields are represented by a base of functions defined on transformed surfaces. The equations are integrated on original surfaces. However modifications are necessary in the calculation of the horizontal derivative. It consists of multiplying them by a scale factor $m$. In the case of the present transform, this factor is:

$$m = \frac{c^2 + 1}{2c} + \frac{c^2 - 1}{2c} \mu'$$

At the pole of dilation ($\mu' = 1$) this factor is $c$. At the pole of contraction ($\mu' = -1$) it is $1/c$. The horizontal wind thus becomes:

$$\vec{v} = m \vec{v}'$$

and the horizontal gradient:

$$\nabla = m \nabla'$$

In going from the real sphere to the transformed sphere, the velocity potential $\chi$ and the stream function $\psi$, which are scalars, are invariant. The main modification consists of solving Poisson equation in a more complicated form:

$$\chi = \Delta^{-1} \frac{D}{m^2}$$
$$\psi = \Delta^{-1} \frac{\zeta}{m^2}$$

where $\Delta^{-1}$ is the same formal operator as $\Delta^{-1}$ but on the transformed sphere. As a consequence, the state variable of the model is: $\zeta' = \zeta/m^2$ and $D' = D/m^2$ (or equivalently $\psi$ and $\chi$).

The equations become then:

$$\frac{\partial \zeta'}{\partial t} = \nabla' \times \left[ \frac{1}{m} \left( \vec{H}_{\theta} - g \frac{\partial \eta}{\partial \rho} \frac{\partial \vec{E}_{\theta}}{\partial \eta} + \vec{S}_{\theta} \right) \right] + K_{\zeta'}$$
$$\frac{\partial D'}{\partial t} = \nabla' \cdot \left[ \frac{1}{m} \left( \vec{H}_{\theta} - g \frac{\partial \eta}{\partial \rho} \frac{\partial \vec{E}_{\theta}}{\partial \eta} + \vec{S}_{\theta} \right) \right] - \Delta' (\Phi + E_c) + K_{D'}$$
3.3 Impact on post-processing

The files produced by ARPEGE-CLIMAT for restarting as well as for post-processing contain the model prognostic variables, even though, in the case of post-processing, $\psi$ and $\chi$ are transformed into $\vec{v}'$. This is also true for momentum fluxes. As a consequence, both components of wind velocity or surface stress have to be multiplied by $m$ before any comparison with observations or other model outputs. This operation can be done, for example, at the same time as the conversion from ARPEGE format to another format.

4 Lagrangian form of the primitive equations

The Lagrangian form of the equations of momentum, thermodynamics and moisture are respectively (2), (3) and (4). The continuity equation (5) has as a Lagrangian form:

$$\frac{d}{dt} \left( \frac{\partial p}{\partial \eta} \right) = -\frac{\partial p}{\partial \eta} \left( D + \frac{\partial \eta}{\partial \eta} \right) - g \frac{\partial F_p}{\partial \eta} \quad (6)$$

It can take another form, more adapted to the semi-Lagrangian advection method:

$$\frac{d}{dt} \left[ \left( \frac{\partial F_p}{\partial \eta} \right) J \right] = -g \frac{\partial F_p}{\partial \eta} \quad (7)$$

where $J$ indicates the Jacobian of the transform which associates the position of a point at time $t$ with its position at a reference time $t_0$.

On the stretched and tilted sphere, the continuity equation (6) becomes:

$$\frac{d}{dt} \left( \frac{\partial p}{\partial \eta} \right) = -\frac{\partial p}{\partial \eta} \left( m^2 D' + \frac{\partial \eta}{\partial \eta} \right) - g \frac{\partial F_p}{\partial \eta}$$

and its Lagrangian form (7):

$$\frac{d}{dt} \left[ \left( \frac{\partial F_p}{\partial \eta} \right) J' \right] = -g \frac{\partial F_p}{\partial \eta}$$

The momentum equation (2) becomes:

$$\frac{d}{dt} \left[ m \vec{v}' \right] + m \left[ 2\Omega \times \vec{v}' + RT \nabla' \ln p + \nabla' \Phi \right] = -g \frac{\partial \eta}{\partial p} \frac{\partial F_p}{\partial \eta} + \vec{S} + \vec{K}$$

The equations for thermodynamics (3) and moisture (4) are formally unchanged.
5 Vertical discretization

5.1 Model vertical levels

The atmosphere is vertically split into $L$ layers, defined by the pressures at their interfaces, which are calculated by:

$$p_{\tilde{\ell}} = A_{\tilde{\ell}} + B_{\tilde{\ell}} p_s \quad \tilde{\ell} = 0, \ldots, L$$  \hfill (8)

$A_{\tilde{\ell}}$ and $B_{\tilde{\ell}}$ are constants which define the vertical coordinate. The vertical distribution of the variables is presented in Figure 1, indices $\ell$ relating to the mid-layers (also named full levels) and indices $\tilde{\ell}$ to the inter-layers (also named half levels).

The values of $A_{\tilde{\ell}}$ and $B_{\tilde{\ell}}$ are imposed to the model. In earlier versions of ARPEGE-CLIMAT they were calculated from analytical functions. In recent cycles, one uses the same vertical discretization as in forecast models (Météo-France or ECMWF).

5.2 Vertical discretization of the equations

The vertical discretization scheme is defined according to Simmons and Burridge (1981). One introduces the operator $\delta$ which represents the variation...
of a variable between the two ends of a layer:

\[ \delta p_{\ell} = p_{\ell} - p_{\ell-1} \]

Index \( \ell \) will be omitted when there is no ambiguity.

**Continuity equation**

The continuity equation is written as (\( F_m \) being the mass flux due to water cycle):

\[
\frac{\partial}{\partial \eta} \left( \frac{\partial p}{\partial t} \right) + \nabla \cdot \left( \vec{v} \frac{\partial p}{\partial \eta} \right) + \frac{\partial}{\partial \eta} \left( \dot{\eta} \frac{\partial p}{\partial \eta} \right) = F_m
\]

For a given layer, one writes it as:

\[
\frac{\partial (\delta p)}{\partial t} = -\nabla \cdot (\vec{v} \delta p) - \delta \left( \dot{\eta} \frac{\partial p}{\partial \eta} \right) + F_m
\]

One will note, in the following, the physical term of the discretized equation in the same way as the corresponding term of the continuous equation.

Summing on the vertical, one obtains the evolution equation of surface pressure:

\[
\frac{\partial p_s}{\partial t} = -\sum_{\ell=1}^{L} \left[ \delta p D + \delta B \vec{v} \cdot \nabla p_s \right] - g(P + E)
\]

since:

\[
\nabla \delta p = \delta B \nabla p_s
\]

The vertical speed is obtained by summing the continuity equation from the top to the current level:

\[
w_{\ell} = \left( \dot{\eta} \frac{\partial p}{\partial \eta} \right)_{\ell} = \sum_{k=1}^{\ell} \left[ -\delta B_k \vec{v}_k \cdot \nabla p_s - \delta p_k D_k + F_{mk} \right] - B_{\ell} \frac{\partial p_s}{\partial t}
\]
Eulerian advection

The vertical advections in momentum, thermodynamics and moisture equations are calculated using the scheme:

\[
\left( w \frac{\partial X}{\partial p} \right)_\ell = \frac{1}{2\delta p} \left[ w_\ell (X_{\ell+1} - X_{\ell}) + w_{\ell-1} (X_{\ell} - X_{\ell-1}) \right]
\]

This scheme ensures conservation of \( X \) and \( X^2 \). It results from the following form of the vertical advection:

\[
w \frac{\partial X}{\partial p} = \frac{\partial X}{\partial p} - X \frac{\partial w}{\partial p}
\]

with the interpolation:

\[
X_\ell = \frac{1}{2} (X_{\ell} + X_{\ell+1})
\]

In the case of the semi-Lagrangian scheme, see Chapter ??.

Hydrostatic equation

The equation of hydrostatic balance is integrated by using the centered scheme:

\[
\Phi_\ell_{-1} = \Phi_\ell - R_\ell T_\ell \ln \frac{p_{\ell-1}}{p_\ell}
\]

Which gives, summing from the surface:

\[
\Phi_\ell = \Phi_s + \sum_{k=L}^{\ell+1} R_k T_k \ln \frac{p_k}{p_{k-1}}
\]

where \( \Phi_s \) indicates surface geopotential.

To calculate the geopotential in the mid-layers one writes:

\[
\Phi_\ell = \Phi_\ell + \alpha_\ell R_\ell T_\ell
\]

where:

\[
\begin{align*}
\alpha_1 &= 1 \\
\alpha_\ell &= 1 - \frac{p_{\ell-1}}{\delta p_\ell} \ln \frac{p_\ell}{p_{\ell-1}}
\end{align*}
\]

The expression of \( \Phi_\ell \) is consistent with the discretization of the form:

\[
\Phi = \frac{\partial p \Phi}{\partial p} + RT
\]
Discretization of the pressure force

For conserving angular momentum, one writes the acceleration term due to the pressure force as:

\[
\nabla \Phi + RT \nabla \ln p = \left[ \nabla \left( \Phi \frac{\partial p}{\partial \eta} \right) - \frac{\partial (\Phi \nabla p)}{\partial \eta} \right] \left( \frac{\partial p}{\partial \eta} \right)^{-1} \]

\[
= \nabla \Phi \left[ \Phi \nabla \left( \frac{\partial p}{\partial \eta} \right) - \frac{\partial (\Phi \nabla p)}{\partial \eta} \right] \left( \frac{\partial p}{\partial \eta} \right)^{-1} \]

The discretization of this term yields:

\[
(RT \nabla \ln p)_{\ell} = R_{\ell} T_{\ell} \frac{1}{\delta p_{\ell}} \left[ \delta B_{\ell} + \frac{C_{\ell}}{\delta p_{\ell}} \ln \frac{p_{\ell}}{p_{\ell-1}} \right] \nabla_{s} \cdot p_{s} \]

with:

\[
C_{\ell} = A_{\ell} B_{\ell-1} - A_{\ell-1} B_{\ell} \]

Term ZRTGR is also used in the calculation of the energy transformation term. It can be written in the simpler form:

\[
\frac{1}{\delta p} \left[ \alpha_{\ell} \delta B_{\ell} + B_{\ell-1} \ln \frac{p_{\ell}}{p_{\ell-1}} \right] \]

Energy transformation term

The energy transformation term is written as:

\[
\frac{RT \omega}{c_p \cdot p} \]

One writes:

\[
\frac{\omega}{p} = \vec{v} \cdot \nabla \ln p - \frac{1}{p} \int_{0}^{p} \left[ \vec{v} \cdot \nabla \left( \frac{\partial p}{\partial \eta} \right) + D \frac{\partial p}{\partial \eta} + F \right] d\eta \]
The first term of the right-hand member is evaluated in the same way as the corresponding term of the momentum equation:

\[
(\vec{v} \cdot \nabla \ln p)_\ell = \frac{1}{\delta p_\ell} \left[ \delta B_\ell + \frac{C_\ell}{\delta p_\ell} \ln \frac{p_\ell}{p_{\ell-1}} \right] \vec{v}_\ell \cdot \nabla p_s
\]

and the second term:

\[
-\frac{1}{\delta p_\ell} \left[ \alpha_\ell (\nabla \cdot (\vec{v}_\ell \delta p_\ell) + F_{m_\ell}) + \left( \ln \frac{p_\ell}{p_{\ell-1}} \right) \sum_{k=1}^{\ell-1} (\nabla \cdot (\vec{v}_k \delta p_k) + F_{m_k}) \right]
\]

where:

\[
\nabla \cdot (\vec{v}_\ell \delta p_\ell) = \vec{v}_\ell \cdot \nabla \delta p_\ell + \delta p_\ell D_\ell
\]

This discretization results from writing the last term in the form:

\[
\frac{1}{p} \int X d\eta = \left( \frac{\partial p}{\partial \eta} \right)^{-1} \frac{\partial \ln p}{\partial \eta} \int X d\eta = \left[ \frac{\partial}{\partial \eta} \left( \ln p \int X d\eta \right) - X \ln p \right] \left( \frac{\partial p}{\partial \eta} \right)^{-1}
\]

in which one calculates the last logarithm of the pressure by:

\[
\ln p = \frac{d}{dp} (p \ln p) - 1
\]
Spectral transforms

1 Introduction

ARPEGE-IFS is a spectral global model. One part of computations is made in spectral space (semi-implicit scheme, horizontal diffusion scheme), the other part in grid-point space on a grid defined by a Gaussian quadrature. It is therefore necessary to perform spectral transforms from spectral space to grid-point space or vice-versa. The present chapter aims at giving a brief summary of the spectral method. For more algorithmic details one can report to Rochas and Courtier (1992) or to Temperton (1991). Computation aspects are described in details in Yessad (2007a). Most subroutines are located in an independent library named TFL.

For a global spectral model, spectral transforms are a combination of a Legendre transform and a Fourier transform. A spectral limited-area model like ALADIN uses a double Fourier representation for spectral fields.

2 Spectral representation

2.1 Spherical harmonics

The spherical Laplacian operator $\Delta$ on a sphere $\Sigma$ of radius $a$ admits as eigenvalues family $-n(n+1)/a^2$ with an order of $2n+1$. The eigenvectors are the surface spherical harmonics. An orthogonal base of an eigenspace is given by:

$$Y_n^m = P_n^m(\mu)e^{im\lambda}$$
where \( \mu \) is the sine of the latitude and \( \lambda \) longitude. The \( P_n^m(\mu) \) are the first-type Legendre polynomials. The standardization of the \( P_n^m(\mu) \) is such as:

\[
\int_{\sigma} Y_n^m(\lambda, \mu) Y_{n'}^{m'}(\lambda, \mu) d\sigma = \delta_{n=n'} \delta_{m=m'} \int_{\sigma} d\sigma
\]

One has then, for \( m \geq 0 \):

\[
P_n^m(\mu) = \sqrt{\frac{(2n+1)(n-m)!}{(n+m)!}} \frac{1}{2^n n!} (1 - \mu^2)^{m/2} \frac{d^{n+m}}{d\mu^{n+m}} (\mu^2 - 1)^n
\]

And for \( m \leq 0 \):

\[
P_n^{-m}(\mu) = P_n^m(\mu)
\]

As mentioned above, the spherical harmonics satisfy:

\[
\Delta Y_n^m = -\frac{n(n+1)}{\alpha^2} Y_n^m
\]

The Laplacian operator is invariant by rotation, his eigenspaces are thus also invariant by rotation. We deduce from it that under the effect of a rotation i.e. a change of pole, the coefficients of the decomposition of a field in spherical harmonics are exchanged at fixed \( n \). For each \( n \), there is a linear transformation (thus a matrix) which makes it possible to make the basic change. This property is preserved by the discretization if truncation is triangular. This is why triangular truncation is said to be isotropic.

### 2.2 Collocation grid

A collocation grid is selected for non-linear calculations which cannot be carried out directly on the coefficients of the spherical harmonics. At each time step, one passes from the spectral coefficients to the grid-point values and reciprocally. From the expression of the \( Y_n^m \), the E-W transforms are Fourier transforms. To use fast Fourier transforms (FFT), one thus needs a regular grid in longitude. In the N-S direction, one uses a Gauss quadrature for the direct transform, therefore the latitudes are not equidistant.

At high latitudes, one takes less points on a latitude circle than Fourier modes, in order to maintain the grid almost isotropic. The collocation grid is said to be reduced (Hortal and Simmons, 1991). The grid is said to be Gaussian quadratic (or simply Gaussian) when the number of latitude
3. Spectral transforms

3.1 Circles is large enough (for a given truncation) so that the Gauss quadrature is exact for any product of two Legendre polynomials in the truncation. This is useful in the case of the Eulerian advection (product of velocity by gradient). When the approximate calculation of the integral is exact only for the Legendre polynomials of the truncation, the grid is said to be linear (Hortal, 1996).

2.3 Spectral transforms

To pass from the spectral coefficients to the grid points values, one uses the two formulas of direct evaluation:

\[ A_m(\mu) = \sum_{n=|m|}^{\infty} A_n^m P_n^m(\mu) \]

and:

\[ A(\lambda, \mu) = \sum_{m=-\infty}^{+\infty} A_m(\mu) e^{im\lambda} \]

where the \( A_m \) are called the Fourier coefficients.

The horizontal derivatives are calculated exactly by using the derivatives of the Legendre functions for the N-S direction and by multiplying by \( im \) for the E-W direction.

From the grid point fields, the Fourier coefficients are determined by:

\[ A_m(\mu) = \frac{1}{2\pi} \int_0^{2\pi} A(\lambda, \mu) e^{-im\lambda} d\lambda \]

a formula which ensures that \( A_0(\mu) \) is the average of field \( A \) along parallel \( \mu \). Integration is carried out numerically by using a fast Fourier transform (FFT). The integral in latitude is:

\[ A_n^m = \frac{1}{2} \int_{-1}^{1} A_m(\mu) P_n^m(\mu) d\mu \]

As mentioned above, we use a Gauss quadrature, discrete version of the preceding integral:

\[ A_n^m = \sum_{k=1}^{K} \omega(\mu_k) A_m(\mu_k) P_n^m(\mu_k) \]
where the $\mu_k$ are the $K$ roots of the Legendre polynomial of degree $K$ and the Gauss weights $\omega(\mu_k)$ are given by:

$$\omega(\mu_k) = \frac{1 - \mu_k^2}{(NP_{N-1})^2}$$

Here, the Legendre polynomial is the one of the mathematicians, the squared norm of which is $1/(2n + 1)$.

3 Horizontal discretization

3.1 Spectral truncation

In practical the expression of $A$ is limited to a finite set of harmonics corresponding to $0 \leq n \leq N$ and $-n \leq m \leq n$. That defines a triangular truncation $N$. The truncated expansion of field $A$ reads:

$$A(\lambda, \mu) = \sum_{m=-N}^{N} \sum_{n=|m|}^{N} A^m_n P^m_n(\mu) e^{i m \lambda}$$

Due to the properties of $P^m_n(\mu)$, expression of $A$ becomes for a real scalar field:

$$A(\lambda, \mu) = \sum_{m=0}^{N} \sum_{n=|m|}^{N} A^m_n P^m_n(\mu) e^{i m \lambda}$$

3.2 Horizontal derivatives

Meridional derivative relative to latitude $\theta$

For a variable $A$, meridional derivative is discretized in spectral space by the following formula:

$$\left(\cos \theta \frac{\partial A}{\partial \theta}\right)_n^m = -(n - 1) e_n^m A_{n-1}^m + (n + 2) e_{n+1}^m A_{n+1}^m$$

where $e_0^m = 0$ and:

$$e_n^m = \sqrt{\frac{n^2 - m^2}{4n^2 - 1}}$$
3. Spectral transforms

Zonal derivative relative to longitude $\lambda$

For a variable $A$, zonal derivative is discretized in spectral space by the following formula:

$$\left( \frac{\partial A}{\partial \lambda} \right)_n^m = i m A_n^m$$

Such a derivation can be made on Fourier coefficients by a multiplication by $i m$.

3.3 Spectral relationships for wind representation

The reduced components of the velocity are obtained by dividing the physical components by the mapping factor $M$. Divergence and vorticity are divided by $M^2$. As reduced divergence $D'$ is obtained from velocity potential $\chi$ by a laplacian operator, and as reduced vorticity $\zeta'$ is obtained similarly from stream function $\psi$, we have in spectral space:

$$D_n^m = -\frac{n(n+1)}{a^2} \chi_n^m$$

$$\zeta_n^m = -\frac{n(n+1)}{a^2} \psi_n^m$$

Relationship between $U'$, $\psi$ and $\chi$:

$$(U' a \cos \theta) = \frac{\partial \chi}{\partial \lambda} - \cos \theta \frac{\partial \psi}{\partial \theta}$$

the spectral discretization of which is:

$$(U' a \cos \theta)_n^m = i m \chi_n^m + (n-1) c_n^m \psi_{n-1}^m - (n+2) c_{n+1}^m \psi_{n+1}^m$$

Relationship between $V'$, $\psi$ and $\chi$:

$$(V' a \cos \theta) = \frac{\partial \psi}{\partial \lambda} + \cos \theta \frac{\partial \chi}{\partial \theta}$$

the spectral discretization of which is:

$$(V' a \cos \theta)_n^m = i m \psi_n^m - (n-1) c_n^m \chi_{n-1}^m + (n+2) c_{n+1}^m \chi_{n+1}^m$$
Relationship between $D'$, $U'$ and $V'$:

$$D' = \frac{1}{a \cos \theta} \left( \frac{\partial U'}{\partial \lambda} + \cos \theta \frac{\partial (V' \cos \theta)}{\partial \theta} \right)$$

which can be rewritten:

$$D' = \frac{1}{a^2 \cos^2 \theta} \left( \frac{\partial (U' a \cos \theta)}{\partial \lambda} + \cos \theta \frac{\partial (V' a \cos \theta)}{\partial \theta} \right)$$

Relationship between $\zeta'$, $U'$ and $V'$:

$$\zeta' = \frac{1}{a \cos \theta} \left( \frac{\partial V'}{\partial \lambda} - \frac{\partial (U' \cos \theta)}{\partial \theta} \right)$$

which can be rewritten:

$$\zeta' = \frac{1}{a^2 \cos^2 \theta} \left( \frac{\partial (V' a \cos \theta)}{\partial \lambda} - \cos \theta \frac{\partial (U' a \cos \theta)}{\partial \theta} \right)$$

Spectral discretizations allow to retrieve easily spectral components of fields $D' a^2 \cos^2 \theta$ and $\zeta' a^2 \cos^2 \theta$, but not directly spectral components of $D'$ and $\zeta'$ (requiring inversion of a penta-diagonal matrix). In fact, the algorithm involved to retrieve spectral coefficients of $D'$ and $\zeta'$ once known values of wind components is slightly different (requiring a division by $a \cos \theta$ in Fourier space), and is described in detail in Temperton (1991).

3.4 Relationship between dimension in spectral space and in grid point space

Quadratic grid, linear grid

Spectral space is defined by a triangular truncation $N$. Grid point space has $ndgl$ latitudes and a maximum number of longitudes equal to $ndlon$. $ndlon$ and $ndgl$ are always even integers: if $ndlon$ is a multiple of 4, $ndgl = ndlon/2$; if $ndlon$ is not a multiple of 4, $ndgl = ndlon/2 + 1$. For a quadratic Gaussian grid, there is a relationship between these parameters to avoid aliasing on quadratic terms.

- If the stretching coefficient $c$ is equal to 1 (no stretching), $N$ is the maximum integer verifying the relationship $3 \times N \leq (ndlon - 1)$. 
• If the stretching coefficient $c$ is greater than 1 (stretching), $N$ is the maximum integer verifying the relationship $3 \ast N \leq \min(2 \ast ndgl - 3, ndlon - 1)$.

In a semi-Lagrangian scheme the advective quadratic terms disappear, so it is possible to use a smaller grid-point space: a linear grid. It is characterized by:

• If the stretching coefficient $c$ is equal to 1 (no stretching), $N$ is the maximum integer verifying the relationship $2 \ast N \leq (ndlon - 1)$.

• If the stretching coefficient $c$ is greater than 1 (stretching), $N$ is the maximum integer verifying the relationship $2 \ast N \leq \min(2 \ast ndgl - 3, ndlon - 1)$.

In ARPEGE-CLIMAT with $c > 1$, some aliasing is allowed, and the same $N$ is taken as in the case $c = 1$.

Admissible dimensions for longitude

The current algorithm for FFT allows integers $ndlon$ which can factorize as $2^{p_2} \ast 3^{p_3} \ast 5^{p_5}$. That limits the possibility of choosing the dimensions in a discontinuous subset of truncations and dimensions for Gaussian grid. In the range compatible with climate multi-year integrations, the admissible sizes (with even number of latitudes) are:

64 72 80 90 96 100 108 120 128 144 150 160 162 180 192 200 216 240 250 256 270 288 300 320 324 360 384 400 432 450 480 486 500 512 540 576 600 640 648 720

Reduced grid

To save memory and computation time (in particular in the physical parametrizations), the number of longitudes per latitude circle is reduced outside the tropics, in order to maintain a quasi-isotropic grid (note that the spectral triangular truncation allows an isotropic representation of the fields, despite the accumulation of grid points near the poles). This optimization is done at the expense of an aliasing error (Williamson and Rosinski, 2000). An algorithm is proposed to compute for a given truncation, the number of longitudes per latitude circle which is the best compromise between accuracy in the spectral transform and an isotropy in the physical parametrizations.
Semi-lagrangian computations

The general purpose of this chapter is to describe the set of equations used, and also the way to integrate the dynamics of the model with the semi-Lagrangian method currently implemented in ARPEGE/IFS. The following points will be described: semi-Lagrangian formulation and discretisation for different sets of equations, semi-Lagrangian trajectory research, horizontal and vertical interpolations done in the semi-Lagrangian scheme, specific geometric problems met in this type of discretisation. An organigramme is provided. An introduction to tangent linear and adjoint code is provided. An example of namelist is provided.

1 Introduction.

1.1 General purpose 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 with the semi-Lagrangian method currently implemented in ARPEGE/IFS. Equations will be written without horizontal diffusion scheme (which is treated in spectral computations), and without Rayleigh friction (which is done in grid-point space) in order to give a clearer presentation of the discretised equations. For additional information about horizontal diffusion scheme, report to documentation (IDDH). Extensions to ALADIN (cycle AL37T1) are not described in detail, but differences are briefly mentioned.

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

- The 2D shallow-water equations model (configuration 201).
The primitive equations hydrostatic model (configuration 1): thin layer and deep layer (according to White and Bromley, 1995) formulations.

The NH-PDVD non-hydrostatic equations model (configuration 1): uses $\hat{Q}$ and $d$ (or $d_4$) as NH prognostic variables.

The NH-GEOGW non-hydrostatic equations model (configuration 1): uses $\Phi$ and $gw$ as NH prognostic variables.

In the current version of this documentation, some points are partly described:

- The option with finite element vertical discretisations $\text{LVERTFE}=.T.$ . When $\text{LVERTFE}=.T.$, the main modifications in the semi-Lagrangian part of the code are the following ones:
  - $\hat{\eta} \frac{\partial \Pi}{\partial \eta}$ is directly computed at full levels so the way of computing $\hat{\eta}$ to find the vertical displacement is modified.
  - All the vertical integrals use a matricial multiplication with special coefficients computed in the setup routine $\text{SUVERTFE1}$ or $\text{SUVERTFE3}$; the vertical integration is done by routine $\text{VERINT}$.
  - All the vertical derivatives use a matricial multiplication with special coefficients computed in the setup routine $\text{SUVERTFE3D}$; the vertical derivation is done by routine $\text{VERDER}$.

- The spline cubic vertical interpolations used when $\text{LVSPILP}=.T.$ (in practical only for ozone, when $\text{YO3\_NL\%LVSPILP}=.T.$ ). The detail of calculation of the interpolation weights is not currently given.

- The “semi-Lagrangian horizontal diffusion” interpolations (SLHD) used when $\text{LSLHD}=.T.$ .

- Modified interpolations when $\text{L3DTURB}=.T.$ .

- The NH-PDVD non-hydrostatic model with prognostic variable $d_4$.

- The NH-PDVD non-hydrostatic model with option $\text{LRDBBC}=.T.$ .

- The NH-GEOGW non-hydrostatic model .

1.2 Distributed memory code.

Some distributed code has been introduced for the semi-Lagrangian scheme, for some convenience expressions such “DM-local" or “DM-global" will be used to describe some distributed memory features.
• Expression “DM-local” for a quantity means “local to the couple of processors \((proca, procb)\)”: each processor has its own value for the quantity. Expression “DM-local computations” means that the computations are made independently in each processor on “DM-local” quantities, leading to results internal to each processor, which can be different from a processor to another one.

• Expression “DM-global” for a quantity means that it has a unique value available in all the processors. Expression “DM-global computations” means that the computations are either made in one processor, then the results are dispatched in all the processors, or the same computations are made in all the processors, leading to the same results in all the processors.

• In a routine description the mention “For distributed memory computations are DM-local” means that all calculations made by this routine are DM-local; the mention “For distributed memory computations are DM-global” means that all calculations made by this routine are DM-global; when no information is provided it means that a part of calculations is DM-local and the other part is DM-global.

1.3 Mass corrector.

Two different mass correctors are coded (but do not work in all configurations), one used at ECMWF, the other one for METEO-FRANCE climatic simulations. They are useful especially for climatic simulations, to correct the lack of conservativity of the semi-Lagrangian scheme. They are not described in this documentation.

1.4 Tangent linear and adjoint codes.

Tangent linear and adjoint codes have been introduced for the shallow-water equations and the 3D hydrostatic model for a subset of options.

1.5 Deep layer equations (according to White and Bromley, 1995).

They have been introduced in the hydrostatic model for the Eulerian scheme and most options of the semi-Lagrangian scheme.
1.6 Deep layer equations (according to Wood and Staniforth, 2003).

They have been introduced in the NH-PDVD non-hydrostatic model for the option \texttt{LGWADV=T} only (in practical, semi-Lagrangian scheme only).

1.7 Modifications since cycle 37.

- New weights for interpolators, linked to \texttt{L3DTURB} or \texttt{NSPLITHO}.
- Cleanings in the code: gather weights and coordinates in new structures (see module intdyn_mod.F90).
- New GFL attribute \texttt{LPHYLIN} to do linear interpolations on physics for GFL.
- New choice \texttt{N(X)LAG=4}: possibility to do linear interpolations on physics for GMV, adiabatic terms being interpolated like for \texttt{N(X)LAG=3}.
- Possibility to separate linear terms and non-linear terms in interpolations for some DDH diagnostics.
- More flexible use of option \texttt{LSVTSM=T} (new variable \texttt{RPRES_SVTSM}).
- \texttt{LARCH2E} instead of \texttt{LARCH} for 2D models (idem for TL and AD).
2 Definition of Eulerian and semi-Lagrangian schemes.

2.1 Eulerian scheme.

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

$$\frac{\partial X}{\partial t} = -U \nabla_3 X + \dot{X}$$  \hspace{1cm} (1)

where $U$ is the 3D wind, $\nabla_3$ 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 satisfies to the CFL (Courant Friedrich Levy) condition everywhere.

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

$$M |V| \frac{Dt}{2} \sqrt{\frac{N(N+1)}{r^2}} < 1$$  \hspace{1cm} (2)

which can be rewritten:

$$M \frac{|V|}{r} \frac{Dt}{2} \sqrt{N(N+1)} < 1$$  \hspace{1cm} (3)

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), $|V|$ is the horizontal wind modulus, $N$ is the truncation, $r$ is the distance between the point and the centre of the Earth.

- For the spectral limited area model ALADIN, the horizontal CFL condition writes as:

$$M \frac{|V|}{r} \frac{Dt}{2} \left(2 \pi \right) \sqrt{\frac{1}{\frac{L_x^2}{a^2 N_m^2} + \frac{L_y^2}{a^2 N_n^2}}} < 1$$  \hspace{1cm} (4)

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), $|V|$ is the horizontal wind modulus, $N_m$ is the zonal truncation, $N_n$ is the meridian truncation, $a$ is the mean Earth radius, $r$ is the distance between the point and the centre of the Earth, $L_x$ (resp. $L_y$) is the zonal (resp. meridian) length of the ALADIN domain taken on a surface iso $r = a$.

The vertical CFL condition writes as:

$$|\dot{\eta}| \frac{Dt}{2} \Delta \eta < 1$$  \hspace{1cm} (5)

For a TL358L46 (triangular truncation 358, linear Gaussian grid, 46 levels) model with stretching coefficient $c = 2.4$, that gives $\Delta t \simeq 2$ mn.
2.2 Semi-Lagrangian scheme.

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

$$\frac{dX}{dt} = \dot{X}$$  \hfill (6)

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 positions of the origin point $O$ and the medium point $M$ along the trajectory and to interpolate some quantities at these points (roughly 1.5 times the cost of the Eulerian scheme in the TL358L46c2.4 model with full French physics). 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.

$D$ is the divergence of the horizontal wind on the $\eta$-coordinates, $\dot{\eta} = \frac{d\eta}{dt}$. Lipschitz criterion writes for a three-time level semi-Lagrangian scheme:

$$\left| D + \frac{\partial \dot{\eta}}{\partial \eta} \right| \frac{Dt}{2} < 1$$  \hfill (7)

Lipschitz criterion writes for a two-time level semi-Lagrangian scheme:

$$\left| D + \frac{\partial \dot{\eta}}{\partial \eta} \right| \frac{\Delta t}{2} < 1$$  \hfill (8)

Expressions “semi-Lagrangian scheme”, “three-time level semi-Lagrangian scheme” and “two-time level semi-Lagrangian scheme” will be from now on abbreviated into “SL scheme”, “3TL SL scheme” and “2TL SL scheme”.

3 The 2D equations.

3.1 Notations for the 2D equations.

- $V$ is the horizontal wind. Its zonal component (on the Gaussian grid) is denoted by $U$. Its meridian component (on the Gaussian grid) is denoted by $V$.
- $D$ is the horizontal wind divergence.
- $\zeta$ is the horizontal wind vorticity.
- $\Phi$ is the equivalent height. $\Phi_s$ is the surface geopotential height (i.e. the orography). $\Phi^*$ is a reference equivalent height which is only used in the semi-implicit scheme and the linear model.
- $\Omega$ is the Earth rotation angular velocity.
- $\nabla$ is the first order horizontal gradient on $\eta$-surfaces.
- $a$ is the Earth radius.
- $(\lambda_{bne}, \theta_{bne})$ are the longitude-latitude coordinates on a tilted and not stretched geometry, the tilting being the same as the one of the computational sphere.
- $k$ is the unit vertical vector. One can write:

$$k = \frac{r}{|r|} = \frac{r}{a}$$

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

Momentum equation.

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

$$\frac{d}{dt} \left( V + \delta_{\varphi} (2\vec{\Omega} \wedge \vec{r}) \right) = \left[ -2(1 - \delta_{\varphi})[(\vec{\Omega} \wedge \vec{V})] - \nabla \Phi \right]$$

Continuity equation.

- Conventional formulation.

$$\frac{d}{dt}(\Phi - (1 - \delta_{TR})\Phi_s) = -(\Phi - \Phi_s)D + \delta_{TR} \nabla \nabla (\Phi_s)$$

- Lagrangian formulation.

$$\frac{d}{dt}((\Phi - \Phi_s)J) = 0$$

$J$ is a “Jacobian” quantity which satisfies to:

$$\frac{dJ}{dt} = -JD$$
4 The 3D equations in spherical geometry (ARPEGE/IFS).

4.1 Notations for the 3D equations.

- \( \mathbf{V} \) is the horizontal wind. Its zonal component (on the Gaussian grid) is denoted by \( U \). Its meridian component (on the Gaussian grid) is denoted by \( V \).
- \( D \) is the horizontal wind divergence.
- \( \zeta \) is the horizontal wind vorticity.
- \( T \) is the temperature.
- \( q \) is the humidity.
- \( q_r \) is the rain.
- \( \Pi \) is the hydrostatic pressure.
- \( \Pi_s \) is the hydrostatic surface pressure.
- \( \Omega \) is the Earth rotation angular velocity.
- \((\lambda_b, \theta_b)\) are the longitude-latitude coordinates on a tilted and not stretched geometry, the tilting being the same as the one of the computational sphere.
- \((\lambda, \theta)\) are the geographical longitude-latitude coordinates.
- \((\Lambda, \Theta)\) are the computational sphere longitude-latitude coordinates.
- \( w \) is the \( z \)-coordinate vertical velocity: \( w = \frac{dz}{dt} \).
- \( \omega = \frac{d\Pi}{dt} \) is the total temporal derivative of the hydrostatic pressure.
- \( p \) is the pressure, \( p_s \) is the surface pressure.
- \( gz \) is the geopotential height.
- \( \Phi \) is the total geopotential. \( \Phi = gz \) in the thin layer equations, but not in the deep layer equations formulation of White and Bromley.
- \( \Phi_s = gz_s \) is the surface geopotential (i.e. the orography).
- \( r \) is the vector directed upwards, the length of which is the Earth radius. The length of this vector is \( r \). In the deep layer equations according to (White and Bromley, 1995), one uses an approximation of this radius, only depending on the hydrostatic pressure ("pseudo-radius").
- \( a \) is the average Earth radius near the surface.
- \( W = \frac{dr}{dt} \) is the pseudo-vertical velocity used in some Coriolis and curvature terms in the deep layer equations according to (White and Bromley, 1995). \( W = 0 \) in the thin layer equations.
- \( \mathbf{i} \) (resp. \( \mathbf{j} \)) is the unit zonal (resp. meridian) vector on the Gaussian grid.
- \( \mathbf{k} \) is the unit vertical vector. One can write:
  \[ \mathbf{k} = \frac{\mathbf{r}}{r_s} = \frac{\mathbf{r}}{r} \]
- \( g \) is the gravity acceleration constant.
- In the case where vertical variations of \( g \) are taken into account, we denote by \( G \) the reference value of \( g \) at \( r_s = a \).
- \( R \) is the gas constant for air and \( R_d \) the gas constant for dry air.
- \( c_p \) is the specific heat at constant pressure for air and \( c_{pd} \) is the specific heat at constant pressure for dry air.
- \( c_v \) is the specific heat at constant volume for air and \( c_{vd} \) is the specific heat at constant volume for dry air.
- \( \nabla \) is the first order horizontal gradient on \( \eta \)-surfaces.
• $\alpha_T$ is a vertical-dependent coefficient used to define a thermodynamic variable $T + \delta_T R \frac{\alpha_T}{\Pi_{ST}}$ less sensitive to orography than temperature $T$. Expression of $\alpha_T$ is:

$$
\alpha_T = B \left( -\frac{R_g}{g} \left[ \frac{dT}{dz} \right]_{ST} \right) T_{ST} \left( \frac{\Pi_{ST}}{\Pi_{st}} \right) \left( \frac{R_g}{g} \left[ \frac{dT}{dz} \right]_{ST} - 1 \right)
$$

(13)

where $B$ is a vertically dependent and horizontally constant quantity which defines the vertical hybrid coordinate (see later paragraph “Definition of the vertical coordinate $\eta$", subsection [6.1]): $B$ varies from 1 to 0 from bottom to top. Subscript “st” stands for “standard atmosphere”.

• $\rho$ is the mass per volume unit of air.

• $M$ is the mapping factor.

• $M$ is a reference mapping factor for the semi-implicit scheme.

• $D_3$ is the 3D divergence used in the NH model. Its expression is given by equation (2).

• $\tau$, $\gamma$, $\nu$, $L^*$, $\partial^*$ are linear operators used in the semi-implicit scheme (for more details, see documentation (IDSI) about semi-implicit scheme).

• $T^*$ is a vertically-constant reference temperature which is used in the semi-implicit scheme and in some non-hydrostatic equations. If LSPRT = T, (use of virtual temperature in spectral transforms instead of real temperature), $T^*$ is used as a reference virtual temperature (same default value).

• $T^a$ is a cold vertically-constant reference temperature which is used in the semi-implicit scheme in the NH vertical divergence equation; it is recommended to have $T^a$ lower than the current temperature.

• $T_{ST}$ is the reference standard atmosphere surface temperature (288.15 K). $\left[ \frac{dT}{dz} \right]_{ST}$ is the standard atmosphere tropospheric gradient of temperature (-0.0065 K/m).

• $\Pi^*$ is a reference hydrostatic pressure and $\Pi_{st}^*$ is a reference hydrostatic surface pressure, which are used in the semi-implicit scheme and in some non-hydrostatic equations. These reference quantities are vertically dependent and “horizontally" (i.e. on $\eta$ surfaces) constant. $\Delta \Pi^*$ are layer depths corresponding to a surface hydrostatic pressure equal to $\Pi^*$.

• $\Pi_{st}^*$ is a reference hydrostatic pressure equal to the surface pressure of the standard atmosphere (101325 Pa, variable VP00). $\Pi_{ST}$ is a reference hydrostatic pressure defined at full levels and half levels corresponding to the surface reference hydrostatic pressure $\Pi_{st}^*$ (stored in array STPRE).

• $\Delta \Phi^*$ is a reference geopotential depth computed on model layers, used in the non-hydrostatic model (more exactly in the non-linear part of the true 3D divergence). $\Delta \Phi^*$ is vertically dependent and “horizontally" (i.e. on $\eta$ surfaces) constant.

• In the Wood and Staniforth deep-layer NH equations, a mass vertically integrated quantity $\bar{\Pi}$ is introduced, in order to hide some metric terms, especially in the continuity equation: $\bar{\Pi}$ replaces $\Pi$ in the adiabatic equations ($\Pi$ becomes, if needed, a diagnostic quantity). The definition of the hybrid vertical coordinate applies for $\bar{\Pi}$, not for $\Pi$. Quantities $\delta$ and $\alpha$ (depths of logarithm of hydrostatic pressure) are replaced by $\bar{\delta}$ and $\bar{\alpha}$ (depths of logarithm of $\bar{\Pi}$).

$$
\frac{\partial \bar{\Pi}}{\partial \Pi} = \frac{r_s^2 \cdot g}{a^2 \cdot g}
$$
4. Semi-lagrangian computations

4.2 The thin layer 3D primitive equation model: Lagrangian formulation.

Momentum equation.

Vectorial form of momentum equation is used. Coriolis force can be treated explicitly ($\delta V = 0$) or implicitly ($\delta V = 1$).

\[
d\left(\vec{V} + \delta V (2\vec{\Omega} \wedge \vec{r})\right) = -2(1 - \delta V)(\vec{\Omega} \wedge \vec{V}) - \nabla \Phi - RT\nabla(\log \Pi) + F_V
\]

$F_V$ is the physical contribution on horizontal wind.

Thermodynamic equation.

\[
d\left(T + \delta T R \alpha T \Phi_s R d T_{ST}\right) = d\left(\delta T R \alpha T \Phi_s R d T_{ST}\right) + RT \frac{\omega}{c_p} \Pi + F_T
\]

$F_T$ is the physical contribution on temperature. When $\delta T R = 1$ the Eulerian treatment of orography is applied and the prognostic variable is replaced by one variable less sensitive to the surface orography. This modification has been proposed by Ritchie and Tanguay (1996). See equation (13) for definition of $\alpha_T$.

Term $d\left(\delta T R \alpha T \Phi_s R d T_{ST}\right)$ only contains advection terms linked to horizontal variations of orography and vertical variations of the coefficient $\alpha_T$.

Continuity equation.

The equation which is discretised is the vertically integrated Lagrangian formulation of continuity equation.

\[
\int_{\eta=0}^{\eta=1} \frac{\partial B}{\partial \eta} \left[ \frac{\log \Pi_e + \delta T R \Phi_s R d T_{ST}}{dt} \right] d\eta = \int_{\eta=0}^{\eta=1} \frac{\partial B}{\partial \eta} \left[ \vec{V} \nabla \left( \log \Pi_e + \delta T R \Phi_s R d T_{ST} \right) \right] d\eta - \frac{\Pi_e}{g} \left[ F_m \right]_{\eta=1} d\eta
\]

Details leading to this formulation is given in part 6.2.3 of the documentation (IDEUL) about model equations and Eulerian dynamics.

Variable $\delta T R$ is 0 or 1; when $\delta T R = 1$ the new variable is less sensitive to the orography (new variable proposed by Ritchie and Tanguay (1996) to reduce orographic resonance).

If one assumes that a volume of air occupied by rainfall drops is not replaced by dry air when drops are falling (case $\delta m = 0$, variable NDPSFI is 0 in NAMPHY), $F_m$ is replaced by zero and $\left[ \frac{\partial B}{\partial \eta} \right]_{\eta=1}$ is equal to zero.

If one assumes that a volume of air occupied by rainfall drops is replaced by dry air when drops are falling (case $\delta m = 1$, variable NDPSFI is 1 in NAMPHY), $F_m$ (diabatic flux) has to be taken in account and $\left[ \frac{\partial B}{\partial \eta} \right]_{\eta=1}$ is non-zero (more details are given in documentation (IDEUL)).

$\left[ \frac{\partial B}{\partial \eta} \right]_{\eta=0}$ is non-zero only when there is an upper radiative boundary condition (LRUBC=.TRUE.).
Advectable GFL variables: moisture, but also for example ozone, liquid water, ice, cloud fraction, TKE, aerosols and extra GFL variables equations.

Equation is written for moisture $q$, and is the same for the other advectable GFL variables.

$$\frac{dq}{dt} = F_q$$  \hspace{1cm} (17)

$F_q$ is the physical contribution on moisture.

Non advectable pseudo-historic GFL variables:

Equation is written for rain $q_r$, and is the same for the other non-advectable GFL variables. Since there is no advection, the SL equation is identical to the Eulerian equation.

$$\frac{\partial q_r}{\partial t} = F_{q_r}$$  \hspace{1cm} (18)

4.3 The deep layer 3D primitive equation model according to White and Bromley, 1995: Lagrangian formulation.

Deep layer equations: basics and new features.

The following modifications are done, according to (White and Bromley, 1995):

- One takes account to the fact, that the distance to the Earth center is no longer a but a radius varying with the vertical. For conveniency (with the $\eta$ vertical coordinate), one approximates the radius by a pseudo-radius $r_s$ which depends only on the hydrostatic pressure $\Pi$. Two vertical lines are no longer parallel, so the section of a vertical column varies with the hydrostatic pressure.
- The vertical velocity is now taken in account in the Coriolis term through a pseudo-vertical velocity $W$ defined by $W = \frac{dr}{dt}$. $W$ also appears in some new curvature terms.
- The total geopotential $\Phi$, which appears in the RHS of the wind equation, is no longer equal to the geopotential height $g_z$.

Details about deep layer equations (definition and expression of $r_s$, $W$, geopotential relationships) is given in the documentation (IDEUL) about model equations and Eulerian dynamics and is not detailed again here. All calculations giving the below formulation of primitive equations is also given in the documentation (IDEUL).

Momentum equation.

Coriolis force can be treated explicitly ($\delta \vec{V} = 0$) or implicitly ($\delta \vec{V} = 1$) in the Lagrangian equation.

$$\frac{d (\vec{V} + \delta \vec{V} (2\vec{\Omega} \wedge \vec{r}))}{dt} = (1-\delta \vec{V}) (-2\vec{\Omega} \wedge \vec{V} - 2\vec{\Omega} \wedge \vec{W} k) - \frac{W}{r_s} \vec{V} - \nabla \Phi - (R T + \mu_s R_i T_i) \nabla (\log \Pi) + \vec{F}_V$$  \hspace{1cm} (19)

$\vec{F}_V$ is the physical contribution on horizontal wind. See documentation (IDEUL) for definition of $\Phi$ and $\mu_s$. 
4. Semi-lagrangian computations

Thermodynamic equation.

Lagrangian tendency:

\[
\frac{d}{dt} \left( T + \delta_{TR} \frac{\alpha T \Phi_s R \alpha T}{\Pi_a T_s} \right) = \frac{d}{dt} \left( \delta_{TR} \frac{\alpha T \Phi_s R \alpha T}{\Pi_a T_s} \right) + RT \frac{\omega}{c_p} \Pi + F_T
\]  

(20)

\(F_T\) is the physical contribution on temperature. When \(\delta_{TR} = 1\) the Eulerian treatment of orography is applied and the prognostic variable is replaced by one variable less sensitive to the surface orography. This modification has been proposed by Ritchie and Tanguay (1996).

This equation is unchanged compared to its expression in the thin layer equations. The only change is the diagnostic expression of \(\omega\) (see documentation (IDEUL) for details about expression and discretisation of \(\omega\)).

Continuity equation.

The equation which is discretised is the vertically integrated Lagrangian formulation of continuity equation.

\[
\int_{\eta_0}^{\eta_1} \frac{\partial B}{\partial \eta} \left[ \log \Pi_s + \delta_{TR} \frac{\Phi_s R}{\Pi_a T_s} \right] d\eta = \int_{\eta_0}^{\eta_1} \frac{\partial B}{\partial \eta} \left[ \frac{r_s}{r_s a} \nabla \right] \left[ \log \Pi_s + \delta_{TR} \frac{\Phi_s R}{\Pi_a T_s} \right] d\eta + \int_{\eta_0}^{\eta_1} \frac{r_s}{r_s a} \nabla \left[ \frac{r_s}{r_s a} \right] \left[ \log \Pi_s + \delta_{TR} \frac{\Phi_s R}{\Pi_a T_s} \right] d\eta \]  

(21)

where \(\frac{\partial \log(\Pi_s)}{\partial t}\) is given by equation:

\[
\frac{\partial \log(\Pi_s)}{\partial t} = - \left[ \frac{a^2}{r_s^2} \right]_{\eta_1}^{\eta_0} \frac{1}{r_s a} \int_{\eta_0}^{\eta_1} \nabla \left[ \frac{r_s}{r_s a} \right] \left[ \log \Pi_s + \delta_{TR} \frac{\Phi_s R}{\Pi_a T_s} \right] d\eta - \frac{1}{r_s a} g \left[ F_m \right]_{\eta_1}^{\eta_0} - \frac{1}{r_s a} g \left[ F_m \right]_{\eta_0}^{\eta_1} \]  

(22)

Advectable GFL variables.

These equations are identical to the thin layer model ones.

Non advectable pseudo-historic GFL variables.

These equations are identical to the thin layer model ones.

4.4 The thin layer 3D NH-PDVD non-hydrostatic model: Lagrangian formulation.

Basics about NH equations.

The system of equations uses an idea of Laprise (1992) to be consistent with the hydrostatic equations: pressure \(p\) is split into an hydrostatic contribution (hydrostatic pressure \(\Pi\)) and an anhydrostatic departure \((p - \Pi)\). \(\Pi\) is still used as an independent prognostic variable. This way of treating the non-hydrostatic equations is compatible with the
use of a terrain-following vertical coordinate ($\eta$). For NPDVAR—2 and NVDVAR—3 the non-hydrostatic system of equations includes two additional prognostic variables: a variable $\hat{Q}$ linked to the pressure departure (its expression is $\hat{Q} = \log \frac{\Pi}{\rho}$), and a vertical divergence $d$. Details about these equations (choice of the two additional prognostic variables, discretisations, linearisation for semi-implicit scheme) can be found in (Bubnová et al., 1995), (Geleyn and Bubnová, 1995), (IDNH), (IDNH2.1), (IDVNH1), (IDVHN2) and (IDVNH3). For the vertical divergence equation, it is also possible to use the prognostic variable:

$$d_4 = d + \chi$$

where:

$$\chi = \frac{p}{m_{\Pi} RT} \nabla \Phi \left( \frac{\partial \nabla}{\partial \eta} \right)$$

Momentum equation.

Vectorial form of momentum equation is used. Coriolis force can be treated explicitly ($\delta \vec{V} = 0$) or implicitly ($\delta \vec{V} = 1$).

$$\frac{d (\vec{V} + \delta \vec{V} (2 \vec{\Omega} \wedge \vec{r}))}{dt} = [-2(1 - \delta \vec{V}) (\vec{\Omega} \wedge \vec{V})] - \frac{\partial p}{\partial \Pi} \nabla \Phi - RT \frac{\nabla (p)}{p} + \vec{F}_V$$  \hspace{1cm} (23)

The pressure gradient term is modified compared to the hydrostatic case. One can see that both non-hydrostatic pressure $p$ and hydrostatic pressure $\Pi$ are used in the RHS (right hand side) of equation (23). Term $\frac{\nabla (\log p)}{p}$ replaces $\nabla (\log p)$, the discretisation of this term is no longer a discretisation of $\nabla (\log p)$ and is slightly more complicated (more details are given in documentation (IDEUL)).

Thermodynamic equation.

$$\frac{dT}{dt} = \frac{RT}{c_v} \left( \frac{dp}{dt} \right) + F_T$$  \hspace{1cm} (24)

The RHS of equation (24) now uses the total pressure $p$. This equation can be rewritten in order to use the true 3D divergence $D_3$ in the RHS.

$$\frac{dT}{dt} = \frac{RT}{c_v} D_3 + \left[ \frac{c_p}{c_v} F_T \right]$$  \hspace{1cm} (25)

Expression of $D_3$ is:

$$D_3 = \nabla \vec{V} + \frac{p}{m_{\Pi} RT} \nabla \Phi \left( \frac{\partial \nabla}{\partial \eta} \right) - \frac{gp}{m_{\Pi} RT} \left( \frac{\partial w}{\partial \eta} \right)$$  \hspace{1cm} (26)

As for hydrostatic temperature equation, this equation can be modified by replacing temperature $T$ by $T + \delta_T \frac{\alpha_T \Phi_s R}{RT}$ which is less sensitive to orography.

$$\frac{d \left( T + \delta_T \frac{\alpha_T \Phi_s R}{RT} \right)}{dt} = \frac{d \left( \delta_T \frac{\alpha_T \Phi_s R}{RT} \right)}{dt} \frac{RT}{c_v} D_3 + \left[ \frac{c_p}{c_v} F_T \right]$$  \hspace{1cm} (27)

See equation (13) for definition of $\alpha_T$. Term $\frac{d \left( \delta_T \frac{\alpha_T \Phi_s R}{RT} \right)}{dt}$ only contains advection terms linked to horizontal variations of orography and vertical variations of coefficient $\alpha_T$. 

Continuity equation.

This equation is unchanged compared to the hydrostatic case and only uses the hydrostatic pressure $\Pi$.

Advectable GFL variables and pseudo-historic non-advectable GFL variables equations.

These equations are unchanged compared to the hydrostatic case.

Pressure departure variable $\hat{Q}$ equation.

This part is valid for $\text{NPDVAR}=2$.

Definition of $\hat{Q}$ is:

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

(28)

Equation of $\hat{Q}$ is:

$$\frac{d\hat{Q}}{dt} = -\frac{c_0}{c_v} \frac{\omega}{\Pi} + \frac{c_0}{c_v} F_T$$

(29)

where $D_3$ is defined by equation (2). The physical contribution of $\hat{Q}$ is linked to the physical contribution of $T$.

Vertical velocity $w$ and vertical divergence $d$ equations.

* Using $d$ as prognostic variable: This part is valid for $\text{NVDVAR}=3$.

The relationship between $d$ and $w$ writes:

$$d = -\frac{gp}{R_d T} \eta \frac{\partial w}{\partial \eta}$$

(30)

Equation of $d$ is:

$$\frac{dd}{dt} = -dD_3 + d\nabla V - \frac{gp}{R_d T} \frac{\partial w}{\partial \eta} \frac{\partial w}{\partial \eta} \left( \frac{\partial V}{\partial \eta} \right) + \frac{gp}{R_d T} \frac{\partial w}{\partial \eta} \left( \nabla w \right) \left( \frac{\partial V}{\partial \eta} \right) + F_d$$

(31)

The physical contribution of $d$ is linked to $F_w$ which is the physical contribution of $w$, and also to $F_m''$ (continuity equation for $\text{NDPSFI}=1$) by the following relationship:

$$F_d = -\frac{d}{\frac{\partial w}{\partial \eta}} F_m'' + \frac{\partial F_w}{\partial \eta}$$

(32)
4. Semi-lagrangian computations

One can notice that the RHS of equation (31) contains the term \( \frac{\partial [ dw/dt \text{ ad} \partial \eta]}{\partial \eta} \) which requires the computation of \( \frac{dw}{dt} \) (at least its adiabatic part).

Equation of \( w \) is:

\[
\frac{dw}{dt} = g \frac{\partial (p - \Pi)}{\partial \Pi} + F_w \tag{33}
\]

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

\[
g \frac{dw_{surf}}{dt} = \frac{dV_{surf}}{dt} \nabla \Phi_s + V_{surf} [V_{surf} \nabla (\nabla \Phi_s)] + F_{w_{surf}} \tag{34}
\]

Value of term \( \frac{dV_{surf}}{dt} \nabla \Phi_s + V_{surf} [V_{surf} \nabla (\nabla \Phi_s)] \) contains the horizontal second derivatives of surface orography and Coriolis term (cf. equation (32) of Geleyn and Bubnová, 1995). Its calculation is rather tricky and is not detailed in this documentation (for more details, see documentation (IDEUL)).

Remark: an alternative formulation is to mix the \( w \) equation in the advection scheme and the \( d \) equation in the other parts of the code.

* **Using \( d_4 \) as prognostic variable:** This part is valid for NVDVAR=4.

Equation of \( d_4 \) is:

\[
\frac{dd_4}{dt} = \frac{dd}{dt} + \frac{d\xi}{dt} \tag{35}
\]

where \( \frac{d\xi}{dt} \) is the Lagrangian total derivative of the quantity \( \xi = \frac{p}{\rho} \nabla \Phi \left( \frac{\partial \eta}{\partial \eta} \right) \). The calculation of \( \frac{d\xi}{dt} \) is not done by a semi-Lagrangian temporal advection but simply by a diagnostic.

4.5 The deep layer NH-PDVD non-hydrostatic model according to Wood and Staniforth, 2003: Lagrangian formulation.

**Basics deep layer equations.**

* **New features brought in the deep layer equations:** The following modifications are done:

- One takes account to the fact, that the distance to the Earth center is no longer \( a \) but a radius \( r_s \) varying with the vertical. No specific approximation is done on this radius. Two vertical lines are no longer parallel, so the section of a vertical column varies with the altitude.
- The vertical velocity \( w \) is now taken into account in the Coriolis term. \( w \) also appears in some new curvature terms.
- Relationship between \( r_s \) and \( w \) is \( \frac{dr_s}{dt} = w \).
The total geopotential $\Phi$ does not appear in the adiabatic equations (it is replaced by $G_z = G(r_s - a)$); it may appear in some energetic quantities; $\Phi$ is not always equal to $G_z$.

A mass vertically integrated quantity $\tilde{\Pi}$ is introduced, in order to hide some metric terms, especially in the continuity equation: $\tilde{\Pi}$ replaces $\Pi$ in the adiabatic equations ($\Pi$ becomes, if needed, a diagnostic quantity). The definition of the hybrid vertical coordinate applies for $\tilde{\Pi}$, not for $\Pi$. Quantities $\delta$ and $\alpha$ (depths of logarithm of hydrostatic pressure) are replaced by $\tilde{\delta}$ and $\tilde{\alpha}$ (depths of logarithm of $\tilde{\Pi}$). The relationship between $\tilde{\Pi}$ and $\Pi$ is:

$$\frac{\partial \tilde{\Pi}}{\partial \Pi} = \frac{r_s^2 G}{a^2 g}$$

and:

$$\tilde{\Pi}_{\text{top}} = \Pi_{\text{top}}$$

The vertical variations of $g$ may be taken into account (optionally): we denote by $g$ the vertically-dependent value and by $G$ the constant reference value ($g = G$ where $r_s = a$).

$\omega$ is now $\omega = \frac{\text{d} \tilde{\Pi}}{\text{d} t}$.

See documentation (IDEUL) for more details about the calculation of $r_s$ and $\tilde{\Pi}$.

**Momentum equation.**

Modifications brought by the deep-layer formulation are:

- Additional Coriolis $w$ terms and curvature $w$ terms: they are similar to those of the White and Bromley formulation, but this is $w$ and not $W$ which appears.
- $\Phi$ is replaced by $G r_s$, and $\Pi$ is replaced by $\tilde{\Pi}$ in the pressure gradient term: these modifications explain why some metric terms appear.
- Contrary to the White and Bromley formulation, there are no $\mu_s$ terms appearing in the pressure gradient term (it appears in the $w$ equation instead).

The momentum equation writes:

$$\frac{d (\tilde{\mathbf{V}} + \delta \tilde{\mathbf{V}} (2 \tilde{\Omega} \wedge \tilde{\mathbf{r}}))}{dt} = [-2(1-\delta \tilde{\Omega}) (\Omega \wedge \tilde{\mathbf{V}} + 2 \tilde{\Omega} \wedge w \tilde{\mathbf{k}})] \cdot \frac{w}{r_s} \tilde{\mathbf{V}} - \frac{r_s^2}{a^2} \frac{\partial p}{\partial \tilde{\Pi}} \nabla [G r_s] - RT \frac{\nabla (p)}{p} + F \tilde{\mathbf{V}}$$

**Thermodynamic equation.**

Equation (27) is still valid, but the calculation of $D_3$ is modified (see documentation (IDEUL) for more details).

**Continuity equation.**

Nature of changes:

- $\Pi$ is changed into $\tilde{\Pi}$.
- Some metric terms appear: replace $\nabla \left( \tilde{\mathbf{V}} \frac{\partial \tilde{\Pi}}{\partial \eta} \right)$ by $\nabla \left( \frac{\text{d}}{r_s} \tilde{\mathbf{V}} \frac{\partial \tilde{\Pi}}{\partial \eta} \right)$. 


4. Semi-lagrangian computations

- $G$ appears in factor of the diabatic term.

For example, the vertically integrated Lagrangian formulation of continuity equation writes:

$$
\int_{\eta=0}^{\eta=1} \frac{\partial B}{\partial \eta} \frac{\partial \left[ \log \tilde{\Pi} + \delta T R \frac{\Phi_s}{R_T} \right]}{\partial \eta} d\eta =
\int_{\eta=0}^{\eta=1} \frac{\partial B}{\partial \eta} \left( \frac{1}{R_T} \right) \nabla \left( \frac{p}{R_T} \frac{\partial \tilde{\Pi}}{\partial \eta} \right) + \nabla \left[ \log \tilde{\Pi} + \delta T R \frac{\Phi_s}{R_T} \right] - \frac{1}{R_T} \left[ \frac{\partial \tilde{\Pi}}{\partial \eta} \right]_{\eta=1} + \frac{1}{R_T} \left[ \frac{\partial \tilde{\Pi}}{\partial \eta} \right]_{\eta=0} - \frac{1}{R_T} G \left( F_m, \eta=1 \right)
$$

Advectable GFL variables and pseudo-historic non-advectable GFL variables equations.

Cf. thin layer equations.

**Pressure departure variable $\hat{Q}$ equation.**

This part is valid for $\text{NPVAR}=2$.

Definition of $\hat{Q}$ is modified:

$$
\hat{Q} = \log \left( \frac{p}{\tilde{\Pi}} \right) + \delta_p \log \left( \frac{\tilde{\Pi}_{ref}}{\Pi_{ref}} \right)
$$

where $\tilde{\Pi}_{ref}$ and $\Pi_{ref}$ are reference values based on the standard atmosphere ($\log(\tilde{\Pi}_{ref}/\Pi_{ref})$ is temporally constant and constant along the $\eta$ surfaces). It is desirable to take a value of $\delta_p$ close to 1.

Equation of $\hat{Q}$ is:

$$
\frac{d\hat{Q}}{dt} = -\frac{c_p}{c_v} D_3 - \frac{\omega}{\tilde{\Pi}} + \delta_p \frac{\partial}{\partial \eta} \left( \log \frac{\tilde{\Pi}_{ref}}{\Pi_{ref}} \right) + \frac{c_p}{c_v} F_T
$$

where $D_3$ is defined by equation (2). The physical contribution of $\hat{Q}$ is linked to the physical contribution of $T$.

**Vertical velocity $w$ and vertical divergence $d$ equations.**

Equation of $w$ is:

$$
\frac{dw}{dt} = -G \mu_s - (g - G) + G \left( \frac{r_s^2}{a^2} - 1 \right) \frac{\partial p}{\partial \Pi} + G \frac{\partial (p - \Pi)}{\partial \Pi} + F_w
$$

The surface condition writes:

$$
\frac{w_{surf}}{dt} = \nabla_{surf} \nabla r_s
$$

and this relationship can be used to compute $\frac{d\Pi_{surf}}{dt}$. Currently only the diagnostic calculation of $w_{surf}$ is used in the code.

Lagrangian tendency of $d$ equation:

$$
\frac{dd}{dt} = -dD_3 + d\nabla \cdot \frac{G p}{R_T \frac{\partial \Pi}{\partial \eta}} \left[ \frac{\partial \left[ \frac{r_s^2}{a^2} \right]}{\partial \eta} \right]_{ad} + \frac{G p}{R_T \frac{\partial \Pi}{\partial \eta}} \left( \nabla \left[ \frac{r_s^2}{a^2} \right] \right) \left( \frac{\partial \nabla}{\partial \eta} \right) + F_d
$$
4.6 The thin layer NH-GEOGW non-hydrostatic model: Lagrangian formulation.

See part 4.4 for momentum, temperature, continuity and GFL equations.

The NH prognostic variables are now $\Phi$ (or $\Phi + \delta_{\text{GEO}} B \Phi_s$) and $gw$ (or $gw + \delta_W B gw_s$). See equation (40) for $gw$ equation. Equation for $\Phi$ is:

$$\frac{d\Phi}{dt} = gw + F_\Phi$$  \hspace{1cm} (42)
5 Discretisation of the equations: general aspects.

This section does not describe in detail iterative centred-implicit schemes and describes only non-iterative schemes in detail. For more details about iterative centred-implicit schemes, see documentation (IDSI).

5.1 Notations.

* Upper index:
  - First integration step: \( + \) (resp. \( m \), \( o \), \( - \)) for \( t + \Delta t \) (resp. \( t + 0.5\Delta t \), \( t \), \( t \)) quantity.
  - Following integration steps: \( + \) (resp. \( m \), \( o \), \( - \)) for \( t + \Delta t \) (resp. \( t + 0.5\Delta t \), \( t \), \( t - \Delta t \)) quantity.

* Lower index: \( F \) (resp. \( M \) and \( O \)) for final (resp. medium and origin) point.

* Particular case of the first timestep in a SL3TL scheme: Written discretisations are valid from the second integration step. \( \Delta t \) 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 \( \text{NFTHER} \) thermodynamic variables.
  - 3D advectable “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”). This class of variables includes for example humidity, liquid water, ice, cloud fraction, ozone, TKE, aerosols, and some extra fields. See documentation (IDEUL) for a comprehensive list of advectable GFL.
  - 3D non advectable pseudo-historic variables. The equation RHS of these variables looks like the one of the 3D advectable “conservative” variables, but there is no advection. They are included in the GFL variables. This class of variables includes for example rain, snow, graupels, hail, convective precipitation flux, stratiform precipitation flux, SRC (second-order flux for AROME), forcings, easy diagnostics, greenhouse gases, reactive gases, moisture convergence, total humidity variation, standard deviation of the saturation depression, convective vertical velocity. See documentation (IDEUL) for a comprehensive list of non advectable pseudo-historic GFL.
  - 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).
5.2 Discretisation for a 3D variable in a 3D model: general case where the RHS has non-zero linear and non-linear terms (GMV variables).

List of equations.

- Momentum equation.
- Temperature equation.
- Pressure departure variable (non-hydrostatic model only).
- Vertical divergence variable (non-hydrostatic model only).

Generic notations.

Generic notation $N(X)\text{LAG}$ stands for:

- $NW\text{LAG}$ for momentum equation.
- $NT\text{LAG}$ for temperature equation.
- $NSP\text{DLAG}$ for pressure departure variable (non-hydrostatic model only).
- $NSV\text{DLAG}$ for vertical divergence (non-hydrostatic model only).

Generic notation $P(X)\text{L0, L9, T1}$ stands for:

- $PUL\text{L0, L9, T1}$ for U-momentum equation.
- $PVL\text{L0, L9, T1}$ for V-momentum equation.
- $PTL\text{L0, L9, T1}$ for temperature equation.
- $PSP\text{DL0, D9, DT1}$ for $\hat{Q}$ equation.
- $PSV\text{DL0, D9, DT1}$ for $d$ equation.

Generic notation $P(X)\text{NLT9}$ stands for:

- $PUL\text{NT9}$ for U-momentum equation.
- $PV\text{NLT9}$ for V-momentum equation.
- $PT\text{NLT9}$ for temperature equation.
- $PSP\text{DNLT9}$ for $\hat{Q}$ equation.
- $PSV\text{DNLT9}$ for $d$ equation.

Generic notation for total term, linear term, non-linear term, physics: $A$ is the total term (sum of dynamical contributions), $B$ is the linear term (treated in the semi-implicit scheme), the difference $A - \beta B$ is the non-linear term. $F$ is the sum of contributions computed in the physical parameterizations.

Description stands for the general case where linear and non-linear terms are gathered in the same buffer, and where no additional splitting is required to do diagnostics or to apply SLHD interpolations to a subset of the terms interpolated by a high-order interpolation. In some particular cases, additional splitting involving separate buffers may be required:

- In the NH model for options where linear terms must be separately interpolated (controlled by variables $\text{LSLINL}$, $\text{LSLINLC1}$ and $\text{LSLINLC2}$). Linear terms are stored in arrays, the name of them has appendix _SI, or in parts of PB1 with pointers, the name of them has appendix _SI.
• Separation between linear and non-linear terms may have to be done for some DDH diagnostics too (if LRSIDDH = T).

• When some diagnostics (for example DDH) impose that evaluation of the dynamics and of the physics at the origin point of the SL trajectory must be done separately in two different buffers (controlled by variable NSPLTHOI set to -1). Buffer P(X)LF9 is used instead of P(X)L9 to store quantities, the interpolation of which is a non-SLHD high-order one.

• When SLHD interpolations, if switched on, are applied only to a subset of the quantities interpolated by a high order interpolation (applied to X(t) or X(t - Δt) but not to the other terms); in particular physics do not use a SLHD interpolation in this case (controlled by variable NSPLTHOI set to 1). Buffer P(X)LF9 is used instead of P(X)L9 to store quantities, the interpolation of which is a non-SLHD high-order one.

Case N(X)LAG = 4 is not described in detail: in this case diabatic terms are interpolated by trilinear interpolations and they enter the buffer containing terms interpolated by trilinear interpolations. Start from discretisations given for N(X)LAG = 3 and move term ΔtF from P(X)L9 to P(X)L0.

Other points.

* **High-order interpolations:** In the following discretisations, “high-order interpolations" means 32 points interpolations for 3D terms (vertical interpolations are cubic), 12 points interpolations for 2D terms.

* **Uncentering:** ε is a first-order “uncentering factor". It allows to remove the noise due to gravity waves (orographic resonance).

* **Vectorial variables:** The following discretisations are written for scalar variables. For vectorial variables (for example the horizontal wind) a rotation operator \( \mathcal{R} \) has to be applied from interpolation point to final point:

  • expression interpolated at \( O \) has to be replaced by \( \mathcal{R}^{OF} \{ \text{this expression} \} _O \).
  • expression interpolated at \( M \) has to be replaced by \( \mathcal{R}^{MF} \{ \text{this expression} \} _M \).

3TL vertical interpolating SL scheme.

Equation

\[
\frac{dX}{dt} = A + \mathcal{F}
\]  \hspace{1cm} (43)

is discretised as follows:

\[
(X - (1 + \epsilon)\Delta t\beta \mathcal{B})_F^+ = \{X^- + (1 - \epsilon)\Delta t[A - \beta \mathcal{B}]_o^o + [(1 - \epsilon)\Delta t\beta \mathcal{B} + 2\Delta t\mathcal{F}]^-\}_O^O + [(1 + \epsilon)\Delta t[A - \beta \mathcal{B}]_e^e]\]  \hspace{1cm} (44)

Buffers content before interpolations for N(X)LAG = 2:
• $P(X)L0$ is not used.

• $P(X)L9$: $X^- + (1 - \epsilon)\Delta t[A - \beta B]^o + [(1 - \epsilon)\Delta t\beta B + 2\Delta tF]^- \Delta t_9$ for high-order interpolation at the origin point $O$.

• $P(X)T1$: $(1 + \epsilon)\Delta t[A - \beta B]^o$ then provisional add of quantity $[(1 + \epsilon)\Delta t\beta B]^o$ before $t + dt$ physics; evaluated at the final point $F$.

Buffers content before interpolations for $N(X)LAG$ = 3:

• $P(X)L0$: $(1 - \epsilon)\Delta t[A - \beta B]^o + [(1 - \epsilon)\Delta t\beta B]^- \Delta t_9$ for trilinear interpolation at the origin point $O$.

• $P(X)L9$: $X^- + [2\Delta tF]^- \Delta t_9$ for high-order interpolation at the origin point $O$.

• $P(X)T1$: $(1 + \epsilon)\Delta t[A - \beta B]^o$ then provisional add of quantity $[(1 + \epsilon)\Delta t\beta B]^o$ before $t + dt$ physics; evaluated at the final point $F$.

2TL vertical interpolating SL scheme: conventional discretisation (LSETTLS=.F.) and first-order uncentering.

The $t + \frac{\Delta t}{2}$ non-linear term $A^m - \beta B^m$ used in the 2TL SL scheme is computed by a linear temporal extrapolation using the $t$ and $t - \Delta t$ quantities at the same location. At the first time integration step, values at $t + \frac{\Delta t}{2}$ are set equal to initial values. This discretisation of the 2TL SL scheme follows (Mc Donald and Haugen, 1992). Quantity $A^o - \beta B^o$ has to be saved in a buffer $P(X)NLTL9$ to be available as $A^- - \beta B^-$ for the following timestep.

Equation (45) is discretised as follows:

\[
(X - (1 + \epsilon)\frac{\Delta t}{2}\beta B)_F^+ = \{X^o + (1 - \epsilon)\frac{\Delta t}{2}[A - \beta B]^o + [(1 - \epsilon)\frac{\Delta t}{2}\beta B + \Delta tF]^o\}_O \\
+ (1 + \epsilon)\frac{\Delta t}{2}[A - \beta B]^o_f
\]

which can be rewritten, once expanded the extrapolation:

\[
(X - (1 + \epsilon)\frac{\Delta t}{2}\beta B)_F^+ = \{X^o + \frac{\Delta t}{2}(1 - \epsilon)A^o + \Delta tF^o\}_O \\
+ 0.5(1 - \epsilon)\frac{\Delta t}{2}[(A - \beta B)^o - (A - \beta B)^-]_O + (1 + \epsilon)\frac{\Delta t}{2}[1.5(A - \beta B)^o - 0.5(A - \beta B)^-]_F
\]

Buffers content before interpolations for $N(X)LAG$ = 2:

• $P(X)L0$ is not used.

• $P(X)L9$: $X^o + \frac{\Delta t}{2}(1 - \epsilon)A^o + 0.5(1 - \epsilon)\frac{\Delta t}{2}[(A - \beta B)^o - (A - \beta B)^-] + \Delta tF^o$ for high-order interpolation at the origin point $O$.

• $P(X)T1$: $(1 + \epsilon)\frac{\Delta t}{2}[1.5(A - \beta B)^o - 0.5(A - \beta B)^-]$ then provisional add of quantity $[(1 + \epsilon)\frac{\Delta t}{2}\beta B]^o$ before $t + dt$ physics; evaluated at the final point $F$.

Buffers content before interpolations for $N(X)LAG$ = 3:

• $P(X)L0$: $\frac{\Delta t}{2}(1 - \epsilon)A^o + 0.5(1 - \epsilon)\frac{\Delta t}{2}[(A - \beta B)^o - (A - \beta B)^-]$ for trilinear interpolation at the origin point $O$.

• $P(X)L9$: $X^o + \Delta tF^o$ for high-order interpolation at the origin point $O$.

• $P(X)T1$: $(1 + \epsilon)\frac{\Delta t}{2}[1.5(A - \beta B)^o - 0.5(A - \beta B)^-]$ then provisional add of quantity $[(1 + \epsilon)\frac{\Delta t}{2}\beta B]^o$ before $t + dt$ physics; evaluated at the final point $F$. 

2TL vertical interpolating SL scheme: conventional discretisation (LSETTTLS=.F.) and pseudo-second order uncentering.

One starts to remove uncentering $\epsilon$ from the nonlinear terms and to apply a second-order uncentering $\epsilon_X$ to linear terms, that yields a term $B^-$ in the discretisation. From property:

$$B^- = 3B^* - 2B^m$$ (47)

one can term $B^-$ and show that discretisation is equivalent to replace $\beta$ by $(1 + \epsilon_X)\beta$. For more details, see part 5 (equations (37) and (38)) of (Simmons and Temperton, 1996). In equation (47), uncentering $\epsilon$ has to be replaced by zero, and $\beta$ has to be replaced by $(1 + \epsilon_X)\beta$. The $t + \frac{\Delta t}{2}$ non-linear term $A^m - (1 + \epsilon_X)\beta B^m$ used in the 2TL SL scheme is computed by a linear temporal extrapolation using the $t$ and $t - \Delta t$ quantities at the same location. At the first time integration step, values at $t + \frac{\Delta t}{2}$ are set equal to initial values and second-order uncentering is replaced by a first order uncentering. This discretisation of the 2TL SL scheme follows (McDonald and Haugen, 1992). Quantity $A^* - (1 + \epsilon_X)\beta B^m$ has to be saved in a buffer $P(X)NLT9$ to be available as $A^- - (1 + \epsilon_X)\beta B^-$ for the following timestep.

Equation (47) is discretised as follows:

$$(X - (1 + \epsilon_X)\frac{\Delta t}{2}\beta B)_{x}^m = \{X_0^* + \frac{\Delta t}{2}[A - (1 + \epsilon_X)\beta B]^m + [(1 + \epsilon_X)\frac{\Delta t}{2}\beta B + \Delta tF]^m\} \circ$$

$$+ \{\frac{\Delta t}{2}[A - (1 + \epsilon_X)\beta B]^m\} \circ$$ (48)

which can be rewritten, once expanded the extrapolation:

$$(X - (1 + \epsilon_X)\frac{\Delta t}{2}\beta B)_{x}^m = \{X_0^* + \frac{\Delta t}{2}A + \Delta tF\} \circ + 0.5 \frac{\Delta t}{2}[(A - (1 + \epsilon_X)\beta B)^m - (A - (1 + \epsilon_X)\beta B)^m] \circ$$

$$+ \frac{\Delta t}{2}[1.5(A - (1 + \epsilon_X)\beta B)^m - 0.5(A - (1 + \epsilon_X)\beta B)^m]\} \circ$$ (49)

Buffers content before interpolations for $N(X)LAG=2$:

- **$P(X)L0$** is not used.
- **$P(X)L9$**: $X^* + \frac{\Delta t}{2}A^* + 0.5 \frac{\Delta t}{2}[(A - (1 + \epsilon_X)\beta B)^m - (A - (1 + \epsilon_X)\beta B)^m] + \Delta tF$ for high-order interpolation at the origin point $O$.
- **$P(X)T1$**: $\frac{\Delta t}{2}[1.5(A - (1 + \epsilon_X)\beta B)^m - 0.5(A - (1 + \epsilon_X)\beta B)^m]$ then provisional add of quantity $[(1 + \epsilon_X)\frac{\Delta t}{2}\beta B]^m$ before $t + \Delta t$ physics; evaluated at the final point $F$.

Buffers content before interpolations for $N(X)LAG=3$:

- **$P(X)L0$**: $\frac{\Delta t}{2}A^* + 0.5 \frac{\Delta t}{2}[(A - (1 + \epsilon_X)\beta B)^m - (A - (1 + \epsilon_X)\beta B)^m]$ for trilinear interpolation at the origin point $O$.
- **$P(X)L9$**: $X^* + [\Delta tF]_m$ for high-order interpolation at the origin point $O$.
- **$P(X)T1$**: $\frac{\Delta t}{2}[1.5(A - (1 + \epsilon_X)\beta B)^m - 0.5(A - (1 + \epsilon_X)\beta B)^m]$ then provisional add of quantity $[(1 + \epsilon_X)\frac{\Delta t}{2}\beta B]^m$ before $t + \Delta t$ physics; evaluated at the final point $F$. 
2TL vertical interpolating SL scheme: stable discretisation (LSET-TLS=.T.) and first-order uncentering.

The $t + \Delta t$ non-linear term $A^m - \beta B^m$ used in the 2TL SL scheme if LSET-TLS=F. is replaced in the case LSET-TLS=.T. by a linear spatio-temporal extrapolation comparable to the one applied to the wind components for the research of trajectory (see formula (101)), except the fact that there is an additional uncentering. Term $A^m - \beta B^m$ is replaced by

$$0.5(1 + \epsilon)[A^m - \beta B^m]_F + 0.5(2 - \epsilon)[A^m - \beta B^m]_O - 0.5[A^m - \beta B^m]_O$$

This type of extrapolation is available only for $N(X)\text{LAG}=3$. At the first time integration step, values at $t + \Delta t$ are set equal to initial values. Quantity $A^m - \beta B^m$ has to be saved in a buffer $P(X)\text{NLT9}$ to be available as $A^m - \beta B^m$ for the following timestep.

Equation (10) is discretised as follows:

$$(X - (1 + \epsilon)\frac{\Delta t}{2}\beta B)^+_F = \{X^o + (2 - \epsilon)\frac{\Delta t}{2}[A - \beta B]^o - \frac{\Delta t}{2}[A - \beta B]^- + [(1 - \epsilon)\frac{\Delta t}{2}\beta B + \Delta t F]^o\}_O$$

which can be rewritten, once expanded the extrapolation:

$$(X - (1 + \epsilon)\frac{\Delta t}{2}\beta B)^+_F = \{X^o + \frac{\Delta t}{2}(1 - \epsilon)A^o + \Delta t F^o\}_O$$

$$+ \frac{\Delta t}{2}((A - \beta B)^o - (A - \beta B)^-) + \frac{\Delta t}{2}(1 + \epsilon)(A - \beta B)^n\}_F (50)$$

Buffers content before interpolations for $N(X)\text{LAG}=3$:

- $P(X)\text{L9}$: $\frac{\Delta t}{2}(1 - \epsilon)A^o + \frac{\Delta t}{2}((A - \beta B)^o - (A - \beta B)^-)$ for trilinear interpolation at the origin point $O$.
- $P(X)\text{L9}$: $X^o + \Delta t F^o$ for high-order interpolation at the origin point $O$.
- $P(X)\text{T1}$: $\frac{\Delta t}{2}(1 + \epsilon)(A - \beta B)^n$ then provisional add of quantity $[(1 + \epsilon)\frac{\Delta t}{2}\beta B]^o$ before $t+dt$ physics; evaluated at the final point $F$.

2TL vertical interpolating SL scheme: stable discretisation (LSET-TLS=.T.) and pseudo-second order uncentering.

In equation (50), uncentering $\epsilon$ has to be replaced by zero, and $\beta$ has to be replaced by $(1 + \epsilon_x)\beta$. At the first time integration step, values at $t + \Delta t$ are set equal to initial values and second-order uncentering is replaced by a first order uncentering. Quantity $A^m - (1 + \epsilon_x)\beta B^m$ has to be saved in a buffer $P(X)\text{NLT9}$ to be available as $A^m - (1 + \epsilon_x)\beta B^m$ for the following timestep.

Equation (10) is discretised as follows:

$$(X - (1 + \epsilon_x)\frac{\Delta t}{2}\beta B)^+_F = \{X^o + \Delta t[A - (1 + \epsilon_x)\beta B]^o - \frac{\Delta t}{2}[A - (1 + \epsilon_x)\beta B]^- + [(1 + \epsilon_x)\frac{\Delta t}{2}\beta B + \Delta t F]^o\}_O$$

$$+ \frac{\Delta t}{2}(A - (1 + \epsilon_x)\beta B)^n\}_F (52)$$

which can be rewritten, once expanded the extrapolation:

$$(X - (1 + \epsilon_x)\frac{\Delta t}{2}\beta B)^+_F = \{X^o + \frac{\Delta t}{2}A^o + \Delta t F^o\}_O$$

$$+ \frac{\Delta t}{2}((A - (1 + \epsilon_x)\beta B)^o - (A - (1 + \epsilon_x)\beta B)^-) + \frac{\Delta t}{2}(A - (1 + \epsilon_x)\beta B)^n\}_F (53)$$

Buffers content before interpolations for $N(X)\text{LAG}=3$:
• $P(X)L_0$: $\frac{\Delta t}{2} \mathcal{A}^v + \frac{\Delta t}{2} [(\mathcal{A} - (1 + \epsilon_x)\beta\mathcal{B})^v - (\mathcal{A} - (1 + \epsilon_x)\beta\mathcal{B})^-]$ for trilinear interpolation at the origin point $O$.

• $P(X)L_9$: $X^o + [\Delta t\mathcal{F}]^o$ for high-order interpolation at the origin point $O$.

• $P(X)T_1$: $\frac{\Delta t}{2} (\mathcal{A} - (1 + \epsilon_x)\beta\mathcal{B})^o$ then provisional add of quantity $[(1 + \epsilon_x)\frac{\Delta t}{2} \beta\mathcal{B}]^o$ before $t+\Delta t$ physics; evaluated at the final point $F$.

Specific treatment for some options in the vertical divergence equation.

* Option LGW ADV=.T.. When this option is activated, the SL scheme treats the Lagrangian equation of $w$ instead of the one of $d$ (or $d_4$). That implies the following steps in the code:

  • Change of variable from $d$ or $d_4$ to $w$.
  • SL explicit treatment of the $w$ equation. For finite difference vertical discretisation ($\text{LVFE}_{\text{GW}}=\text{F}$) $w$ is given at half levels: that means that the SL trajectory must be computed also for trajectories ending at half levels. For finite element vertical discretisation ($\text{LVFE}_{\text{GW}}=\text{T}$) $w$ is given at full levels. There is a specific treatment for $w_{\text{surf}}$ (diagnostic condition at $t + \Delta t$).
  • Calculation of the linear terms of the $d$ equation for the semi-implicit scheme.
  • Conversion from $w^+$ into $d^+$ or $d_4^+$. If $\text{LVFE}_{\text{GW}}=\text{T}$, this conversion is done by applying the vertical derivation to the temporal increment of $w$ to obtain the temporal increment of $d$.
  • Add the linear terms to $d^+$ or $d_4^+$.

It is necessary to keep the prognostic variable $d$ or $d_4$ in the linear model: $w$ in the linear model would lead to linear instabilities.

This option allows to remove spurious chimneys above slopes, and also spurious noise in some “bubble” tests.

* Option NVDVAR=4. In this case we must discretize the term $\frac{\partial}{\partial t}$, and this is not done by a SL treatment of the equation of $X$ (the RHS of this equation is not easy to compute).

If there is no predictor-corrector scheme activated, this term is diagnosed as follows:

• Three-time level semi-Lagrangian scheme:

  $$\frac{d\mathcal{X}}{dt} = \frac{0.5\mathcal{X}_{+} + 0.5\mathcal{X}_{-} - \mathcal{X}_{o}}{\Delta t} \quad (54)$$

• Two-time level semi-Lagrangian scheme:

  $$\frac{d\mathcal{X}}{dt} = \frac{\mathcal{X}_{+} - \mathcal{X}_{o}}{0.5\Delta t} \quad (55)$$

$\mathcal{X}_{M}$ is a symbolic denotation for the extrapolated $t + 0.5\Delta t$ value of $\mathcal{X}$. The way of extrapolating depends on $\text{LSETTTL}$ and is the same as for the other terms evaluated at $t + 0.5\Delta t$ (there is no interpolation at $M$).

If $\text{LPC}_{\text{FULL}}=.T.$ and $\text{LPC}_{\text{NESC}}=.F.$ these discretisations also apply to the predictor step; in this case there are two options:

• Option 1 ($\text{ND4SYS}=1$): include all the contributions of the evolution of $\mathcal{X}$ in the predictor-corrector scheme iterations.
Option 2 (ND4SYS=2): include only the advective contributions evolution of $\mathbf{x}$ in the predictor-corrector scheme iterations; after the last step of the corrector step, include the non advective processes.

In some cases (ECMWF configurations with lagged physics) option 1 may generate instabilities above slopes.

* Assumptions at the surface (velocities). Calculation of $\frac{d}{dt} \mathbf{w}_{surf}$ for the layer $l = L$ requires the calculation of $\frac{d\mathbf{w}_{surf}}{dt}$. There are two options to compute $\frac{d\mathbf{w}_{surf}}{dt}$, controlled by the variable LRDBBC of NAMdyn.

- For LRDBBC=F, one simply discretizes equation (33) with a SL treatment. The RHS of this equation contains $\frac{d\mathbf{v}_{surf}}{dt}$ and $\mathbf{v}_{surf}$, and the assumptions currently done about these quantities are:
  \[
  \mathbf{v}_{surf} = \mathbf{v}_{l=L},
  \]
  \[
  \frac{d\mathbf{v}_{surf}}{dt} = \frac{\partial \mathbf{v}}{\partial t} \big|_{l=L,ad} \big, \text{with an explicit treatment of the Coriolis term, even if LADVF=T. or LIMPF=T.}
  \]

- An alternate discretisation (option LRDBBC=T.) is to evaluate $\frac{d\mathbf{w}_{surf}}{dt}$ by:
  
  \[\text{SL3TL:} \quad \frac{d\mathbf{w}_{surf}}{dt} = \mathbf{w}_{l=L}^{+} - \mathbf{w}_{l=L}^{-} + \mathbf{F} \delta t \] (56)
  
  \[\text{SL2TL:} \quad \frac{d\mathbf{w}_{surf}}{dt} = \mathbf{w}_{l=L}^{+} - \mathbf{w}_{l=L}^{-} \big/ \Delta t \] (57)
  which requires additional interpolations to compute $\mathbf{w}_{l=L}^{-}$.

  This option allows to remove spurious chimneys above slopes. $\mathbf{w}^+$ is computed by:
  \[
  \mathbf{w}^+ = \mathbf{v}_{l=L,\text{prov}} \nabla \Phi
  \]
  where $\mathbf{v}_{l=L,\text{prov}}$ is the provisional $t + \Delta t$ value of $\mathbf{v}_{l=L}$ computed just after the interpolations (this is this provisional value which is used as input for the lagged physics).

  The treatment of $\frac{d}{dt} \mathbf{w}$ for the layer $l = L$ is done according to the following steps:

  - Calculation of $\left[ \frac{d}{dt} \right]_{l=L,\text{lrdbbc}=\text{F}}$ (as if LRDBBC were F), $\left[ \frac{d\mathbf{w}_{surf}}{dt} \right]_{\text{lrdbbc}=\text{F}}$ and $\left[ \frac{d\mathbf{w}_{surf}}{dt} \right]_{\text{lrdbbc}=\text{T}}$.
  - Calculation of $\left[ \frac{d}{dt} \right]_{l=L,\text{lrdbbc}=\text{T}}$ by the following formula:
    \[
    \left[ \frac{d}{dt} \right]_{l=L,\text{lrdbbc}=\text{T}} = \left[ \frac{d}{dt} \right]_{l=L,\text{lrdbbc}=\text{F}} - \left( \left[ \frac{d\mathbf{w}_{surf}}{dt} \right]_{\text{lrdbbc}=\text{T}} - \left[ \frac{d\mathbf{w}_{surf}}{dt} \right]_{\text{lrdbbc}=\text{F}} \right) \frac{\eta \nabla \Pi + \mathbf{R}_{\Delta t} \nabla \Pi}{R_{\Delta t}} \big|_{l=L}
    \]
    where $MM$ is the decentered middle of $O$ and $F$ (that requires a specific interpolation of $\left[ \frac{\eta \nabla \Pi + \mathbf{R}_{\Delta t} \nabla \Pi}{R_{\Delta t}} \right]_{l=L}$ at $O$).

  For more details about these calculations and the additional interpolations required, see documentation (IDVNH3), especially the parts 1 to 3.

  The above formulae are valid for thin layer equations. For WS2003 deep-layer equations change $\Pi$ into $\tilde{\Pi}$, $g$ into $G$, $w$ into $(r^2/a^2)w$ in the above formulae.
5.3 Discretisation for a 3D variable in a 3D model: particular case where the RHS has zero linear and non-linear terms (advectable GFL variables).

List of equations.

Humidity equation, and for example:

• Ozone equation.
• Liquid water equation.
• Ice equation.
• Cloudiness equation.
• TKE.
• Aerosols equation.
• Extra GFL variables equations.

See documentation (IDEUL) for a comprehensive list of advectable GFL variables.

Generic notations.

Generic notation \( P(X)L9 \), \( P(X)T1 \) stands for:

• \( PGFL9 \), \( PGFLT1 \) for GFL variables.

Generic notation for total term, linear term, non-linear term, physics: \( A \) is the total term (sum of dynamical contributions), \( B \) is the linear term (treated in the semi-implicit scheme), the difference \( A - \beta B \) is the non-linear term. \( F \) is the sum of contributions computed in the physical parameterizations. In the present case \( A \) and \( B \) are equal to zero.

Description stands for the general case where diabatic and adiabatic terms are gathered in the same buffer, and where no additional splitting is required to do diagnostics or to apply SLHD interpolations to a subset of the terms interpolated by a high-order interpolation. In some particular cases, additional splitting involving separate buffers may be required:

• When some diagnostics (for example DDH) impose that evaluation of the dynamics and of the physics at the origin point of the SL trajectory must be done separately in two different buffers (controlled by variable NSPLTHOI set to -1). Buffer \( P(X)LF9 \) is used instead of \( P(X)L9 \) to store quantities, the interpolation of which is a non-SLHD high-order one.
• When diabatic terms use a different interpolation from the adiabatic one. That can be the case for non-zero values of \( NSPLTHOI \), attribute LPHYLIN set to T. Buffer \( P(X)LF9 \) is used instead of \( P(X)L9 \) to store diabatic terms to be interpolated.

Other points.

* High-order interpolations: In the following discretisations, “high-order interpolations” means: 32 points interpolations for 3D terms (vertical interpolations are cubic), 12 points interpolations for 2D terms. For ozone, vertical cubic interpolations can be replaced by vertical Hermite cubic interpolations (switch YO3_NL%LHV in NAMGFL), or vertical spline cubic interpolations (switch YO3_NL%LVSPLIP in NAMGFL).
**Uncentering:** \( \epsilon \) is a first-order “uncentering factor”. It allows to remove the noise due to gravity waves (orographic resonance).

### 3TL vertical interpolating SL scheme.

Equation (10) is discretised as follows:

\[
X_F^+ = \{X^- + [2\Delta t F^-]\}_O
\]

Buffers content before interpolations:

- **P(X)L9**: \( X^- + [2\Delta t F^-] \) for high-order interpolation at the origin point \( O \).
- **P(X)T1** contains zero; evaluated at the final point \( F \).

### 2TL vertical interpolating SL scheme.

At the first time integration step, values at \( t + \frac{\Delta t}{2} \) are set equal to initial values. This discretisation of the 2TL SL scheme follows (Mc Donald and Haugen, 1992).

Equation (10) is discretised as follows:

\[
X_F^+ = \{X^o + [\Delta t F]^o\}_O
\]

Buffers content before interpolations:

- **P(X)L9**: \( X^o + [\Delta t F]^o \) for high-order interpolation at the origin point \( O \).
- **P(X)T1** contains zero; evaluated at the final point \( F \).

**Remark for iterative centred-implicit schemes:** For options where this type of scheme involves the momentum equation (this is the case for the option LPC\_FULL\_T.) \( X_F^+ \) has to be recomputed at all iterations of the iterative centred-implicit scheme since the origin point \( O \) is recomputed at each iteration.

### Non advectable pseudo-historic GFL variables.

For these variables the discretization always writes (3TL SL and 2TL SL):

\[
X_F^+ = \{X^o + [\Delta t F]^o\}_F
\]

and there are never interpolations.
4. Semi-lagrangian computations

5.4 Discretisation for a 2D variable in a 3D model (GMVS variables, for example continuity equation).

The equation which is now discretised is:

\[ [\mathcal{R}_{\text{inte}}(\text{top,surf})] \left( \frac{W_{\text{vei}} dX}{\Delta \eta} \right) = [\mathcal{R}_{\text{inte}}(\text{top,surf})] \left( \frac{W_{\text{vei}} A}{\Delta \eta} \right) + [\mathcal{R}_{\text{inte}}(\text{top,surf})] \left( \frac{W_{\text{vei}} F}{\Delta \eta} \right) \] (62)

where:

\[ [\mathcal{R}_{\text{inte}}(\text{top,surf})] \left( \frac{W_{\text{vei}}}{\Delta \eta} \right) = 1 \] (63)

and \([\mathcal{R}_{\text{inte}}(\text{top,surf})]\) is the vertical integral matrical operator (the scalar product \([\mathcal{R}_{\text{inte}}(\text{top,surf})] \langle X \rangle\) is the discretisation of \(\int_{\eta=0}^{\eta=1} \eta d\eta\), \(\langle X \rangle\) is the vector containing the layer values of \(X\): \((X_1; X_2; ...; X_l; ...; X_L)\)).

In the thin layer equations or in the Wood and Staniforth formulation of deep-layer equations, expression of \(W_{\text{vei}}\) at full levels is:

\[ |W_{\text{vei}}| = \Delta B_l \] (64)

In the deep layer equations (White and Bromley, 1995), expression of \(W_{\text{vei}}\) at full levels is:

\[ |W_{\text{vei}}| = \left[ \frac{\Delta B_l \left[ \frac{\Delta \eta}{\Delta \eta} \right]^2}{[\mathcal{R}_{\text{inte}}(\text{top,surf})] \left( \frac{\Delta B_l \left[ \frac{\Delta \eta}{\Delta \eta} \right]^2}{\Delta \eta} \right)} \right] \] (65)

When the finite element vertical discretisation is activated (\texttt{LVERTFE=.T.}), \([\mathcal{R}_{\text{inte}}(\text{top,surf})]\) is a vector tricky to compute, it is precomputed in the setup routine \texttt{SUVERTFE1} or \texttt{SUVERTFE3} and stored in the array \texttt{RINTE} of \texttt{YOMCVER}. Vertical integrations are done in routine \texttt{VERINT}. For more details see the part of the appendix of documentation (IDEUL) which explains the computation of matrix \([\mathcal{R}_{\text{inte}}]\). When the finite difference vertical discretisation is activated (\texttt{LVERTFE=.F.}), \([\mathcal{R}_{\text{inte}}(\text{top,surf})]\) is simply the vector of coordinates \((|\Delta \eta_1|; |\Delta \eta_2|; ...; |\Delta \eta_l|; ...; |\Delta \eta_L|)\) so:

- equation (62) rewrites:

  \[ \sum_{i=1}^{L} |W_{\text{vei}}| \left( \frac{dX}{dt} \right)_i = \sum_{i=1}^{L} |W_{\text{vei}}| \cdot A_i + \sum_{i=1}^{L} |W_{\text{vei}}| \cdot F_i \] (66)

- equation (65) rewrites:

  \[ |W_{\text{vei}}| = \left[ \frac{\Delta B_l \left[ \frac{\Delta \eta}{\Delta \eta} \right]^2}{\sum_{k=1}^{L} |\Delta B_k| \left[ \frac{\Delta \eta}{\Delta \eta} \right]^2} \right] \] (67)

- routine \texttt{VERINT} is not used in this case.

List of equations.

- Continuity equation.
Generic notations.

Generic notation $N(X)_{LAG}$ stands for:

- $N_{LAG}$ for continuity equation.

Generic notations $P(X)_{2D}$ (2D term), $P(X)_{T1}$ (2D term), $P(X)_{3D}$ (3D term), stand for:

- $P_{2D}$, $PS_{T1}$, $P_{CL}$ for continuity equation.

Generic notation $P(X)_{NLT}$ (3D term) stands for:

- $PS_{NLT}$ for continuity equation.

Generic notation for total term, linear term, non linear term, physics:

- $A$ is the total term (sum of dynamical contributions): it is assumed to be a 3D term (sum of 3D and 2D contributions).
- $B$ is the linear term (treated in the semi-implicit scheme): it is assumed to be a 2D term (vertical integral of a 3D term).
- the difference $A - \beta B$ is the non-linear term, considered as a 3D term.
- $F$ is the sum of contributions computed in the physical parameterizations; it is assumed to be a 2D term (vertical integral of a 3D term).

Description stands for the general case where linear and non-linear terms are gathered in the same buffer, and where no additional splitting is required to do diagnostics or to apply SLHD interpolations to a subset of the terms interpolated by a high-order interpolation. In some particular cases, additional splitting involving separate buffers may be required (see in part 5.2).

Other points.

* **High-order interpolations:** In the following discretisations, “high-order interpolations” means: 32 points interpolations for 3D terms (vertical interpolations are cubic), 12 points interpolations for 2D terms.

* **Uncentering:** $\epsilon$ is a first-order “uncentering factor”. It allows to remove the noise due to gravity waves (orographic resonance).

* **Horizontal interpolation of 2D terms:** Since the horizontal position of the interpolation point is vertical dependent, horizontal interpolations of 2D quantities have to be done for each layer. For example, when interpolating a 2D surface variable at the origin point, $[R_{inte}]_{(top, surf)} \left[ \frac{W_{x1}}{\Delta \eta} \right]_P [surface \ quantity]_o$ has no reason to be equal to $[surface \ quantity]_{o(\eta=1)}$, these quantities are generally different: this is $[R_{inte}]_{(top, surf)} \left[ \frac{W_{x1}}{\Delta \eta} \right]_P [surface \ quantity]_o$ which has to be computed. $\left[ \frac{W_{x1}}{\Delta \eta} \right]_P [surface \ quantity]_{o(l)}$, $l = 1$ to $L$. For $LVERTFE=.F$. this is $\sum_{l=1}^{L} [W_{x1}]_P [surface \ quantity]_{o(l)}$. 

3TL vertical interpolating SL scheme.

Equation (62) is discretised as follows:

\[
(X - (1 + \epsilon)\Delta t [\beta B])^+ = [R_{\text{inte}}(\text{top, surf})] \left( \frac{W_{\text{top, surf}}}{\Delta t} \right)_F \{X^+ + (1 + \epsilon)\Delta t [A - \beta B]^+ + [(1 - \epsilon)\Delta t [\beta B] + 2\Delta t F] \}^O + [(1 + \epsilon)\Delta t [A - \beta B]^+ \}^O
\]  

Buffers content before interpolations for \(N(X)\)LAG=2:

- \(P(X)\)TL9: \((1 - \epsilon)\Delta t [A - \beta B]^+ + [(1 - \epsilon)\Delta t [\beta B]^+ \text{ for high-order interpolation at the origin point } O(l)\).
- \(P(X)\)TL9: \((X^- + 2\Delta t [F]^+) \text{ for horizontal high-order interpolation at the origin point } O(l)\).
- \(P(X)\)TL1: \((1 + \epsilon)\Delta t [A - \beta B]^+ \text{ then provisional add of quantity } [(1 + \epsilon)\Delta t [\beta B]^+ \text{ before } t + dt \text{ physics; evaluated at the final point } F\).

Buffers content before interpolations for \(N(X)\)LAG=3:

- \(P(X)\)TL9: \((1 - \epsilon)\Delta t [A - \beta B]^+ + [(1 - \epsilon)\Delta t [\beta B]^+ \text{ for trilinear interpolation at the origin point } O(l)\).
- \(P(X)\)TL9: \((X^- + 2\Delta t [F]^+) \text{ for horizontal high-order interpolation at the origin point } O(l)\).
- \(P(X)\)TL1: \((1 + \epsilon)\Delta t [A - \beta B]^+ \text{ then provisional add of quantity } [(1 + \epsilon)\Delta t [\beta B]^+ \text{ before } t + dt \text{ physics; evaluated at the final point } F\).

2TL vertical interpolating SL scheme: conventional discretisation \((\text{LSETTLS}=-.F.)\) and first-order uncentering.

The \(t + \frac{\Delta t}{2}\) non-linear term \(A^m - \beta B^m\) used in the 2TL SL scheme is computed by a linear temporal extrapolation using the \(t\) and \(t - \Delta t\) quantities at the same location. At the first time integration step, values at \(t + \frac{\Delta t}{2}\) are set equal to initial values. This discretisation of the 2TL SL scheme follows (Mc Donald and Haugen, 1992). Quantity \(A^m - \beta B^m\) has to be saved in a buffer \(P(X)\)NL9 to be available as \(A^m - \beta B^m\) for the following timestep.

Equation (62) is discretised as follows:

\[
(X - (1 + \epsilon)\frac{\Delta t}{2} [\beta B])^+ = [R_{\text{inte}}(\text{top, surf})] \left( \frac{W_{\text{top, surf}}}{\Delta t} \right)_F \{X^+ + (1 - \epsilon)\frac{\Delta t}{2} [A - \beta B] + [(1 - \epsilon)\frac{\Delta t}{2} [\beta B] + \Delta t F] \}^O + [(1 + \epsilon)\frac{\Delta t}{2} [A - \beta B]^+ \}^O
\]  

which can be rewritten, once expanded the extrapolation:

\[
(X - (1 + \epsilon)\frac{\Delta t}{2} [\beta B])^+ = [R_{\text{inte}}(\text{top, surf})] \left( \frac{W_{\text{top, surf}}}{\Delta t} \right)_F \{X^+ + (1 - \epsilon)\frac{\Delta t}{2} [A + \Delta t F] \}^O + 0.5[R_{\text{inte}}(\text{top, surf})] \left( \frac{W_{\text{top, surf}}}{\Delta t} \right)_F \{1 - \epsilon\frac{\Delta t}{2} [(A - \beta B)^+ - (A - \beta B)] O + \frac{\Delta t}{2} [(1 + \epsilon)(1.5(A - \beta B)^+ - 0.5(A - \beta B)] \}
\]  

Buffers content before interpolations for \(N(X)\)LAG=2:
Equation (62) is discretised as follows:

1. \( P(\text{X3D})L9: (1 - \epsilon) \frac{\Delta t}{2} [1.5(A - \beta B)^\circ - 0.5(A - \beta B)^-] + [(1 - \epsilon) \frac{\Delta t}{2} \beta B]^\circ \) for high-order interpolation at the origin point \( O(l) \).

2. \( P(\text{X2D})9: X^\circ + [\Delta t F]^\circ \) for horizontal high-order interpolation at the origin point \( O(l) \).

3. \( P(\text{X})T1: (1 + \epsilon) \frac{\Delta t}{2} [1.5(A - \beta B)^\circ - 0.5(A - \beta B)^-] \) then provisional add of quantity \([(1 + \epsilon) \frac{\Delta t}{2} \beta B]^\circ\) before \( t + \Delta t \) physics; evaluated at the final point \( F \).

4. \( \epsilon \) remark: another possibility is to add all the content of \( P(\text{X2D})9 \) to \( P(\text{X3D})L9 \) in order to remove one horizontal interpolation.

Buffers content before interpolations for \( N(\text{X})\text{LAG}=3 \):

1. \( P(\text{X3D})L9: (1 - \epsilon) \frac{\Delta t}{2} [1.5(A - \beta B)^\circ - 0.5(A - \beta B)^-] + [(1 - \epsilon) \frac{\Delta t}{2} \beta B]^\circ \) for trilinear interpolation at the origin point \( O(l) \).

2. \( P(\text{X2D})9: X^\circ + [\Delta t F]^\circ \) for horizontal high-order interpolation at the origin point \( O(l) \).

3. \( P(\text{X})T1: (1 + \epsilon) \frac{\Delta t}{2} [1.5(A - \beta B)^\circ - 0.5(A - \beta B)^-] \) then provisional add of quantity \([(1 + \epsilon) \frac{\Delta t}{2} \beta B]^\circ\) before \( t + \Delta t \) physics; evaluated at the final point \( F \).

2TL vertical interpolating SL scheme: conventional discretisation (LSETTILLS=..F.) and pseudo-second order uncentering.

One starts to remove uncentering \( \epsilon \) from the nonlinear terms and to apply a second-order uncentering \( \epsilon_X \) to linear terms, that yields a term \( B^- \) in the discretisation. From property given by formula (67), one can remove term \( B^- \) and show that discretisation is equivalent to replace \( \beta \) by \((1 + \epsilon_X)\beta\). For more details, see part 5 (equations (37) and (38)) of (Simmons and Temperton, 1996). In equation (69), uncentering \( \epsilon \) has to be replaced by zero, and \( \beta \) to be replaced by \((1 + \epsilon_X)\beta\). The \( t + \frac{\Delta t}{2} \) non-linear term \( A^m - (1 + \epsilon_X)\beta B^m \) used in the 2TL SL scheme is computed by a linear temporal extrapolation using the \( t \) and \( t - \Delta t \) quantities at the same location. At the first time integration step, values at \( t + \frac{\Delta t}{2} \) are set equal to initial values and second-order uncentering is replaced by a first order uncentering. This discretisation of the 2TL SL scheme follows (Mc Donald and Haugen, 1992). Quantity \( A^\circ - (1 + \epsilon_X)\beta B^\circ \) has to be saved in a buffer \( P(\text{X})\text{NL}T9 \) to be available as \( A^- - (1 + \epsilon_X)\beta B^- \) for the following timestep.

Equation (69) is discretised as follows:

\[
\begin{align*}
(X - (1 + \epsilon_X) \frac{\Delta t}{2} \beta B)^\circ &= \left[ R_{\text{inte}}(\text{top,surf}) \right] \left[ \frac{\partial B}{\partial \phi} \right]_f \{ X^\circ + \frac{\Delta t}{2} [A - (1 + \epsilon_X) \beta B]^m + [(1 + \epsilon_X) \frac{\Delta t}{2} \beta B + \Delta t F]^\circ \} \bigg|_o + \left( \frac{\Delta t}{2} [A - (1 + \epsilon_X) \beta B]^m \right)_f \bigg|_o \\
&= \left[ R_{\text{inte}}(\text{top,surf}) \right] \left[ \frac{\partial B}{\partial \phi} \right]_f \left\{ X^\circ + \frac{\Delta t}{2} [A - (1 + \epsilon_X) \beta B + \Delta t F]^\circ \right\} \bigg|_o + \left( \frac{\Delta t}{2} [A - (1 + \epsilon_X) \beta B]^m \right)_f \bigg|_o
\end{align*}
\]

which can be rewritten, once expanded the extrapolation:

\[
\begin{align*}
(X - (1 + \epsilon_X) \frac{\Delta t}{2} \beta B)^\circ &= \left[ R_{\text{inte}}(\text{top,surf}) \right] \left[ \frac{\partial B}{\partial \phi} \right]_f \left\{ X^\circ + \frac{\Delta t}{2} [A - (1 + \epsilon_X) \beta B + \Delta t F]^\circ \right\} \bigg|_o \\
&+ 0.5 \left[ R_{\text{inte}}(\text{top,surf}) \right] \left[ \frac{\partial B}{\partial \phi} \right]_f \left\{ A - (1 + \epsilon_X) \beta B^o - (A - (1 + \epsilon_X) \beta B^-)^o \right\} \bigg|_o \\
&\quad + \frac{\Delta t}{2} [1.5(A - (1 + \epsilon_X) \beta B)^o - 0.5(A - (1 + \epsilon_X) \beta B^-)^o]_f \bigg|_o
\end{align*}
\]

Buffers content before interpolations for \( N(\text{X})\text{LAG}=2 \):
4. Semi-lagrangian computations

- **P(X3D)L9**: $\Delta_t^2[1.5(A - (1 + \epsilon X)\beta B)^{\nu} - 0.5(A - (1 + \epsilon X)\beta B)^{-}] + [(1 + \epsilon X)\Delta_t^2 \beta B]^{\nu}$ for high-order interpolation at the origin point $O(l)$.
- **P(X2D)9**: $X^\nu + [\Delta t F]^\nu$ for horizontal high-order interpolation at the origin point $O(l)$.
- **P(X)T1**: $\Delta_t^2[1.5(A - (1 + \epsilon X)\beta B)^{\nu} - 0.5(A - (1 + \epsilon X)\beta B)^{-}]$ then provisional add of quantity $[(1 + \epsilon X)\Delta_t^2 \beta B]^{\nu}$ before $t + dt$ physics; evaluated at the final point $F$.
- **remark**: another possibility is to add all the content of $P(X2D)9$ to $P(X3D)L9$ in order to remove one horizontal interpolation.

Buffers content before interpolations for $N(X)LAG=3$:

- **P(X3D)L9**: $\Delta_t^2[1.5(A - (1 + \epsilon X)\beta B)^{\nu} - 0.5(A - (1 + \epsilon X)\beta B)^{-}] + [(1 + \epsilon X)\Delta_t^2 \beta B]^{\nu}$ for trilinear interpolation at the origin point $O(l)$.
- **P(X2D)9**: $X^\nu + [\Delta t F]^\nu$ for horizontal high-order interpolation at the origin point $O(l)$.
- **P(X)T1**: $\Delta_t^2[1.5(A - (1 + \epsilon X)\beta B)^{\nu} - 0.5(A - (1 + \epsilon X)\beta B)^{-}]$ then provisional add of quantity $[(1 + \epsilon X)\Delta_t^2 \beta B]^{\nu}$ before $t + dt$ physics; evaluated at the final point $F$.

2TL vertical interpolating SL scheme: stable discretisation (LSETTLS=T.) and first-order uncentering.

The $t + \Delta t$ non-linear term $A^m - \beta B^m$ used in the 2TL SL scheme if LSETTLS=T. is replaced by a linear spatio-temporal extrapolation comparable to the one applied to the wind components for the research of trajectory (see formula [101]), except the fact that there is an additional uncentering. Term $A^m - \beta B^m$ is replaced by $0.5(1 + \epsilon)[A^\nu - \beta B^\nu]_0 + 0.5(2 - \epsilon)[A^\nu - \beta B^\nu]_O - 0.5[A^\nu - \beta B^\nu]_O$.

This type of extrapolation is available only for $N(X)LAG=3$. At the first time integration step, values at $t + \Delta t$ are set equal to initial values. Quantity $A^\nu - \beta B^\nu$ has to be saved in a buffer $P(X)NLT9$ to be available as $A^\nu - \beta B^\nu$ for the following timestep.

Equation (62) is discretised as follows:

$$
(X - (1 + \epsilon)\frac{\Delta t}{2} \beta B)^\nu_f
= \left[R_{inter\,(top, surf)} \left\langle \frac{W_{seq}}{\Delta t} \right\rangle \right] F \{X^\nu + (2 - \epsilon)\frac{\Delta t}{2} [A - \beta B]^\nu - [(1 + \epsilon)\frac{\Delta t}{2} \beta B + \Delta t F]^\nu\} + \{(1 + \epsilon)\frac{\Delta t}{2} [A - \beta B]^\nu\} \}
$$

which can be rewritten:

$$
(X - (1 + \epsilon)\frac{\Delta t}{2} \beta B)^\nu_f
= \left[R_{inter\,(top, surf)} \left\langle \frac{W_{seq}}{\Delta t} \right\rangle \right] F \{X^\nu + (1 - \epsilon)\frac{\Delta t}{2} A + \Delta t F\}^\nu\} + \{(1 + \epsilon)\frac{\Delta t}{2} [A - \beta B]^\nu\}
$$

Buffers content before interpolations for $N(X)LAG=3$:

- **P(X3D)L0** is not used.
Equation (62) is discretised as follows:

- \( P(X3D)L9 \): \((2 - \epsilon) \frac{\Delta}{\Delta^2}[A - \beta B]^o - \frac{\Delta}{\Delta^2}[A - \beta B]^- + [(1 - \epsilon) \frac{\Delta}{\Delta^2} \beta B]^o \) for trilinear interpolation at the origin point \( O(l) \) (which can be rewritten: \((1 - \epsilon) \frac{\Delta}{\Delta^2} A + \frac{\Delta}{\Delta^2} \{(A - \beta B)^o - [A - \beta B]^- \})\).
- \( P(X2D)0 \) is not used.
- \( P(X2D)9 \): \( X^o + [\Delta t F]^o \) for horizontal high-order interpolation at the origin point \( O(l) \).
- \( P(X)T1 \): \((1 + \epsilon) \frac{\Delta}{\Delta^2}[A - \beta B]^o \) then provisional add of quantity \([(1 + \epsilon) \frac{\Delta}{\Delta^2} \beta B]^o \) before \( t + dt \) physics; evaluated at the final point \( F \).

2TL vertical interpolating SL scheme: stable discretisation (LSET-TLS=T.) and pseudo-second order uncentering.

In equation (73), uncentering \( \epsilon \) has to be replaced by zero, and \( \beta \) has to be replaced by \((1 + \epsilon x) \beta \). At the first time integration step, values at \( t + \frac{\Delta}{2} \) are set equal to initial values and second-order uncentering is replaced by a first order uncentering. Quantity \( A^o = (1 + \epsilon x) \beta B^o \) has to be saved in a buffer \( P(X)NLT9 \) to be available as \( A^o = (1 + \epsilon x) \beta B^o \) for the following timestep.

Equation (62) is discretised as follows:

\[
(X - (1 + \epsilon x) \frac{\Delta}{\Delta^2} \beta B)_p^+ = [R_{inte}(top, surf)] \left( \left[ \frac{\Delta}{\Delta^2} \right]_F \{X^o + \Delta t[A - (1 + \epsilon x) \beta B]^o - \frac{\Delta}{\Delta^2}[A - (1 + \epsilon x) \beta B]^- + [(1 + \epsilon x) \frac{\Delta}{\Delta^2} \beta B + \Delta \frac{\Delta}{\Delta^2} A - (1 + \epsilon x) \beta B]^o \} \right)
\]

which can be rewritten:

\[
(X - (1 + \epsilon x) \frac{\Delta}{\Delta^2} \beta B)_p^+ = [R_{inte}(top, surf)] \left( \left[ \frac{\Delta}{\Delta^2} \right]_F \{X^o + \Delta t[A - (1 + \epsilon x) \beta B]^o - \frac{\Delta}{\Delta^2}[A - (1 + \epsilon x) \beta B]^- \} \right) + \left[ \frac{\Delta}{\Delta^2} \right]_F \{A - (1 + \epsilon x) \beta B]^o \}
\]

Buffers content before interpolations for \( N(X)LAG = 3 \):

- \( P(X3D)L0 \) is not used.
- \( P(X3D)L9 \): \( \Delta t[A - (1 + \epsilon x) \beta B]^o - \frac{\Delta}{\Delta^2}[A - (1 + \epsilon x) \beta B]^- + [(1 + \epsilon x) \frac{\Delta}{\Delta^2} \beta B]^o \) for trilinear interpolation at the origin point \( O(l) \) (which can be rewritten: \( \frac{\Delta}{\Delta^2} A + \frac{\Delta}{\Delta^2} \{(A - (1 + \epsilon x) \beta B)^o - [A - (1 + \epsilon x) \beta B]^- \})\).
- \( P(X2D)0 \) is not used.
- \( P(X2D)9 \): \( X^o + [\Delta t F]^o \) for horizontal high-order interpolation at the origin point \( O(l) \).
- \( P(X)T1 \): \( \frac{\Delta}{\Delta^2}[A - (1 + \epsilon x) \beta B]^o \) then provisional add of quantity \([(1 + \epsilon x) \frac{\Delta}{\Delta^2} \beta B]^o \) before \( t + dt \) physics; evaluated at the final point \( F \).

5.5 Discretisation for a 2D variable in a 2D model.

The content of the part (5.2) is generally valid, but there are particular remarks.
List of equations.

- Momentum equation.
- Continuity equation.

Generic notations.

Generic notation $N(X)\text{LAG}$ stands for:

- $NW\text{LAG}$ for momentum equation.
- $NV\text{LAG}$ for continuity equation. Negative values of $NV\text{LAG}$ are used for Lagrangian formulation of continuity equation (the following discretisations apply to the absolute value of $NV\text{LAG}$), only $NV\text{LAG}=-2$ is available in this case.

Generic notation $P(X)L0$, $P(X)L9$, $P(X)T1$ stands for:

- $PUL0$, $PUL9$, $PUT1$ for U-momentum equation.
- $PVL0$, $PVL9$, $PVT1$ for V-momentum equation.
- $PSPL0$, $PSPL9$, $PSPT1$ for continuity equation.

Generic notation $P(X)NLT9$ stands for:

- $PUNLT9$ for U-momentum equation.
- $PVNLT9$ for V-momentum equation.
- $PSPNLT9$ for continuity equation.

Generic notation for total term, linear term, non linear term: $A$ is the total term (sum of dynamical contributions), $B$ is the linear term (treated in the semi-implicit scheme), the difference $A - \beta B$ is the non-linear term.

Other points.

* High-order interpolations: In the following discretisations, “high-order interpolations” means 12 points interpolations.

* Uncentering: $\epsilon$ is a first-order “uncentering factor”. It allows to remove the noise due to gravity waves (orographic resonance).

* Vectorial variables: The following discretisations are written for scalar variables. For vectorial variables (for example the horizontal wind) a rotation operator $R$ has to be applied from interpolation point to final point:

- expression interpolated at $O$ has to be replaced by $R^{OF}\{\text{this expression}\}_O$.
- expression interpolated at $M$ has to be replaced by $R^{MF}\{\text{this expression}\}_M$. 
3TL SL scheme.

Equation:

\[
\frac{dX}{dt} = A \quad (77)
\]

is discretised as follows:

\[
(X - (1 + \epsilon)\Delta t \beta B)_{P(X)} = \{X^o + [(1 - \epsilon)\Delta t A - (1 - \epsilon)\Delta t \beta B] + [(1 - \epsilon)\Delta t \beta B]^o \} \big|_{O} \\
+ \{[(1 + \epsilon)\Delta t A - (1 + \epsilon)\Delta t \beta B]^o \} \big|_{F}
\]  

Buffers content before interpolations for \(N(X)LAG=2\):

- **P(X)L0**: not used.
- **P(X)L9**: \(X^o + [(1 - \epsilon)\Delta t A - (1 - \epsilon)\Delta t \beta B] + [(1 - \epsilon)\Delta t \beta B]^o \) for high-order interpolation at the origin point \(O\).
- **P(X)T1**: \([(1 + \epsilon)\Delta t A - (1 + \epsilon)\Delta t \beta B]^o \); evaluated at the final point \(F\).

Buffers content before interpolations for \(N(X)LAG=3\):

- **P(X)L0**: \([(1 - \epsilon)\Delta t A - (1 - \epsilon)\Delta t \beta B] + [(1 - \epsilon)\Delta t \beta B]^o \) for bilinear interpolation at the origin point \(O\).
- **P(X)L9**: \(X^o \) for high-order interpolation at the origin point \(O\).
- **P(X)T1**: \([(1 + \epsilon)\Delta t A - (1 + \epsilon)\Delta t \beta B]^o \); evaluated at the final point \(F\).

2TL SL scheme: conventional discretisation (LSETTLS=.F.) and first-order uncentering.

The \(t + \Delta t\) non-linear term \(A^o - \beta B^m\) used in the 2TL SL scheme is computed by a linear temporal extrapolation using the \(t\) and \(t - \Delta t\) quantities at the same location. At the first time integration step, values at \(t + \frac{\Delta \t}{2}\) are set equal to initial values. This discretisation of the 2TL SL scheme follows (McDonald and Haugen, 1992). Quantity \(A^o - \beta B^m\) has to be saved in a buffer \(P(X)NLT9\) to be available as \(A^o - \beta B^m\) for the following timestep, when non-zero (i.e. only for continuity equation, if \(\beta=1\)).

Equation (77) is discretised as follows:

\[
(X - (1 + \epsilon)\frac{\Delta t}{2} \beta B)_{P(X)} = \{X^o + [(1 - \epsilon)\frac{\Delta t}{2} A - (1 - \epsilon)\frac{\Delta t}{2} \beta B] + [(1 - \epsilon)\frac{\Delta t}{2} \beta B]^o \} \big|_{O} \\
+ \{[(1 + \epsilon)\frac{\Delta t}{2} A - (1 + \epsilon)\frac{\Delta t}{2} \beta B]^o \} \big|_{F}
\]  

Buffers content before interpolations for \(N(X)LAG=2\):

- **P(X)L0**: not used.
- **P(X)L9**: \(X^o + [(1 - \epsilon)\frac{\Delta t}{2} A - (1 - \epsilon)\frac{\Delta t}{2} \beta B] + [(1 - \epsilon)\frac{\Delta t}{2} \beta B]^o \) for high-order interpolation at the origin point \(O\).
- **P(X)T1**: \([(1 + \epsilon)\frac{\Delta t}{2} A - (1 + \epsilon)\frac{\Delta t}{2} \beta B]^o \); evaluated at the final point \(F\).

Buffers content before interpolations for \(N(X)LAG=3\):

- **P(X)L0**: \([(1 - \epsilon)\frac{\Delta t}{2} A - (1 - \epsilon)\frac{\Delta t}{2} \beta B] + [(1 - \epsilon)\frac{\Delta t}{2} \beta B]^o \) for bilinear interpolation at the origin point \(O\).
- **P(X)L9**: \(X^o \) for high-order interpolation at the origin point \(O\).
- **P(X)T1**: \([(1 + \epsilon)\frac{\Delta t}{2} A - (1 + \epsilon)\frac{\Delta t}{2} \beta B]^o \); evaluated at the final point \(F\).
* Remark for momentum equation: $\beta$ can only take the value 1 and in this case the non-linear term $A - B$ is zero; simplifications are made in the code; arrays PUNLT9 and PVNLT9 are useless and not allocated.

2TL SL scheme: stable discretisation (LSETTLS=.T.) and first-order uncentering.

The $t + \frac{\Delta t}{2}$ non-linear term $A^m - \beta B^m$ used in the 2TL SL scheme if LSETTLS=.F. is replaced in the case LSETTLS=.T. by a linear spatio-temporal extrapolation comparable to the one applied to the wind components for the research of trajectory (see formula 101), except the fact that there is an additional uncentering. Term $A^m - \beta B^m$ is replaced in the case $\beta = 1$.

Equation (77) is discretised as follows:

$$
(X - (1 + \epsilon) \frac{\Delta t}{2} \beta B)^o_p = \{X^o + [(2 - \epsilon) \frac{\Delta t}{2} A - (2 - \epsilon) \frac{\Delta t}{2} \beta B]^o - [(1 + \epsilon) \frac{\Delta t}{2} A - \frac{\Delta t}{2} \beta B]^o + [(1 + \epsilon) \frac{\Delta t}{2} A - (1 + \epsilon) \frac{\Delta t}{2} \beta B]^o\}_o + [(1 + \epsilon) \frac{\Delta t}{2} A - (1 + \epsilon) \frac{\Delta t}{2} \beta B]^o\}_p 
$$

Buffers content before interpolations for N(X)LAG=3:

- P(X)L0: $[(2 - \epsilon) \frac{\Delta t}{2} A - (2 - \epsilon) \frac{\Delta t}{2} \beta B]^o - [(1 + \epsilon) \frac{\Delta t}{2} A - \frac{\Delta t}{2} \beta B]^o$ for bilinear interpolation at the origin point $O$.
- P(X)L9: $X^o$ for high-order interpolation at the origin point $O$.
- P(X)T1: $[(1 + \epsilon) \frac{\Delta t}{2} A - (1 + \epsilon) \frac{\Delta t}{2} \beta B]^o$; evaluated at the final point $F$.

* Remark for momentum equation: $\beta$ can only take the value 1 and in this case the non-linear term $A - B$ is zero; simplifications are made in the code; arrays PUNLT9 and PVNLT9 are useless and not allocated.

2TL SL scheme: pseudo-second order uncentering.

In cycle 37T1 of ARPEGE/IFS the pseudo-second order uncentering is not coded in the shallow-water model.

5.6 Additional vertical derivatives.

If $\delta T_R$ is non-zero, discretisation of temperature equation needs to compute the vertical advection $\left(\int \frac{\partial \alpha_T}{\partial \eta} \right)$ (at full levels) of $\alpha_T$. Layers values of $\alpha_T$ (array RCORDIF) are used to define $T + \delta T_R \frac{\alpha_T}{\alpha_T}$, but half level values of $\alpha_T$ (array RCORDDH) are used to compute vertical advection.

There is something similar in the $\hat{Q}$ equation if $\delta P$ is non-zero (Wood and Staniforth deep-layer version of the NH-PDVD model, use of an array RCORPDH).
5.7 Implicit treatment of some non-linear terms in the case of NH-PDVD non-hydrostatic model.

No longer existing option.

5.8 Case when some variables are evaluated at half levels.

The prognostic variables and the RHS of equations are generally evaluated at full levels in the discretisation. In the NH-PDVD non-hydrostatic model there is an option allowing to advect \( w \) (at half levels) instead of \( d \) (at full levels). The previous general considerations valid to layer variables also apply to half level variables, but a "half level" trajectory has now to be computed (the origin \( O \) now matches a half level final point \( F \)).

- Horizontal displacement at half levels: the coordinates of the half level-trajectory interpolation point are computed as the average (with a vertical weight taking account of \( \eta \)) of the coordinates of the adjacent full level-trajectory interpolation points; this average is a lon-lat average on the computational sphere in spherical geometry, and a x-y average on the projection plane in plane geometry (ALADIN).

- Vertical displacement at half levels: the vertical coordinate of the half level-trajectory interpolation point is computed as the average (with a vertical weight taking account of \( \eta \)) of the vertical coordinates of the adjacent full level-trajectory interpolation points; no vertical displacement if there is no vertical displacement for the two adjacent full levels.

- No complete iterative recalculation of trajectory is done for half-level trajectories.

5.9 Remarks for spline cubic vertical interpolations.

In this case the vertical interpolation uses all model levels and can be written as the product of two vertical interpolations: the first one uses all model levels and can be done at \( F \) in the unlagged grid-point calculations (the intermediate quantity obtained is stored in the array \( P(X)SPL9 \)), the second one is a 4 points interpolation, done in the lagged grid-point calculations in the interpolation routine. Interpolation routine uses both \( P(X)SPL9 \) (for interpolations) and \( P(X)L9 \) to apply a monotonicity constraint.
6 Computation of medium and origin points.

Preliminary remark: the subsections (6.1), (6.2) and (6.3) are detailed for a spherical geometry and for the trajectory which is used in the advection of full level variables; the subsection (6.4) gives informations about the other cases (for example plane geometry and half level variables).

6.1 Medium point $M$ (subroutines LARMES and LARMES2).

Trajectories are great circles on the geographical sphere. The computation of the medium point $M$ location of the Lagrangian trajectory is performed by an iterative method described by Robert (1981) and adapted to the sphere by M. Rochas. In a 3TL SL scheme, the particle is at the point $M$ at the instant $t (t + \Delta t/2$ for the first integration step). In a 2TL SL scheme, the particle is at the point $M$ at the instant $t + \Delta t/2$. $M$ is at the middle position of the origin point $O$ and the final point $F$. Algorithm is described for deep layer equations; in the thin layer equations, replace simply $r_s$ by $a$ in formulae. For convenience, equations are written with the angular velocity $V/r_s$, but actually this is rather $(a/r_s)V$ which is used in the code. Parts (6.1), (6.1) and (6.1) are valid for non implicit iterative schemes. Part (6.1) is valid for a class of iterative centred-implicit schemes where the momentum equation is treated in an iterative centred-implicit manner (this is the case of the option $\text{LPC~FULL}$ in ALADIN-NH).

* Notations:

- $R^{MF}$ is the rotation operator from medium point to final point (see section [10]).
- $R^{OF}$ is the rotation operator from origin point to final point (see section [11]).
- $r^F = CF$ (C Earth centre, $F$ final point).
- $r^M = CM$ ($M$ medium point).
- $a = r, \lambda$.
- $\phi^{MF}$: angle $(CM, CF)$.
- $\theta^F, \lambda^F$: latitude, longitude on the geographical sphere of $F$.
- $\theta^M, \lambda^M$: latitude, longitude on the geographical sphere of $M$.
- $V^M$: interpolated horizontal wind at $M$ (wind at $t$ in 3TL SL scheme, $t + 0.5\Delta t$ in 2TL SL scheme).
- $V^O$: interpolated horizontal wind at $O$ (wind at $t$ in 3TL SL scheme, $t + 0.5\Delta t$ in 2TL SL scheme).
- $a$ is the average Earth radius near the surface.
- $\Delta t$: time-step.
- $\Delta t$: time-step.
- $\delta t$: time-step.
- In a 3TL SL scheme, $\delta t = 0.5\Delta t$ at the first integration step, $\delta t = \Delta t$ at the following integration steps (leap-frog scheme).
- $L$: number of layers of the model.
- $A, B$ define hydrostatic pressure on the $\eta$ levels ($\Pi = A + B\Pi_s$, where $\Pi_s$ is the hydrostatic surface pressure).
- $\Pi_s$ is a reference hydrostatic pressure equal to the surface pressure of the standard atmosphere (variable $\text{VP00}$).
- $\Pi_{ST}$ is a reference hydrostatic pressure defined at full levels and half levels corresponding to the surface reference hydrostatic pressure $\Pi_s$ ($\Pi_{ST} = A + B\Pi_s$): stored in array $\text{STPRE}$. 
**Definition of the vertical coordinate \( \eta \):** Research of medium point needs an exact definition of the vertical coordinate \( \eta \). For the half level number \( \ell \) (\( \ell \) between 0 and \( L \)), \( \eta_\ell \) is defined by:

$$
\eta_\ell = \frac{A_\ell}{\Pi_\ell} + B_\ell \tag{81}
$$

if \text{LREGETA}=.F. (in namelist \text{NAMCT0}), and:

$$
\eta_\ell = \frac{\ell}{L} \tag{82}
$$

if \text{LREGETA}=.T. .

For the layer number \( l \) (\( l \) between 1 and \( L \)), \( \eta_l \) is defined by:

$$
\eta_l = 0.5(\eta_l + \eta_{l-1}) \tag{83}
$$

A specific definition of \( \eta \) may be required for the VFE operators if \text{LVERTFE}=.T.: it is controled by the key \text{LVFE_REGETA}.

Conventional algorithm (always used for 3TL SL scheme, case \text{LSETTLST=LELTRA=.F. for 2TL SL scheme}).

**Extrapolation of the wind for 2TL SL scheme:** The quantity \([V/r_s]_M\) at \( t + 0.5\Delta t \) used in the 2TL SL scheme is computed by a linear temporal extrapolation using the \( t \) and \( t - \Delta t \) winds at the same location.

**Algorithm:** The medium point is defined by the following iterative scheme: for the iteration \( k + 1 \):

$$
[r/r_s]_{M, k+1} = [r/r_s]^F \cos \phi_k - \frac{R_M([V/r_s]_{M,k})}{|[V/r_s]_{M,k}|} \sin \phi_k \tag{84}
$$

where:

$$
\phi_k = \frac{\delta t}{|[V/r_s]_{M,k}|} \tag{85}
$$

\( \phi \) is a small angle:

$$
\sin \phi \simeq \phi - \frac{\phi^3}{6} \tag{86}
$$

and:

$$
\cos \phi \simeq 1 - \frac{\phi^2}{2} \tag{87}
$$

This approximation allows to simplify some calculations and avoids the occurrence of a division by 0 in the formula defining \( r_{M, k+1} \). Of course:

$$
\sin \phi \simeq \phi \left(1 - \frac{\phi^2}{6}\right) \simeq |[V/r_s]| \cdot \delta t \left(1 - \frac{\phi^2}{6}\right) \tag{88}
$$
4. Semi-lagrangian computations

Thus:

\[
[r/r_s]_M^{k+1} = [r/r_s]_M^F \left( 1 - \frac{\phi_k^2}{2} \right) - R^{MF}([V/r_s]_M^F)\delta t \left( 1 - \frac{\phi_k^2}{2} \right) \]  

(89)

On the vertical, for 3D model:

\[
\eta_1^M = \eta^F - \delta t \dot{\eta}_0^M
\]  

(90)

**First iteration:** Let us start with \( M_0 = F \), \([V/r_s]_0^M = [V/r_s]^F \), \( \phi_0 = \delta t \mid [V/r_s]^F \), \( \dot{\eta}_0^M = \dot{\eta}^F \). Horizontal wind \( V \) has components \( (u, v) \). Thus

\[
sin \theta_1^M = \sin \theta^F \cos \phi_0 - \frac{[v/r_s]^F}{[V/r_s]^F} \cos \theta^F \sin \phi_0
\]  

(91)

\[
\cos \theta_1^M \cos(\lambda_1^M - \lambda^F) = \cos \theta^F \cos \phi_0 + \frac{[v/r_s]^F}{[V/r_s]^F} \sin \theta^F \sin \phi_0
\]  

(92)

\[
\cos \theta_1^M \sin(\lambda_1^M - \lambda^F) = - \frac{[u/r_s]^F}{[V/r_s]^F} \sin \phi_0
\]  

(93)

\[
\eta_1^M = \eta^F - \delta t \dot{\eta}_0^F
\]  

(94)

This defines the coordinates of \( M_1 \). Then \( [u/r_s], [v/r_s], \dot{\eta} \) are interpolated at this point, that gives \([V/r_s]_1^M \) and \( \dot{\eta}_1^M \). Tri-linear interpolations are used in the 3D primitive equation model, horizontal 12 points interpolations are used in the 2D shallow-water model (see section 12).

**Following iterations:** Let us denote by \( V' (u', v') = R^{MF} (V_k^M) \).

\[
sin \theta_{k+1}^M = \sin \theta^F \cos \phi_k - \frac{[v'/r_s]}{[V'/r_s]} \cos \theta^F \sin \phi_k
\]  

(95)

\[
\cos \theta_{k+1}^M \cos(\lambda_{k+1}^M - \lambda^F) = \cos \theta^F \cos \phi_k + \frac{[v'/r_s]}{[V'/r_s]} \sin \theta^F \sin \phi_k
\]  

(96)

\[
\cos \theta_{k+1}^M \sin(\lambda_{k+1}^M - \lambda^F) = - \frac{[u'/r_s]}{[V'/r_s]} \sin \phi_k
\]  

(97)

\[
\eta_{k+1}^M = \eta^F - \delta t \dot{\eta_k}^M
\]  

(98)

This defines coordinates of \( M_{k+1} \). Then, if it is not the last iteration, \( [u/r_s], [v/r_s], \dot{\eta} \) are interpolated at this point, that gives \([V/r_s]_{k+1}^M \) and \( \dot{\eta}_{k+1}^M \). Tri-linear interpolations are used in the 3D primitive equation model, horizontal 12 points interpolations are used in the 2D shallow-water model (see section 12). This iterative algorithm quickly converges: 3 iterations are generally enough.
Stable algorithm for 2TL SL scheme (LSETTLST=.T., LELTRA=.F. in NAMDYN).

* Extrapolation of the wind: The previous algorithm with LSETTLST=.F. can sometimes generate instability (especially when applied to the vertical displacement) so a stable algorithm has been developed by M. Hortal at ECMWF. For more details about theoretical aspects see (Hortal, 1998), (Hortal, 2002). The basic idea is to replace the purely temporal extrapolation by a spatio-temporal extrapolation:

\[ R^{MF}[V/r_s]^M(t + 0.5\Delta t) = 1.5R^{NF}[V/r_s]^N(t) - 0.5R^{OF}[V/r_s]^O(t - \Delta t) \]  

where \( N \) is the position of the particle at time \( t \) for a particle which goes from the origin point \( O \) at time \( t - \Delta t \) to \( M \) at time \( t + 0.5\Delta t \). Assuming that the wind is constant along the trajectory one can write:

\[ ON = 2NM = 0.5NF \]

and evaluate the angular velocity \( R^{NF}[V/r_s]^N(t) \) by \( 2/3R^{MF}[V/r_s]^M(t) + 1/3R^{OF}[V/r_s]^O(t) \) or \( 1/3[V/r_s]^F(t) + 2/3R^{OF}[V/r_s]^O(t) \). Expression of \( [V/r_s]^M(t + 0.5\Delta t) \) becomes:

\[ R^{MF}[V/r_s]^M(t + 0.5\Delta t) = 0.5[V/r_s]^F(t) + 0.5R^{OF}(2[V/r_s]^O(t) - [V/r_s]^O(t - \Delta t)) \]

The same type of extrapolation is done for the \( \eta \)-coordinate vertical velocity. The algorithm of research of trajectory uses directly the RHS of this equation, and for all iterations the origin point \( O \) is computed instead of the medium point \( M \).

* Algorithm: The origin point is defined by the following iterative scheme: for the iteration \( k + 1 \):

\[ [r/r_s]^O_{k+1} = [r/r_s]^F \cos \phi_k - \frac{0.5[V/r_s]^F(t) + 0.5R^{OF}(2[V/r_s]^O(t) - [V/r_s]^O(t - \Delta t))}{0.5[V/r_s]^F(t) + 0.5R^{OF}(2[V/r_s]^O(t) - [V/r_s]^O(t - \Delta t))} \sin \phi_k \]

where:

\[ \phi_k = 2\delta t \mid 0.5[V/r_s]^F(t) + 0.5R^{OF}(2[V/r_s]^O(t) - [V/r_s]^O(t - \Delta t)) \mid \]

Approximations given by equations (90), (97) and (98) are still valid (change \( \delta t \) by \( 2\delta t \) in (98)), thus:

\[ [r/r_s]^O_{k+1} = [r/r_s]^F \left(1 - \frac{\phi_k^2}{2}\right)(0.5[V/r_s]^F(t) + 0.5R^{OF}(2[V/r_s]^O(t) - [V/r_s]^O(t - \Delta t)))(2\delta t)\left(1 - \frac{\phi_k^2}{6}\right) \]

On the vertical, for 3D model:

\[ \eta^O_{k+1} = \eta^F - 2\delta t(0.5\eta^F(t) + 0.5(2\eta^O(t) - \eta^O(t - \Delta t))) \]

* First iteration: One starts with \( M_0 = F \), \([V/r_s]^O(t)\) as a first guess for the spatio-temporally extrapolated horizontal angular velocity, \( \phi_0 = 2\delta t \mid [V/r_s]^F(t) \mid \), \( \eta^F(t) \) as a first guess for the spatio-temporally extrapolated \( \eta \)-coordinate vertical wind. Remark that quantities at \( t \) are taken as a first guess and not quantities at \( (t + 0.5\Delta t) \), contrary to the case LSETTLST=.F. Use equations (91) to (94) replacing \( \delta t \) by \( 2\delta t \) and the superscript \( M \) by \( O \). This defines the coordinates of \( O_1 \); \( (2[V/r_s](t) - [V/r_s](t - \Delta t)) \) and \( (2\eta(t) - \eta(t - \Delta t)) \) are interpolated at this point, that allows to compute the wind components which will be used for the next iteration.
4. Semi-lagrangian computations

* Following iterations: \([V/r_s]\) (of coordinates \([u/r_s, v/r_s]\)) is a generic notation for \((0.5)\left(\frac{V}{r_s}\right)_k^H(t) + 0.5\mathbf{C}^F(2\left(\frac{V}{r_s}\right)_k^O(t) - \left(\frac{V}{r_s}\right)_k^O(t - \Delta t))\). For horizontal displacement use equations (95) to (97) replacing \(\delta t\) by \(2\delta t\) and the superscript \(M\) by \(O\). For vertical displacement use equation

\[
\eta_{k+1}^O = \eta^V - 2\delta t(0.5\eta^F(t) + 0.5(2\eta_k^O(t) - \dot{\eta}_k^O(t - \Delta t)) (106)
\]


* Extrapolation of the horizontal angular velocity: The quantity \([V/r_s]\) at \(t + 0.5\Delta t\) used in the 2TL SL scheme is computed using the RHS of the \([V/r_s]\) equation with explicit formulation of Coriolis term. We denote this RHS by \(RHS_{AV}\) in subsection (6.1). The extrapolated value of \([V/r_s]\) is given by:

\[
[V/r_s]_{ext} = [V/r_s](t) + 0.5\Delta tRHS_{AV} (107)
\]

* Extrapolation of the vertical velocity: The RHS is assumed to be zero, so:

\[
\eta_{ext} = \eta(t) (108)
\]

There is no extrapolation at all.

* Algorithm: The medium point is defined by the following iterative scheme: for the iteration \(k + 1\), a provisional position of the origin point is computed using equations (93) to (95), replacing “\(M^*\)” by “\(O^*\)” and “\(\delta t^*\)” by “\(\Delta t^*\)”.

* First iteration: Let us start with \(O_0 = F\), \([V/r_s]_0^O = [V/r_s]_{ext}, \varphi_0 = \Delta t \left| [V/r_s]_{ext}\right|, \eta_0^O = \eta_{ext}\). Horizontal angular velocity \([V/r_s]_{ext}\) has components \((u/r_s, v/r_s)\). Use equations (91) to (94), replacing “\(M^*\)” by “\(O^*\)” and “\(\delta t^*\)” by “\(\Delta t^*\)” to define coordinates of \(O_1\). Then \(u/r_s, v/r_s, \eta\) are interpolated at this point, that gives \([V/r_s]_1^O\) and \(\eta_1^O\) (subscript “ext” is omitted). Tri-linear interpolations are used in the 3D primitive equation model, horizontal 12 points interpolations are used in the 2D shallow-water model (see section 12).

* Following iterations: Use equations (96) to (98), replacing “\(M^*\)” by “\(O^*\)” and “\(\delta t^*\)” by “\(\Delta t^*\)” to define coordinates of \(O_{k+1}\). Then \(u/r_s, v/r_s, \eta\) are interpolated at this point, that gives \([V/r_s]_{k+1}^O\) and \(\eta_{k+1}^O\) (subscript “ext” is omitted). Tri-linear interpolations are used in the 2D shallow-water model (see section 12). This iterative algorithm quickly converges: 3 iterations are generally enough.

Algorithm used with the iterative centred-implicit schemes.

* Preliminary remarks: Iterative centred-implicit schemes are used to improve stability and it has been shown that this type of scheme has to be used when non-hydrostatic equations are advected by a SL2TL scheme. There are several manners to do iterative centred-implicit schemes; these schemes are not described in detail here; for some of them which are not obsolete, see documentation (IDSI). When the momentum equation is treated by an iterative centred-implicit scheme, the semi-Lagrangian trajectory has to be recomputed at each iteration of the iterative centred scheme and interpolations have to be done again. In this case the algorithm of research trajectory is modified. The case is currently encountered with a SL2TL scheme and LPC\_FULL=.T.. The following algorithm will be described for a SL2TL scheme but it can be extended to a SL3TL scheme (replacing instant \(t\) by instant \(t - \Delta t\).
Extrapolation of the wind if non-extrapolating option (LPC_NESCT=.F.):

No extrapolation is done. The first iteration of the iterative centred-implicit scheme uses $[\mathbf{V}/r_s](t)$ and $\eta(t)$. The following iterations of the iterative centred-implicit scheme use the $[\mathbf{V}/r_s](t+\Delta t)$ and $\eta(t+\Delta t)$ of the previous iteration to start the research of trajectory. As for the "LSETTLST*" option the algorithm computes the origin point $O$.

Extrapolation of the wind if extrapolating option (LPC_NESCT=.T.):

The only difference with the previous case is for the first iteration of the iterative centred-implicit scheme: the wind which is used is now $1.5[\mathbf{V}/r_s](t) - 0.5[\mathbf{V}/r_s](t - \Delta t)$ and $1.5\eta(t) - 0.5\eta(t - \Delta t)$.

Algorithm: One denotes by index "$k$" the numbering of the SL-trajectory research algorithm iteration, and by index "$(i)$" the number of the iterative centred-implicit scheme iteration (in variable NCURRENT_ITER).

The origin point is defined by the following iterative scheme: for the iteration $k+1$ of the SL-trajectory research algorithm:

$$[r/r_s]_{k+1}^O[i] = [r/r_s]^F[i] \cos \phi_k[i] - \frac{0.5R^O [\mathbf{V}/r_s]^O(t) + 0.5 [\mathbf{V}/r_s]^F(t + \Delta t)]_{(i-1)}}{0.5R^O [\mathbf{V}/r_s]^O(t) + 0.5 [\mathbf{V}/r_s]^F(t + \Delta t)]_{(i-1)}}\sin \phi_k[i]$$

where:

$$[\phi_k]_{(i)} = 2\delta t | 0.5R^O [\mathbf{V}/r_s]^O(t) + 0.5 [\mathbf{V}/r_s]^F(t + \Delta t)]_{(i-1)}$$

Approximations given by equations (56), (57) and (58) are still valid (change $\delta t$ by $2\delta t$ in (58)), thus:

$$[r/r_s]_{k+1}^O[i] = [r/r_s]^F[i] \left(1 - \frac{\phi_k^2[i]}{2}\right) - \left(0.5R^O [\mathbf{V}/r_s]^O(t) + 0.5 [\mathbf{V}/r_s]^F(t + \Delta t)]_{(i-1)}\right)(2\delta t) \left(1 - \frac{\phi_k}{2}\right)$$

On the vertical, for 3D model:

$$[\eta_{k+1}]_{(i)} = \eta^F - 2\delta t \left(0.5\eta^O(t) + 0.5 [\eta^F(t + \Delta t)]_{(i-1)}\right)$$

For $i = 0$: $[[\mathbf{V}/r_s](t + \Delta t)]_{(i=0)} = [\mathbf{V}/r_s](t)$ and $[\eta(t + \Delta t)]_{(i=0)} = \eta(t)$ if LPC_NESCT=.T.; $[[\mathbf{V}/r_s](t + \Delta t)]_{(i=0)} = 2[\mathbf{V}/r_s](t) - [\mathbf{V}/r_s](t - \Delta t)$ and $[\eta(t + \Delta t)]_{(i=0)} = 2\eta(t) - \eta(t - \Delta t)$ if LPC_NESCT=.F. .

First iteration of the research of SL trajectory: One starts with $M_0 = F$, $[[\mathbf{V}/r_s]^F(t + \Delta t)]_{(i=1)}$ as a first guess for the spatio-temporally extrapolated horizontal wind, $[\phi_0]_{(i)} = 2\delta t | [[\mathbf{V}/r_s]^F(t + \Delta t)]_{(i=1)}|$, $[\eta^F(t + \Delta t)]_{(i=1)}$ as a first guess for the spatio-temporally extrapolated $\eta$-coordinate vertical wind. Use equations (97) to (97) replacing $\delta t$ by $2\delta t$ and the superscript $M$ by $O$ in the SL-trajectory research. This defines the coordinates of $[O_1]_{(i)}$: $[\mathbf{V}/r_s](t)$ and $\eta(t)$ are interpolated at this point, that allows to compute the wind components which will be used for the next iteration of the research of SL trajectory.
4. Semi-lagrangian computations

* Following iterations of the research of SL trajectory: \[ V' / r_s \] (of co-
ordinates \([ u' / r_s, v' / r_s ] \)) is a generic notation for \( 0.5 R^P [ V / r_s ]^0(t) + 0.5 [ V / r_s ]^P(t + \Delta t) \) \(_{(i-1)}\).
For horizontal displacement use equations (95) to (97) replacing \( \delta t \) by \( 2 \delta t \) and the super-
script \( M \) by \( O \) otherwise. For vertical displacement use equation

\[
\eta^O_{k+1}(i) = \eta^P - 2\delta t \left( 0.5 \eta^O (t) + 0.5 [ \eta^P(t + \Delta t) ]_{(i-1)} \right)
\]  

(113)

* Alternate stable algorithm for 2TL SL scheme (LEL TRA=.T. in 
NAMdyn): For each iteration of the iterative centred-implicit scheme t he 
algorithm is the same as for explicit schemes; at each iteration the RHS o 
of the momentum equation is updated with the "provisional" \( t + \Delta t \) information computed at the previous iteration.

6.2 Origin point \( O \) (subroutines LARMES and LAINOR2).

In a 3TL SL scheme, the particle is at the point \( O \) at the instant \( t - \Delta t \) (\( t \) for the first integration step). In a 2TL SL scheme, the particle is at the point \( O \) at the instant \( t \).

\( O \) is on the same great circle arc (on the geographical sphere) as \( M \) and \( F \) and the 
length of \( OF \) is twice the length of \( MF \). If angle \( \hat{\phi}(r / r_s M, r / r_s F) \) is small (less than \( 10^\circ \), 
what is generally satisfied), one can write for horizontal displacement:

\[
[r / r_s]^O - [r / r_s]^F \simeq 2([r / r_s]^M - [r / r_s]^F)
\]  

(114)

For vertical displacement one can always write:

\[
\eta^O - \eta^F = 2(\eta^M - \eta^F)
\]  

(115)

One denotes by:

- \( \phi = ([r / r_s]^M, [r / r_s]^F) \)
- \([ V' / r_s ] \) (of coordinates \([ u' / r_s, v' / r_s ] \)) the last interpolated horizontal velocity.

Using the following identities:

\[
\cos 2\phi = 2\cos^2 \phi - 1
\]  

(116)

\[
\sin 2\phi = 2\sin \phi \cos \phi
\]  

(117)

the origin point horizontal coordinates can be computed by:

\[
\sin \theta^O = \sin \theta^F \cos 2\phi - 2\cos \phi \left[ \frac{[ u' / r_s ]}{[ V' / r_s ]} \cos \theta^F \sin \phi \right]
\]  

(118)

\[
\cos \theta^O \cos(\lambda^O - \lambda^F) = \cos \theta^F \cos 2\phi + 2\cos \phi \left[ \frac{[ v' / r_s ]}{[ V' / r_s ]} \sin \theta^F \sin \phi \right]
\]  

(119)

\[
\cos \theta^O \sin(\lambda^O - \lambda^F) = -2\cos \phi \left[ \frac{[ u' / r_s ]}{[ V' / r_s ]} \sin \phi \right]
\]  

(120)

Terms in brackets are already computed to determine \( M \).
6.3 Refined recomputation of point $O$.

* **Option L2TLFF for RW2TLFF=1:** Option (switch L2TLFF in YOMDYN) controls recomputation of the origin point using the average between the angular velocity at the origin point and the provisional $t + \Delta t$ angular velocity, according to the algorithm previously described. Only term $2(\Omega \wedge \mathbf{a}k)$ is computed (always analytically) at this improved position of $O$ (so L2TLFF is active only if LADVF=.T. or LADVVF=.T.). Refined recomputation of point $O$ is available only in a limited set of options. In the following sections 7, 8 and 9 discretised equations are written with notation at this improved position of $O$ and its position for $R W2TLFF = 1$. Refined recomputation of point $O$ is given by $R W2TLFF = 1/3$ (Yessad, internal paper in French).

* **Options L2TLFF for RW2TLFF between 0 and 1:** In this case the improved position $O N$ of the origin point $O$ is computed as a linear interpolation between $O$ and its position for $R W2TLFF = 1$ on a great circle bow on geographical sphere. For $R W2TLFF = 0$, $O N = O$. For an idealised straight displacement with a wind of constant acceleration and constant direction, one can show that the "exact" position of the origin point is found above the top of the atmosphere ($O N = .T.$). A correction $(2(\Omega \wedge \mathbf{a}k)(O_{\text{improved}}) - (2\Omega \wedge k)(O))$ is analytically computed and added to wind equation to find the "improved" value of $V^F(t + \Delta t)$. In the deep layer equations and in the cases where a multiplicative factor $r_s / a$ is required, this factor remains interpolated at $O$ and is never re-interpolated at the refined origin point (this is too tricky to code and too expensive also). Computations are currently made in routine LAPINEB and LADINE.

6.4 Remarks.

* **Shallow water model:** Computations remain valid for horizontal coordinates (there is no vertical movement, equations containing $\eta$ have not to be considered). There are remaining some old features (compute $M$ in LARMES2 even for LPC_FULL=.T., LELTRA=.T. or SETTTLST=.T., then $O$ in LAINOR2).

* **Trajectory going out of atmosphere for trajectories ending at a layer final point:** If the origin point $O$ is found above the top of the atmosphere (resp. under the ground) it is put on the top of the atmosphere (resp. on the ground). Then the position of the medium point is recomputed (when necessary), that gives necessary a point between the bottom and the top of the atmosphere. At last the origin point is bounded by a vertical position $\eta_0$ between the top of the atmosphere and the layer $l = 1$. according to the namelist variable VETAOX in NAMDYN (resp. between the layer $l = L$ and the ground, according to the namelist variable VETAOX in NAMDYN). Upper bound of $O$ is $\eta_0 = \eta = L + 1 - \text{VETAOX} \frac{(\eta_{l=1} - \eta_{l=0})}{(\eta_{\text{max}} - \eta_{\text{min}})}$. Lower bound of $O$ is $\eta_0 = \eta = L + 1 - \text{VETAOX} \frac{(\eta_{\text{max}} - \eta_{\text{min}})}{(\eta_{l=1} - \eta_{l=0})}$.

* **Case of interpolations applied to half level variables:** That produces for example in the non-hydrostatic scheme when the half level variable $w$ is advected instead of the full level vertical divergence variable (option LGWADV=.T.). In this case one needs to define an origin $O(\tilde{l})$ for a half level trajectory (which ends at a half level final point $F(\tilde{l})$). The following rules are applied to compute such a kind of trajectory:
  
  - The horizontal displacement from $F(\tilde{l})$ is a weighted average of the horizontal displacements from $F(l = \tilde{l})$ and $F(l)$ (see below for vertical displacement).
  - At the top (resp. bottom) the horizontal displacement from $F(\tilde{l} = 0)$ (resp. $F(\tilde{l} = L)$) is equal to the horizontal displacement from $F(l = 1)$ (resp. $F(l = L)$).
The rule applied to the horizontal displacement is also applied to the vertical displacement for \( l \) between 1 and \( L - 1 \). For example, if \( l \) is between 1 and \( L - 1 \):

\[
\eta_{O(l)} = + \left( 1 - \frac{\eta_{O(l)} - \eta_{O(l+1)}}{\eta_{O(l+1)} - \eta_{O(l)}} \right) \eta_{O(l+1)} + \frac{\eta_{O(l)} - \eta_{O(l)}}{\eta_{O(l+1)} - \eta_{O(l)}} \eta_{O(l)}
\]

A particle coming from the top or the bottom has no vertical displacement (that assumes that \( \dot{\eta} = 0 \) at the top and the bottom, and so that excludes the options \( \text{LRUBC} = \text{T} \), \( \text{NDPSFI} = 1 \) and \( \Pi_{\text{top}} > 0 \) which are not consistent with the constraint \( (\dot{\eta}_{\text{top}} = 0; \dot{\eta}_{\text{surf}} = 0)) \).

If the trajectory goes above the top of the atmosphere, it is bounded at the top of the atmosphere.

If the trajectory goes below the surface, it is bounded at the surface.

Computation of the position of \( O(l) \) is done in routine \text{LARCINHA}.

\* **Plane geometry (ALADIN):** Computation of the SL trajectory is made on the projected plane geometry. \text{ELARMES} and \text{ELARMES2} are called instead of \text{LARMES} and \text{LARMES2}.

\* **Treatment of \( \dot{\eta} \) in the upper stratosphere:** For some options of the code the horizontal interpolations applied on \( \dot{\eta} \) are replaced by "least-square" interpolations; that allows to remove some instabilities. See documentation (IDSVTSM) for more details.
7 The SL discretisation of the 2D shallow-water system of equations (spherical geometry).

7.1 Momentum equation.

* Definition of $X$, $A$ and $B$.

$$X = \vec{V} + \delta \vec{V} (2\vec{f} \wedge \vec{r})$$  \hspace{1cm} (121)

$$A = [-2(1 - \delta \vec{V})(\vec{f} \wedge \vec{V})] - \nabla \Phi$$  \hspace{1cm} (122)

$$B = -\nabla \Phi$$  \hspace{1cm} (123)

* Remarks.

- In the cycle 37T1 of ARPEGE/IFS Coriolis term can be treated explicitly ($\delta \vec{f} = 0$) or implicitly ($\delta \vec{f} = 1$). Use switch LADV in namelist NAMDyn. LADV = .F. corresponds to ($\delta \vec{f} = 0$). LADV = .T. corresponds to ($\delta \vec{f} = 1$). If LADV = .T., term $(2\vec{f} \wedge \vec{r})$ is analytically computed.

- For a limited set of options, term $(2\vec{f} \wedge \vec{r})$ can be recomputed at an improved position of the origin point (RW2TLFF > 0 in NAMDyn).

- Coriolis term can also be put in the semi-implicit scheme by tuning $\beta_{Co}$ (which has a sense only if LADV = .F.). Values $\beta_{Co} = 0$ (LIMPF = .F. in namelist NAMDyn) and $\beta_{Co} = 1$ (LIMPF = .T.) are available in the cycle 37T1. Caution: do not use LIMPF = .T. in variable resolution in cycle 37T1 (formulation of spectral computations is not correct in this case for the semi-implicit scheme).

- If $\beta = 1$, the non-linear term $[\nabla (2\vec{f} \wedge \vec{V})] - \beta (-\nabla \Phi)$ is zero.

7.2 Continuity equation.

Conventional formulation (positive value of NVLAG in the namelist NAMDyn).

* Definition of $X$, $A$ and $B$.

$$X = (\Phi - \Phi_s)$$  \hspace{1cm} (124)

$$A = -(\Phi - \Phi_s)D + \delta_{TR} \vec{V} \nabla (\Phi_s)$$  \hspace{1cm} (125)

$$B = -\Phi^* M^2 D'$$  \hspace{1cm} (126)

$\delta_{TR} = 1$ plays the same role as the "Tanguy-Ritchie" modification for 3D model.
Lagrangian formulation (negative value of NVLAG in the namelist NAMDYN).

* Definition of $X$, $A$ and $B$.

\[ X = (\Phi - \Phi_s)J \]  
\[ A = 0 \]  
\[ B = -\Phi^* M J' \]  

where $J$ is a "Jacobian" quantity defined by its Lagrangian derivative (see equation (12)).

* Calculation of $J$ if 3TL scheme. Jacobian quantities are computed by: $J^- = (1 - \Delta t D)/(1 + \Delta t D)$, $J^* = 1/(1 + \Delta t D^*)$ and $J^+ = 1$.

* Discretisation if 2TL scheme. In cycle 37T1 of ARPEGE/IFS the option NVLAG=-2 is not coded for the 2TLSL scheme.

7.3 Quantities to be interpolated.

When researching the medium point by an iterative algorithm, the interpolation at the medium point (or the origin point LSETTLST=.T. in NAMDYN) of the two components of the horizontal wind is needed: a 12 points interpolation is used. For other quantities to be interpolated, see section 5. For more details about interpolations, see section 12.
8 The SL discretisation of the 3D primitive equation model.

Remark: the detailed discretisation of each part of the RHS is given in the documentation (IDEUL). Some notations used in the expression of linear term $B$ (like $\tau$, $\gamma$, $\mu$, $\nu$) are given in the documentation (IDSI).

8.1 Thin layer formulation of the momentum equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values.

$$X = \vec{V} + \delta \vec{V} (2\vec{\Omega} \wedge \vec{r})$$  \hspace{1cm} (130)

$$A = -2(1 - \delta \vec{V})(\vec{\Omega} \wedge \vec{V}) - \nabla \Phi - RT \nabla (\log \Pi)$$  \hspace{1cm} (131)

$$B = -\nabla \left[ \gamma T + \frac{R_d T^*}{\Pi^*} \Pi_n \right] + \beta_{Co} [-2(1 - \delta \vec{V})(\vec{\Omega} \wedge \vec{V})]$$  \hspace{1cm} (132)

$$F = F_V$$  \hspace{1cm} (133)

Top:

$$V_{\eta=0} = V_{l=1}$$  \hspace{1cm} (134)

Bottom if $\delta m = 0$:

$$V_{\eta=1} = V_{l=L}$$  \hspace{1cm} (135)

Bottom if $\delta m = 1$:

$$V_{\eta=1} = 0$$  \hspace{1cm} (136)

* Remarks.

- In the cycle 37T1 of ARPEGE/IFS Coriolis term can be treated explicitly ($\delta \vec{V} = 0$) or implicitly ($\delta \vec{V} = 1$). Use switch `LADVF` in namelist `NAMMDYN`. `LADVF=F.` corresponds to ($\delta \vec{V} = 0$). `LADVF=T.` corresponds to ($\delta \vec{V} = 1$). If `LADVF=T.`, term $(2\vec{\Omega} \wedge \vec{r})$ is analytically computed.

- For a limited set of options, term $(2\vec{\Omega} \wedge \vec{r})$ can be recomputed at an improved position of the origin point ($RW2TLFF>0$ in `NAMMDYN`).

- Coriolis term can also be put in the semi-implicit scheme by tuning $\beta_{Co}$ (which has a sense only if `LADVF=F.`). Values $\beta_{Co} = 0$ (`LIMPF=F.` in namelist `NAMMDYN`) and $\beta_{Co} = 1$ (`LIMPF=T.`) are available in the cycle 37T1. Caution: do not use `LIMPF=T.` in variable resolution in cycle 37T1 (formulation of spectral computations is not correct in this case for the semi-implicit scheme).
8.2 White and Bromley deep layer formulation of the momentum equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values.

$$X = \vec{V} + \delta_\varphi (2\Omega \wedge \vec{r})$$

$$A = (1 - \delta_\varphi)(-2\Omega \wedge \vec{V} - 2\Omega \wedge W) - \frac{W}{r_s} \vec{V} - \nabla \Phi - (RT + \mu_s R_d T) \nabla (\log \Pi)$$

$$B = -MV' [\gamma T + R_d T^* \log(\Pi_s)] + \beta_{Co} [-2(1 - \delta_\varphi)(\Omega \wedge \vec{V})]$$

$$F = F_V$$

Top:

$$V_{\eta=0} = V_{l=1}$$

Bottom if $\delta m = 0$:

$$V_{\eta=1} = V_{l=L}$$

Bottom if $\delta m = 1$:

$$V_{\eta=1} = 0$$

* Remarks.

- In the cycle 37T1 of ARPEGE/IFS Coriolis term can be treated explicitly ($\delta_\varphi = 0$) or implicitly ($\delta_\varphi = 1$). Use switches LADVF and LADVFW in namelist NAM-DYN. (LADVF:LADVW)=(F:T) corresponds to ($\delta_\varphi = 0$). (LADVF:LADVW)=(T:T) corresponds to ($\delta_\varphi = 1$). If (LADVF:LADVW)=(T:F) term $(2\Omega \wedge ak)$ is treated implicitly and the remaining Coriolis term is treated explicitly. If (LADVF:LADVW)=(F:T) term $(2\Omega \wedge (r_s - a)k)$ is treated implicitly and the remaining Coriolis term is treated explicitly. If at least LADVF=T or LADVFW=T, term $(2\Omega \wedge ak)$ is analytically computed. If LADVFW=T, $r_s/a$ (at $t - \Delta t$ if SL3TL or $t$ if SL2TL) has to be interpolated at $O$, and $r_s/a$ has also to be computed at $t + \Delta t$ using a provisional value of $\log \Pi_s$ at $t + \Delta t$.

- For a limited set of options, term $(2\Omega \wedge ak)$ can be recomputed at an improved position of the origin point (RW2TLFF>0 in NAMdyn).

- The horizontal part of the Coriolis term can also be put in the semi-implicit scheme by tuning $\beta_{Co}$ (which has a sense only if LADVF=F). Values $\beta_{Co} = 0$ (LIMPF=F in namelist NAM-DYN) and $\beta_{Co} = 1$ (LIMPF=T) are available in the cycle 37T1. Caution: do not use LIMPF=T in variable resolution in cycle 37T1 (formulation of spectral computations is not correct in this case for the semi-implicit scheme). The combination (LIMPF:LADVW)=(T:T) is possible but does not eliminate completely a residual explicit term linked to Coriolis force in the RHS of the momentum equation.
8.3 Thermodynamic equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values.

\[
X = T + \delta_{TR} \frac{\alpha_T \Phi_s}{R_d T^{ST}} \tag{144}
\]

\[
A = \frac{RT \omega}{c_p} \frac{\alpha_T}{\Pi} \bar{V} \nabla (\Phi_s) + \delta_{TR} \frac{\Phi_s}{R_d T^{ST}} \left( \frac{d \alpha_T}{d \eta} \right) \tag{145}
\]

\[
B = -\tau (M' D') \tag{146}
\]

\[
F = F_T \tag{147}
\]

Top:

\[
T_{\eta=0} = T_{\eta=1} \tag{148}
\]

Bottom if $\delta m = 0$:

\[
T_{\eta=1} = T_{\eta=L} \tag{149}
\]

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

\[
T_{\eta=1} = T_s \tag{150}
\]

8.4 Thin layer formulation of the continuity equation.

* Definition of $X$, $A$, $B$, and $F$.

\[
X = \log \Pi_s + \delta_{TR} \frac{\Phi_s}{R_d T^{st}} \tag{151}
\]

\[
A = -\frac{1}{\Pi_s} \int_{\eta=0}^{\eta=1} \nabla \left( \bar{V} \frac{\partial \Pi}{\partial \eta} \right) d\eta + \bar{V} \nabla \left[ \log \Pi_s + \delta_{TR} \frac{\Phi_s}{R_d T^{st}} \right] \tag{152}
\]

\[
B = -\frac{M^2}{M^2} \nu D \tag{153}
\]

\[
F = \left( \frac{F_m}{\Pi_s} \right) \tag{154}
\]

* Remarks:

- $A$ is a sum of 3D terms (the advection term) and 2D terms (the other terms).
- $B$ and $F$ are 2D terms (vertical integrals).
8.5 White and Bromley deep layer formulation of the continuity equation.

* Definition of $X$, $A$, $B$, and $F$.

$$X = \log \Pi_s + \delta_T \frac{\Phi_a}{R_d T_{st}}$$ (155)

$$A = -\frac{a^2}{r_s a} \int_{\eta=0}^{\eta=1} \left( \frac{r_s a}{a} \nabla \frac{\partial \Pi_s}{\partial \eta} \right) d\eta + \frac{a}{r_s a} \int_{\eta=0}^{\eta=1} \left( \frac{r_s a}{a} \nabla \frac{\partial \Pi_s}{\partial \eta} \right) - \frac{1}{\Pi_s} \left[ \eta \frac{\partial \Pi_s}{\partial \eta} \right]_{\eta=1} + \frac{1}{\Pi_s} \left[ \frac{r_s a}{a} \nabla \frac{\partial \Pi_s}{\partial \eta} \right]_{\eta=0}$$ (157)

$$B = -\nu \left( \bar{M}^T D' \right)$$

$$F = \left( \frac{F_m}{\Pi_s} \right)$$ (158)

* Remarks:

- $A$ is a sum of 3D terms (the advection term) and 2D terms (the other terms).
- $B$ and $F$ are 2D terms (vertical integrals).

8.6 Moisture equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values.

$$X = q$$ (159)

$$A = 0$$ (160)

$$B = 0$$ (161)

$$F = F_q$$ (162)

Top:

$$q_{\eta=0} = q_{\eta=1}$$ (163)

Bottom if $\delta m = 0$:

$$q_{\eta=1} = q_{\eta=L}$$ (164)

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

$$q_{\eta=1} = q_{pearl}$$ (165)
8.7 Other advectable GFL variables.

Equations are discretised as for humidity equation. 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 $\delta m = 1$; quantities other than $q$ are assumed to be zero at the surface in case $\delta m = 1$.

8.8 Case of lagged physics.

All the previous discretisations have been written with not lagged physics (interpolated at the origin point $O$).

- For lagged physics ($\text{LAGPHY} = \text{T.}$, $\text{LSLPHY} = \text{F.}$): the previous discretisations are done without physics, then the provisional

\[
(X^+ - (1 + \epsilon_x) \frac{\Delta t}{2} B^+ + (1 + \epsilon_x) \frac{\Delta t}{2} B^o)_F
\]

or

\[
(X^+ - (1 + \epsilon) \frac{\Delta t}{2} B^+ + (1 + \epsilon) \frac{\Delta t}{2} B^o)_F
\]

is used as input to the lagged physics.

- For split physics used at ECMWF ($\text{LEPHYS} = \text{T.}$, $\text{LAGPHY} = \text{T.}$, $\text{LSLPHY} = \text{T.}$): one part of the physics is interpolated at the origin point ($t$ or $t - \Delta t$ physics according to $\text{LTWOTL}$), the remainder is evaluated at the final point ($t + \Delta t$ physics). The physical contribution is put in a separate interpolation buffer (name $P(X)P9$) and tri-linearly interpolated. The way to compute the non-lagged contribution is different than the way used at METEO-FRANCE: compute it at the previous timestep as a lagged contribution, then saving it (by the routine $\text{GPSAVTEND}$) from one timestep to the following one (where it is restored by calling the routine $\text{GPGETTEND}$ and added to the interpolation buffer by calling the routine $\text{GPADDSLPHY}$). Partition between the non-lagged and lagged contribution is done by a linear partition of coefficient $\text{RSLWX}$.

8.9 Quantities to be interpolated (computation under subroutine LACDYN).

Research of trajectory.

When researching the medium point by an iterative algorithm, the interpolation at the medium point (or the origin point $\text{LSETTLST} = \text{T.}$. $\text{LELTRA} = \text{T.}$ or $\text{LPC\_FULL} = \text{T.}$) of $([U/r_s], [V/r_s], \dot{\eta})$ is needed: a tri-linear interpolation is performed. For more details about interpolations, see section 12.

RHS of equations.

The list of quantities to be interpolated has been described in subsections 5.2, 5.3 and 5.4 for each type of equation.
Additional quantities to be interpolated at the origin point if RW2TLFF > 0.

The two components of the $[V/r_s]$ at time $t$ (if 2TL SL scheme) or $t - \Delta t$ (if 3TL SL scheme) when not available after the other interpolations (for example if not lagged physics). These additional interpolations are useless if lagged physics (or adiabatic run) and $NWLAG = 3$. 
9 The SL discretisation of the 3D non hydrostatic model.

Remarks: Some notations used in the expression of linear term \( B \) (like \( \tau, \gamma, \mu, \nu \)) are given in the documentation (IDSI).

9.1 Thin layer NH-PDVD model.

Momentum equation.

* Definition of \( X, A, B \) and \( F \), top and bottom values.

\[
X = V + \delta \tau (2\Omega \wedge \vec{r}) \\
A = [-2(1 - \delta \tau)\Omega \wedge \vec{V}] - \frac{\partial p}{\partial \Pi} \nabla \Phi - RT \frac{\nabla(p)}{p} \\
B = -\nabla \left[ \gamma T - T^* (\gamma \dot{P}) + \frac{R_d T^*}{H_s^*} \Pi_s + R_d T^* \dot{P} \right] + \beta c_v [-2(1 - \delta \tau)\Omega \wedge \vec{V}] \\
F = F_V
\]

Top:

\[
V_{\eta=0} = V_{\eta=1} \quad (170)
\]

Bottom if \( \delta m = 0 \):

\[
V_{\eta=1} = V_{\eta=L} \quad (171)
\]

Bottom if \( \delta m = 1 \):

\[
V_{\eta=1} = 0 \quad (172)
\]

Remarks about implicit formulations of the Coriolis term are generally valid for the non-hydrostatic equations.

Temperature equation.

* Definition of \( X, A, B \) and \( F \), top and bottom values.

\[
X = T + \delta T_R \frac{\alpha T \Phi_s}{R_d T_{ST}} \\
A = \frac{RT}{c_v} D_3 + \delta T_R \frac{\alpha T}{R_d T_{ST}} \vec{V} \nabla(\Phi_s) + \delta T_R \frac{\Phi_s}{R_d T_{ST}} \left( \eta \frac{d \alpha T}{d \eta} \right) \\
F = F_V
\]
4. Semi-lagrangian computations

\[ B = - \frac{R_d T^*}{c_v d} [\bar{M}^2 D' + d] \]  
(175)

\[ F = \left[ \frac{c_p}{c_v} f_T \right] \]  
(176)

Top:
\[ T_{\eta=0} = T_{l=1} \]  
(177)

Bottom if \( \delta m = 0 \):
\[ T_{\eta=1} = T_{l=L} \]  
(178)

Bottom if \( \delta m = 1 \) (output of physics):
\[ T_{\eta=1} = T_s \]  
(179)

**Continuity equation.**

Same discretisation as in the hydrostatic case, see part [8.4].

**Moisture equation.**

Same discretisation as in the hydrostatic case, see part [8.6].

**Other advectable GFL variables equations.**

Same discretisation as in the hydrostatic case, see part [8.7].

**Pressure departure variable equation.**

*Definition of \( X, A, B \) and \( F \), top and bottom values, case NPDVAR=2.*

\[ X = \hat{Q} \]  
(180)

\[ A = - \frac{c_p}{c_v} D_3 - \frac{\omega}{\Pi} \]  
(181)

\[ B = - \left[ \frac{c_p d}{c_v d} (\bar{M}^2 D' + d) - \frac{c_p d}{R_d T^* \tau (\bar{M}^2 D')} \right] \]  
(182)

\[ F = \frac{c_p}{c_v T} f_T \]  
(183)

Top:
\[ \hat{Q}_{\eta=0} = 0 \]  
(184)

Bottom if \( \delta m = 0 \):
\[ \hat{Q}_{\eta=1} = \hat{Q}_{l=L} \]  
(185)

Bottom if \( \delta m = 1 \): not yet coded.
Vertical divergence equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values, case NVD-VAR=3, LGWADV=.F. .

\( X = d \) \hspace{1cm} (186)

\[
A = -dD_3 + d\nabla V - \frac{gp}{R_dT_d} \frac{\partial [\frac{\partial P}{\partial \eta}]_{ad}}{\partial \eta} + \frac{gp}{R_dT_d} (\nabla w) \left( \frac{\partial V}{\partial \eta} \right) \] \hspace{1cm} (187)

\[
B = -\frac{g^2}{R_dT_d} (L^*Q) \] \hspace{1cm} (188)

\[
F = -d \left[ \frac{\partial P}{\partial \eta} \right] F'_{m} - \left[ \frac{gp}{R_dT_d} \frac{\partial F_w}{\partial \eta} \right] \] \hspace{1cm} (189)

Top:
\[
d_{\eta=0} = d_{t=1} \] \hspace{1cm} (190)

Bottom if $\delta_m = 0$:
\[
d_{\eta=1} = d_{t=L} \] \hspace{1cm} (191)

Bottom if $\delta_m = 1$: not yet coded.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values, case NVD-VAR=3, LGWADV=.T. .

\( X = w \) \hspace{1cm} (192)

\[
A = g \frac{\partial (P - \Pi)}{\partial \Pi} \] \hspace{1cm} (193)

\[
B = B_w \] \hspace{1cm} (194)

where $B_w$ matches the following relationships:
\[
[B_w]_{surf} = 0 \] \hspace{1cm} (195)

\[
[B_w]_{l} = -gT_a \sum_{k=L}^{k=L} \left( \frac{[\Delta \Pi^*]_{k}}{[\Pi^*]_{k}} \right) \left[ L^*Q \right]_{l} \] \hspace{1cm} (196)

\[
[\Delta B_w]_{l} = gT_a \left[ \frac{[\Delta \Pi^*]_{l}}{[\Pi^*]_{l}} \right] \left[ L^*Q \right]_{l} \] \hspace{1cm} (197)
4. Semi-lagrangian computations

The relationship between \( B_d \) and \( B_w \) writes:

\[
B_d = - \frac{g\Pi^*}{R_d T^*} \frac{\partial B_w}{\partial \eta} \tag{198}
\]

Discretization of equation \( 198 \) at full levels writes:

\[
[B_d]^t = - \frac{g}{R_d T^*} \left[ \frac{\Pi^*}{\Delta \Pi^*} \right]^t [\Delta B_w]^t \tag{199}
\]

\[ F = F_w \tag{200} \]

Top:
- \text{LVERTFE=-F.: } w_{\eta=0} \text{ is computed by the general formula giving } w \text{ at half levels.}
- \text{LVERTFE=-T.: FD treatment for } w_{\eta=0} \text{ (cf. above).}

Bottom if \( \delta m = 0 \):

\[ w_{\eta=1} = V_{\text{surf}} \nabla z_{\text{surf}} \tag{201} \]

Bottom if \( \delta m = 1 \): not yet coded.

Case of lagged physics.

See part \( 8.3 \).

Quantities to be interpolated (computation under subroutine LAC-DYN).

See section \( 5 \).

9.2 Wood and Staniforth deep layer NH-PDVD model.

Momentum equation.

\* Definition of \( X, A, B \) and \( F \), top and bottom values.

\[
A = [-2(1 - \delta \dot{V})(\Omega \times V + 2\Omega \times w_k)] - \frac{w}{r_s} V - \frac{r_s^2}{a^2} \frac{\partial p}{\partial \Pi} \nabla [Gr_s] - RT \frac{\nabla [p]}{p} \tag{202}
\]

Equations \( 166, 169, 170, 171, 172 \) remain valid. Replace \( \Pi \) by \( \tilde{\Pi} \) in equation \( 168 \).

Temperature equation.

See part \( 9.1 \).
4. Semi-lagrangian computations

Continuity equation.

Start from the equations given in part [S7], apply the following changes:

- $\Pi$ is changed into $\tilde{\Pi}$.
- Some metric terms appear: replace $\nabla \left( V \frac{\partial \Pi}{\partial \eta} \right)$ by $\nabla \left( s \frac{\partial \tilde{\Pi}}{\partial \eta} \right)$.
- $G$ appears in factor of the diabatic term.

Moisture equation.

Same discretisation as in the hydrostatic case, see part [S7].

Other advectable GFL variables equations.

Same discretisation as in the hydrostatic case, see part [S7].

Pressure departure variable equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values, case NPDVAR=2.

\[
X = \dot{\bar{Q}} + \delta_P \log(\bar{\Pi}_{\text{ref}}/\Pi_{\text{ref}}) \tag{203}
\]

\[
A = -\frac{c_p}{c_v} D_3 - \frac{\omega}{\Pi} + \delta_P \eta \frac{\partial \log(\bar{\Pi}_{\text{ref}}/\Pi_{\text{ref}})}{\partial \eta} \tag{204}
\]

Top:

\[
[\dot{\bar{Q}} + \delta_P \log(\bar{\Pi}_{\text{ref}}/\Pi_{\text{ref}})]_{\eta=0} = 0 \tag{205}
\]

Bottom if $\delta m = 0$:

\[
[\dot{\bar{Q}} + \delta_P \log(\bar{\Pi}_{\text{ref}}/\Pi_{\text{ref}})]_{\eta=1} = [\dot{\bar{Q}} + \delta_P \log(\bar{\Pi}_{\text{ref}}/\Pi_{\text{ref}})]_{\eta=L} \tag{206}
\]

Bottom if $\delta m = 1$: not yet coded.

Equations [182], [183] remain valid.

Vertical velocity equation.

* Definition of $X$, $A$, $B$ and $F$, top and bottom values, case NVDVAR=3, LGWADV=.T.

\[
X = w \tag{207}
\]

\[
A = -G\mu_s - (g - G) + G \left( \frac{r_s^2}{a^2} - 1 \right) \frac{\partial p}{\partial \Pi} + G \frac{\partial (p - \bar{\Pi})}{\partial \Pi} \tag{208}
\]

Linear terms, physics, top and bottom values: cf. thin layer formulation, simply change $\Pi$ by $\bar{\Pi}$. 
4. Semi-lagrangian computations

* Definition of $X$, $A$, $B$ and $F$, top and bottom values, case NVD-VAR=3, LGWADV=.F.

$$X = d$$ (209)

$$A = -dD_3 + d\nabla V - \frac{Gp}{RdT_0\frac{\partial}{\partial \eta}} \frac{\partial [\frac{2[r^2/a^2 w]}{\eta}]_{ml}}{\partial \eta} + \frac{Gp}{RdT_0\frac{\partial}{\partial \eta}} \left( \nabla \left[ \frac{r^2}{a} w \right] \right) \left( \frac{\partial V}{\partial \eta} \right)$$ (210)

Linear terms, physics, top and bottom values: cf. thin layer formulation, simply change $\Pi$ by $\bar{\Pi}$.

Case of lagged physics.

See part (8.8).

Quantities to be interpolated (computation under subroutine LAC-DYN).

See section (5).

9.3 Thin layer NH-GEOGW model.

Linear term $B$ is generally different, not detailed. See documentation (IDS I) for more details. We just mention the differences with the NH-PDVD model for $A$ and $X$.

Momentum equation, temperature equation, continuity equation, GFL equations.

See part 9.1 for terms $A$ and $X$.

Vertical velocity equation.

If $\delta W = 0$, see part 9.2. This is $gw$ which is actually the prognostic variable.

An alternate possibility is to take $gw - \delta W Bg_{w_{surf}}$ as prognostic variable: in this case additional terms are present in the RHS.

Geopotential equation.

$$X = \Phi$$ (211)

$$A = gw$$ (212)

Top: not used for VFE discretisation.

Bottom:

$$\Phi_{\eta=1} = \Phi_u$$ (213)

An alternate possibility is to take $\Phi - \delta_{GEO} B\Phi_u$ as prognostic variable: in this case additional terms are present in the RHS.
10 \( \mathcal{R} \) operator.

10.1 No tilting.

To transport a vector along a trajectory (part of a great circle) from an origin point \( O \) to a final point \( F \) the following operator \( \mathcal{R}^{OF} \) is defined:

\[
V' = \mathcal{R}^{OF}(V)
\]

where \( V' \) has coordinates \((u', v')\), \( V \) has coordinates \((u, v)\), and the relationship between \((u, v)\) and \((u', v')\) is:

\[
\begin{pmatrix}
  u' \\
  v'
\end{pmatrix} = \begin{pmatrix}
  p & q \\
  -q & p
\end{pmatrix} \begin{pmatrix}
  u \\
  v
\end{pmatrix}
\]

where:

\[
p = \frac{iF^O + jF^O}{1 + kF^O} = \frac{\cos \theta^F \cos \theta^O + \sin \theta^F \sin \theta^O \cos(\lambda^F - \lambda^O)}{1 + \cos \phi}
\]

\[
q = \frac{iF^O + jF^O}{1 + kF^O} = \frac{(\sin \theta^F + \sin \theta^O) \sin(\lambda^F - \lambda^O)}{1 + \cos \phi}
\]

(Notations \( \theta^O, \theta^F, \lambda^O, \lambda^F, \phi \): see section 6).

\( p \) and \( q \) verify the following identity:

\[
p^2 + q^2 = 1
\]

Computation of \( p \) and \( q \) is made in subroutine LARCHE.

10.2 Tilting.

The coordinates of \( V' \) and \( V \) are linked by the following relationship:

\[
\begin{pmatrix}
  u' \\
  v'
\end{pmatrix} = \begin{pmatrix}
  GNORDM & GNORDL \\
  -GNORDL & GNORDM
\end{pmatrix} \begin{pmatrix}
  p & q \\
  -q & p
\end{pmatrix} \begin{pmatrix}
  \cos \alpha & -\sin \alpha \\
  \sin \alpha & \cos \alpha
\end{pmatrix} \begin{pmatrix}
  u \\
  v
\end{pmatrix}
\]

where:

\[
\cos \alpha = \frac{2c}{A \cos \theta^O} \left[ \sin \theta_p \cos \theta^O - \sin \theta^O \cos \theta_p \cos(\lambda^O - \lambda_p) \right]
\]

\[
\sin \alpha = \frac{2c}{A \cos \theta^O} \left[ \cos \theta_p \sin(\lambda^O - \lambda_p) \right]
\]

\[
A = (1 + c^2) + (1 - c^2)(\sin \theta_p \sin \theta^O + \cos \theta_p \cos \theta^O \cos(\lambda^O - \lambda_p))
\]

and where:
• $c$ is the stretching coefficient.
• $\Theta^O$ is the latitude on the computational sphere of the origin point $O$.
• $(\theta_p, \lambda_p)$ are the latitude and longitude on the geographical sphere of the stretching pole.
• $p$ and $q$ are computed like in the not tilted case (in subroutine \texttt{LARCHE}).
• $\cos \alpha$ and $\sin \alpha$ are also computed in subroutine \texttt{LARCHE}).
• $(GNORDL, GNORDM)$ are the coordinates in the computational sphere of the unit vector directed towards the true north, computed in subroutine \texttt{SUGEM2}.

10.3 Plane geometry (ALADIN).

The curvature of the Earth is now taken into account in computing an operator $R^{OF}$ in the routine \texttt{ELARCHE} instead of computing curvature terms. Expressions of $p$ and $q$ are different from the ones of ARPEGE and are not detailed here.
11 Computation of longitudes and latitudes on the computational sphere.

For interpolations it is necessary to compute \((θ^O, Λ^O)\), latitude and longitude of the interpolation point \(O\) in the computational sphere. The iterative algorithm allowing to find \(O\) gives \((θ^O, Λ^O)\), latitude and longitude in the geographical sphere (more exactly \(\sin θ^O, \cos θ^O \cos Λ^O - Λ^F\) and \(\cos θ^O \sin Λ^O - Λ^F\) where \((θ^F, Λ^F)\) are the coordinates of the final point on the geographical sphere). Transform formulae giving \((Θ, Λ)\) on the computational sphere once knowing \((θ, λ)\) on the geographical sphere are given by equations (223) to (225).

\[
\sin Θ = \frac{(1 - c^2) + (1 + c^2)(\sin θ_p \sin θ + \cos θ_p \cos θ \cos(λ - λ_p))}{A} \tag{223}
\]
\[
\cos Θ \cos Λ = \frac{2c(\cos θ_p \sin θ - \sin θ_p \cos θ \cos(λ - λ_p))}{A} \tag{224}
\]
\[
\cos Θ \sin Λ = \frac{2c \cos θ \sin(λ - λ_p)}{A} \tag{225}
\]

where:

- \(A = (1 + c^2) + (1 - c^2)(\sin θ_p \sin θ + \cos θ_p \cos θ \cos(λ - λ_p))\)
- \(c\) is the stretching coefficient.
- \((θ_p, λ_p)\) are the latitude and longitude on the geographical sphere of the stretching pole.
- Computation of \(Θ, Λ\) is made in subroutine LARCHE.

* Plane geometry (ALADIN): The SL trajectory is already computed on the computational grid, so equivalent transformation formulae from geographical space to computational space are useless.
12 Interpolations and weights computations.

12.1 Interpolation grid and weights (subroutine LASCAW).

Horizontal interpolation grid and weights for bi-linear interpolations.

* Definitions: * A 16 points horizontal grid is defined as it is shown in figure 12.1, but only 4 of these 16 points are used in the interpolations. The interpolation point $O$ (medium or origin point) is between $B_1$, $C_1$, $B_2$, and $C_2$. $\Lambda$ and $\Theta$ are the longitudes and latitudes on the computational sphere. The following weights are defined as follows:

- zonal weight number 1:
  $$ZDLO_1 = \frac{\Lambda_O - \Lambda_{B_1}}{\Lambda_{C_1} - \Lambda_{B_1}}$$

- zonal weight number 2:
  $$ZDLO_2 = \frac{\Lambda_O - \Lambda_{B_2}}{\Lambda_{C_2} - \Lambda_{B_2}}$$

- meridian weight:
  $$ZDLAT = \frac{\Theta_O - \Theta_{B_1}}{\Theta_{B_2} - \Theta_{B_1}}$$

* Computations:

- The weights $ZDLO_1$ and $ZDLO_2$ are computed then stored in the array $PDLO$.
- The weight $ZDLAT$ is computed then stored in the array $PDLAT$.
- The memory address (in SL arrays) of the data concerning the points $A_1$ and $A_2$ are computed then stored in the array $KL0$ or $KLH0$. Memory address of the data concerning the points $B_1$, $B_2$, $C_1$, and $C_2$ can be easily computed in interpolations routines knowing these ones of $A_1$ and $A_2$.
- Interpolations use data of points $B_1$, $B_2$, $C_1$, and $C_2$.

Vertical interpolation grid and weights for vertical linear interpolations.

* Definitions: * A 4 points vertical grid is defined as it is shown in figure 12.2, but only 2 of these 4 points are used in the interpolations. The interpolation point $O$ (medium or origin point) is between $T_{l+1}$ and $T_{l+2}$. The vertical weight is defined by:

$$ZDVER = \frac{\eta_O - \eta_{T_{l+1}}}{\eta_{T_{l+2}} - \eta_{T_{l+1}}}$$

* Computations:

- The weight $ZDVER$ is computed then stored in the array $PDVER$.
- The level number $l$ of $T_l$ is stored in the array $KLEV$.
- Interpolations use data of points $T_{l+1}$ and $T_{l+2}$.

* Remark: * The same formulae and computations are valid for half level data, simply replace the layer index $l$ by the half level index $\bar{l}$. 

---

4. Semi-lagrangian computations 107
Horizontal interpolation grid and weights for 12 points cubic interpolations.

* Definitions: A 16 points horizontal grid is defined as it is shown in figure 1, but only 12 of these 16 points are used in the interpolations. The interpolation point $O$ (medium or origin point) is between $B_1$, $C_1$, $B_2$ and $C_2$. The following weights are defined as follows:

- zonal linear weights for latitudes 0, 1, 2, 3:
  
  \[ ZDLO0 = \frac{\Lambda_O - \Lambda_{B_0}}{\Lambda_{C_0} - \Lambda_{B_0}} \]
  \[ ZDLO1 = \frac{\Lambda_O - \Lambda_{B_1}}{\Lambda_{C_1} - \Lambda_{B_1}} \]
  \[ ZDLO2 = \frac{\Lambda_O - \Lambda_{B_2}}{\Lambda_{C_2} - \Lambda_{B_2}} \]
  \[ ZDLO3 = \frac{\Lambda_O - \Lambda_{B_3}}{\Lambda_{C_3} - \Lambda_{B_3}} \]

- zonal cubic weights for latitude 1:
  
  \[ ZCLO11 = f_1(ZDLO1) \]
  \[ ZCLO12 = f_2(ZDLO1) \]
  \[ ZCLO13 = f_3(ZDLO1) \]

where:

- $f_1(\alpha) = (\alpha + 1)(\alpha - 2)(\alpha - 1)/2$
- $f_2(\alpha) = -(\alpha + 1)(\alpha - 2)\alpha/2$
- $f_3(\alpha) = \alpha(\alpha - 1)(\alpha + 1)/6$

- zonal cubic weights for latitude 2:
  
  \[ ZCLO21 = f_1(ZDLO2) \]
  \[ ZCLO22 = f_2(ZDLO2) \]
  \[ ZCLO23 = f_3(ZDLO2) \]

- meridian cubic weights:
  
  \[ ZCLA1 = \frac{(\Theta_O - \Theta_{B_0})(\Theta_O - \Theta_{B_2})(\Theta_O - \Theta_{B_3})}{(\Theta_{B_1} - \Theta_{B_0})(\Theta_{B_1} - \Theta_{B_2})(\Theta_{B_1} - \Theta_{B_3})} \]
  \[ ZCLA2 = \frac{(\Theta_O - \Theta_{B_0})(\Theta_O - \Theta_{B_1})(\Theta_O - \Theta_{B_3})}{(\Theta_{B_2} - \Theta_{B_0})(\Theta_{B_2} - \Theta_{B_1})(\Theta_{B_2} - \Theta_{B_3})} \]
  \[ ZCLA3 = \frac{(\Theta_O - \Theta_{B_0})(\Theta_O - \Theta_{B_1})(\Theta_O - \Theta_{B_2})}{(\Theta_{B_3} - \Theta_{B_0})(\Theta_{B_3} - \Theta_{B_1})(\Theta_{B_3} - \Theta_{B_2})} \]

* Computations:

- The zonal linear weights $ZDLO0$, $ZDLO1$, $ZDLO2$ and $ZDLO3$ are computed then stored in the array PDLO.
- The zonal cubic weights $ZCLO0$, $ZCLO1$, $ZCLO2$, $ZCLO3$, $ZCLO21$, $ZCLO22$, $ZCLO23$ are computed then stored in the array PCLLO.
- The meridian cubic weights $ZCLA1$, $ZCLA2$ and $ZCLA3$ are computed then stored in the array PCLLO. One can notice that the denominators of $ZCLA1$, $ZCLA2$ and $ZCLA3$ do not depend on coordinates of $O$ and can be pre-computed in the subroutine SULEG, in array RIPI.
- The memory address (in SL arrays) of the data concerning the points $A_0$, $A_1$, $A_2$ and $A_3$ are stored in the array KLO or KLH0. Once knowing these addresses one can easily retrieve the other points addresses in the interpolations routines.
- Interpolations use data of the following 12 points $B_0$, $B_1$, $B_2$, $B_3$, $C_0$, $C_1$, $C_2$, $C_3$, $D_1$, $D_2$, $D_3$ and $C_3$. 


* Extension to diffusive interpolations (SLHD): The way to take account of the diffusive properties of interpolations has been redesigned, and now it is completely contained in the calculation of cubic weights: we pass from conventional cubic interpolations to diffusive SLHD cubic interpolations simply by changing the way of computing the cubic weights. The comprehensive way of computing the SLHD cubic weights is not provided in this documentation (because it leads to rather tricky formulae), but details about the calculations can be found in documentation (IDSLIF2) and looking in routine LASCAW (ELASCAW in ALADIN). We just give a sum-up of the way to compute the cubic weights.

- Meridian diffusive cubic weights:
  \[ Z_{CLA1_{slhd}} = Z_{CLA1} + Z_{INCRM1} \]
  \[ Z_{CLA2_{slhd}} = Z_{CLA2} + Z_{INCRM2} \]
  \[ Z_{CLA3_{slhd}} = Z_{CLA3} + Z_{INCRM3} \]
  where each of increments \( Z_{INCRM1}, Z_{INCRM2} \) and \( Z_{INCRM3} \) is a linear combination of cubic Lagrangian weights \( Z_{CLA1}, Z_{CLA2} \) and \( Z_{CLA3} \) and diffusive weights (linear for \( LSLHD\_OLD=.T., \) quadratic otherwise).

  Coefficients of these linear combinations depend on:
  - Constants in time \( C_{sldw} \), describing Laplacian smoother.
  - A pre-computed quantity \( \kappa \) (computed in \( GP\_KAPPA \)) depending on horizontal flow deformation, ranging from 0 to 1; \( \kappa \) is equal to 1 if \( LSLHD\_STATIC=.T. \).
  - Lower and upper bounds (\( \kappa_{min} \) and \( \kappa_{max} \)) respectively stored in variables \( SLHDKMIN \) and \( SLHDKMAX \), used to construct limit interpolators (0 - cubic Lagrange, 1 - linear/quadratic; they are not restricted to range 0-1).

- Zonal diffusive cubic weights for latitude number 1:
  \[ Z_{CLO11_{slhd}} = Z_{CLO11} + Z_{INCRL11} \]
  \[ Z_{CLO12_{slhd}} = Z_{CLO12} + Z_{INCRL12} \]
  \[ Z_{CLO13_{slhd}} = Z_{CLO13} + Z_{INCRL13} \]
  where each of increments \( Z_{INCRL11}, Z_{INCRL12} \) and \( Z_{INCRL13} \) is a linear combination of cubic Lagrangian weights \( Z_{CLO11}, Z_{CLO12} \) and \( Z_{CLO13} \) and diffusive weights (linear for \( LSLHD\_OLD=.T., \) quadratic otherwise).

  Coefficients of these linear combinations depend on:
  - Constants like \( C_{slhdepsh} \).
  - \( \kappa \) (see above).
  - Lower and upper bounds \( \kappa_{min} \) and \( \kappa_{max} \) (see above).

- Zonal diffusive cubic weights for latitude number 2:
  \[ Z_{CLO21_{slhd}} = Z_{CLO21} + Z_{INCRL21} \]
  \[ Z_{CLO22_{slhd}} = Z_{CLO22} + Z_{INCRL22} \]
  \[ Z_{CLO23_{slhd}} = Z_{CLO23} + Z_{INCRL23} \]
  where each of increments \( Z_{INCRL21}, Z_{INCRL22} \) and \( Z_{INCRL23} \) is a linear combination of cubic Lagrangian weights \( Z_{CLO21}, Z_{CLO22} \) and \( Z_{CLO23} \) and diffusive weights (linear for \( LSLHD\_OLD=.T., \) quadratic otherwise).

  Coefficients of these linear combinations depend on \( C_{slhdepsh}, \kappa, \kappa_{min}, \kappa_{max} \) like the zonal weights for latitude number 1.

Several options of SLHD smoothing are available, according to values of the keys \( LSLHDQUAD \) and \( LSLHD\_OLD \).

In part **12** cubic interpolations are written with conventional cubic weights: replace conventional cubic weights by their SLHD counterpart for diffusive cubic interpolations.
Vertical interpolation grid and weights for vertical cubic 4 points interpolations.

A 4 points vertical grid is defined as it is shown in figure 12.2. The interpolation point \( O \) (medium or origin point) is between \( T_{l+1} \) and \( T_{l+2} \). The vertical weights are defined by:

\[
ZCVE_1 = \frac{(\eta_O - \eta_{T_l})(\eta_O - \eta_{T_{l+2}})(\eta_O - \eta_{T_{l+3}})}{(\eta_{T_{l+1}} - \eta_{T_l})(\eta_{T_{l+1}} - \eta_{T_{l+2}})(\eta_{T_{l+1}} - \eta_{T_{l+3}})}
\]

\[
ZCVE_2 = \frac{(\eta_O - \eta_{T_l})(\eta_O - \eta_{T_{l+1}})(\eta_O - \eta_{T_{l+3}})}{(\eta_{T_{l+2}} - \eta_{T_l})(\eta_{T_{l+2}} - \eta_{T_{l+1}})(\eta_{T_{l+2}} - \eta_{T_{l+3}})}
\]

\[
ZCVE_3 = \frac{(\eta_O - \eta_{T_l})(\eta_O - \eta_{T_{l+1}})(\eta_O - \eta_{T_{l+2}})}{(\eta_{T_{l+3}} - \eta_{T_l})(\eta_{T_{l+3}} - \eta_{T_{l+1}})(\eta_{T_{l+3}} - \eta_{T_{l+2}})}
\]

* Computations:

- The vertical weights \( ZCVE_1, ZCVE_2 \) and \( ZCVE_3 \) are computed then stored in the array \( PVINTW \). One can notice that the denominators of \( ZCVE_1, ZCVE_2 \) and \( ZCVE_3 \) do not depend on coordinates of \( O \) and can be pre-computed in the subroutine \( SUVERT \), in the array \( VCUICO \).
- The level number \( l \) of \( T_l \) is stored in the array \( KLEV \).
- Interpolations use data of points \( T_l, T_{l+1}, T_{l+2} \) and \( T_{l+3} \).

* Remark: The same formulae and computations are valid for half level data, simply replace the layer index \( l \) by the half level index \( l' \), and use array \( VCUICOH \).

* Extension to diffusive interpolations (SLHD): Like we do for horizontal weights, the semi-Lagrangian diffusion can be taken into account by modifying vertical cubic weights as follow:

\[
ZCVE_{1\text{slhd}} = ZCVE_1 + ZINCRV_1
\]

\[
ZCVE_{2\text{slhd}} = ZCVE_2 + ZINCRV_2
\]

\[
ZCVE_{3\text{slhd}} = ZCVE_3 + ZINCRV_3
\]

Expressions giving vertical increments \( ZINCRV_1 \) to \( ZINCRV_3 \) have a shape similar to those of horizontal increments, and still depend on quantities like \( \kappa, \kappa_{\text{min}}, \kappa_{\text{max}} \).

In part 12.2 cubic interpolations are written with conventional cubic weights; replace conventional cubic weights by their SLHD counterpart for diffusive cubic interpolations.

Vertical interpolation grid and weights for vertical cubic Hermite interpolations.

This part is valid only for interpolations of full level data.

A 4 points vertical grid is defined as it is shown in figure 12.2. The interpolation point \( O \) (medium or origin point) is between \( T_{l+1} \) and \( T_{l+2} \).

First weights to compute vertical derivatives at layers \( l+1 \) and \( l+2 \) are computed. For a variable \( X \), \( \frac{\partial X}{\partial \eta} \) is computed as close as possible as \( \left( \frac{\eta_{T_{l+3}} - \eta_{T_l}}{\eta_{T_{l+3}} - \eta_{T_{l+1}}} \right) \frac{\partial \Pi}{\partial \eta} \), but with additional approximations allowing to avoid horizontal interpolations for term \( \left( \frac{\eta_{T_{l+3}} - \eta_{T_l}}{\eta_{T_{l+3}} - \eta_{T_{l+1}}} \right) \).
4. Semi-lagrangian computations

• For layers other than the first or the last layer, discretisation follows:

\[
\left( \frac{\partial X}{\partial \eta} \right)_{l+1} = 0.5 \frac{(X_{l+2} - X_l)}{\eta_{l+1} - \eta_l}
\]  

(226)

• For layer \( l = 1 \), discretisation assumes that \( \left( \frac{\partial \eta}{\partial \eta} \right)_{l=0} = 0 \) (valid only if \( \text{LRUBC} = \text{F.} \)); discretisation follows:

\[
\left( \frac{\partial X}{\partial \eta} \right)_{l=1} = \frac{(X_{l=2} - X_{l=1})}{\eta_{l=1} - \eta_{l=0}}
\]  

(227)

• For layer \( l = L \), discretisation assumes that \( \left( \frac{\partial \eta}{\partial \eta} \right)_{l=L} = 0 \); discretisation follows:

\[
\left( \frac{\partial X}{\partial \eta} \right)_{l=L} = \frac{(X_{l=L} - X_{l=L-1})}{\eta_{l=L} - \eta_{l=L-1}}
\]  

(228)

The following weights are computed:

• For an interpolation point included between layers 2 and \( L - 1 \) (\( l \geq 1 \) and \( l \leq L - 3 \)):

\[
VDERW_{11} = 0.5 \frac{(\eta_{l+2} - \eta_{l+1})}{\eta_{l+1} - \eta_l}
\]

\[
VDERW_{21} = 0.5 \frac{(\eta_{l+2} - \eta_{l+1})}{\eta_{l+1} - \eta_{l=0}}
\]

\[
VDERW_{12} = 0.5 \frac{(\eta_{l+2} - \eta_{l+1})}{\eta_{l+2} - \eta_{l+1}}
\]

\[
VDERW_{22} = 0.5 \frac{(\eta_{l+2} - \eta_{l+1})}{\eta_{l+2} - \eta_{l+1}}
\]

• For an interpolation point included between layers 1 and 2:

\[
VDERW_{11} = 0
\]

\[
VDERW_{21} = \frac{(\eta_{l=2} - \eta_{l=1})}{\eta_{l=1} - \eta_{l=0}}
\]

\[
VDERW_{12} = 0.5 \frac{(\eta_{l=2} - \eta_{l=1})}{\eta_{l=2} - \eta_{l=1}}
\]

\[
VDERW_{22} = 0.5 \frac{(\eta_{l=2} - \eta_{l=1})}{\eta_{l=2} - \eta_{l=1}}
\]

such case is extended to the case where the interpolation point is between the top and the first layer; in this case the interpolation becomes an extrapolation.

• For an interpolation point included between layers \( L - 1 \) and \( L \):

\[
VDERW_{11} = \frac{(\eta_{l=L} - \eta_{l=L-1})}{\eta_{l=L-1} - \eta_{l=L-2}}
\]

\[
VDERW_{21} = \frac{(\eta_{l=L} - \eta_{l=L-1})}{\eta_{l=L-1} - \eta_{l=L-2}}
\]

\[
VDERW_{12} = \frac{(\eta_{l=L} - \eta_{l=L-1})}{\eta_{l=L} - \eta_{l=L-1}}
\]

\[
VDERW_{22} = 0
\]

such case is extended to the case where the interpolation point is between the last layer and the ground; in this case the interpolation becomes an extrapolation.
Computations:

- The vertical weights $V_{DERW11}, V_{DERW21}, V_{DERW12}$ and $V_{DERW22}$ are computed then stored in the array $PV_{DERW}$.
- The weight $Z_{DVER}$ is computed then stored in the array $PDVER$ (see subsubsection [12.1]).
- Functions $f_{H1}(Z_{DVER})$ to $f_{H4}(Z_{DVER})$ (involved in any Hermite cubic interpolation), where:
  
  - $f_{H1}(\alpha) = (1 - \alpha)^2(1 + 2\alpha)$
  - $f_{H2}(\alpha) = \alpha^2(3 - 2\alpha)$
  - $f_{H3}(\alpha) = \alpha(1 - \alpha)^2$
  - $f_{H4}(\alpha) = -\alpha^2(1 - \alpha)$

  are computed and stored in array $PHVW$.
- The level number $l$ of $T_l$ is stored in the array $KLEV$.
- Interpolations use data of points $T_i, T_{i+1}, T_{i+2}$ and $T_{i+3}$.

Vertical interpolation grid and weights for vertical cubic spline interpolations.

This part is valid only for interpolations of full level data.

A 4 points vertical grid is defined as it is shown in figure [12.2]. The interpolation point $O$ (medium or origin point) is between $T_{i+1}$ and $T_{i+2}$. The algorithm of calculation of the vertical weights will be given in a later version of the documentation.

Computations:

- Vertical interpolation is the product of two operators. The first one uses all the layers and is done in the unlagged part of the grid-point calculations by a routine VSPLTRANS and needs to compute top and bottom values and vertical derivatives of the field to be interpolated, and also the inversion of a tridiagonal matrix (routine TRIDIA); for this operation it is necessary to use some coefficients stored in the arrays $RVSPTRI$ and $RVSPC$ and pre-computed in the set-up subroutine $SUVSPLIP$; the original field is stored in the buffer $P(X)L9$ and the intermediate result after this first part is stored in the buffer $P(X)SPL9$. The second part uses 4 points and is done in the interpolation routine itself.
- The vertical weights $Z_{CVE0}, Z_{CVE1}, Z_{CVE2}$ and $Z_{CVE3}$ necessary for the second part are computed then stored in the array $PVINTWS$. Some part of the calculations can be pre-computed in the set-up subroutine $SUVSPLIP$ (arrays $RFAA, RFBB, RFCC$ and $RFDD$).
- The level number $l$ of $T_l$ is stored in the array $KLEV$.
- Interpolations use data of points $T_i, T_{i+1}, T_{i+2}$ and $T_{i+3}$, stored in the intermediate array $P(X)SPL9$.

Interpolation grid and weights for tri-linear interpolations.

A 64 points grid is defined as it is shown in figure [12.4], but only 8 of these 64 points are used in the interpolations. The interpolation point $O$ (medium or origin point) is between $B_{1,i+1}, C_{1,i+1}, B_{2,i+1}, C_{2,i+1}, B_{1,i+2}, C_{1,i+2}, B_{2,i+2} and C_{2,i+2}$. For the two levels $l+1$ and $l+2$ see subsubsection [12.1] corresponding to bi-linear horizontal interpolations for weights computations. For weights needed for vertical interpolations ($Z_{DVER}$) see subsubsection [12.1] corresponding to linear vertical interpolations.
4. Semi-lagrangian computations

- The memory address (in SL arrays) of the data concerning the points A0,l, A1,l, A2,l and A3,l is computed then stored in the array KL0. Once knowing these addresses one can easily retrieve the other points addresses in the interpolations routines.
- Interpolations use data of points B1,l+1, C1,l+1, B2,l+1, C2,l+1, B1,l+2, C1,l+2, B2,l+2 and C2,l+2.

* Remark: The same formulae and computations are valid for half level data, simply replace the layer index l by the half level index l/2.

Interpolation grid and weights for 32 points interpolations.

A 64 points grid is defined as it is shown in figure 12.4, but only 32 (48 if vertical spline cubic interpolations) of these 64 points are used in the interpolations. The interpolation point O (medium or origin point) is between B1,l+1, C1,l+1, B2,l+1, C2,l+1, B1,l+2, C1,l+2, B2,l+2 and C2,l+2. For the two levels l and l + 3 see subsubsection 12.1 corresponding to bi-linear horizontal interpolations for weights computations. For the two levels l + 1 and l + 2 see subsubsection 12.1 corresponding to 12 points horizontal interpolations for weights computations. For weights needed for vertical interpolations see subsubsection 12.1 for vertical cubic interpolations, see subsection 12.1 for vertical Hermite cubic interpolations, see subsection 12.1 for vertical spline cubic interpolations.

- The memory address (in SL arrays) of the data concerning the points A0,l, A1,l, A2,l and A3,l is computed then stored in the array KL0. Once knowing these addresses one can easily retrieve the other points addresses in the interpolations routines.
- Interpolations use data of the following 32 points B3,l, C3,l, B4,l, C4,l, B0,l+1, C0,l+1, A1,l+1, B1,l+1, C1,l+1, D1,l+1, A2,l+1, B2,l+1, C2,l+1, D2,l+1, B3,l+1, C3,l+1, B0,l+2, C0,l+2, A1,l+2, B1,l+2, C1,l+2, D1,l+2, A2,l+2, B2,l+2, C2,l+2, D2,l+2, B3,l+2, C3,l+2, B1,l+3, C1,l+3, B2,l+3 and C2,l+3.

* Remarks:

- The same formulae and computations are valid for half level data, simply replace the layer index l by the half level index l/2.
- For vertical spline cubic interpolations a 12 points grid is used on each level l, l + 1, l + 2, l + 3 (total = 48 points used).

Horizontal interpolation grid and weights for 16 points linear least-square fit interpolations.

A 16 points horizontal grid is defined as it is shown in figure 12.3. The interpolation point O (medium or origin point) is between B1, C1, B2 and C2. The interpolation is replaced by a linear least-square fit minimisation of a first order polynomial in each direction (first zonal interpolations, then meridian interpolations).

The weights used are the same ones as for the 4 points bilinear horizontal interpolation, but the zonal linear weights are required for the 4 latitudes of the 16 points grid (ZDL00, ZDL01, ZDL02, ZDL03, ZDLAT). The actual weights used in one direction (for example the meridian direction) are respectively: 0.4 – 0.3ZDLAT, 0.3 – 0.1ZDLAT, 0.2 + 0.1ZDLAT, 0.1 + 0.3ZDLAT.
Computations:

- The weights ZDLO0, ZDLO1, ZDLO2 and ZDLO3 are computed then stored in the array PDLO.
- The meridian weight ZDLAT is computed then stored in the array PDLAT.
- The memory address (in SL arrays) of the data concerning the points A0, A1, A2 and A3 are stored in the array KL0 or KLH0. Once knowing these addresses one can easily retrieve the other points addresses in the interpolations routines.
- Interpolations use data of the following 16 points A0, B0, C0, D0, A1, B1, C1, D1, A2, B2, C2, D2, A3, B3, C3, D3.

Horizontal interpolation grid and weights for 32 points linear least-square fit interpolations.

A 32 points grid is defined as it is shown in figure 12.4 where only the two intermediate layers are retained. The interpolation point O (medium or origin point) is between B1, l+1, C1, l+1, B2, l+1, C2, l+1, B1, l+2, C1, l+2, B2, l+2 and C2, l+2. For the two levels l + 1 and l + 2 see subsubsection 12.1 corresponding to 16 points linear least-square fit horizontal interpolations for weights computations. For weights needed for vertical interpolations, only the linear weight ZDVER is required; see subsubsection 12.1 for vertical linear interpolations.

- Interpolations use data of the following 32 points A0, f+1, B0, f+1, C0, f+1, D0, f+1, A1, f+1, B1, f+1, C1, f+1, D1, f+1, A2, f+1, B2, f+1, C2, f+1, D2, f+1, A3, f+1, B3, f+1, C3, f+1, D3, f+1, A0, f+2, B0, f+2, C0, f+2, D0, f+2, A1, f+2, B1, f+2, C1, f+2, D1, f+2, A2, f+2, B2, f+2, C2, f+2, D2, f+2, A3, f+2, B3, f+2, C3, f+2, D3, f+2.

Remarks:

- The same formulae and computations are valid for half level data, simply replace the layer index l by the half level index l/2.

Plane geometry (ALADIN).

All previous formulae for weight computation can be used for an irregular latitude spacing and a different number of points on each longitude. The ALADIN grid has a horizontal regular spacing, so the previous formulae can be simplified and array RIPI is no longer necessary. ELASCAW is called instead of LASCW and is cheaper in CPU time.

12.2 Interpolations.

Bilinear interpolation (subroutine LAIDL).

See figure 12.1 and subsubsection 12.1 for definition of ZDLO1, ZDLO2, ZDLAT and points B1, C1, B2 and C2.

For a quantity X, are computed successively:

- a linear interpolation on the longitude number 1:
  \[ X_1 = X_{B1} + ZDLO1(X_{C1} - X_{B1}). \]
- a linear interpolation on the longitude number 2:
  \[ X_2 = X_{B2} + ZDLO2(X_{C2} - X_{B2}). \]
- a meridian linear interpolation:
  \[ X_{\text{interpolated}} = X_1 + ZDLAT(X_2 - X_1). \]
Tri-linear interpolation (subroutine LAITLI).

For layers \( l + 1 \) and \( l + 2 \) (see figure 12.1) bilinear horizontal interpolations give two interpolated values \( X_{l+1} \) and \( X_{l+2} \) (see subsubsection 12.2). Then the final interpolated value is given by the following expression:

\[
X_{\text{interpolated}} = X_{l+1} + ZDVER(X_{l+2} - X_{l+1})
\]

* Remark: The same formulae and computations are valid for half level data, simply replace the layer index \( l \) by the half level index \( \tilde{l} \).

Horizontal 12 points interpolation (subroutine LAIDDI).

See figure 12.3 and subsubsection 12.1 for definition of \( ZDLO0 \), \( ZDLO1 \), \( ZDLO2 \), \( ZDLO3 \), \( ZCLA1 \), \( ZCLA2 \) and \( ZCLA3 \) and points \( B_0 \), \( C_0 \), \( A_1 \), \( B_1 \), \( C_1 \), \( D_1 \), \( A_2 \), \( B_2 \), \( C_2 \), \( D_2 \), \( B_3 \) and \( C_3 \).

For a quantity \( X \), are computed successively:

- a linear interpolation on the longitude number 0:
  \[
  X_0 = X_{B_0} + ZDLO0(X_{C_0} - X_{B_0}).
  \]
- a cubic 4 points interpolation on the longitude number 1:
  \[
  X_1 = X_{A_1} + ZCLO11(X_{B_1} - X_{A_1}) + ZCLO12(X_{C_1} - X_{A_1}) + ZCLO13(X_{D_1} - X_{A_1}).
  \]
- a cubic 4 points interpolation on the longitude number 2:
  \[
  X_2 = X_{A_2} + ZCLO21(X_{B_2} - X_{A_2}) + ZCLO22(X_{C_2} - X_{A_2}) + ZCLO23(X_{D_2} - X_{A_2}).
  \]
- a linear interpolation on the longitude number 3:
  \[
  X_3 = X_{B_3} + ZDLO3(X_{C_3} - X_{B_3}).
  \]
- a meridian cubic 4 points interpolation:
  \[
  X_{\text{interpolated}} = X_0 + ZCLA1(X_1 - X_0) + ZCLA2(X_2 - X_0) + ZCLA3(X_3 - X_0).
  \]

There is a shape-preserving option: after cubic 4 points interpolations on longitudes number 1 and 2, \( X_1 \) is bounded between \( X_{B_1} \) and \( X_{C_1} \) and \( X_2 \) is bounded between \( X_{B_2} \) and \( X_{C_2} \); after meridian cubic 4 points interpolation \( X_{\text{interpolated}} \) is bounded between \( X_1 \) and \( X_2 \). Use of switches \( LQMW \) (momentum equation), \( LQMT \) (temperature equation), \( LQMP \) (continuity equation), \( LQMSPD \) (pressure departure variable equation), \( Y[X]_{\text{NL%LQM}} \) for GFL variables allow to use shape-preserving option.

Cubic 4 points vertical interpolation.

See figure 12.2 and subsubsection 12.1 for definition of \( ZCVE1 \), \( ZCVE2 \) and \( ZCVE3 \). The cubic 4 points vertical interpolation gives the final interpolated value:

\[
X_{\text{interpolated}} = X_l + ZCVE1(X_{l+1} - X_l) + ZCVE2(X_{l+2} - X_l) + ZCVE3(X_{l+3} - X_l)
\]

* Remark: The same formulae and computations are valid for half level data, simply replace the layer index \( l \) by the half level index \( \tilde{l} \).

Cubic Hermite vertical interpolation.

This part is valid for interpolations of full level variables. See figure 12.2 and subsubsection 12.1 for definition of \( VDERW11 \), \( VDERW21 \), \( VDERW12 \) and \( VDERW22 \). See subsubsection 12.1 for definition of \( ZDVER \). See subsubsection 12.1 for definition of functions \( f_{H1} \) to \( f_{H4} \). The cubic Hermite vertical interpolation gives the final interpolated value:

\[
X_{\text{interpolated}} = f_{H1}(ZDVER(X_{l+1} + f_{H2}(ZDVER)X_{l+2}) + f_{H3}(ZDVER)(VDERW11(X_{l+1} - X_l) + VDERW21(X_{l+2} - X_{l+1}))) + f_{H4}(ZDVER)(VDERW12(X_{l+2} - X_{l+1}) + VDERW22(X_{l+3} - X_{l+2}))
\]
Spline cubic 4 points vertical interpolation.

This part is valid for interpolations of full level variables. See figure 12.2 and subsection 12.2 for definition of $ZCVE_0$, $ZCVE_1$, $ZCVE_2$ and $ZCVE_3$. The spline cubic 4 points vertical interpolation gives the final interpolated value:

$$X_{\text{interpolated}} = ZCVE_0 \times XRP_l + ZCVE_1 \times XRP_{l+1} + ZCVE_2 \times XRP_{l+2} + ZCVE_3 \times XRP_{l+3}$$

where $XRP$ is the re-profiled version of $X$ available in $P(X)$SPL9. A monotonic constraint can be added, bounding $X_{\text{interpolated}}$ between $X_{l+1}$ and $X_{l+2}$.

32 points 3D interpolation with vertical cubic 4 points interpolation (subroutine LAITRI).

For layers $l$ and $l+3$ (see figure 12.4) bilinear horizontal interpolations give two interpolated values $X_l$ and $X_{l+3}$ (see subsubsection 12.2). For layers $l+1$ and $l+2$ (see figure 12.4) 12 points horizontal interpolations give two interpolated values $X_{l+1}$ and $X_{l+2}$ (see subsubsection 12.2). The final interpolated value $X_{\text{interpolated}}$ is a cubic 4 points vertical interpolation of $X_l$, $X_{l+1}$, $X_{l+2}$ and $X_{l+3}$ (see subsubsection 12.2).

There are shape-preserving options for horizontal or both horizontal and vertical interpolations.

Use of switches LQMW (momentum equation), LQMT (temperature equation), LQMP (continuity equation), LQMSPD (pressure departure variable equation), LQMSPD (vertical divergence equation), Y[X]_NL%LQM (GFL equations), allows to use shape-preserving option for both horizontal and vertical interpolations.

Use of switches LQMHW (momentum equation), LQMHT (temperature equation), LQMHP (continuity equation), LQMHSV (pressure departure variable equation), LQMHSV (vertical divergence equation), Y[X]_NL%LQMH (GFL equations), allows to use shape-preserving option for horizontal interpolations.

* Remark: The same formulae and computations are valid for half level data, simply replace the layer index $l$ by the half level index $\tilde{l}$.

32 points 3D interpolation with vertical cubic Hermite interpolation (subroutine LAIHVT).

This part is valid for interpolations of full level variables. For layers $l$ and $l+3$ (see figure 12.4) bilinear horizontal interpolations give two interpolated values $X_l$ and $X_{l+3}$ (see subsubsection 12.2). For layers $l+1$ and $l+2$ (see figure 12.4) 12 points horizontal interpolations give two interpolated values $X_{l+1}$ and $X_{l+2}$ (see subsubsection 12.2). The final interpolated value $X_{\text{interpolated}}$ is a cubic Hermite vertical interpolation of $X_l$, $X_{l+1}$, $X_{l+2}$ and $X_{l+3}$ (see subsubsection 12.2).

There are shape-preserving options for horizontal or both horizontal and vertical interpolations.

Use of switch Y[X]_NL%LQM (GFL equations), allows to use shape-preserving option for both horizontal and vertical interpolations.

Use of switch Y[X]_NL%LQMH (GFL equations), allows to use shape-preserving option for horizontal interpolations.
48 points 3D interpolation with vertical cubic spline interpolation (subroutine LAITVSPCQM).

This part is valid for interpolations of full level variables. This type of interpolation is activated for GFL variables when \texttt{Y[X]_NL%LVSPLI=*.T.} in \texttt{NAMGFL} (currently for ozone only). Contrary to what is done for the other 32 points interpolations routines, the vertical interpolations are performed first. 12 vertical interpolations are done on the verticals matching the following points: \(B_{0,i+1}, C_{0,i+1}, A_{i+1}, B_{i+1}, C_{i+1}, D_{i+1}, A_{2,i+1}, B_{2,i+1}, C_{2,i+1}, D_{2,i+1}, C_{3,i+1},\) A monotonic constraint is added for the lower levels (currently the 9 lower levels). The projection horizontal on the level of the interpolation point provides a 12-points grid. A 12 points interpolation is done on this projection (see part 12.2). A final monotonic constraint is added: the interpolated value of \(X\) is bounded between the value of \(X\) at the points \(B_{1,i+1}, C_{1,i+1}, B_{2,i+1}, C_{2,i+1}, B_{1,i+2}, C_{1,i+2}, B_{2,i+2}, C_{2,i+2}\), and the overshoots/undershoots found are dispatched on upper levels to ensure as possible it can be done conservation properties.

Horizontal 16 points linear least-square fit interpolation.

See figure 12.3 and subsubsection 12.1 for definition of \(ZDLO0, ZDLO1, ZDLO2, ZDLO3, ZDLAT\) and points \(A_0, B_0, C_0, D_0, A_1, B_1, C_1, D_1, A_2, B_2, C_2, D_2, A_3, B_3, C_3, D_3\). Let us define:

- \(f_1(\alpha) = 0.4 - 0.3\alpha\)
- \(f_2(\alpha) = 0.3 - 0.1\alpha\)
- \(f_3(\alpha) = 0.2 + 0.1\alpha\)
- \(f_4(\alpha) = 0.1 + 0.3\alpha\)

For a quantity \(X\), are computed successively:

- a linear least-square fit 4 points interpolation on the longitude number \(lon\) for \(lon = 0, 1, 2, 3:\)
  \[ X_{lon} = f_1(ZDLOlon)X_{Alon} + f_2(ZDLOlon)X_{Blon} + f_3(ZDLOlon)X_{Clon} + f_4(ZDLOlon)X_{Dlon}. \]
- a meridian linear least-square fit 4 points interpolation:
  \[ X_{\text{interpolated}} = f_1(ZDLAT)X_0 + f_2(ZDLAT)X_1 + f_3(ZDLAT)X_2 + f_4(ZDLAT)X_3. \]

32 points 3D interpolation with linear least-square fit horizontal interpolations and vertical linear interpolations (subroutine LAISMOO).

For layers \(l+1\) and \(l+2\) (see figure 12.4) 16 points linear least-square fit horizontal interpolations give two interpolated values \(X_{l+1}\) and \(X_{l+2}\) (see subsubsection 12.2). The final interpolated value is then given by the following expression:

\[ X_{\text{interpolated}} = X_{l+1} + ZDVER(X_{l+2} - X_{l+1}) \]

* **Remark:** In the current usage of this interpolation (for \(\hat{\eta}\) in the upper stratosphere) there is an additional smoothing done in routine \texttt{LAISMOA} before the interpolation by \texttt{LAISMOO}.

* **For more information:** See the internal paper (IDSVTSM).

12.3 Code structures to store weights.

Two structures have been created in module intdyn_mod.F90:

- structure TLSCAW for linear weights.
- structure TRSCAW for non-linear weights.
Figure 12.1: Interpolation horizontal grid for bilinear interpolations.
$x_1(B_1) = x_2(B_2) = 0; \ x_1(C_1) = x_2(C_2) = 1.$
Figure 12.2: Interpolation vertical grid for linear and cubic vertical interpolations.
\[ T_l \]

level \( l \)

\[ T_{l+1} \]

\( z = 0 \)

level \( l+1 \)

\( O \bullet \quad z = ZDVER \)

\[ T_{l+2} \]

\( z = 1 \)

level \( l+2 \)

\[ T_{l+3} \]

level \( l+3 \)

\( z \)
Figure 12.3: Interpolation horizontal grid for 12 points interpolations.
4. Semi-lagrangian computations

\[ x_0(B_0) = x_1(B_1) = x_2(B_2) = x_3(B_3) = 0; \quad x_0(C_0) = x_1(C_1) = x_2(C_2) = x_3(C_3) = 1. \]
Figure 12.4: Interpolation grid for tri-linear and 32 points interpolations.
- points used in 32 points interpolations.
- points not used in 32 points interpolations.
13 Computation of $\dot{\eta}$ at full levels.

* **General expression:** $\dot{\eta}$ is needed to find the height of the medium and origin points.

$\dot{\eta}$ can be written:

$$\dot{\eta} = \left( \frac{\eta}{\frac{\partial \Pi}{\partial \eta}} \right) \frac{\partial \eta}{\partial \Pi}$$  \hspace{1cm} (229)

* **Discretisation at full levels for LVERTFE=.F.:** $\left( \frac{\dot{\eta}}{\frac{\partial \Pi}{\partial \eta}} \right)$ is provided at half levels, and $\Delta \eta$ and $\Delta \Pi$ are provided at full levels. Discretisation of (229) is:

$$\dot{\eta}_l = 0.5 \left[ \left( \frac{\eta}{\frac{\partial \Pi}{\partial \eta}} \right)_l + \left( \frac{\eta}{\frac{\partial \Pi}{\partial \eta}} \right)_{l-1} \right] \frac{[\Delta \eta]_l}{[\Delta \Pi]_l} \hspace{1cm} (230)$$

* **Discretisation at full levels for LVERTFE=.T.:** $\left( \frac{\dot{\eta}}{\frac{\partial \Pi}{\partial \eta}} \right)$ is provided at full levels, and $\Delta \eta$ and $\Delta \Pi$ are provided at full levels. Discretisation of (229) is:

$$\dot{\eta}_l = \left( \frac{\eta}{\frac{\partial \Pi}{\partial \eta}} \right)_l \frac{[\Delta \eta]_l}{[\Delta \Pi]_l} \hspace{1cm} (231)$$

* **Wood and Staniforth deep-layer NH-PDVD model:** $\Pi$ must be replaced by $\tilde{\Pi}$ in formulae (229), (230) and (231).
14 Lateral boundary conditions.

14.1 Extra longitudes.

Let us denote by $LX$ the number of longitudes (in the array NLOENG for each latitude in the code). For a quantity $X$, let us define:

- $X$(longitude number 0) = $X$(longitude number $LX$).
- $X$(longitude number $LX + 1$) = $X$(longitude number 1).
- $X$(longitude number $LX + 2$) = $X$(longitude number 2).

These extra computations are necessary for all interpolated fields. For distributed memory computations are done when making the halo (routine SLCOMM+SLCOMM2A which exchange data with other processors).

14.2 Extra latitudes.

Let us denote by $lx$ the number of latitudes (NDGLG in the code): latitudes number -1,0,$lx + 1$,lx + 2 are respectively the symmetric of latitudes number 2,1,$lx + 1$,lx − 1. These extra computations are necessary for all interpolated fields. For distributed memory computations are done in SLEXTPOL.

14.3 Vertical boundary conditions in the 3D model.

* Vertical linear interpolations for layer variables at the medium point: The medium point has a vertical coordinate always included between $\eta_{L-1}$ and $\eta_L$ in case of vertical interpolating scheme. Therefore no extrapolated values are needed.

* Vertical cubic 4 points interpolations for layer variables at the origin point: When the origin point is above the layer number 2 (resp. below the layer number $L - 1$), the vertical cubic 4 points interpolations using data of the layers number 1, 2, 3 (resp. $L - 2$, $L - 1$, $L$) and the extra-layer number 0 (resp. $L + 1$) are degenerated into linear interpolations between the layers number 1 and 2 (resp. $L - 1$ and $L$). The extrapolated values at the extra-layer number 0 (resp. $L + 1$) are always multiplied by a weight equal to 0 and are set to 0 in subroutine LAVARO. This algorithm extends itself to the case where the origin point is between the top (resp. surface) and the layer number 1 (resp. $L$), but in this case the interpolation using data of the layers number 1 and 2 (resp. $L - 1$ and $L$) becomes an extrapolation.

* Vertical cubic 4 points interpolations for half level variables at the origin point: When the origin point is above the half level number 1 (resp. $L - 1$), the vertical cubic 4 points interpolations using data of the half levels number -1, 0, 1, 2 (resp. $L - 2$, $L - 1$, $L$ and $L + 1$) are degenerated into linear interpolations between the half levels numbers 0 and 1 (resp. $L - 1$ and $L$).
* Vertical cubic Hermite interpolations for layer variables at the origin point: When the origin point is above the layer number 2 (resp. \( L - 1 \)), interpolation is still a vertical cubic Hermite one, computation of vertical derivatives is modified for layer number 1 (resp. \( L \)). This algorithm extends itself to the case where the origin point is between the top (resp. ground) and the layer number 1 (resp. \( L \)), but in this case the interpolation using data of the layers number 1 and 2 (resp. \( L - 1 \) and \( L \)) becomes an extrapolation. For more details see subsection 12.1.

* Vertical cubic spline interpolations for layer variables at the origin point: Some top and bottom values are computed (the algorithm of computation will be provided in a later version of this documentation) and the vertical interpolation always uses 4 points.
15 2D shallow water and 3D models organigrammes.

15.1 Interpolation routines.

All calculations are DM-local.

* Bilinear and trilinear interpolations:
  - LAIDL: bi-linear 3D interpolations.
  - LAIL: tri-linear 3D interpolations.

* 12 points and 32 points interpolations:
  - LAID: 12 points horizontal interpolation.
  - LAIH: 32 points 3D interpolations, with vertical cubic 4 points interpolations.
  - Remark: LAIL is also used for SLHD diffusive interpolations (with modified pre-computed weights).

* 32 points interpolations with Hermite cubic on the vertical:
  - LAIHVT: 32 points 3D interpolations, with vertical cubic Hermite interpolations.

* 48 points interpolations with vertical splines:
  - LAVSPCQM: Shape-preserving routine for 48 points 3D interpolations, with 12 points horizontal interpolations and vertical cubic spline interpolations.

* Other routines:
  - LAISOO: 32 points interpolation by linear least-square fit.
  - LAISMOA: preliminary smoothing applied on quantities interpolated by the routine LAISOO.
  - LAITRE_GFL: interface routine for interpolations done on GFL variables.
  - LAITRE_GMV: interface routine for interpolations done on GMV variables.
15.2 Routines common to 3D and 2D organigrammes.

All calculations are DM-local.

- **GP_MODEL**: interface for grid-point calculations.
- **LACONE**: computes analytically \((2\Omega \wedge \mathbf{a})\).
- **LASCAW**: computes weights and interpolation grid.
- **LASCAW_CLO**: computes zonal high-order weights (and also meridian high-order weights in ALADIN).
- **LASCAW_CLA**: computes meridian high-order weights (spherical geometry).

15.3 Routines specific to 2D organigramme.

All calculations done in CPG2, CPG2LAG and under these routines are DM-local.

- **LACDYNWSHW**: computation of the SL quantities, and semi-implicit scheme quantities.
- **CPG2LAG**: lagged grid-point computations; manages interpolations and computation of \(t + \Delta t\) quantities.
- **LARMES2**: computation of medium point by an iterative algorithm for 2D model. Calls **LARCIN2**. **ELARMES2** is called instead of **LARMES2** for plane geometry.
- **LAINOR2**: computation of origin point. Calls **LARCIN2**.
- **LARCIN2**: manages computation of medium/origin point coordinates (on the computational sphere), weights for interpolations, and interpolations.
- **LARCH2**: computes latitude, longitude of the medium/origin point coordinates on the computational sphere, \(p\) and \(q\) of the \(\mathbf{R}\) operator matrix. **ELARCH2** is called instead of **LARCH2** for plane geometry.

15.4 Routines specific to 3D organigramme.

All calculations done in CPG, CPGLAG and under these routines are DM-local.

- **CPG_GP, CPG_DIA, CPG_DYN, CPG_END**: different parts of the grid-point calculations (resp. beginning including call to some GP_routines, accumulation of diabatic tendencies in NMI, diagnostics, unlagged dynamics, end).
- **EC_PHYS**: interface for ECMWF physics.
- **EC_PHYS_LSLPHY**: interface for ECMWF split physics.
- **GPGETTEND**: get the physical tendencies for ECMWF physics if split physics.
- **GPADDSLPHY**: adds the physical tendencies to the interpolation buffers for ECMWF physics if split physics.
- **GP_KAPPA**: computes the deformation and \(\kappa\), required in the SLHD interpolations. Called only if **LSMHD** = .T. .
- **LACDYN**: computation of the SL quantities, and semi-implicit scheme quantities.
- **LARCH2**: computes latitude, longitude of the medium/origin point coordinates on the computational sphere, \(p\) and \(q\) of the \(\mathbf{R}\) operator matrix. **ELARCH2** is called instead of **LARCH2** for plane geometry.
- **LASURE**: some initialisations.
- **LASSIE**: computation of semi-implicit linear terms for hydrostatic equations.
- **LANHSI**: computation of semi-implicit linear terms for non-hydrostatic equations.
4. Semi-lagrangian computations

• **LANHSIB**: fills the $P[X]SI$ arrays in the NH model when $LGWADV=\cdot T.\cdot$

• **LAVENT**: computation of quantities to be interpolated (wind) for trajectory research.

• **LATTEX**: computation of quantities to be interpolated for 3D variables equations (GFL and GMV).

• **LATTEX_TNT**: called by **LATTEX** for a SL3TL scheme, part of the code which is the same for all variables.

• **LATTEX_DNT**: called by **LATTEX** for a SL2TL scheme, part of the code which is the same for all variables.

• **LATTES**: computation of quantities to be interpolated for 3D variables equations (currently continuity equation).

• **LA VENT**: computation of quantities to be interpolated (wind) for trajectory research.

• **LARCINHA**: the same as **LARCINA** but for half level data.

• **LARCINB**: manages interpolations needed at the origin point in the equations RHS; apply to interpolations on layer data.

• **LASCAY_VINTW**: computes vertical high-order interpolation weights.

• **GNHG2SVD**: manages the conversions from $gw$ towards $d$ in the non-hydrostatic model when the key $LGWADV$ is put to $\cdot T.\cdot$

• **MF_PHYS_PREP**: preparation of input data for **MF_PHYS**.

• **MF_PHYS**: interface for MF physics.

• **STORE_TRAJ_SLAG**: stores the trajectory for AD, TL in a buffer (in order to be read in TL/AD integrations).

• **VSPLTRANS**: re-profiling of the quantity to be interpolated prior to vertical cubic spline interpolations (calls **TRIDIA** to invert tridiagonal linear systems).

• **VERINT**: does vertical integrals for vertical finite elements discretisation.

15.5 Set-up routines.

• **SUSLB**: initialise pointers of SL buffers and other quantities used in the SL scheme. **SUSLB** is called by **SUSC2B**. DM-global calculations.

• **SUSC2B**: computes different dimensions for SL buffers. DM-global and DM-local calculations.
4. Semi-lagrangian computations

- **SLCSET** + **SLRSET**: computes some distributed memory environment for interpolations buffers. **SLCSET** is called by **SUSC2B**, **SLRSET** calls parallel/SET2PE. DM-global and DM-local calculations.

- **SUSLAD1**: initialise data structures for SL adjoint.
- **SUSLAD2**: initialise constant arrays for SL adjoint interpolation.
- **SUSLAD3**: optimise halo for SL adjoint interpolation.
- **SUVERT**: computes some variables linked to vertical coordinate and also some coefficients for vertical integrals and SL vertical interpolations.
- **SUVSPLIP**: computes some variables linked to cubic spline vertical interpolations.

15.6 Organigramme for 2D model.

control/STEP0 -> control/SCAN2H -> control/SCAN2M -> control/GP_MODEL ->
* adiab/CPG2 ->
  - adiab/GPTF2
  - adiab/LACDYNSHW
  - dia/WRSLTRAJ2
* parallel/SLCOMM and parallel/SLEXTPOL
* adiab/CPG2LAG ->
  - adiab/LADINE ->
    * adiab/LARMS2 -> adiab/LARCl2 ->
      - adiab/LARCHE2 + adiab/LASCAW (adiab/ELASCAW in ALADIN)
        -> adiab/LASCAW_CLA, adiab/LASCAW_CLA
      - adiab/LAIDDI
    * adiab/LAINOR2 -> adiab/LARCl2 ->
      - adiab/LARCHE2 + adiab/LASCAW (adiab/ELASCAW in ALADIN)
        -> adiab/LASCAW_CLA, adiab/LASCAW_CLA
      - adiab/LAIDDI
    * adiab/LACONE
    * adiab/LARCHE2
  - adiab/GPTF1
15.7 Organigramme for 3D model.

* Organigramme under STEPO.

```
control/STEP -> control/SCAN2H “control/SCAN2M “control/GP_MODEL “adiab/CPG “adiab/CPG_GP (see documentation (IDEUL) for details)
- dia/GPINTDDH
- phys_ec/EC_PHYS (unlagged ECMWF physics, organigramme not detailed)
- phys_ec/EC_PHYS_LSLPHY (split ECMWF physics) “adiab/CPGETTEND
* adiab/GPADSLPHY
- phys_dmm/MF_PHYS_PREP
- phys_dmm/MF_PHYS (unlagged MF physics, organigramme not detailed)
- adiab/CPG_DIA “(routines for some diagnostics, organigramme not detailed)
- adiab/CPDYSLHIA “(semi-Lagrangian diagnostics)
- adiab/CPG_DYN “adiab/LACDYN “adiab/LASURE
- adiab/LASSIE ( -> SITNU, SIGAM, GPRCP) or LANHSI ( -> SIPTP, SIDD, SISEVE, GPRCP)
- adiab/LAVENT
- adiab/LATEGX “(adiab/LATEGX_TNT or adiab/LATEGX_DNT)
* utility/VSPPLTRANS “xla/external/linalg/TRIDIA
- adiab/LATEGX_KAPP “adiab/GP_KAPP
- adiab/LATEGX_BBC
- adiab/LATEGX “utility/VERINT
- adiab/LAVABO
- adiab/LANHSI and adiab/LANHSIB for NH and LGWADV=T
* adiab/CP_FORCING (RHS of equations for 1D model with LSFORC=T)
* sinvent/VDFLICZ “(see documentation (IDEUL) for details)
* module/traj_semilag_mod/STORE_TRAJ_SLAG (organigramme not detailed)
- adiab/CPG_END “(see documentation (IDEUL) for details)
* adiab/CALL_SL “adiab/CALL_SL->
- parallel/SLCOMM and parallel/SLEXTPOL (ARPEGE) or parallel/ESLEXTPOL (ALADIN)
- adiab/LAPINEA “adiab/LAPINEA->
* adiab/LARMES (adiab/ELARMES in ALADIN) “adiab/LARMES “adiab/LARINA
- adiab/LARINA (see below) “adiab/LAISMOA
- adiab/LARINA “adiab/LARJA + adiab/LASCW (adiab/ELARJA + adiab/ELASCW in ALADIN)
- adiab/LASCW_CLA, adiab/LASCW_CLD, adiab/LASCW_VINTW
- adiab/LAITLI
- adiab/LAISMO
* adiab/LARINA “adiab/LARJA + adiab/LASCW (adiab/ELARJA + adiab/ELASCW in ALADIN)
- adiab/LASCW_CLA, adiab/LASCW_CLD, adiab/LASCW_VINTW
* adiab/LARINA “adiab/LASCW_CLA, adiab/LASCW_CLD, adiab/LASCW_VINTW
- parallel/SLCOMM2A and parallel/SLEXTPOL (ARPEGE) or parallel/ESLEXTPOL (ALADIN)
if on-demand SL communications asked for
- adiab/LAPINEA “adiab/LAPINEA->
* adiab/LARINB “adiab/LARINB “adiab/LAITRE_GMV “adiab/LAITRI
- adiab/LAITRE_GFL “adiab/LAITRI, adiab/LAITHVT, adiab/LAITVSPCQM
- adiab/LAITLI
- adiab/LAIDDI
* adiab/LARINHB “adiab/LAITRE_GMV “adiab/LAITRI
- adiab/LAITLI
* some adiab/GP... and adiab/GNH... routines.
* adiab/LARCH
```
* adiab/LACONE
* utility/VERINT
* adiab/GNHGW2SVD for NH and LGADV=T
* adiab/CPGLAC (organigramme not detailed, see documentation (IDEUL))
* phys_ec/EC_PHYS_DRV (organigramme not detailed)
16 Tangent linear and adjoint codes.

16.1 3D hydrostatic model.

Available options.

Code is available for the following subset of options in the cycle 37T1 of ARPEGE/IFS:

- Two-time level semi-Lagrangian scheme.
- NVLAG=3, NWLAG=3, NTLAG=3.
- Non monotonic horizontal interpolations.
- LSLPHY=.F.
- All Y[X]_NL%LHV=.F.
- All Y[X]_NL%LVSPAP=.F.
- LSLHD=.F.

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 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
  \[ Z = XY_5 + X_5 Y \]
  using some trajectory variables subscripted by index “5”.
- There is additionally a “trajectory” code which is a copy of the direct code, but applied to the trajectory variables subscripted by index “5”.
- For the interpolation grid indices that appear in the weight computation routine (LASCWTL) and the interpolation routines, there is only one version and the names of variables are not subscripted by index “5”. The “trajectory”-coordinates (subscripted by index “5”). are used to compute the indices of the interpolation grid.
- Concerning the interpolation buffers “PB1...” there are several ways to deal with them, according to the value of variable NTRSLTYPE (in module YOMTRSL, see paragraph about this module). The ADTL-trajectory is stored in the array TRAJSL. For NTRSLTYPE=2 (the default value and the only value coded in the 3D model), some quantities (essentially some quantities of buffer PB1...) are stored in the direct model and read in the TL model, so their recomputation is not necessary in the TL model.
Organigramme of TL code.

* Direct code modification: **CPG_DYN** calls **STORE_TRAJ_SLAG** to store a subset of the trajectory in configurations for which the TL code is called (at least the configurations 131, 401, 501 and 801). This call is activated if **NTRSLTYPE** = 2. **STORE_TRAJ_SLAG** is a subroutine which writes data on a buffer.

**TL code organigramme under SCAN2HTL:**

```
control/SCAN2HTL -> control/SCAN2MTL -> control/GP_MODEL_TL ->
  * adiab/CPGTL ->
    - Allocations
    - adiab/CPSGP (for trajectory) ->
      - some routines reading trajectory (RDPHTRAJM, RDPHTRSF, module/traj_surface_mod/GET_TRAJ_SFC)
      - adiab/GPMFPC, adiab/GPMFPC_GMVS, adiab/SPV (for trajectory)
      - some adiab/CPG.. routines computing intermediate grid-point quantities for trajectories.
    - adiab/CPG_GP_TL ->
      - utility/SC2RDG (organigramme not detailed)
      - adiab/GP2FT
      - adiab/GPMFPC, adiab/GPMFPC_GMVS (for model variables)
      - some adiab/CPG..TL routines computing intermediate grid-point quantities for model variables.
    * adiab/CPG_END_TL ->
      - utility/SC2WRG (organigramme not detailed)
      - adiab/LACDYNTL
      - adiab/LASURE
      - adiab/LASPIETL
      - adiab/LAVNNTL
      - adiab/LATTEXTL -> adiab/LATTEX_DNT
      - adiab/Latte_KAPPATL -> adiab/GP_KAPPATL
      - adiab/LATTESTL -> utility/VERINT
      - adiab/LAVABULTL
      - sinvct/VDIFLCTL (organigramme not detailed)
      - module/traj_semilag_mod/GET_TRAJ_SLAG (if NTRSLTYPE = 2)
    - adiab/CMP_END_TL ->
      - adiab/GPMFPC, adiab/GPMFPC_GMVS (for model variables)
      - utility/SC2WRG (organigramme not detailed)
      - adiab/GPMFPC, adiab/GPMFPC_GMVS (for trajectory)
      - Deallocations
    * adiab/CALL_SL_TL ->
      - parallel/SLCOMMT and parallel/SLEXTPOL (DM code only)
      - adiab/LAPINEAS ->
        - adiab/LARMESS (adiab/ELARMESS in ALADIN) -> adiab/LARCINA ->
          - adiab/LARCH + adiab/LASCW (adiab/ELARCH + adiab/ELASCW in ALADIN) ->
            - adiab/LASCW_CLA, adiab/LASCW_CLO, adiab/LASCW_VINTW
            - adiab/LAITLI
        - adiab/LARCINA ->
          - adiab/LARCH + adiab/LASCW (adiab/ELARCH + adiab/ELASCW in ALADIN) ->
            - adiab/LASCW_CLA, adiab/LASCW_CLO, adiab/LASCW_VINTW
      - adiab/LAPINEATL ->
        - adiab/LARMESTL (adiab/ELARMESTL in ALADIN) -> adiab/LARCIATL ->
          - adiab/LARCHETL (adiab/ELARCHETL in ALADIN)
          - adiab/LASCWTL (adiab/ELASCWTL in ALADIN) ->
            - adiab/LASCW_CLA, adiab/LASCW_CLO, adiab/LASCW_VINTW_TL
            - adiab/LAITLI TL
        - adiab/LARCIATL ->
          - adiab/LARCHETL (adiab/ELARCHETL in ALADIN)
          - adiab/LASCWTL (adiab/ELASCWTL in ALADIN) ->
            - adiab/LASCW_CLA, adiab/LASCW_CLO, adiab/LASCW_VINTW_TL
      - parallel/SLCOMM2A and parallel/SLEXTPOL (DM code only and on-demand SL communications asked)
      - adiab/LAPINEBTL ->
        - adiab/LAITL ->
          - adiab/LAITRE_GMV_TL -> adiab/LAITRITL
```
- adiab/LAILITL
- adiab/LAIDDITL
* utility/VERINT
* adiab/LACONETL
* adiab/LARCHETL (adiab/ELARCHETL in ALADIN)
* utility/VERINT
* phys_ec/EC_PHYS_TL (organigramme not detailed)
* adiab/CPGLAGTL -> (organigramme not detailed, see documentation (IDEUL))
Additional remarks:

- **GET_TRAJ_SLAG** reads the TL trajectory, which is stored in arrays under-scripted by index "5".
- For subroutines, the name of which ends by "5", see their corresponding direct code routines, the explanation of which is given in the section describing the direct code. These routines compute the trajectory which is required in the TL calculations for some quantities, the trajectory of which is not computed in the direct code.
- For subroutines, the name of which ends by "TL", see their corresponding direct code routines, the explanation of which is given in the section describing the direct code.
- **LATTTEXTL** and **LATTESTL**: quantities $P(X)T15$ are not computed, excepted in the case $L2TLFF=.T$, where $PUT15$ and $PVT15$ are computed. $PB1...-buffers are directly recovered by a call to **GET_TRAJ_SLAG** under $CPGTL$ if $NTRSLTYPE=2$.
- **LARMESTL**: computation of the semi-Lagrangian displacement yields a duplication of the code into a "differentiated code" and a "ADTL-trajectory code" (for example an instruction like $Z = XY$ yields the "differentiated code" $Z = XY_5 + X_5Y$ and the "ADTL-trajectory code" $Z_5 = X_5Y_5$). This is the reason for which one can find in routine **LARCHETL** two sets of coordinates $(PLON, PLAT)$ and $(PLON5, PLAT5)$ and two sets of $(p, q)$ matrices. Only the "5" quantities are used to compute the indices of the interpolation grid. On the contrary both sets of coordinates are used to compute the weights in **LASCAWTTL** because both sets of weights are needed in the linear tangent versions of the interpolation routines.
- **LAPINEBTL**: computation of $P(X)T1$ needs both the arrays $Z(X)95$ and $Z(X)9$.
- **LARCHETL**: has optimisations which are not present in the direct code. First the code for not tilted geometry has been splitted into a code for stretched geometry (also valid for unstretched geometry) and a code for unstretched geometry. Additionally some functions such as **MAX**, **MIN**, **MOD**, have been replaced by tests with sometimes additional variables. The consequence is that the TL code looks not straightforward to compare with the direct code.

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

$$X = X + \hat{Y}MY$$

in the adjoint code. Some particular features appear in the code of the semi-Lagrangian scheme, and the adjoint code is not completely the mirror of the linear tangent code.
Organigramme of AD code.

* AD code organigramme under SCAN2MAD:

control/SCAN2HAD -> control/SCAN2MAD -> control/GP_MODEL_AD ->
  adiab/CPG5 -> module/traj_semilag_mod/GET_TRAJ_SLAG
  adiab/CPGLAGAD (arborescence not detailed, see documentation (IDEUL))
  phys_ec/EC_PHYS_AD (arborescence not detailed)
  adiab/PRE_SLADREP (case LSLADREP=T) ->
  - parallel/SLCOMM2A (case LSLONDEM=T)
  - parallel/SLCOMM (case LSLONDEM=F)
  adiab/CALL_SL_AD ->
  - parallel/SLCOMM and parallel/SLEXTPOL (DM code only)
  - adiab/LAPINE5 ->
    - adiab/LAITLI
    - adiab/LAITRI
  - adiab/LACONE
  - adiab/LARCHED
  - utility/VERINTAD
  - adiab/LARCHED
  - adiab/LARCHED
  - adiab/LARCHED
  - some adiab/LAI..._INIT routines (LSLADREP=T)
  - adiab/LAITLIAD
  - adiab/LAITRI
  - adiab/LAITLIAD
  - some adiab/LAI..._INIT routines (LSLADREP=T)
  - adiab/LAITLIAD
  - adiab/LAITRI
  - some adiab/LAI..._INIT routines (LSLADREP=T)
  - adiab/LAITLIAD
  - some adiab/LAI..._INIT routines (LSLADREP=F)
  - adiab/LAITLIAD
  - some adiab/LAI..._INIT routines (LSLADREP=F)
  - adiab/LAITLIAD
  - adiab/LATTE_KAPPAAD -> adiab/GP_KAPPAAD
  - adiab/LATTEXAD -> adiab/LATTEX_DNT_AD
  - adiab/LAVENTAD
  - adiab/LASSIEAD

adib/CPGAD ->

  - Allocations
  - some routines reading trajectory (RDPHTRAJM, RDPHTRSF, module/traj_surface_mod/GET_TRAJ_SFC)
  - some adiab/GP.. routines computing intermediate grid-point quantities for trajectories.
  - adib/CPG_END_AD ->
    - utility/SC2RDG (organigramme not detailed)
    - adiab/GPMPFC, adiab/GPMPFC_GMVS, adiab/GP_SPV (for trajectory)
  - adib/CPG_ZERO_AD
  - adib/CPG_DYN_AD ->
    - some routines reading trajectory (RDPHTRAJM, RDPHTRSF, module/traj_surface_mod/GET_TRAJ_SFC)
    - some adiab/GP.. routines computing intermediate grid-point quantities for trajectories.
4. Semi-lagrangian computations

- phys_dmn/MF_PHYSAD (organigramme not detailed)
- adiab/CPG_GP_AD
  - some adiab/GP..AD routines computing intermediate grid-point quantities for model variables
  - adiab/GPMPFC, adiab/GPMPFC_GMVS (for model variables)
  - adiab/GPTZAD
  - utility/SC2WRG (organigramme not detailed)
- adiab/GPMPFC, adiab/GPMPFC_GMVS (for trajectory)
4. Semi-lagrangian computations

* Additional remarks:

- For subroutines, the name of which ends by "5", see their corresponding direct code routines, the explanation of which is given in the section describing the direct code. These routines compute the trajectory which is required in the AD calculations for some quantities, the trajectory of which is not computed in the direct code.

- For subroutines, the name of which ends by "AD", see their corresponding direct code routines, the explanation of which is given in the section describing the direct code.

- The storage of the ADTL-trajectory variables (indiced by "5") is not fully done in the direct code, because of the huge amount of data to store. So some quantities which have normally to be stored in the direct code, are in this case recomputed in the adjoint code and stored in arrays, the name of which is ending by "5" (sometimes "6" or "7"). This recomputation is done on all data concerning the iterative calculation of the semi-Lagrangian displacement (medium and origin points) and the weights and grid for interpolations. One can notice that, when computing the position of the mid-point, all intermediate positions are stored at each iteration (contrary to the direct code which only retains the last iteration). The consequence is that the adjoint code calls the routines LARMES5 (slightly modified version of LARMES), LARCINBS, LARCHE and LACONE.

- In LASCW, only the computation of interpolation weights has an adjoint code, but not the index-array for the interpolation grid (arrays KL0 and KLH0).

- In interpolation routines (LAIDDIAD, LAIDLIAD, LAITRIAD, LAITLIAD), adjoint codes have some optimisations not present in the direct code and the TL code, in order to avoid memory conflicts (and also specific ALADIN versions). The main feature of these optimisation is the use of an intermediate array PZINC in the adjoint code of the interpolation routines. Memory transfer in PSLBUF1 is done only in LARCINAAD and LARCINBAD once all adjoints of interpolation routines have been called.

- CPG5 is a very simplified version of CPG which currently only reads trajectory.

- LARCHEAD: has optimisations which are not present in the direct code and does not follow completely LARCHETL (no splitting between stretched and unstretched geometry). Some functions such as MAX, MIN, MOD, have been replaced by tests with sometimes additional real or logical variables. The consequence is that the AD code looks not straightforward to compare with the direct and TL codes.

- The adjoint interpolation routines have options of reproducibility which are not present in the direct and TL codes, and which need to call intermediate LAI..._INIT routines.

16.2 3D non hydrostatic model.

Adjoint code is not yet available in the cycle 37T1 of ARPEGE/IFS.

16.3 2D shallow-water model.

Available options.

Code is available for the following subset of options in the cycle 37T1 of ARPEGE/IFS:

- Two-time level semi-Lagrangian scheme.
- Spherical geometry with no tilting.
- Unstretched geometry.
• Conventional formulation for continuity equation.
• Non monotonic horizontal interpolations.
• Eulerian treatment of orography (RCMSLP0=1).
• “On demand” processor communications not activated.

Basics about TL code.

See part 16.1.

Organigramme of TL code.

* Direct code modification:  CPG2 calls WRSLTRAJ2 to store the trajectory in configurations where the TL code is called (configurations 421 and 521). WRSLTRAJ2 is a subroutine which writes data on a buffer.

* TL code organigramme under SCAN2HTL:

control/SCAN2HTL -> control/SCAN2MTL -> control/GP_MODEL_TL ->
* adiab/CPG2TL ->
  - utility/RDSLTRAJ2 (if NTRSLTYPE = 1) -> utility/EMPTB
  - adiab/GPTF2
  - adiab/LACDYNSHWTL ->
  - utility/RDSLTRAJ2 (if NTRSLTYPE = 2) -> utility/EMPTB
* adiab/CPG2LAGTL ->
  - adiab/LADINETL ->
  * adiab/LARMES2TL -> adiab/LARCI2TL -> (see below)
  * adiab/LAINOR2TL -> adiab/LARCI2TL -> (see below)
  * adiab/LARCHE2TL
  * adiab/LACONETL
  - adiab/GPTF1

Organigramme below LARCI2TL:

adiab/LARCI2TL ->
* adiab/LARCHE2TL + adiab/LASCAWT (adiab/ELASCAWT in ALADIN)
  -> adiab/LASCAW_CLA_TL, adiab/LASCAW_CLO_TL
  * adiab/LAIDDITL or adiab/LAIDLITL

* Additional remarks:

• RDSLTRAJ2 reads the TL trajectory, which is stored in arrays underscripted by index “5”.
• For subroutines, the name of which ends by “TL”, see their corresponding direct code routines, the explanation of which is given in the section describing the direct code.
• LACDYNSHWTL: quantities P(X)/T15 are not computed. One only computes PB1...-buffers P(X)/L95 and P(X)/L05 if NTRSLTYPE is zero or 1; otherwise the PB1...-buffers are directly recovered by a call to RDSLTRAJ2 under CPG2TL if NTRSLTYPE=2.
4. Semi-lagrangian computations

- \textbf{LARMES2TL} and \textbf{LAINOR2TL}: computation of the semi-Lagrangian displacement yields a duplication of the code into a “differentiated code” and a “ADTL-trajectory code” (for example an instruction like \( Z = XY \) yields the “differentiated code” \( Z = X_5 Y_5 + X_5 Y \) and the “ADTL-trajectory code” \( Z_5 = X_5 Y_5 \)). This is the reason for which one can find in routine \textbf{LARCHETL} two sets of coordinates \((PLON, PLAT)\) and \((PLON5, PLAT5)\) and two sets of \((p,q)\) matrices. Only the “5” quantities are used to compute the indices of the interpolation grid. On the contrary both sets of coordinates are used to compute the weights in \textbf{LASCAWTL} because both sets of weights are needed in the linear tangent versions of the interpolation routines.

- \textbf{LADINETL}: computation of \( P(X)T1 \) needs both the arrays \( Z(X)95 \) and \( Z(X)9 \).

- \textbf{LARCHETL}: cf. \textbf{LARCHETL}.

\textbf{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

\[ X = X + \text{t}MY \]

in the adjoint code. Some particular features appear in the code of the semi-Lagrangian scheme.

- The storage of the ADTL-trajectory variables (indiced by “5”) is not fully done in the direct code, because of the huge amount of data to store. So some quantities which have normally to be stored in the direct code, are in this case recomputed in the adjoint code and stored in arrays, the name of which is ending by “5” (sometimes “6” or “7”). This recomputation is done on all data concerning the iterative calculation of the semi-Lagrangian displacement (medium and origin points) and the weights and grid for interpolations. One can notice that, when computing the position of the mid-point, all intermediate positions are stored at each iteration (contrary to the direct code which only retains the last iteration). The consequence is that the adjoint code will call the routines \textbf{LARMES25} (slightly modified version of \textbf{LARMES2} which could be merged with \textbf{LARMES2} in the future), \textbf{LAINOR2}, \textbf{LARCIN2}, \textbf{LARCHET}, \textbf{LACONE} and \textbf{LASCAY}. Concerning the not lagged dynamics one finds a sequence \textbf{CPG25} \( \rightarrow \) \textbf{LACDYNSHW} before the call to \textbf{CPG2LAGAD} under \textbf{SCAN2HAD}. \textbf{CPG25} is a simplified version of \textbf{CPG2} which calls \textbf{LACDYNSHW} and gets some \text{t} - \text{t}\Delta\text{t} variables by calling \textbf{RDSLTRAJ2}.

- In \textbf{LASCAY}, only the computation of interpolation weights has an adjoint code, but not the index-array for the interpolation grid (arrays \textbf{KL0} and \textbf{KLH0}).

\textbf{Organigramme of AD code.}

* AD code organigramme under \textbf{SCAN2MAD}:

\begin{verbatim}
control/SCAN2HAD -> control/SCAN2MAD -> control/GP_MODEL_AD ->
* adiab/CPG25 ->
  - utility/RDSLTRAJ2 -> utility/EMPTB
  - adiab/LACDYNSHW
* adiab/CPG2LADAG ->
  - adiab/GPTF1AD
* adiab/LADAD ->
  - adiab/LADINEAD ->
    * adiab/LARMES25 -> adiab/LARCIN2 -> (see below)
\end{verbatim}
4. Semi-lagrangian computations

* adiab/LAINOR2 -> adiab/LARCIN2 -> (see below)
* adiab/LACONE
* adiab/LARCHE2
* adiab/LACONEAD
* adiab/LARCHE2AD
* adiab/LAINOR2AD -> adiab/LARCIN2AD -> (see below)
* adiab/LARMES2AD -> adiab/LARCIN2AD -> (see below)
* adiab/CPG2AD ->
  - adiab/LACDYNSHWAD
  - adiab/GPTF2AD

Organigramme below LARCIN2:

adiab/LARCIN2 ->
  * adiab/LARCHE2 + adiab/LASCAW (adiab/ELASCAW in ALADIN)
  -> adiab/LASCAW_CLA, adiab/LASCAW_CL0
  * adiab/LAIDDI or adiab/LAIDLI

Organigramme below LARCIN2AD:

adiab/LARCIN2AD ->
  * adiab/LARCHE2 + adiab/LASCAW (adiab/ELASCAW in ALADIN)
  -> adiab/LASCAW_CLA, adiab/LASCAW_CL0
  * adiab/LAIDDIAAD or adiab/LAIDLIAAD
  * adiab/LASCAWAD + adiab/LARCHE2AD (adiab/ELASCAWAD in ALADIN)
  -> adiab/LASCAW_CLA_AD, adiab/LASCAW_CL0_AD

* Additional remarks:

  • Some direct code is called again under SCAN2MAD, LADINEAD and LARCIN2AD.
  • LARMES25 is a version of LARMES2, the only differences are actually:
    – only the semi-lagrangian displacement is computed; no call of interpolations
      at the mid-point for the RHS of the equations.
    – all the intermediate positions of the mid-point are stored in the iterative
      algorithm.
  • CPG25 is a very simplified version of CPG2.
  • LADAD: interface for interpolations (equivalent of CALL_SL_AD in the 3D
      model).
  • LARCHE2AD: same remarks as for the hydrostatic 3D model.
17 Some distributed memory features.

17.1 Case LEQ_REGIONS=F.

Calculations packets.

* Grid-point computations: 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 processor treats NGPTOT points (a part of the Gaussian grid points). The total amount of grid-points for all the processors is NGPTOTG. The maximum value of NGPTOT is NGPTOTMX. In the grid-point space there is a subdivision of the current processor grid-points into NGPBLKS=int[(NGPTOT+NPROMA-1)/NPROMA] packets of length NPROMA (the useful number of values in each packet is lower or equal than NPROMA). These packets do not contain neither extra-longitudes nor polar or extra-polar latitudes data. A NPROMA-packet does not always contain a set of complete latitudes. This subdivision into NPROMA-packet both concern not lagged and lagged computations. Currently all the not lagged calculations are made for the int[(NGPTOT+NPROMA-1)/NPROMA] packets of length NPROMA before calling the lagged computations for the int[(NGPTOT+NPROMA-1)/NPROMA] packets of length NPROMA. int[(NGPTOT+NPROMA-1)/NPROMA] is stored in the variable NGPBLKS. More details will be given later for the data transmission for horizontal interpolations. One 2D field has NGPTOTG points divided into NPRGPNS*NPRGPEW sets of NGPTOT points treated by each processor. NGPTOT does not take account of the extra-longitudes and the extra-polar latitudes. All these variables take account of the reduced Gaussian grid. It is assumed and hardcoded that there are one western extra-longitude and two eastern extra-longitudes. The DM-global longitude $jlon=1$ is always the “Greenwich” meridian of the computational sphere. All the vertical levels and the variables corresponding to a same grid-point are treated by the same processor. There are necessary transpositions (reorganisation of data) between grid point computations and Fourier transforms because Fourier transforms need complete latitudes.

* Additional remarks about the LEQ_REGIONS environment variables. Variables N_REGIONS_NS, N_REGIONS and N_REGIONS_EW are used even when LEQ_REGIONS=F but in this case:

- N_REGIONS_NS=NPRGPNS.
- N_REGIONS=NPRGPEW everywhere.
- N_REGIONS_EW=NPRGPEW.

Transmission of data necessary for semi-Lagrangian horizontal interpolations from the non lagged grid-point computations towards the lagged grid-point computations: interpolation buffers.

Description is done for the 3D model (the 2D model uses an obsolescent dataflow structure and does not work any longer).

First one associates to each interpolation point, which is generally not a model grid-point, an “associated” model grid-point which currently satisfies to the following rule: for the semi-Lagrangian scheme one associates to the interpolation point the corresponding
Interpolations use data of points which are not necessary on the same latitude and longitude as the interpolation point and the arrival grid-point of the semi-Lagrangian trajectory. Thus interpolation routines need to have access to a limited number of surrounding latitudes and longitudes which are not necessary treated by the current processor. First the not lagged computations are done for the int\(\lfloor \frac{\text{NGPTOT} + \text{NPROMA} - 1}{\text{NPROMA}} \rfloor\) packets of length \(\text{NPROMA}\), then the lagged computations are done for the int\(\lfloor \frac{\text{NGPTOT} + \text{NPROMA} - 1}{\text{NPROMA}} \rfloor\) packets of length \(\text{NPROMA}\) interpolation points. The number of surrounding latitudes and longitudes rows necessary for interpolations but which do not belong to the current processor is precomputed in the subroutine \texttt{SUSC2B} (variable \texttt{NSLWIDE}). This is a sort of “halo” belonging to some other processors. Due to the “halo” there is still need to split calculations into not lagged ones and lagged ones. Quantities to be interpolated are computed in the non-lagged part and interpolations are performed in the lagged part. In the non-lagged part, only data of the current processor (without any extra-longitudinal data nor polar and extra-polar data) are computed. For all the \(\text{NPROMA}\)-packages treated by the current processor these data are stored in the arrays \texttt{PB1} in \texttt{CPG} (corresponding to local array \texttt{ZSLBUF1} in the routine \texttt{GP\_MODEL}). The first dimension of \texttt{PB1} is \texttt{NASLB1}, which is the total number of points one needs for the interpolations (\texttt{NASLB1} is greater than \texttt{NGPTOT}).

The second dimension of \texttt{PB1} is \texttt{NFLDSL1}: this is the number of 2D quantities to be interpolated. Inside \texttt{CPG}, an intermediate array \texttt{ZSLBUF1AU} is used and data are transferred in \texttt{PB1}: this memory transfer uses a precomputed intermediate array \texttt{NSLCORE}. Then some communication routines are called in \texttt{CALL\_SL} to constitute the halo. \texttt{SLCOMM} does processor communication to fill the halo (receives and sends data from some other processors). When all the \texttt{NASLB1} dataset is constituted, the lagged part \texttt{CALL\_SL} is called which do horizontal interpolations for all the data to be interpolated for the current processor.

* Particular case of the “on demand” processor communication: \texttt{SLCOMM} and \texttt{SLEXTPOL} are still called with specific options, and do the communications only on the part of the buffer which contains the information to find the semi-Lagrangian displacement. For the RHS of equations, the interpolations need to communicate a subset of points, this subset can be known only when all the interpolations grids have been computed (in \texttt{LASCAW}): communications are currently done by \texttt{SLCOMM2A} and \texttt{SLEXTPOL} called by \texttt{CALL\_SL} just before the interpolations. \texttt{SLCOMM2A} and \texttt{SLEXTPOL} have less points to exchange so the model integration is slightly less expensive.

Terms which must be evaluated at \(F\) are stored in \texttt{GFLT1}, \texttt{GMVT1}, \texttt{GMVT1S} (for the RHS of equations) and \texttt{PB2} (\texttt{ZSLBUF2} in \texttt{GP\_MODEL}) for additional intermediate terms. See documentation (\texttt{IDEUL}) for more details about \texttt{PB2} and the dataflow in the grid-point calculations.

17.2 Case \texttt{LEQ\_REGIONS=}T.

This case is relevant only when \texttt{NPRGPEW}>1 (B-level parallelisation at least in the grid-point calculations). This is an optimised version of the \texttt{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 \( N_{\text{REGIONS NS}} \) bands. \( N_{\text{REGIONS NS}} \) can be slightly different from \( \text{NPRGPNS} \).
- for each band \( j \text{roca} \), the B-level grid-point distribution splits the band into \( N_{\text{REGIONS}(j \text{roca})} \) zones: the minimum value of \( N_{\text{REGIONS}} \) is at the poles of the computational sphere (equal to 1 in the examples provided by Mozdzynski); the maximum value of \( N_{\text{REGIONS}} \) is at the equator of the computational sphere and this maximum is equal to \( N_{\text{REGIONS EW}} \). The meridian variations of \( N_{\text{REGIONS}} \) are highly correlated to those of \( \text{NLOENG} \).
- In the examples provided by Mozdzynski, \( \text{NPRGPNS} = \text{NPRGPEW} = \text{NPRTRW} = \text{NPRTRV} \) and we notice that \( N_{\text{REGIONS NS}} \) is slightly below \( \text{NPRGPNS} \), and that \( N_{\text{REGIONS EW}} \) is slightly below \( 2 \times \text{NPRGPEW} \).
18 Specific SL variables in pointer modules, modules and namelists.

PTR... and YOM... are modules in cycle 37T1.

18.1 Modules for declarations.

Pointer PTRSLB1.

Pointers for quantities to be interpolated. All variables of PTRSLB1 are DM-global.

- **RPARSRL1**: parity array for extra-polar latitudes extensions. 1 for scalar quantities, -1 for vectors.
- **MSL... quantities**: pointers computed in SUSLB to retrieve the fields in interpolation buffer. Pointers ending by '9' are for quantities interpolated at the origin point of the SL trajectory, pointers ending by '0' are for quantities interpolated at the medium (or sometimes origin) point of the SL trajectory, pointers ending by 'L' are for the wind components interpolated to find the trajectory. For more details, see later.
- **MSLBUF1**: pointer used for the whole interpolation buffer.
- **Pointers for quantities interpolated at the origin point**: MSLB1U9 (U-wind equation), MSLB1V9 (V-wind equation), MSLB1T9 (temperature equation), MSLB1GFL9 (GFL variables equations), MSLB1GFLSP9 (GFL variables equations, if spline cubic vertical interpolations only), MSLB1PD9 (pressure departure variable equation), MSLB1VD9 (vertical divergence equation), MSLB1NHX9 (X-term), MSLB1Y9 (semi-implicit term used in the non-hydrostatic system), MSLB1C9 (3D term of continuity equation), MSLB1SP9 (2D term of continuity equation), MSLB1DPH9 and MSLB1GWS9 are pointers for additional quantities used when LRDBBC=.T. in the NH model (for more details, see paper (IDVNH3)).
- **Pointers for second quantity interpolated linearly at the origin point if required**: MSLB1U0 (U-wind equation), MSLB1V0 (V-wind equation), MSLB1T0 (temperature equation), MSLB1PD0 (pressure departure variable equation), MSLB1VD0 (vertical divergence equation), MSLB1C0 (3D term of continuity equation), MSLB1SP0 (2D term of continuity equation).
- **Pointers for separate linear terms interpolated at the origin point if required**: MSLB1U9_SI (U-wind equation), MSLB1V9_SI (V-wind equation), MSLB1T9_SI (temperature equation), MSLB1PD9_SI (pressure departure variable equation), MSLB1VD9_SI (vertical divergence equation), MSLB1C9_SI (3D term of continuity equation).
- **Pointers for (aU/r; aV/r; \dot{\eta}) used for the SL trajectory research**: MSLB1URL, MSLB1VRL, MSLB1UR0, MSLB1VRO, MSLB1WR0, MSLB1WRA. Pointers ending by '0' are equal to pointers ending by 'L + 1' (for ex MSLB1UR0 = MSLB1URL + 1). MSLB1WRA is used only for the option LSVTSM=.T...
- **Pointers for (aU/r; aV/r) used for the recalculation of the origin point (RW2TLFF>0) when useful**: MSLB1UR9 and MSLB1VR9.
- **Pointers for additional quantities used in the ECMWF split physics**: MSLB1UP9 (U-wind equation), MSLB1VP9 (V-wind equation), MSLB1TP9 (temperature equation), MSLB1GFLP9 (GFL variables equations).
- **Pointer MSLB1RSSA9**: r_x/a to be interpolated at O.
- **Pointers (applicable only when NSPLTHO1 different from 0) for second part of the quantity to be interpolated at O by high order interpolation**: MSLB1UF9 (U-wind equation), MSLB1VF9 (V-wind equation), MSLB1TF9 (temperature equation), MSLB1VD9 (vertical divergence equation), MSLB1GFLSPF9, MSLB1GFLF9 (GFL variables equations).
4. Semi-lagrangian computations

Pointer PTRSLB15.

Pointers for quantities to be interpolated: trajectory used in the tangent linear and adjoint models. All variables of PTRSLB15 are DM-global.

- **RP ARSL15**: parity array for extra-polar latitudes extensions. 1 for scalar quantities, -1 for vectors.
- **MSL... quantities**: pointers computed in SUSLB to retrieve the fields in interpolation buffer. Pointers ending by '95' are for quantities interpolated at the origin point of the SL trajectory, pointers ending by '05' are for quantities interpolated at the medium (or sometimes origin) point of the SL trajectory. For more details, see later.
- **MSLBUF15**: pointer used for the whole interpolation buffer.
- **Pointers for quantities interpolated at the origin point**: MSLB1U95 (U-wind equation), MSLB1V95 (V-wind equation), MSLB1T95 (temperature equation), MSLB1GFL95 (GFL variables equations), MSLB1C95 (3D term of continuity equation), MSLB1SP95 (2D term of continuity equation).
- **Pointers for quantities interpolated at the medium point (or second quantity interpolated linearly at the origin point if required)**: MSLB1U05 (U-wind equation), MSLB1V05 (V-wind equation), MSLB1T05 (temperature equation), MSLB1C05 (3D term of continuity equation), MSLB1SP05 (2D term of continuity equation).
- **Pointers for \((aU/r_s; aV/r_s; \dot{\eta})\) used for the SL trajectory research**: MSLB1UR05, MSLB1VR05, MSLB1WR05.
- **Pointers for \((aU/r_s; aV/r_r)\) used for the recalculation of the origin point (RW2TLFF>0) when useful**: MSLB1UR95 and MSLB1VR95.

Pointer PTRSLB2.

Used to communicate information from not lagged to lagged computations (also used for Eulerian scheme) for quantities which cannot be stored in GMVT1, GMVT1S and GFT1. All variables of PTRSLB2 are DM-global. MSLBUF2 is the pointer used for the whole buffer. (X) is a generic notation for each variable, see the equivalent MSLB1(X)9 where (X) = U, V, etc.

- **Name is MSLB2(X)1** for buffers containing \(t + dt\) quantities at the model gridpoints.
- **MSLB2(X)15** are used in the TL and AD codes (storage of trajectory).
- **MSLB2(X)SI** are used to save explicit components to the semi-implicit scheme which have to be added in the lagged computations (after the \(t + dt\) physics for example).
- **MSLB2(X)SI5** are used to save explicit components to the semi-implicit scheme which have to be added in the lagged computations in the TL and AD codes (storage of trajectory).
- **MSLB2(X)RL** are used for \((aU/r_s; aV/r_s; \dot{\eta})\) used for the SL trajectory research.
- **MSLB2(X)RL5** are used for \((aU/r_s; aV/r_s; \dot{\eta})\) used for the SL trajectory research in the TL and AD codes (storage of trajectory).
- **MSLB2VCVWEI**: weight \(W_{vei}\) used in the vertical integrals (SL form of the continuity equation).
- **MSLB2KAPPAM** is used for \(\kappa_*\) required in the SLHD interpolations.
- **MSLB2KAPPAM5**: cf. MSLB2KAPPAM but for trajectory.
- **MSLB2KAPPAM** is used for horizontal momentum exchange coef. used in 3D turbulence.
- **MSLB2KAPPAM** is used for horizontal heat exchange coef. used in 3D turbulence.
- **MSLB2GWF, MSLB2GDW, MSLB2GWS**: to store \(gw\) and \(-g\Delta w\).
YOMCT0.

Contains control variables. The following variables also present in namelist NAMCT0 are used in the semi-Lagrangian scheme. All the following variables are DM-global.

- **LSLAG**: T. = SL scheme; F. = Eulerian scheme.
- **LREGETA**:
  - F. (default value): \( \eta_\ell = A / \Pi_s + B \).
  - T.: \( \eta_\ell = 1/L \).

  for the definition of \( \eta \) used in the semi-Lagrangian trajectory and interpolations.
- **LVFE_REGETA**: cf. LREGETA but for the definition of \( \eta \) used in the VFE operators.
- **LTWOTL**: T. = 2TL SL scheme; F. = 3TL SL scheme.
- **LSPRT**:
  - F. (default value at METEO-FRANCE): use of temperature in spectral space.
  - T. (default value at ECMWF): use of virtual temperature in spectral space.
- **LRUBC**: if T., possibility of having a non-zero value of \( \left( \frac{\partial \eta}{\partial \Pi} \right)_{\eta=0} \). Default value is false.
- **LPC_FULL**: if T., full non-incremental iterative centred-implicit scheme.
- **LPC_CHEAP**: if T., cheap version of LPC\_FULL = T (no iteration on the SL trajectory).
- **LPC_NESC**: if T., non-extrapolating SL2TL SI scheme for equations RHS; \( X(t+dt/2) = X(t) \) during predictor.
- **LPC_NESCT**: if T., non-extrapolating SL2TL SI scheme for SL trajectory research.
- **LVERCOR**: T./F.: deep/thin layer equations. Default value is .F.
- **LVERTFE**: T./F.: finite element/conventional vertical discretisation. Default value is T. at ECMWF, F. at METEO-FRANCE.

The following variables (some of them are in namelist NAMPAR0) can be useful when running distributed memory jobs: NPRGPNS, NPRGPEW, NPRTRW, NPRTRV, NPROC, NPRTRN, NPRTRNS, LMPOFF, LMPDIAG, NOUTPUT. See documentation (IDDM) for more details about these variables.

YOMCVER.

See documentation (IDEUL) for more details. The following variables are used in the SL scheme for vertical spline cubic interpolations.

! LVSPLIP : T. if vertical spline cubic SL interpolations for O3.
! RVSPTRI,RVSPC: are used to re-profile the field to be interpolated in routine VSPLTRANS.
! RFAA,RFBB,RFCC,RFDD: are used in the computation of the vertical weights.
! VRDETAI: ratio (\( \text{eta}(\bar{lbar})-\text{eta}(\bar{lbar}-1) \))/(\( \text{eta}(\bar{lbar}-1)-\text{eta}(\bar{lbar}-2) \)),
! is used in the interpolation routine LAITVSPLCQM to
! ensure monotonicity and conservation properties.

The following variables are used in the SL scheme for the option "Stratospheric vertical trajectory smoothed".

! LSVTSM : stratospheric vertical trajectory smoothed (SL scheme).
! RPRES_SVTSM : smoothing done for standard pressure < RPRES_SVTSM.
4. Semi-lagrangian computations

YOMDIM.

Contains dimensioning variables. The following variables can be useful in the semi-Lagrangian scheme.

* Variables computed in SUDIM1+SUDIM2:
  - **NDLON**: maximum number of Gaussian longitudes (DM-global).
  - **NFLEVG**: number of model levels (DM-global).
  - **NPROMA**: working dimension of grid-point packets for grid-point computations (DM-global, default value depends on configuration). When a positive value of NPROMA is given in the namelist, LOPTPROMA is set to .T.; when a negative value of NPROMA is given in the namelist, LOPTPROMA is set to .F. and the model takes the absolute value of NPROMA.
  - **NGPBLKS**: number of NPROMA-subpackets in a processor (DM-local).
  - **NDGSAH**: modified lower bound for latitude (DM-local).
  - **NDGENH**: modified upper bound for latitude (DM-local).
  - **NRLEVX**: NRLEVX is used to dimension the arrays NVAUTF and NVAUTH in SUVERT. NVAUTH (resp. NVAUTF) gives the index l of the layer (resp. the index l of the half level) immediately above the level defined by the rounded vertical coordinate η: NVAUTF(INT(NRLEVX*η))=l. NVAUTH(INT(NRLEVX*η))=l. NVAUTF and NVAUTH are useful to find the interpolation grid, and to compute vertical weights: they allow to avoid expensive computations in the loops where are computed the vertical weights. The default value of NRLEVX is 100000 (i.e. the precision of η is five significant digits). NRLEVX has to be significantly greater than the maximal value of 1/(Δη). Do not forget to increase significantly NRLEVX when using option LREGETA=.F. and a lot of layers in the stratosphere!

* Namelist NAMDIM: The following variables of YOMDIM are in namelist NAMDIM: NPROMA, NRLEVX.

YOMDYNA and namelist NAMDYNA.

* **LGWADV**: use "w" as advective variable instead of the vertical divergence variable in the vertical divergence variable equation.
* **NGWADVSI**: when LGWADV, alternate treatments for linear terms.
  - NGWADVSI=1: linear terms set to 0 in LATTEX, LATTES for all eqns; linear terms evaluated at O and linear terms evaluated at F added to the RHS of equations in OPENDR.
  - NGWADVSI=2 (relevant for LSETTLS but not for LPC_NESC): linear terms set to 0 in LATTEX, LATTES for all eqns; linear terms evaluated at O added to the RHS of equations in LAPINEB (after the call to GNNHGW2SVD); linear terms evaluated at F added to the RHS of equations in OPENDR.
* **NPDVAR**: switch for type of variable used for pressure departure
  - 2: q_hat = ln(p/pi)
  - 3: Phi - B*Phi_surf
  - 4: Phi
* **NVDVAR**: switch for type of variable used for pseudo vertical divergence
  - 3: d3 = -g (p/(m.Rd.T)) d_parcs w/d_parcs eta
  - 4: d4 = d3 + X = d3 + (p/m.R.T) nabla_phi d_parcs V/d_parcs eta
  - 5: gw-B*gw_s
  - 6: gw
* **ND4SYS**: switch for the way of treatment of term NHX in the NH d4 equation.
  - ND4SYS=1: all contributions of D(NHX)/Dt are treated at the level of CPG_DYR+CALL_SL, and X is updated via GPXX in CPG_GP.
ND4SYS=2 (SL only): only the advective terms are treated at the level of CPG_DYN+CALL_SL; additional contributions are done in CPGLAG, and X(t+dt) is updated in CPGLAG.

* LRDBBC : .T. if S.-L. diagnostic BBC active for NH (if LSLAG=.T. only)
  .F. if eulerian development of NH BBC in LSLAG.

* LGWOPT1 : NH_GEOGW model only. Way to compute "dver" from "gw" in the RHS of temperature equation.
  .F. => vertical derivative applied to "gw".
  .T. => vertical derivative applied to pseudo NH departure of "gw".

! ------ Deep-layer equations: Wood and Staniforth formulation ---------------

* LRWSDLW : "w" inertia terms taken into account.
* LRWSDLR : metric terms taken into account (elsewhere than in the conversions between "dver" and "w").
* LRWSDLR2 : metric terms taken into account also in the conversions between "dver" and "w" (requires LRWSDLR=T).
* LRWSDLG : vertical variations of "g" taken into account (requires LRWSDLR=T)

! ------ SLHD horizontal diffusion -------------------------------------------

* LSLHD_W : switch for SLHD of horizontal flow.
* LSLHD_T : switch for SLHD of temperature.
* LSLHD_SPD : switch for SLHD of (NH) pressure departure.
* LSLHD_SVD : switch for SLHD of (NH) vertical divergence.
* LSLHD : internal model switch for semi-Lagrangian diffusion computation (not present in NAMDYNA).
* LSLHD_OLD : use old SLHD interpolator (cubic Lagrange mixed with linear).
* LSLHD_STATIC : do not diagnose kappa from horizontal flow deformation, use static value instead:
  SLHDKMIN for undiffused fields
  SLHDKMAX for diffused fields
* LSLHDQUAD : internal model switch indicating need to precompute quadratic weights.
* SLHDKMIN : minimum value for the kappa function (kappa_min).
* SLHDKMAX : maximum value for the kappa function (kappa_max).
* SLHDEPSH : dimensionless strength of horizontal Laplacian smoothing (C_sldhdepsh).
* SLHDEPSV : dimensionless strength of vertical Laplacian smoothing.

! ------ 3D turbulence ------------------------------------------------------

* L3DTURB : main key to activate 3D turbulence

! ------ Diagnostics --------------------------------------------------------

* LSLDIA : switch on semi-lagrangian dynamics diagnostics.

---

YOMDYN and namelist NAMDYN.

All the following variables are DM-global.

- **BETADT**: variable $\beta$.
- **NVLAG**: controls interpolations done in continuity equation. For 2D model, positive values are used for conventional formulation of continuity equation, negative values are used for Lagrangian formulation of continuity equation.
- **NWLAG**: controls interpolations done in momentum equation.
- **NTLAG**: controls interpolations done in temperature equation.
- **NSPDLAG**: controls interpolations done in pressure departure variable equation.
- **NSVVLAG**: controls interpolations done in vertical divergence equation.
- **NSPLTHOI**: key to split high order interpolations ($0 = no\; split;\; 1 = split\; with\; non-diffusive\; interpolation\; for\; physics;\; 2 = split\; without\; changing\; the\; nature\; of\; interpolations$).
- **LSPLOTHOIFGFL**: if T, allows to interpolate separately GFL variables and their diabatic tendency.
- **NSLDMK**: number (dimension) of used horizontal non-linear weights in the model.
- **JPSLDMK**: maximum possible value for NSLDMK.
4. Semi-lagrangian computations

- **LETRA**: T: alternate stable way of computing the SL trajectory (currently coded in the 2D shallow-water model and the 3D hydrostatic model only).
- **LSETTL**: way of computing the extrapolations in the 2TL SL scheme (stable algorithm if T).
- **LSETTLST**: way of computing the SL trajectory in the 2TL SL scheme (stable algorithm if T).
- **NITMP**: number of iterations for SL trajectory research.
- **VETAON** and **VETAOX**: Control the vertical bounding of the origin interpolation point $O$ for vertical interpolating semi-Lagrangian scheme. If $O$ is above the first layer, the final position of $O$ is bounded by $\eta_O = \eta_{n=1} + (\text{VETAON} - 1)(\eta_{n=1} - \eta_{n=0})$. If $O$ is below the last layer, the final position of $O$ is bounded by $\eta_O = \eta_{n=L} + (1 - \text{VETAOX})(\eta_{n=L} - \eta_{n=L})$.
- **RC_PD1**: coefficient $\delta$ (definition of $\dot{Q}$ in the Wood and Staniforth deep-layer NH-ÏVD model).
- **VMAX1**: default value is 0.9 times the default value of VMAX2. If $aU/r_s$ or $aV/r_s$ is above VMAX1 there is a warning in subroutines LARMES and LARMES2. This value is slightly modified in SUSC2B after recomputation.
- **VMAX2**: If $aU/r_s$ or $aV/r_s$ is above VMAX2 the model calls ABO1 in subroutines LARMES and LARMES2. VMAX2 is used for the calculation of the number of SL buffers in SUSC2B. This value is slightly modified in SUSC2B after recomputation.
- **VESL**: First-order uncentering factor $\epsilon$.
- **XIDT**: Second-order accurate uncentering factor $\epsilon_X$ for linear terms, available only in the 3D model. Controls the “alternative averaging” or “pseudo second-order uncentering” for linear terms in 2TL SL scheme. XIDT is applied only to linear scheme (contrary to VESL which applies also to non linear terms). XIDT and VESL cannot be both non zero. “Pseudo second-order uncentering” is theoretically as accurate as a second-order uncentering coefficient for linear terms, but does not need any computation of anterior point. XIDT must be equal to 0 when LVERTFE= .T. .
- **LQMHW, LQMHT, LQMHP, LQMHS, LQMHSVD**: Use of switches LQMHW (momentum equation), LQMHT (temperature equation), LQMHP (continuity equation) LQMHS (pressure departure variable equation) LQMHSVD (vertical divergence equation) allow to use shape-preserving option (in high order interpolations) for horizontal when set to .T.. Default value is .T. for LQMHT, .F. for the other switches.
- **LQM, LQMT, LQMP, LQMSP, LQMSV**: Use of switches LQM (momentum equation), LQMT (continuity equation), LQMP (pressure departure variable equation), LQMSP (vertical divergence equation) allow to use shape-preserving option (in 32 points interpolations) when set to .T.. Default value is .T. for LQM, LQMP, .F. for LQMT.
- **LADV, LADVFW and LIMPF**:
  - (LADV,LIMPF)=(.T.,.F.): (default value at METEO-FRANCE if 2TL SL scheme): implicit formulation for Coriolis term (Rechas formulation) in momentum equation.
  - (LADV,LIMPF)=(.F.,.T.): (default value at ECMWF for a 2TLSL scheme) Coriolis term in the semi-implicit scheme. This option can be used only in not stretched and not tilted mode.
  - In the deep layer equations, one can activate the switch LADV to treat implicitly the term $-2\Omega \times \mathbf{k}$. In this case, default value is the same as LADV.
- **RW2TLFF**: non-zero/zero: Refined/not refined computation of $(2\Omega \times \mathbf{a}k)$. Controls recomputation of the origin point using the average between the wind at the origin point and the provisional $t + \Delta t$ wind. Implict Coriolis term $(2\Omega \times \mathbf{a}k)$ is recomputed at this new position of the origin point. Active only if LADV is true. This option is currently coded only with the following options:
2TL SL scheme or vertical interpolating 3TL SL scheme with implicit formulation of Coriolis term.

NWLAG = 3 in NAMdyn.

When \( RW2TLFF \) is different from 0 or 1, the position of the new origin point \( ON \) is given by a linear interpolation with coefficient \( RW2TLFF \) between the position for \( RW2TLFF = 1 \) and the position for \( RW2TLFF = 0 \).

**Temporal filters:**

- REPS1 and REPS2: Asselin temporal filter coefficients for momentum, temperature, continuity, extra GFL variables equations.
- REPSM1 and REPSM2: Asselin temporal filter coefficients for humidity, liquid water, ice, cloudiness, ozone, pressure departure variable and vertical divergence equations.
- REPSPI: Asselin temporal filter coefficient for soil variables.

- RCMSLP0: Controls the “Tanguay-Ritchie" option (Ritchie and Tanguay, 1996) in continuity and temperature equations (value of \( \delta_T R \)). RCMSLP0 contains \( \delta_T R \).

- NCURRENT _ ITER: current iteration for iterative centred-implicit scheme (not in namelist).

- NSITER: total number of iterations required for an iterative centred-implicit scheme.

- All variables used in the SLHD interpolations:
  - SLHDA: Scaling factor of the deformation \( \kappa_a \).
  - SLHDA0: Namelist variable allowing to compute SLHDA; scaling factor of the deformation without the model resolution correction \( \kappa_{a0} \).
  - SLHDB: Exponent of the deformation. Contains \( \kappa_b \).
  - SLHDD0: Treshold for deformation tensor enhancement. Contains \( DEF_0 \).
  - SLHDDIV: Weight for including horizontal divergence into deformation \( (C_{slhddiv}) \).
  - SLHDRATDDIV: Nondimensional enhancer of divergence based diffusion \( (C_{slhdratddiv}) \).
  - SLHDHOR: Switch for computing flow deformation:
    - 0 - along eta levels (sloped)
    - 1 - along pressure levels (quasi horizontal)
  - ZSLHDP1: exponent used in the calculation of \( \kappa_a \).
  - ZSLHDP3: exponent used in the calculation of \( DEF_0 \).

- NITPRHS: number of iterations in GNH _ CONV _ PRHS (case LRWSDLR = T) to find \( r_s / a \).

**YOMGEM.**

Contains some quantities involved in the semi-Lagrangian scheme or in the SLHD diffusion. See inside this module. These quantities are not in a namelist.

**YOMGFL.**

Module containing grid-point arrays for GFL fields. See documentation (IDEUL) for more details.

**YOMGMV.**

Module containing grid-point arrays for GMV fields. See documentation (IDEUL) for more details.
4. Semi-lagrangian computations

YOMLASCW.

- **KPDUP**: for optimisation on NEC platform in routine LASCAW.

YOMLEG.

Contains some quantities involved in the semi-Lagrangian scheme or in the SLHD diffusion. For example **RSLDW** ($C_{\text{sldw}}$), **RSLD** ($C_{\text{sld1}}$, $C_{\text{sld2}}$ and $C_{\text{sld3}}$).

- **RSLDW** ($C_{\text{sldw}}$).
- **RSLD** ($C_{\text{sld1}}$, $C_{\text{sld2}}$ and $C_{\text{sld3}}$).
- **R3DTW**: weights for 3D turbulence Laplacian smoother in latitude.

See inside this module. These quantities are not in a namelist.

YOMMASK.

Variables specific to the distributed memory environment when the “on demand” processor communications are requested for the interpolation buffers in the semi-Lagrangian scheme. For more details see documentation about distributed memory features.

YOMMP.

Variables specific to the distributed memory environment. For more details see documentation about distributed memory features.

YOMSC2.

Contains parameters used to control vectorisation and memory space. Some of these parameters are used in the semi-Lagrangian scheme. No variable in namelist.

- **NSLWIDE**: number of rows the model lagged part runs behind (DM-global).
- **NFLDLSLB1** (resp. **NFLDLSLB2**): number of fields in buffer **SLBUF1** (resp. **SLBUF2**). DM-global variables.
- **NDIST**: start address for each row in grid-point calculations (DM-local).
- **NDIEND**: end address for each row in grid-point calculations (DM-local).
- **NCIST**: start address for each row of grid-point field in buffer (DM-local).
- **NCIEND**: end address for each row of grid-point field in buffer (DM-local).

YOMSEP.

Variables for optimisation of the adjoint code of the semi-Lagrangian scheme. The following variables are DM-global.

- **NVSEPC**: number of vertical blocks of layers for vertical loops in the cubic interpolation routines.
- **NVSEPL**: number of vertical blocks of layers for vertical loops in the linear interpolation routines.
- **LFINVDSEP**: computation of **NVSEPC** and **NVSEPL** is done if .TRUE.. Variables **NVSEPC** and **NVSEPL** are in **NAMdyn**.
YOMSLD.

The following variables are DM-global.

- **SLHDP**: contains $|\Delta x|_3$ (or maybe $\sqrt{2}|\Delta x|_3$).

YOMSLPHY.

- **LSLPHY** (in **NAMC**T0): for split physics (one part at $t - \Delta t$, one part at $t + \Delta t$); can be used only at ECMWF with the ECMWF package; default value is .FALSE. at METEO-FRANCE.
- **RSLWX**: level of implicitness of semi-Lagrangian/physics (parameter value, value is 0.5).
- **SAVTEND**: buffer to store the physical tendencies.
- **Some pointers** $M[X]_\text{SAVTEND}$, $M[X]_\text{SAVTEND}_S$, $M\text{SAVTEND}_S$, for $[X]=U,V,T,\text{SA T}$.

YOMSLREP.

Used for bit reproducibility of the semi-Lagrangian adjoint code ("SLAD").

```plaintext
! LSLADREP : CONTROL BIT REPRODUCIBILITY (in NAMCTo)
! : T - SLAD IS BIT REPRODUCIBLE WHEN NUMBER OF PROCESSORS
! : OR PARTITIONING IS CHANGED
! : F - SLAD IS NOT BIT REPRODUCIBLE WHEN NUMBER OF PROCESSORS
! : OR PARTITIONING IS CHANGED (default value at MF)
! LVECADIN : CONTROL VECTOR/SCALAR FOR ADJOINT INTERPOLATION
! (default value at MF is .T.)
! NSLMAP : USED FOR REMAPPING POINTS IN SL BUFFER SO THAT THEY
! : ARE UNIQUE
! NADCORE : INNER HALO FOR ADJOINT INTERPOLATIONS
! : ARE UNIQUE
! LSCLCORE : T - IF NSLCORE POINT, F - IF NOT
! NGPTOTAD : NUMBER OF GRID POINTS IN INNER HALO
! MASK_SLTOT : SUM OF SL BUFFER POINTS USED IN TRAJECTORY CALC.
! : USED TO OPTIMISE NGPTUTAD AND THEREBY IMPROVE PERFORMANCE
! RSASIGN : USED FOR CORRECTING SYMMETRIC / ANTISYMMETRIC PROPERTIES
! : (MIRRORED LATITUDES NEAR POLES)
! GEMUAD : SEMI-LAGRANGIAN ADJOINT VERSION OF GEMU
! GESLOAD : SEMI-LAGRANGIAN ADJOINT VERSION OF GESLO
! GMAD : SEMI-LAGRANGIAN ADJOINT VERSION OF GM
! GNORDLAD : SEMI-LAGRANGIAN ADJOINT VERSION OF GNORDL
! GNORDMAD : SEMI-LAGRANGIAN ADJOINT VERSION OF GNORDM
! GSQM2AD : SEMI-LAGRANGIAN ADJOINT VERSION OF GSQM2
! RCOLONAD : SEMI-LAGRANGIAN ADJOINT VERSION OF RCOLON
! RSILONAD : SEMI-LAGRANGIAN ADJOINT VERSION OF RSILON
! RINDXAD : SEMI-LAGRANGIAN ADJOINT VERSION OF RINDX
! RINDYAD : SEMI-LAGRANGIAN ADJOINT VERSION OF RINDY
```

YOMTRSL.

Buffer for trajectory array at time $t - dt$ (used for tangent-linear and adjoint of semi-Lagrangian) No variable in namelist.

- **NLENGTHSLB**: length of buffer for 3D fields.
- **NTRSLTYPE**: trajectory storage method.
4. Semi-lagrangian computations

0 if no storage (only works for LEFLA=TRUE.)

1 if minimum storage: trajectory is stored for prognostic variables and non linear terms at $t - \Delta t$. Calculation of buffers "PB1..." has a TL code similar to the remaining calculations.

2 if more storage and less computation: trajectory is stored for buffers "PB1..." themselves in the direct code; this trajectory can be directly read in the TL code. This is the only available possibility in the 3D model.

Default value is 2.

- NPCKSLSTraj: packing factor ($\leq 2$) for SL trajectory.
- TRAJSL: buffer for 3D fields to store the ADTL-trajectory.

YOM_YGFL (and namelist NAMGFL).

Contains the descriptors of GFL arrays. See documentation (IDEUL) for more details. The following attributes are used in the semi-Lagrangian scheme:

- LADV: T/F: horizontal advection switched on/off.
- LHV: Hermite cubic vertical interpolations.
- LVSPLIP: .T. if vertical spline cubic SL interpolations.
- LSLHD: .T. if SLHD diffusion switched on.
- LQM: .T. if monotonic interpolators.
- LQMH: .T. if horizontally monotonic interpolators.

18.2 Modules for type definition.

- TYPE_GFLS: derived types for describing the GFL structure (types TYPE_GFLD, TYPE_GFL_COMP, TYPE_GFL_NAML).
- TYPE_GMVS: derived types for GMV (types TYPE_T0, TYPE_T9, TYPE_T1, TYPE_PH9, TYPE_AUX).

For more details see comments inside these routines.

18.3 Modules containing subroutines.

- GFL_SUBS_MOD: contains the following routines:
  - DEFINE_GFL_COMP: setup individual GFL field (YGFL,YGFLC).
  - SET_GFL_ATTRIB: add attributes to previously setup GFL components (YGFLC).
  - PRINT_GFL: print GFL attributes.
  - DEACT_CLOUD_GFL: deactivate prognostic cloud variables.
  - REACT_CLOUD_GFL: reactivate prognostic cloud variables.
  - FALSIFY_GFLC: set field descriptors to false.
  - NOADVECT_GFLC: switch off advection.
  - COPY_GFLC_GFLC: copy field descriptors.

- GMV_SUBS_MOD: contains the following routines:
  - SETUP_GMV: calls SETUP_T0, SETUP_T9, SETUP_T1, SETUP_PH9, SETUP_AUX, computes NDIMGMV and NDIMGMVS, allocates GMV and GMVS.
  - SETUP_T0: computes the pointers used for the part of GMV and GMVS containing data at time $t$ (YT).
  - SETUP_T9: computes the pointers used for the part of GMV and GMVS containing data at time $t - \Delta t$ (YT9).
  - SETUP_T1: computes the pointers used for GMVT1 and GMVT1S (YT1).
  - SETUP_PH9: computes the pointers used for the part of GMV and GMVS containing data at time $t - \Delta t$ for physics: pointer (YPH9).
  - SETUP_PH9_PRED: cf. SETUP_PH9 but for predictor step if LPC_FULL=T.
  - SETUP_PH9_CORR: cf. SETUP_PH9 but for corrector step if LPC_FULL=F.
- TRAJECTORY_MOD: manages trajectory for TL and AD runs in a unified way; contains the following routines:
  - ALLOCATE_TRAJECTORY
- DEALLOCATE_TRAJECTORY
- READ_TRAJECTORY
- INIT_TRAJECTORY

calls the following modules:
- TRAJ_MAIN_MOD
- TRAJ_PHYSICS_MOD
- TRAJ_SEMILAG_MOD
- TRAJ_SURFACE_MOD

* TRAJ_MAIN_MOD: manages main trajectory; contains the following routines:
  - ALLOCATE_TRAJ_MAIN
  - DEALLOCATE_TRAJ_MAIN
  - STORE_TRAJ_MAIN
  - GET_TRAJ_SPEC
  - GET_TRAJ_GRID
  - READ_TRAJ_GRID
  - READ_TRAJ_SPEC

* TRAJ_PHYSICS_MOD: manages physics trajectory; contains the following routines:
  - ALLOCATE_TRAJ_PHYS
  - DEALLOCATE_TRAJ_PHYS
  - STORE_TRAJ_PHYS
  - GET_TRAJ_PHYS

* TRAJ_SEMILAG_MOD: manages semi-Lagrangian trajectory; contains the following routines:
  - ALLOCATE_TRAJ_SLAG
  - DEALLOCATE_TRAJ_SLAG
  - STORE_TRAJ_SLAG
  - GET_TRAJ_SLAG

* TRAJ_SURFACE_MOD: manages surface trajectory; contains the following routines:
  - ALLOCATE_TRAJ_SFC
  - DEALLOCATE_TRAJ_SFC
  - STORE_TRAJ_SFC
  - STORE_TRAJ_CST
  - FORCE_SFC_TRAJ
  - GET_TRAJ_SFC
  - READ_TRAJ_SFC
19 An example of operational SL2TL namelist for ARPEGE.

* Introduction: In order to avoid to have to recompile all the code when changing the value of a variable (for example a dimension), as it was the case in the old PERIDOT/EMERAUDE system, some variables are provided in different namelists, so it is possible to change the value in the namelist without recompiling the code. All variables which have to be modified compared to the default value can be specified in the namelist, but there is also an option where a limited amount of variables are specified in a command line and where the model retrieves some data (for example the repartition of vertical levels) by reading the frame of some ARPEGE files.

* Example of namelist and command line for the ARPEGE operational forecast: The example provided below is the forecast namelist "namelistf" which has been used in the ARPEGE (TL798L70c2.4 stretched version) operational suite on 01/12/2010, with minor modifications (adapt the namelist for cycle 37T1, remove activation of FULL-POS diagnostics). This suite used the cycle 36t1_op1. The namelist is:

```
&NACIETEO
&NACOBS
&NACTAN
&NACTEX
&NACVEG
&NADOCK
&NAAEAEM7
&NAAEAER
&NAAECOAPHY
&NAEPHY
&NAAERAD
LRRTM=.TRUE.,
LSRTM=.FALSE.,
HRADFR=1.,
NSW=6,
RLWINHF=0.9,

/NAIMPO
/NALORI

/NAMAFN
TFP_I%LLGP=.TRUE.,
TFP_L%LLGP=.TRUE.,

/NAMARPHY
/NANCA
/NANCAPE

/NAMCFU
LCUMFU=.TRUE.,
LPDUTP=.TRUE.,
LPLP1=.TRUE.,
LPLP2=.TRUE.,
LF=.TRUE.,
LFS=.TRUE.,
LFSOIL=.TRUE.,
LMODN=.TRUE.,
LNEPAR=.TRUE.,
LSTRD=.TRUE.,
LSTR=.TRUE.,
LFRAC=.TRUE.,
LRAYD=.TRUE.,

/NANCHET
```


4. Semi-lagrangian computations

&HAMCHK
&HAMCLA
&HAMCLDP
&HAMCLI
&HAMCLDP15
&HAMCLTC
&HAMDCK
&HACOM
&HAMCOUPLD4

&HAMCT0
LFBDAP=.TRUE.,
LFDBAP=.FALSE.,
LTWTL=.TRUE.,
LRPR=.TRUE.,
NDHFZTS(0)=-9,
NDHFZTS(1)=-6,
NDHFZTS(2)=-12,
NDHFZTS(3)=-18,
NDHFZTS(4)=-24,
NDHFZTS(5)=-36,
NDHFZTS(6)=-48,
NDHFZTS(7)=-60,
NDHFZTS(8)=-72,
NDHFZTS(9)=-96,
NFRHHSF=1,
NFRHHS=1,
NFRHDS=1
NFRHDI=1,
NHISTS(0)=-28,
NHISTS(1)=-0,
NHISTS(2)=-3,
NHISTS(3)=-6,
NHISTS(4)=-9,
NHISTS(5)=-12,
NHISTS(6)=-15,
NHISTS(7)=-18,
NHISTS(8)=-21,
NHISTS(9)=-24,
NHISTS(10)=-27,
NHISTS(11)=-30,
NHISTS(12)=-33,
NHISTS(13)=-36,
NHISTS(14)=-39,
NHISTS(15)=-42,
NHISTS(16)=-45,
NHISTS(17)=-48,
NHISTS(18)=-51,
NHISTS(19)=-54,
NHISTS(20)=-60,
NHISTS(21)=-66,
NHISTS(22)=-72,
NHISTS(23)=-78,
NHISTS(24)=-81,
NHISTS(25)=-84,
NHISTS(26)=-90,
NHISTS(27)=-96,
NHISTS(28)=-102,
NSDITS(0)=-1,
NSDITS(1)=0,
NSDITS(2)=0,
NDFFRT=.FALSE.,
LDFFRT=.FALSE.,
LDFFR=.FALSE.,
LWPGRA=FALSE.,
LVEGTE=.TRUE.,
LVEGTE=.TRUE.,
LAPR=.TRUE.,
LPRU=.TRUE.,
LPRY=.TRUE.,
NFRGDI=1,
NFRGDS=1,
NFRGDI=1,
LVERTFE=.TRUE.,
NVECH=3,
NFRSDI=1,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
LAPR=.TRUE.,
NFRHLD=1,
4. Semi-lagrangian computations

NRES=0,
/ &NAMCUMF
/ &NAMCUMFS
/ &NAMVNH
  GTADJS=10800.,
  XDEPTH=1.,
  XDEPTH_D=4000.,
  XTPERT=0.3,
  XENTR=0.013,
/ &NAMDDN
  LHDEFZ=.TRUE.,
  LHDEK=.TRUE.,
  LHDZON=.TRUE.,
  NDHKD=30,
/ &NAMDFI
/ &NAMDIM
  NPROMA=-3582,
/ &NAMDPHY
/ &NAMDY
  LADVF=.TRUE.,
  LSETTL=.TRUE.,
  LSETTLS=.TRUE.,
  NTLAG=3,
  NVLAG=3,
  RCMSLP0=1.,
  REPST1=0.,
  REPST2=0.,
  REPS1=0.,
  REPS2=0.,
  SIPP1=0.,
  SIPR=100000.,
  SITR=350.,
  NDLNP=0.,
  VESTL=0.,
  XIDL=0.,
  NADTLFF=1.,
  NADAMPDIV=5.,
/ &NAMDNA
/ &NAMDNSCORE
/ &NANEMIS_CONF
/ &NANENKF
/ &NANFA
  NBITCS=30,
  NBITPG=30,
  NSTRON=-1,
  YFALNBITS=16,
  YFAINNBITS=16,
  YFARKNBITS=16,
  YFASKNBITS=16,
  YFATKEBNBITS=24,
/ &NANFPC
/ &NANFPDO
/ &NANFPDY2
/ &NANFPDYH
/ &NANFPDYI
/ &NANFPDPY
/ &NANFPDPYS
/ &NANFPDYT
/ &NANFPDPYV
/ &NANFF
/ &NANFPG
Semi-lagrangian computations

&HAMFPIOS
&HAMFPHY
&HAMFPSC2
&HAMFPSC2_DEP
&HAMGEM
  REFLKUD=5000.,
&HAMGFL
  Y1_NLNLPGINGP= TRUE.,
  Y1_NLNLGP= TRUE.,
  Y1_NLNLTI= TRUE.,
  Y1_NLNLPHY= FALSE.,
  Y1_NLNLREQIN= 1,
  Y1_NLNLREQOUT= TRUE.,
  Y1_NLNLADV= TRUE.,
  Y1_NLNLQM= TRUE.,
  Y1_NLNLGPINGP= TRUE.,
  Y1_NLNLGP= TRUE.,
  Y1_NLNLTI= TRUE.,
  Y1_NLNLPHY= TRUE.,
  Y1_NLNLREQIN= 1,
  Y1_NLNLREQOUT= TRUE.,
  Y1_NLNLADV= TRUE.,
  Y1_NLNLQM= TRUE.,
&HAMGOM
&HAMGRID
&HANGWD
&HANGWMS
&HAMHCP
&HAMHLOPT
&HAMINI
  NEINI=-0,
&HAMIOI
&HAMIOS
&HAMJBCODES
&HAMJFH
&HAMJG
&HAMJG
&HAMLCZ
4. Semi-lagrangian computations

`&NAMLSFORC`

`&NAMMARS`

`&NAMMCC`

`&NAMMCCU`

`&NAMMCCD`

`&NAMMODERR`

`&NAMMTS`

`&NAMMWAVE`

`&NAMMNI`

`&NAMNPROF`

`&NAMNUD`

`&NAMOBS`

`&NAMONEDVAR`

`&NAMOPH`

CFNHWF='ECHIS',
LINC=.TRUE.,

`&NAMPAR0`

NGSIPUT=1,
NPRGCV=2,
NPRGCVS=2,
NPRGCVH=1,
NPRGC=1,
MP_TYPE=2,
MBX_SIZE=128000000,

`&NAMPAR1`

LSPLIT=.TRUE.,
MISTRIN=2,
MISTRUTE=2,
NCOLMATT=1638400,
LEQ_REGIONS=.FALSE.,

`&NAMPARAR`

CGMIXLEN='AY',
LAERODES=.TRUE.,
LAEROLAN=.TRUE.,
LAEROSEA=.TRUE.,
LAEROSOO=.TRUE.,
LCONDWT=.TRUE.,
LDIFCONS=.TRUE.,
LFPCOR=.TRUE.,
LNEWD=.TRUE.,
LNOIAS=.TRUE.,
LO3ABC=.TRUE.,
LPROCLD=.TRUE.,
LRAY=.FALSE.,
LRAYFM=.TRUE.,
LRAYLV=.TRUE.,
LCVPPKF=.TRUE.,
LECDEEP=.TRUE.,
LECSHAL=.TRUE.,
LECT=.TRUE.,
LFLUSO=.TRUE.,
LNEBECT=.FALSE.,
LNSMLIS=.FALSE.,
LQ3FL=.TRUE.,
LQI3REL=.TRUE.,

`&NAMPHY0`

EDD=1.,
EDK=1.,
GCVNU=5.E-05,
GCVPSI=1.,
GCVPSIE=1.,
4. Semi-lagrangian computations

GVD5C = 5.4,
GVD5E = 0.05,
GVD5AI = 0.5,
GVD5BC = 1.1,
GVD5SC = 5,
GVD5AF = 0.25,
REVA5X = 2.0E-07,
RIC5LM = 0.5,
TDD5P = 0.6,
TDD5P = 0.
USUR5C = 0.175,
USUR5CL = 1,
USUR5D = 0.1,
V2O5C = 1.0E-4,
X5LM = 0.5,
X5LM = 5000.,
XHM5LM = 10.,
AL5MA5X = 10000.,
GC5V5MIN = 30000.,
RF5ACH5M = 1.2,
RF5L5CHE = 0.25,
RF5F5TAO = 3600.,
RP5RTH = 1.,
RQC5RT52 = 0.,
RQC5V5MIN = 1.0E-5,
SX5B5CD = 1.,
AL5MA5VE = 0.,
RQC5R5NS = 0.,
RQC5R5SM = 1.,
TV5I = 0.08,
TF5I = 0.02,
TF5V = 1.5,

&NAM5PHY1
ALE5HN = 0.65,
ALCR5IN = 0.75,
EM5G5LA = 0.95,
EM5MER = 0.99,

&NAM5PHY2
FACE5AF = 3.5,
XM5UL5AF = 0.,
L5R5A5F5E5 = True.,
HT5K5ER5AF = 20.0,

&NAM5PHY3
&NAM5PHYS
&NAM5PONG
&NAM5PPC
&NAM5PRE
&NAM5RAD515
&NAM5C5F
&NAM5C5DEF
&NAM5RES
&NAM5B5GR5I
&NAM5AI5NC
&NAM5R5IP
&NAM5SAT5S
&NAM5SC5C
&NAM5SC5EN
&NAM5SC5M
&NAM5SE5KF
&NAM5SENS
&NAM5SIM5PH5L
&NAM5SK5F
&NAM5SP5DOT
&NAM5SS5MI
4. Semi-lagrangian computations

// &NAMSTA
// &NAMSTOPH
// &NAMSWI
// &NAMTESTVAR
// &NAMTLEVEL
// &NAMTOPH
ETCN1M=5000.,
ETNEBU=5000.,
ETFLUI=5000.,
XDMTP=2.E-08,
XDMTPB=800.,
XDMTPX=4.E-07,
XDMUP=680.,
XDMUX=2.E-06,
// &NAMTRAJP
// &NAMTRANS
// &NANTS
// &NANVAR
// &NANVARBC
// &NANVARBC_AIREP
// &NANVARBC_ALLSKY
// &NANVARBC_RAD
// &NANVARBC_TCW
// &NANVARBC_TD3
// &NANVAREPS
// &NANVDOZ
// &NANVFIP
// &NANVRTL
// &NANV1
// &NANWFK
// &NANXFU
LXCLP=.TRUE.,
LXCLS=.TRUE.,
LXFU=.TRUE.,
LXHCL5=.TRUE.,
LXICV=.TRUE.,
LXHNP=.TRUE.,
LXHNTT=.TRUE.,
LXQCLS=.TRUE.,
LXQCL5=.TRUE.,
LXTGST=.TRUE.,
LXTTCLS=.TRUE.,
LTXNEB=.TRUE.,
LTXQICE=.TRUE.,
LTXQICE1=.TRUE.,
LTXGST=.TRUE.,
// &NAM_CANAPE
// &NAM_DISTRIBUTED_VECTORS
// &NAPHLC
// &NEMCT0
// &NENDIM
// &NENDDYN
// &NEMELCOA
// &NEMELCOB
// &NEMFPEZO
4. Semi-lagrangian computations

Explanation of some elements of this namelist:

- **NAMCFU**: LCUMFU=.TRUE. means that the CFU diagnostics are activated.
- **NAMCT0**: in this namelist one finds some information about the temporal advance scheme and the frequency of some outputs. For example LTWOTL=.TRUE., means that the two-time level semi-Lagrangian scheme is used; NFRHIS=1, NHISTS(0)=-28, NHISTS(1)=-0, NHISTS(2)=-3, ..., NHISTS(28)=-102, means that the output of historical files is asked for 28 times during this 102h range model integration, and that the occurrence number 1 (resp. 2, ..., 28) is at range 0h (resp. 3h, ..., 102h).
- **NAMDYN**: in this namelist one finds information about Asselin temporal filter, and some options of the semi-Lagrangian scheme.
- **NAMINI**: in this namelist one finds information relative to the initialization; NEINI=0 means that there is no initialization.
- **NAMPAR0** and **NAMPAR1**: in this namelist one finds information relative to the distributed memory architecture. In our case there are 2 processors in the first level of distribution and one processor in the second one.
- **NAMPHY, NAMPHY0, NAMPHY1, NAMPHY2, NAMPHY3, NAMTOPH**: give information about the set of physical parameterizations and the coefficients used in the physics.
- **NAMXFU**: LXFU=.TRUE. means that the XFU diagnostics are activated.

The previous namelist must be completed with a command line to be specified in the script.

- **NCONF=001**: means that the job runs in configuration 1.
- **VERSION=meteo**: means that the job runs with the default METEO-FRANCE options and not the ECMWF options.
- **CNMEXP=ARPE**: four letters code of the job, which appears in the local name of some files; for example the local name of the initial conditions file in this case has the name ICMSHARPEINIT.
- **TSTEP=600.**: means that the timestep is 600 s.
- **NSTOP=h102**: means that the forecasts has a range of 102 h.
- **ADVEC=sli**: means that the advection scheme is a semi-Lagrangian one.

The reader will remark that some information does not appear in the above namelist and command line: the truncation, the geometry (stretching, high resolution pole, type (reduced or not) of Gaussian grid), the number of latitudes of the Gaussian grid, the maximum number (and also the number for each latitude) of longitudes of the Gaussian grid, the number of layers and the definition of these layers. In our example where a command line is used, all this information is read in the frame of the initial conditions file.

* **ALADIN**: The namelist looks like the one used in ARPEGE, with some additional elements, the name of them generally starting by NEM. For example in the forecast namelist "namelistsfc" of ALADIN-FRANCE which was operational on 01/12/2010, the reader will find additional elements starting by NEM... and some additional information about the lateral coupling (for example the information TEFRC1=10800. present in NEMELBC0B means that the coupling frequency is 3h=18000s).

In the command line one finds the additional instruction DOMAIN=aladin meaning that the model is now ALADIN.
20 Bibliography.

The following list is far to be complete. More detailed references are available in some of the following papers (in particular Staniforth and Côté, 1991), especially for papers published before 1991. Most of the recent papers have been published in *Mon. Wea. Rev.*

20.1 Publications.


4. Semi-lagrangian computations

• Mawson, M. H., 1995: Implementation of semi-Lagrangian advection in the next
generation U.K. Met Office Unified Model. *Forecasting Research Division Scientific

model using a vector semi-Lagrangian finite-difference scheme: Part II: version with

• Nair, R. D., and B. Machenhauer, 2002: The mass-conservative cell-integrated

• Oliveira, A., and A. M. Baptista, 1995: A comparison of integration and interpo-

Wea. Rev.*, 123, 3042-3058.

3318-3330.

• Priestley, A., 1993: A quasi-conservative version of the semi-Lagrangian advection

• Pudykiewicz, J. A., and A. Kallaur, 1997: Semi-Lagrangian modelling of tropo-

• Purser, R. J., and L. M. Leslie, 1991: An efficient interpolation procedure for
high-order three-dimensional semi-Lagrangian models. *Mon. Wea. Rev.*, 119,
2492-2498.

Rev.*, 122, 745-756.

• Purser, R. J., and L. M. Leslie, 1997: High-order generalized Lorenz N-cycle
Wea. Rev.*, 125, 1261-1276.

• Qian, J. H., F. H. M. Semazzi, and J. S. Scroggs, 1998: A global nonhydrostatic
semi-Lagrangian atmospheric model with orography. *Mon. Wea. Rev.*, 126, 747-
771.

• Ramachandran, D. N., and B. Machenhauer, 2002: The mass-conservative cell-
integrated semi-Lagrangian advection scheme on the sphere. *Mon. Wea. Rev.*, 130,
649-667.

• Rančič, M., 1992: Semi-Lagrangian piecewise biparabolic scheme for two-dimensional

• Rančič, M., 1995: An efficient, conservative, monotonic remapping for semi-Lagrangian

• Ritchie, H., 1986: Eliminating the interpolation associated with the semi-Lagrangian

• Ritchie, H., 1988: Application of the semi-Lagrangian method to a spectral model

• Ritchie, H., 1991: Application of the semi-Lagrangian method to a multilevel spec-

• Ritchie, H., and C. Beaudoin, 1994: Approximations and sensitivity experiments
with a baroclinic semi-Lagrangian spectral model. *Mon. Wea. Rev.*, 122, 2391-
2399.

• Ritchie, H., C. Temperton, A. Simmons, M. Hortal, T. Davies, D. Dent, and M.
Hamrud, 1995: Implementation of the semi-Lagrangian method in a high resolution

4. Semi-lagrangian computations


20.2 Some internal notes and other ARPEGE notes.

- (TDECDYN) 2010: IFS technical documentation (CY36R1). Part III: dynamics and numerical procedures. Available at "http://www.ecmwf.int/research/ifsdocs/".
- (TDECTEC) 2010: IFS technical documentation (CY36R1). Part VI: technical and computational procedures. Available at "http://www.ecmwf.int/research/ifsdocs/".
- Mc Donald, A., 1996: Controlling noise in two time level semi-Lagrangian semi-implicit integrations; comments on some new developments. *HIRLAM technical report nr 24*, 4-10.


• (IDBAS) Yessad, K., 2011: Basics about ARPEGE/IFS, ALADIN and AROME in the cycle 37T1 of ARPEGE/IFS (internal note).

• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note).

• (IDDH) Yessad, K., 2011: Horizontal diffusion in the cycle 37T1 of ARPEGE/IFS (internal note).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note).

• (IDEUL) Yessad, K., 2011: Integration of the model equations, and Eulerian dynamics, in the cycle 37T1 of ARPEGE/IFS (internal note).


20.3 Workshop proceedings.

• Workshop proceedings on semi-Lagrangian methods, 6-8 November 1995, 264pp. Available at ECMWF.
This chapter describes two types of schemes allowing to stabilize the numerical discretizations of the models: the semi-implicit scheme and the iterative centred-implicit scheme (sometimes called predictor-corrector scheme). An algorithmic description to different types of equations, and some technical information (organigramme) are provided.

1 Introduction.

1.1 Interest of semi-implicit and iterative centred-implicit schemes.

For both hydrostatic and non-hydrostatic models it is necessary to treat implicitly the linear terms source of (fast moving) gravity waves to ensure a good stability; hence the resolution of equations involve the inversion of a linear system leading to a Helmholtz equation: inversion of such a system is more convenient to do in spectral space. For non-hydrostatic models the semi-implicit scheme is generally not sufficient (especially for a two-time level semi-Lagrangian scheme) and some non-linear terms have also to be treated implicitly; for that one uses an iterative centred-implicit (abbreviated into “ICI”) scheme. The iterative centred-implicit schemes are often called “predictor-corrector” schemes, but in a theoretical point of view one has normally to reserve this appellation for a subset of iterative centred-implicit schemes with only one iteration. The iterative centred-implicit scheme can have an incremental formulation or a non-incremental formulation. The one which is coded in ARPEGE/ALADIN (for both hydrostatic and non-hydrostatic models) and which will be retained and described is a non-incremental one. All the additional calculations generated by an iterative centred-implicit scheme are mainly done in the grid-point calculations and in the spectral transforms.

1.2 Distributed memory.

Some distributed memory features are now introduced in the code and will be briefly described. For convenience one uses some generic appellations.
• Expression “DM-local” for a quantity means “local to the couple of processors \((proca,procb)\)”: each processor has its own value for the quantity. Expression “DM-local computations” means that the computations are done independently in each processor on “DM-local” quantities, leading to results internal to each processor, which can be different from a processor to another one.

• Expression “DM-global” for a quantity means that it has a unique value available in all the processors. Expression “DM-global computations” means that the computations are either done in one processor, then the results are dispatched in all the processors, or the same computations are done in all the processors, leading to the same results in all the processors.

• In a routine description the mention “For distributed memory computations are DM-local” means that all calculations done by this routine are DM-local; the mention “For distributed memory computations are DM-global” means that all calculations done by this routine are DM-global; when no information is provided it means that a part of calculations are DM-local and the other part is DM-global.

• Expression “main” processor currently refers to the processor number 1: \((proca,procb)=(1,1)\).

1.3 The different models described.

• 2D shallow-water model.

• 3D primitive equations model (denoted as HYD).

• 3D NH model with \(\hat{Q}\) and \(d\) or \(d_4\) NH prognostic variables (denoted as NH-PDVD).

• 3D NH model with \(\Phi\) and \(g_\nu\) NH prognostic variables (denoted as NH-GEOGW).

1.4 Other restrictions of this documentation.

• In the non-hydrostatic model with \(\hat{Q}\) and \(d\) or \(d_4\) NH prognostic variables, only the option with \(\hat{Q}\) and \(d\) as prognostic variables (options NPDVAR=2, NVDVAR=3) is currently described. The choice of \(d_4\) as prognostic variable NVDVAR=4 does not change the expression of linear terms.

• Linear systems are written for thin layer equations: in practical, they don’t change for deep layer equations because the radius \(r\) is linearised around a reference value equal to \(a\).

For the (Wood and Staniforth, 2003) NH deep-layer system, hydrostatic pressures \((\Pi,\Pi_s)\) must be replaced by mass-integrated coordinates \((\tilde{\Pi},\tilde{\Pi}_s)\) in the linear system.

1.5 Modifications since cycle 37.

• Minor changes on the organigramme (for example call \((E)SPCM\) directly from STEPO).
2 Notations.

- $M$ is the mapping factor. $\bar{M}$ is a reference mapping factor for semi-implicit computations. $\bar{M} = c$ (stretching factor) if semi-implicit scheme with reduced divergence ($\text{LSIDG} = \text{F.}$ in $\text{YOMDYN}$). $M$ is $\bar{M}$ (mapping factor) if semi-implicit scheme with unreduced divergence ($\text{LSIDG} = \text{T.}$ in $\text{YOMDYN}$).

- $a$ is the Earth mean radius.

- $r$ is the radius. The reference value of $r$ for the linearisation is the mean radius $a$. In the thin layer equations, $r = a$ everywhere. In the $\text{LVERCOR} = \text{T}$ (White and Bromley, 1995) deep-layer equations, $r$ is replaced by a pseudo-radius $r_s$ depending only on the hydrostatic pressure. All the equations involving the radius will be written with the denotation $r$.

- $V$ is the horizontal geographical wind. Its zonal component is $U$. Its meridian component is $V$.

- $D$ is the unreduced divergence of horizontal wind, $D'$ is the reduced divergence. $D$ and $D'$ are linked by the relationship $D = (a/r) \ast M^2 \ast D'$.

- $\zeta$ is the unreduced vorticity of horizontal wind, $\zeta'$ is the reduced vorticity. $\zeta$ and $\zeta'$ are linked by the relationship $\zeta = (a/r) \ast M^2 \ast \zeta'$.

- $w$ is the $z$-coordinate vertical velocity: $w = \frac{dz}{dt}$.

- $T$ is the temperature. $T^*$ is a vertically-constant reference temperature which is used in the semi-implicit scheme and in some non-hydrostatic equations. Default value is 300 K or 350 K according to configuration (for more details see subsection 11.7 for variable $\text{SITR}$). IF $\text{LSPRT} = \text{T.}$ (use of virtual temperature in spectral transforms instead of real temperature), $T^*$ is used as a reference virtual temperature (same default value).

- $T_s^*$ is a cold vertically-constant reference temperature which is used in the semi-implicit scheme in the NH vertical divergence equation; it is recommended to have $T_s^*$ lower than the current temperature.

- $q$ is the humidity.

- $\Pi$ is the hydrostatic pressure, $\Pi_s$ is the hydrostatic surface pressure. $\Pi^*$ is a reference hydrostatic pressure and $\Pi_s^*$ is a reference hydrostatic surface pressure, which are used in the semi-implicit scheme and in some non-hydrostatic equations. These reference quantities are vertically dependent and “horizontally” (i.e. on $\eta$ surfaces) constant. Default value of $\Pi_s^*$ is generally between 800 hPa and 1000 hPa. $\Delta\Pi^*$ are layer depths corresponding to a surface hydrostatic pressure equal to $\Pi_s^*$.

- $\Pi_s^{st}$ is a reference hydrostatic pressure equal to the surface pressure of the standard atmosphere (variable $\text{VP00}$). Default value is 101325 Pa.

- $\omega = \frac{d\Pi}{dt}$ is the total temporal derivative of the hydrostatic pressure (vertical velocity in hydrostatic pressure coordinate).

- $p$ is the pressure, $p_s$ is the surface pressure.

- $\hat{Q}$ is the pressure departure variable. Expression of $\hat{Q}$ is:

$$\hat{Q} = \log \frac{p^*}{\Pi^*} \quad (1)$$

- $gz$ is the geopotential height.

- $\Phi$ is the total geopotential (equivalent height in the shallow-water model), $\Phi_s$ is the surface geopotential (i.e. the orography). In the thin layer equations, $\Phi = gz$. $\Phi_s$ is assumed to be always equal to $gz$. $\Phi^*$ is a reference equivalent height which is only used in the shallow-water model (semi-implicit scheme). Default value of $\Phi^*$ is 100000 J/kg. $\Delta\Phi^*$ is a reference geopotential depth computed on model levels.
• \( \Omega \) is the Earth rotation angular velocity.
• \( r \) is the vector directed upwards, the length of which is the Earth radius \( a \).
• \( g \) is the gravity acceleration constant, assumed to be vertically constant in the current documentation. For the (Wood and Staniforth, 2003) deep-layer NH equations with vertical variations of \( g \), only the reference value of \( g \) (vertically constant) is taken into account in the semi-implicit scheme.
• \( R \) is the gas constant for air and \( R_d \) the gas constant for dry air.
• \( c_p \) is the specific heat at constant pressure for air and \( c_{pd} \) is the specific heat at constant pressure for dry air.
• \( c_v \) is the specific heat at constant volume for air and \( c_{vd} \) is the specific heat at constant volume for dry air.
• \( \nabla \) is the unreduced first order horizontal gradient on \( \eta \)-surfaces. \( \nabla' \) is the reduced first order horizontal gradient. These two operators are linked by the relationship \( \nabla = \left( a/r \right) * M * \nabla' \).
• \( D_3 \) is the true 3D divergence. In the thin layer equations, expression of \( D_3 \) is:
\[
D_3 = \nabla \Phi + \frac{p}{\frac{\partial \Pi}{\partial \eta} RT} \left( \frac{\partial V}{\partial \eta} \right) - \frac{\partial p}{\frac{\partial \Pi}{\partial \eta} RT} \left( \frac{\partial w}{\partial \eta} \right)
\]
(2)
• \( d \) is the vertical divergence. In the thin layer equations, the relationship between \( d \) and the height-coordinate vertical velocity \( w \) is:
\[
d = - \frac{gp}{\frac{\partial \Pi}{\partial \eta} R_d T} \left( \frac{\partial w}{\partial \eta} \right)
\]
(3)
• Variable \( d_4 = d + \frac{p}{\frac{\partial \Pi}{\partial \eta} RT} \nabla \Phi \left( \frac{\partial X}{\partial \eta} \right) \) can be also used as prognostic variable.
• \( L \): number of layers of the model.
• \( A, B \) define hydrostatic pressure on the \( \eta \) levels ( \( \Pi = A + B \Pi_s \), where \( \Pi_s \) is the hydrostatic surface pressure).
• \( \beta \) coefficient for the semi-implicit scheme (between 0 and 1).
• \( \gamma, \tau, \nu, \mu, \partial^*, L^* \) and \( T^* \) are generic notations for linear operators (see subsection 4.2).
• \( H, C, N \) are intermediate constants used in the semi-implicit scheme of the non-hydrostatic model. Definitions are respectively:
\[
H = \frac{R_d T^*}{g}
\]
(4)
\[
C = \frac{R_d T^* c_{pd}}{c_{vd}}
\]
(5)
\[
N = \frac{g}{\sqrt{c_{pd} T^*}}
\]
(6)
• For a variable \( X \) defined at full levels, \( \langle X \rangle \) is the vector of coordinates \( \langle X_1; \ldots; X_l; \ldots; X_L \rangle \).
• \( R_{inte} \) is the vertical integration operator used in the case LVERTFE=1, T:
  - \( \int_{\eta=1}^{\eta=0} X d\eta \) is discretised by \([R_{inte}]_{(top,surf)} \langle X \rangle \).
  - \( \int_{\eta=0}^{\eta=1} X d\eta \) is discretised by \([R_{inte}]_{(top,t)} \langle X \rangle \).
  - \( \int_{\eta=1}^{\eta=1} X d\eta \) is discretised by \([R_{inte}]_{(t,surf)} \langle X \rangle \).
• \( R_{deri} \) is the vertical first-order derivative operator used in the case where VFE are also applied to derivatives.
3 General considerations.

3.1 Advection schemes.

* Explicit Eulerian equations: In Eulerian form of equations, the time dependency equation of a variable $X$ writes as:

$$\frac{\partial X}{\partial t} = -U \nabla_3 X + A + F$$  \hspace{1cm} (7)

where $U$ is the 3D wind, $\nabla_3$ is the 3D gradient operator, $A$ is the dynamical contribution, and $F$ is the physical contribution. $X(t + \Delta t)$ is computed knowing $X(t - \Delta t)$ at the same grid point.

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

$$\frac{dX}{dt} = A + F$$  \hspace{1cm} (8)

In a three-time level semi-Lagrangian scheme (abbreviated into 3TL SL 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 $t - \Delta t$. In a two-time level semi-Lagrangian scheme (abbreviated into 2TL SL 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 $t$.

3.2 Semi-implicit treatment of linear terms (case where there is no iterative centred-implicit scheme).

* Adding of a semi-implicit correction: In all cases the linear terms source of gravity waves must be treated implicitly, in order to allow time-steps compatible with an operational use of the model. Expression of the linear terms is obtained assuming a definition of a reference state. The reference state is defined by a dry resting isotherm atmosphere in hydrostatic balance, reference orography is zero. Equations (7) and (8) become respectively (9) and (10):

  - Eulerian scheme:

$$\frac{\partial X}{\partial t} = -U \nabla_3 X + A + F + [SI correction]$$  \hspace{1cm} (9)

  - Semi-Lagrangian scheme:

$$\frac{dX}{dt} = A + F + [SI correction]$$  \hspace{1cm} (10)

* Discretisation of equations (9) and (10): Equations (9) and (10) give the following discretized equations, where $\Delta t$ is the time step, $B$ is the linear term source of gravity waves, $\beta$ is a tunable parameter ($\beta = 0$ corresponds to an explicit formulation, $\beta = 1$ to an implicit formulation):
Semi-implicit spectral computations and predictor-corrector schemes

- Eulerian scheme:
  \[ [S I_{\text{correction}}] = -\beta B^t + \frac{\beta}{2} B^{t-\Delta t} + \frac{\beta}{2} B^{t+\Delta t} \]  
  (11)

- Three-time level semi-Lagrangian (3TL SL) scheme (without uncentering factor):
  \[ [S I_{\text{correction}}] = -\beta B^t + \frac{\beta}{2} B^{t-\Delta t} + \frac{\beta}{2} B^{t+\Delta t} \]  
  (13)

  \[ X^{t+\Delta t} = X^{t-\Delta t} + 2\Delta t(A + F) - 2\beta \Delta tB^t + \beta \Delta tB^{t-\Delta t} \]  
  (12)

- Two-time level semi-Lagrangian (2TL SL) scheme (without uncentering factor):
  \[ [S I_{\text{correction}}] = -\beta B^{t+0.5\Delta t} + \frac{\beta}{2} B^t + \frac{\beta}{2} B^{t+\Delta t} \]  
  (15)

  \[ X^{t+\Delta t} = X^{t-\Delta t} + 2\Delta t(A + F) - \beta \Delta tB^{t+0.5\Delta t} + 0.5\beta \Delta tB^t \]  
  (16)

where \( X^{t+\Delta t} - \beta \Delta tB^{t+\Delta t} \) is computed at the final grid point of the semi-Lagrangian trajectory, \( X^{t-\Delta t} \) is computed at the origin point of the semi-Lagrangian trajectory, \( -2\beta \Delta tB^t \) is computed as an average between the origin and final points of the trajectory, \( A \) is computed either at the medium point or as an average between the origin and final points of the trajectory. If there is a first-order uncentering factor \( \epsilon \) replace \( \Delta t \) by \((1 - \epsilon)\Delta t\) terms at the origin point, \( \Delta t \) by \((1 + \epsilon)\Delta t\) terms at the final point. For more details see documentation (IDSL) about the semi-Lagrangian scheme.

- Two-time level semi-Lagrangian (2TL SL) scheme (without uncentering factor):
  \[ X^{t+\Delta t} = X^{t-\Delta t} + 2\Delta t(A + F) - \beta \Delta tB^{t+0.5\Delta t} + 0.5\beta \Delta tB^t \]  
  (16)

where \( X^{t+\Delta t} = 0.5\beta \Delta tB^{t+\Delta t} \) is computed at the final grid point of the semi-Lagrangian trajectory, \( X^t \) and \( 0.5\beta \Delta tB^t \) are computed at the origin point of the semi-Lagrangian trajectory, \( -\beta \Delta tB^{t+0.5\Delta t} \) and \( A \) are computed either at the medium point or as an average between the origin and final points of the trajectory. If there is a first-order uncentering factor \( \epsilon \) replace \( \Delta t \) by \((1 - \epsilon)\Delta t\) terms at the origin point, \( \Delta t \) by \((1 + \epsilon)\Delta t\) terms at the final point. For more details see documentation (IDSL) about the semi-Lagrangian scheme.

\( B^{t+0.5\Delta t} \), \( B^t \) and \( B^{t-\Delta t} \) are computed in grid point space. The right-hand side members of equations (12), (14) and (16) are computed in grid point space, then transformed into spectral space. Entering spectral space a system of equations of the following type must be solved:

\[ X^{t+\Delta t} - \beta \Delta tB^{t+\Delta t} = X^* \]  
(17)

for a leap-frog scheme, and:

\[ X^{t+\Delta t} - 0.5\beta \Delta tB^{t+\Delta t} = X^* \]  
(18)

for a two-time level semi-Lagrangian scheme, where \( X^* \) is known and \( X^{t+\Delta t} \) is unknown. Now the spectral computations to solve this system of equations are described for a primitive equations 3D model, a 2D shallow water model and several NH 3D models.
3.3 Iterative centred-implicit schemes and combination with semi-implicit schemes.

**Purpose.**

In some cases (especially in the non-hydrostatic models), the model with a semi-implicit treatment of linear terms may remain unstable, hence a treatment by an iterative centred-implicit scheme may be necessary. In the following description one sticks to non-incremental formulations.

**Iterative centred-implicit scheme.**

* Algorithm: The total number of iterations is denoted by $N_{\text{iter}}$.

- The iteration number ($i = 0$) computes an estimation $X_{\text{iter}}^{t+\Delta t}$ of $X^{t+\Delta t}$ with a normal semi-implicit scheme. Horizontal diffusion can be done optionally at this stage.
- Iterations ($i > 0$): the $i$-th iteration ($i > 0$) computes $X_{\text{iter}}^{t+\Delta t}$ (after inversion of Helmholtz equation) knowing $X_{\text{iter}}^{t+\Delta t}$. Horizontal diffusion is always done at the last iteration, it can be done optionally at the other iterations. The final value of $X_{\text{iter}}^{t+\Delta t}$ is equal to $X_{\text{iter}}^{t+\Delta t}$.
- In the cycle 37T1 of ARPEGE/IFS and the cycle AL37T1 of ALADIN, this scheme is controlled by the key LPC\_FULL=. and is coded in the Eulerian scheme and the two-time level semi-Lagrangian scheme only. It is compulsory for the non-hydrostatic model with a SL2TL scheme to ensure stability (but not compulsory with the hydrostatic model with the two-time level semi-Lagrangian scheme).
- For unlagged physics, the physics has to be computed for the iteration ($i = 0$) only. For lagged physics, there are several possible options in test, but in practical the only one which works is:
  - adiabatic treatment of iterations 0 to $N_{\text{iter}} - 1$.
  - diabatic treatment of iteration $N_{\text{iter}}$.

* Discretisation of algorithm:

- First iteration ($i = 0$): One has to start from the discretisations of equations for a model with no iterative centred-implicit formulation (see documentations (IDEUL) and (IDSL)). For a leap-frog scheme the calculations are the same ones. For a two-time level semi-Lagrangian scheme, $(A - \beta \Delta t B)^{[i+0.5\Delta t]}$ is assumed to be equal to $(A - \beta \Delta t B)^{[i]}$ if no extrapolation is done (case LPC\_NESC=.T.), and to $1.5(A - \beta \Delta t B)^{[i]} - 0.5(A - \beta \Delta t B)^{[i-\Delta t]}$ if extrapolation is done (case LPC\_NESC=.F.). For a SL2TL case with no uncentering factor that yields the following discretisations (physics is assumed to be unlagged):
  - no extrapolation:
    \[
    (X_{\text{iter}}^{t+\Delta t})_{(i=0)}^{O} = \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} - 0.5\Delta t D]}{F} + \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} - 0.5\Delta t D]}{F} + \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} + 0.5\Delta t D]}{F} + \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} + 0.5\Delta t D]}{F}
    \]
  - for the origin and final points of the semi-Lagrangian trajectory, which can be rewritten:
    \[
    (X_{\text{iter}}^{t+\Delta t})_{(i=0)}^{O} = \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} - 0.5\Delta t D]}{F} + \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} + 0.5\Delta t D]}{F} + \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} - 0.5\Delta t D]}{F} + \frac{[X_{\text{iter}}^{t} + 0.5\Delta t A^{t} + 0.5\Delta t D]}{F}
    \]
extrapolation: discretisation is identical to the case with no iterative centred-
implicit scheme and conventional extrapolation (type \texttt{LSETTLS}=.F.); the
RHS terms other than \( X^t \) and \( F^t \) can be replaced by a “spatio-temporal”
average; see documentation (IDSL).

- Following iterations \((i > 0)\): The general iteration writes (no spatio-temporal
physics):

- Eulerian scheme (\(ADV\) stands for advection terms):
  \[
  X_{(i)}^{t + \Delta t} - \Delta t \beta B_{(i)}^{t + \Delta t} = X_{(i)}^{t - \Delta t} + 2 \Delta t ADV^t + [\Delta t A_{(i-1)}^{t + \Delta t} - \Delta t \beta B_{(i-1)}^{t + \Delta t}] + [\Delta t A_{(i-1)}^{t - \Delta t} - \Delta t \beta B_{(i-1)}^{t - \Delta t}] + \Delta t \beta B_{(i)}^{t - \Delta t} + 2 \Delta t F_{(i)}^{t - \Delta t}
  \]
  which can be rewritten:
  \[
  X_{(i)}^{t + \Delta t} - \Delta t \beta B_{(i)}^{t + \Delta t} = \Delta t A_{(i-1)}^{t + \Delta t} - \Delta t \beta B_{(i-1)}^{t + \Delta t} + [X_{(i)}^{t - \Delta t} + 2 \Delta t ADV^t + \Delta t A_{(i-1)}^{t - \Delta t} + 2 \Delta t F_{(i)}^{t - \Delta t}]
  \]

- Three-time level semi-Lagrangian scheme (without uncentering factor):
  \[
  [X_{(i)}^{t + \Delta t} - \Delta t \beta B_{(i)}^{t + \Delta t}]_F = X_{(i)}^{t - \Delta t} + [\Delta t A_{(i-1)}^{t + \Delta t} - \Delta t \beta B_{(i-1)}^{t + \Delta t}]_F + [\Delta t A_{(i-1)}^{t - \Delta t} - \Delta t \beta B_{(i-1)}^{t - \Delta t}]_O(t) + [\Delta t \beta B_{(i-1)}^{t - \Delta t} + 2 \Delta t F_{(i-1)}^{t - \Delta t}]_O(t)
  \]
  The iterative centred-implicit algorithm also applies to re-compute the semi-
Lagrangian trajectory (see documentation (IDSL) for more details), the po-

tion of the origin point at the \(i\)-th (resp \(i - 1\)-th) iteration is \(O(i)\) (resp.
\(O(i - 1)\)).

- Two-time level semi-Lagrangian scheme (without uncentering factor):
  \[
  [X_{(i)}^{t + \Delta t} - \Delta t \beta B_{(i)}^{t + \Delta t}]_F = X_{(i)}^{t} + [0.5 \Delta t A_{(i-1)}^{t + \Delta t} - 0.5 \Delta t \beta B_{(i-1)}^{t + \Delta t}]_F + [0.5 \Delta t A_{(i-1)}^{t} - 0.5 \Delta t \beta B_{(i-1)}^{t}]_O(t) + [0.5 \Delta t \beta B_{(i-1)}^{t} + \Delta t F_{(i-1)}^{t}]_O(t)
  \]
  which can be rewritten:
  \[
  [X_{(i)}^{t + \Delta t} - \Delta t \beta B_{(i)}^{t + \Delta t}]_F = X_{(i)}^{t} + [0.5 \Delta t A_{(i-1)}^{t + \Delta t} - 0.5 \Delta t \beta B_{(i-1)}^{t + \Delta t}]_F + [0.5 \Delta t A_{(i-1)}^{t} + \Delta t F_{(i-1)}^{t}]_O(t)
  \]
  The iterative centred-implicit algorithm also applies to re-compute the semi-
Lagrangian trajectory (see documentation (IDSL) for more details), the po-

tion of the origin point at the \(i\)-th (resp \(i - 1\)-th) iteration is \(O(i)\) (resp.
\(O(i - 1)\)). Remark: when the iterative centred-implicit scheme is used, the
stable extrapolation \texttt{LSETTLS}=.T. is never involved.

\* \textbf{Cheap version of this algorithm:} In a semi-Lagrangian scheme, it is possible
not to iterate the position of the origin point \(O\) (i.e \(O(i) = O(i-1)\)):
this cheap version is activated if \texttt{LPC\_CHEAP}=.T. . It is coded only for a non-extrapolating SL2TL
scheme. In this case the quantity to be interpolated (i.e. \(0.5 \Delta t A_{(i-1)}^{t} + \Delta t F_{(i-1)}^{t}\)) needs to
be interpolated at the predictor step only. It is then stored in a buffer and re-used at
the corrector steps without any interpolation.

\section*{3.4 Introduction of uncentering for semi-Lagrangian schemes}

Averages along the semi-Lagrangian trajectory will be weighted by \((1 - \epsilon)\) at the origin
point and \((1 + \epsilon)\) at the final point. If the uncentering coefficient \(\epsilon\) is horizontally constant
the algorithms remain valid, replacing \(\beta\) by \((1 + \epsilon)\).
5. Semi-implicit spectral computations and predictor-corrector schemes

3.5 Plane geometry (ALADIN).

Particular features of ALADIN when differing from ARPEGE/IFS are not described in detail, only brief comments are mentioned. For more details see ALADIN documentation. Concerning the semi-implicit and iterative centred-implicit algorithms one can consider that the major part of this documentation is still valid for ALADIN; the main differences with ARPEGE/IFS are:

- The shallow-water model is not coded in ALADIN.
- Option \texttt{LESIDG}=T. replaces \texttt{LSIDG}=T. in ALADIN, and is available only with the tilted-rotated Mercator projection. This option is useful only when the horizontal variations of the mapping factor are significant. See (IDESIDG) about its implementation.
- Option \texttt{LIMPF}=T. is not coded in ALADIN.
- In the hydrostatic (resp. NH) model, spectral part of the semi-implicit scheme is performed in routine \texttt{ESPCSI} instead of \texttt{SPCSI} (resp. \texttt{SPNHSI} instead of \texttt{SPNHSI}) in ALADIN; the algorithm is the same as in ARPEGE but the truncation of the spectral representation is elliptic and not triangular.

3.6 Finite elements on the vertical.

The option with finite element vertical discretisations is coded for the hydrostatic model and partly for the NH models. For VFE, the main modifications in the semi-implicit scheme are the following ones:

- The discretisation of $\Pi$, $\alpha$ and $\delta$ at full levels is different.
- The model avoids as possible to compute quantities at half levels; all vertical integrals directly provide quantities at full levels.
- The vertical integrals contained in some linear operators ($\gamma$, $\tau$ and $\nu$) are discretised differently, as a matricial multiplication with special coefficients (contained in the matrix $R_{\text{inte}}$) computed in the setup routine \texttt{SUVERTFE1} or \texttt{SUVERTFE3}; the vertical integration is done by routine \texttt{VERINT}.
- In NH models there are also vertical derivatives with a specific treatment. The NH-GEOGW model is coded only with VFE (for integrals and derivatives) because this is the only way to keep all the prognostic variables at full levels. The spectral part of the semi-implicit scheme is not designed to mix half level and full level variables.
- In the NH-PDVD model, it is possible to switch on the VFE only in the explicit model, and to keep the finite differences vertical discretisation in the linear model (that allows to ensure constraint C1, see below).
4 Prognostic variables and quantities involved in the semi-implicit scheme.

4.1 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 the components of the horizontal wind $V$, temperature $T$, and the two additional non-hydrostatic variables in a non-hydrostatic model. Details about equations of the NH variables in the NH-PDVD model (choice of the two additional prognostic variables, discretisations, linearisation for semi-implicit scheme) can be found for example in (IDNH2.1).

- **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”). This class of variables includes for example humidity $q$, liquid water, ice, cloud fraction, ozone, and some extra fields.

- **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).

Only the GMV and GMVS variables appear in the semi-implicit scheme. In the shallow-water 2D model, only GMV variables exist, this class of variables includes the components of the horizontal wind $V$, and the equivalent height $\Phi - \Phi_s$ (continuity equation).

4.2 Quantities used for vertical discretisations and linear operators.

* **Operators “alpha” and “delta”:** these operators are used for discretisations of some vertical integrals. They have a different expression according to the value of variables NDLNPR, LVERTFE.

- For $LVERTFE=.F., NDLNPR=0$ and a layer $l$ between 2 and $L$ (and also $l = 1$ if the pressure at the top of the model is not zero), $\alpha^*$ and $\delta^*$ are discretised as follows at full levels:

\[
\alpha^*_l = 1 - \frac{\Pi^*_l}{\Delta \Pi^*_l} \log \left( \frac{\Pi^*_l}{\Pi^*_l} \right) \tag{19}
\]

\[
\delta^*_l = \log \left( \frac{\Pi^*_l}{\Pi^*_l} \right) \tag{20}
\]

- For $LVERTFE=.F., NDLNPR=0$ and the layer $l = 1$ if the pressure at the top of the model is zero:

\[
\alpha^*_{l=1} = 1 \text{ at METEO-FRANCE.}
\]
5. Semi-implicit spectral computations and predictor-corrector schemes

- $\alpha^*_{i=1} = \log(2)$ at ECMWF.
- $\delta^*_{i=1}$ has in theory an infinite value, but in the code it is computed with a top pressure equal to 0.1 Pa to provide a finite value.

For LVERTFE=$\text{F.}$, NDLNPR=$\text{1}$ and a layer $l$ between 2 and $L$ (and also $l = 1$ if the pressure at the top of the model is not zero), $\alpha^*$ and $\delta^*$ are discretised as follows at full levels:

$$\alpha^*_l = 1 - \sqrt{\frac{\Pi^*_l - \Pi^*_{l-1}}{\Pi^*_l}} = 1 - \frac{\Pi^*_{l-1}}{\Pi^*_l}$$

$$\delta^*_l = \frac{\Delta \Pi^*_l}{\Pi^*_l} = \frac{\Delta \Pi^*_l}{\sqrt{\Pi^*_l - \Pi^*_{l-1}}}$$

$\Pi^*$ is discretised as follows at full levels:

$$\Pi^*_{l} = \sqrt{\Pi^*_l \Pi^*_{l-1}}$$

Notation $\alpha^*_{i=1}$ is used for quantity $1 - \frac{\Pi^*_{l-1}}{\Pi^*_l}$ instead of notation $\beta^*_{i=1}$ of (Bubnová et al., 1995).

For LVERTFE=$\text{F.}$, NDLNPR=$\text{1}$ and the layer $l = 1$ if the pressure at the top of the model is zero:

- $\alpha^*_{i=1} = 1$ and $\alpha^*_{i=0} = 1$.
- $\delta^*_{i=1} = 1 + \frac{cP_d}{\kappa_d}$
- $\Pi^*_{i=1} = \frac{\Delta \Pi^*_{i=1}}{\kappa_{i=1}}$.

For LVERTFE=$\text{T.}$, NDLNPR=$\text{0}$ and a layer $l$ between 1 and $L$, $\alpha^*$ and $\delta^*$ are discretised as follows at full levels:

$$\alpha^*_l = \frac{\Pi^*_l - \Pi^*_i}{\Pi^*_l}$$

$$\delta^*_l = \frac{\Delta \Pi^*_l}{\Pi^*_l}$$

where $\Pi^*_l = A_l + B_l \Pi^*_0$. See documentation (IDEUL) for computation of $A_l$ and $B_l$ in this case. Formulae (24) and (25) provide finite values of $\alpha^*_l$ and $\delta^*_l$ even if the pressure at the top of the model is zero. $\alpha^*_1$ is not used in the SI scheme in this case because vertical integrals are directly provided at full levels without the intermediate state of interlayer data.

* Linear operator “$\gamma$”: this operator is applied to temperature and pressure departure variable to compute linear term in momentum equation. Denotation $\gamma$ has the same meaning as the denotation $R_d \ast \mathbf{G}^*$ of (Bubnová et al., 1995) or (IDNH2.1). For a variable $Z$, $(\gamma Z)$ is a discretisation of vertical integral

$$\int_\eta^1 \left( \frac{\partial \Pi^*_l}{\Pi^*_l} \right) R_d Z d\eta$$

Expression of this discretisation is:
5. Semi-implicit spectral computations and predictor-corrector schemes

- Case : \( \text{LVERTFE} = \text{T.} \):
  \[
  (\gamma Z)_l = [R_{\text{inte}}]_{(l, \text{surf})} \left\langle \frac{R_d Z \delta^\ast}{\Delta \eta} \right\rangle
  \]  
  \( (26) \)

- Case : \( \text{LVERTFE} = \text{F.} \):
  \[
  (\gamma Z)_l = \alpha^\ast_l R_d Z_l + \sum_{k=l+1}^{L} R_d Z_k \delta^\ast_k
  \]  
  \( (27) \)

Remark: if \( \text{LSPRT} = \text{T.} \), \( \gamma \) is applied to virtual temperature instead of real temperature.

* Linear operator “\( \tau \)”: this operator is applied to divergence to compute linear term in temperature equation. Denotation \( \tau \) has the same meaning as the denotation \( \frac{R_d T^*}{c_p d} \ast \Pi^* \) of (Bubnová et al., 1995) and (IDNH2.1). For a variable \( Z \), \( (\tau Z) \) is the discretisation of vertical integral \( \frac{R_d T^*}{c_p d} \int_0^1 \frac{\partial \Pi^*}{\partial \eta} Z d\eta \). Expression of \( (\tau Z) \) is:

- Case : \( \text{LVERTFE} = \text{T.} \):
  \[
  (\tau Z)_l = \frac{R_d T^*}{c_p d} \left[ \Delta \Pi^* \right]_{(l, \text{top})} \left\langle \frac{\Delta \Pi^* Z}{\Delta \eta} \right\rangle
  \]  
  \( (28) \)

Remark: according to the expression of \( \delta^\ast \) in this case, this equation can be rewritten:
  \[
  (\tau Z)_l = \frac{R_d T^*}{c_p d} \frac{1}{\Pi^*_l} \left[ R_{\text{inte}} \right]_{(\text{top}, l)} \left\langle \frac{\Delta \Pi^* Z}{\Delta \eta} \right\rangle
  \]

- Case : \( \text{LVERTFE} = \text{F.} \):
  \[
  (\tau Z)_l = \frac{R_d T^*}{c_p d} \left[ \alpha^\ast_l Z_l + \sum_{k=1}^{l-1} \Delta \Pi^*_k Z_k \right]
  \]  
  \( (29) \)

Remark: if \( \text{LSPRT} = \text{T.} \), \( R_d T^* \) becomes \( \frac{R_d^2}{\Pi^*_c} T^* \) in expression of \( (\tau Z) \).

* Linear operator “\( \nu \)”: this operator is applied to divergence to compute linear term in continuity equation. Denotation \( \nu \) has the same meaning as the denotation \( \Pi^* N^* \) of (Bubnová et al., 1995) and (IDNH2.1). For a variable \( Z \), definition of \( (\nu Z) \) is:

- Case : \( \text{LVERTFE} = \text{T.} \):
  \[
  (\nu Z) = \frac{1}{\Pi^*_c} \left[ R_{\text{inte}} \right]_{(\text{top}, \text{surf})} \left\langle \frac{\Delta \Pi^* Z}{\Delta \eta} \right\rangle
  \]  
  \( (30) \)

- Case : \( \text{LVERTFE} = \text{F.} \):
  \[
  (\nu Z) = \frac{1}{\Pi^*_c} \sum_{l=1}^{L} \Delta \Pi^*_l Z_l
  \]  
  \( (31) \)

\( (\nu Z) \) is the discretisation of vertical integral \( \frac{1}{\Pi^*_c} \int_0^1 \frac{\partial \Pi^*}{\partial \eta} Z d\eta \).
5. Semi-implicit spectral computations and predictor-corrector schemes

* Linear operator “μ”: This operator is applied to log(Πₙ) to compute linear term in momentum equation. For a variable Z, Definition of (μZ) is:

\[(μZ) = R₉T^∗Z\]  \hspace{1cm} (32)

(μZ) is applied to log(Πₙ).

Remarks:

• if LSPRT = .T., \(R₉T^∗\) becomes \(\frac{p_{n}^{2}}{F}T^∗\) in expression of (μZ).
• In some documentations one sometimes finds the expanded expression \(R₉T^∗I\) instead of μ (where I is the identity matrix).

* Linear operator “\(\partial^∗\)" for non-hydrostatic model: this operator is the following first-order derivative:

\[\partial^∗Z = \frac{∂Z}{∂logΠ^∗}\] \hspace{1cm} (33)

This operator is currently only used in the VFE NH-GEOGW scheme, discretisation involves the operator \([R^*_\text{deri}]\). Actually \(\partial^∗\) and \(\partial^∗ + 1\) appear, and we cannot exclude a priori the possibility to use two slightly different first order derivative operators, in order to ensure good properties for the product \(\partial^∗(\partial^∗ + 1)\) (see the following paragraph).

* Linear operator “\(L^∗\)" for non-hydrostatic model: this operator, called Laplacian operator, is applied to the pressure departure variable to compute linear term in the vertical divergence equation (NH-PDVD model). It also appears in the NH-GEOGW scheme. \((L^∗Z)\) is the vertical double derivative \[Π^∗\frac{∂}{∂Π^∗}(Π^∗Z)\]; this is equivalent to write \(L^∗ = \partial^∗(\partial^∗ + 1)\).

In the linear NH-PDVD model, only \(L^∗Z\) appears, but not the separate first-order operators \(\partial^∗\) and \(\partial^∗ + 1\). This linear operator comes from the RHS of the vertical divergence equation, where there is a first vertical derivative to provide the RHS of the gw equation, then a second vertical derivative to provide the RHS of the divergence equation. It must have the following discretisation to ensure stability and that imposes some constraints to discretize the vertical derivatives which are present in the RHS of the gw and vertical divergence equations. For FD discretisation, the stability is ensured at least when the top hydrostatic pressure is zero but is not guaranteed if the top hydrostatic pressure is not zero. For a more detailed discussion about the vertical discretisation required, see chapter 4 of (IDNH2.1), where the constraint giving this vertical discretisation is denoted by “C2", and in particular the part 4.3.2.

FD discretisation of \((L^∗Z)\) (NH-PDVD model only):

• Case LVERTFE = .F.:
  - Layers 2 to \(L - 1\):

\[(L^∗Z)_{l} = A^*_{l}Z_{l-1} + B^*_{l}Z_{l} + C^*_{l}Z_{l+1}\] \hspace{1cm} (34)

Expressions of \(A^*\), \(B^*\) and \(C^*\) are:

\[A^*_{l} = \frac{Π^*_{l-1}}{δ^*_{l}(Π^*_{l} - Π^*_{l-1})}\] \hspace{1cm} (35)
\( B_i^* = -\frac{1}{\delta_i^*} \left( \frac{\Pi_i^*}{\Pi_i^* - \Pi_{i-1}^*} + \frac{\Pi_i^*}{\Pi_{i+1}^* - \Pi_i^*} \right) \)  \( \quad \) (36)

\( C_i^* = \frac{\Pi_{i+1}^*}{\delta_i^* (\Pi_{i+1}^* - \Pi_i^*)} \)  \( \quad \) (37)

Note that equation (34) can be rewritten:

\( (L^* Z)_l = A_i^*(Z_{l-1} - Z_l) + C_i^*(Z_{l+1} - Z_l) \)  \( \quad \) (38)

– Layer 1: The quantity \( Z \) to which is applied \( L^* \) is assumed to be zero at the top of the model, that means that the term \( A_i^*(Z_0 - Z_1) \) has to be replaced by \(-A_i^* Z_1\). In practical, \( A_i^* \) has to be set to zero if the top hydrostatic pressure is zero, and the general formula valid for any non-zero top pressure is:

\( A_i^* = \frac{\Pi_{\text{top}}^*}{\delta_i^* (\Pi_i^* - \Pi_{\text{top}}^*)} \)  \( \quad \) (39)

\( C_i^* \) matches the general expression:

\( C_i^* = \frac{\Pi_2^*}{\delta_i^* (\Pi_2^* - \Pi_i^*)} \)  \( \quad \) (40)

and, if the top hydrostatic pressure is zero:

\( B_i^* = -C_i^* \)  \( \quad \) (41)

This upper condition is stable at least when the top hydrostatic pressure is zero.

That leads to the following formula for \((L^* Z)_1\):

\( (L^* Z)_1 = -A_1^* Z_1 + C_1^*(Z_2 - Z_1) \)  \( \quad \) (42)

– Layer \( L \): The quantity \( Z \) to which is applied \( L^* \) is assumed to be constant below the full level \( l = L \), that means that the term \( C_L^*(Z_{L+1} - Z_L) \) has to be replaced by 0. In practical, \( C_L^* \) is set to zero. \( A_L^* \) matches the general expression:

\( A_L^* = \frac{\Pi_{L-1}^*}{\delta_L^* (\Pi_L^* - \Pi_{L-1}^*)} \)  \( \quad \) (43)

Application of formula (38) would lead to \( B_L^* = -A_L^* \) but we actually use the formula (34) with a slightly different expression for \( B_L^* \):

\( B_L^* = -\frac{\Pi_L^*}{\delta_L^* (\Pi_L^* - \Pi_{L-1}^*)} \)  \( \quad \) (44)

That leads to the following formula for \((L^* Z)_L\):

\( (L^* Z)_L = A_L^*(Z_{L-1} - Z_L) - A_L^* \left( \frac{\Pi_L^*}{\Pi_{L-1}^*} - 1 \right) Z_L \)  \( \quad \) (45)

VFE discretisation of \((L^* Z)\) in the NH-PDVD model: it is not computed as the product of two first-order derivatives, but uses a second-order operator \([R_{\text{dderi}}]\), the eigenvalues of which should have some properties. Some studies are still required to provide a better \([R_{\text{dderi}}]\) than the one which is currently coded (which does not give sufficient stability).

VFE discretisation of \((L^* Z)\) in the NH-GEOGW model: it is currently (provisionally) computed as the product of two first-order derivatives \((L^* = \partial^* (\partial^* + 1))\).
5. Semi-implicit spectral computations and predictor-corrector schemes

* Linear operator “$T^*$” for NH-PDVD model: This is a combination of linear operators $L^*$, $τ$ and $γ$. Definition of $T^*$ is:

$$T^* = \frac{g^2 L^* \left( \frac{c_{pd}^2}{\tau^2} \tau γ - \frac{c_{pd}}{\tau} \frac{c_{pd}^2}{\tau} τ - \frac{c_{pd}}{\tau} γ \right)}{R_0 N^2 C^2}$$  \tag{46}$$

Using definitions for $C$ and $H$ (see equations (5) and (4)) this equation can be rewritten:

$$T^* = \frac{L^* T^* \left( c_{pd} τ γ - \frac{c_{pd}}{T^*} C^2 τ - C^2 γ \right)}{H^2 N^2 C^2}$$  \tag{47}$$

or

$$T^* = \frac{L^* \left( T^* c_{pd} τ γ - \frac{c_{pd}}{T^*} C^2 τ - C^2 T^* γ \right)}{H^2 N^2 C^2}$$  \tag{48}$$

If a FD vertical discretisation is applied to $L^*$, $T^*$ is a tridiagonal operator, like $L^*$.

If FD vertical discretisation, expression of the elements of the associated matrix is:

$$T^*_{(l,l)} = 1 - \frac{1}{δ^*_l} \left( \frac{Π^*_l}{Π^*_l - Π^*_{l-1}} + \frac{Π^*_l}{Π^*_l + 1 - Π^*_l} \right) (δ^*_l - 2α^*_l) \tag{49}$$

$$T^*_{(1,1)} = 1 \tag{50}$$

$$T^*_{(l,l-1)} = \frac{1}{δ^*_l} \left( \frac{Π^*_l}{Π^*_l - Π^*_{l-1}} \right) (δ^*_{l-1} - 2α^*_{l-1}) \tag{51}$$

$$T^*_{(l,l+1)} = \frac{1}{δ^*_l} \left( \frac{Π^*_l}{Π^*_l + 1 - Π^*_l} \right) (δ^*_{l+1} - 2α^*_{l+1}) \tag{52}$$

with some particular expressions at the top and the bottom.

If one denotes by $Q^*$ the linear diagonal operator, the diagonal elements of which are defined by:

$$Q^*_{(l,l)} = (δ^*_l - 2α^*_l) \tag{53}$$

and, if the top hydrostatic pressure is zero:

$$Q^*_{(1,1)} = 0 \tag{54}$$

thus $T^*$ can be rewritten:

$$T^* = (I + L^* Q^*) \tag{55}$$
4.3 Relationships between some linear operators.

In the NH-PDVD model, operators $\nu$, $\gamma$ and $\tau$ SHOULD be linked by the following relationship if one wants the complete elimination of variables in order to provide a "one variable" Helmholtz equation computing the vertical divergence, and all the following parts of this documentation assumes that this condition is fulfilled. For more details see section 4 of (IDNH2.1) where this condition is denoted by "C1". The adimensioned formulation of constraint "C1" writes:

$$\nu = \frac{1}{R_d} \gamma + \frac{c_{pd}}{R_dT^*} \gamma - \frac{c_{pd}}{R_d^2T^*} \tau$$

(56)

This equation can be multiplied by $\frac{c_{pd}}{c_{vd}}$ and rewritten as follows:

$$COR = 0$$

(57)

where:

$$COR = \frac{c_{vd}}{R_d} \gamma - \frac{c_{vd}}{R_d} \gamma - \frac{c_{vd}}{R_dT^*} \tau + \frac{c_{vd}}{c_{pd}} \nu$$

(58)

Note that this condition is fulfilled at least for LVERTFE=.F. (FD discretisation of operators $\nu$, $\gamma$ and $\tau$) and NDLNPR=1 (this is the reason of using the NDLNPR=1 discretisation in a NH model), and is SURELY NOT fulfilled for LVERTFE=.F. and NDLNPR=0.

For LVERTFE=.T., this condition is not fulfilled. This is the reason why there is a possibility to keep LVERTFE=.F. and NDLNPR=1 in the linear model, even if VFE are switched on in the explicit model.

Additional studies (see for example Bénard, 2004) show that, when the constraint C1 is not fulfilled, it is not possible to fulfill it simply by changing the definition of one of the operators $\nu$, $\gamma$ or $\tau$.

- Changing the definition of $\nu$ to match formula (56) does not ensure any longer the fact that $\nu$ is a 2D operator.
- Changing the definition of $\gamma$ to match $COR = 0$ does not work because that provides an expression with the inverse of a non-invertible matrix.
- Changing the definition of $\tau$ to match $COR = 0$ has none of the shortcomings of the two previous proposals, but provides a linear operator generating spurious noise.
5 Semi-implicit scheme, no Coriolis term in the semi-implicit scheme.

Equations are written for a leap-frog scheme (Eulerian scheme of three-time level semi-Lagrangian scheme). For a two-time level semi-Lagrangian scheme replace $\Delta t$ by $0.5\Delta t$.

5.1 3D hydrostatic primitive equations model.

* Expression of the linear term $B$ for GMV and GMVS variables:
  
  - Continuity equation ($X = \log(\Pi_s)$):
    \[
    B = -\nu(M^2 D')
    \]
    (59)
  
  - Divergence equation ($X = D'$):
    \[
    B = -\nabla'^2(\gamma T + \mu \log \Pi_s)
    \]
    (60)
  
  - Vorticity equation ($X = \zeta'$):
    \[
    B = 0
    \]
    (61)
  
  - Temperature equation ($X = T$):
    \[
    B = -\tau(M^2 D')
    \]
    (62)

* System to be solved: Equations are written for $\log(\Pi_s)$ as a prognostic variable for continuity equation.

  \[
  \log(\Pi_s)_{t+\Delta t} + \beta \Delta t M^2 \nu D'_{t+\Delta t} = P^*
  \]
  (63)

  \[
  D'_{t+\Delta t} + \beta \Delta t \nabla'^2(\gamma T_{t+\Delta t} + \mu \log(\Pi_s)_{t+\Delta t}) = D'^*
  \]
  (64)

  \[
  T_{t+\Delta t} + \beta \Delta t M^2 \tau D'_{t+\Delta t} = T^*
  \]
  (65)

$P^*$, $D'^*$, $T^*$ correspond to $X^*$ defined in equation (17) and are available in spectral arrays (SPSP, SPDIV, SPT) at the beginning of the spectral computations. Equations (63) to (65) yield (66) (Helmholtz equation):

  \[
  (I - \beta^2 \Delta t^2 B \nabla'^2 M^2)D'_{t+\Delta t} = D'^* - \beta \Delta t \nabla'^2(\gamma T^* + \mu P^*)
  \]
  (66)

where $B = \gamma \tau + \mu \nu$ is a matricial operator $L \ast L$ (precomputed in routines SUDYN, SUBMAT and stored in the array SIB).

When $M = M$ (cases LSIDG=?, T or LESIDG=?, T) it is more convenient to rewrite equation (66) as:

  \[
  (\nabla'^{-2} - \beta^2 \Delta t^2 B M^2)D'_{t+\Delta t} = \nabla'^{-2}D'^* - \beta \Delta t(\gamma T^* + \mu P^*)
  \]
  (67)

which shows a symmetric matricial operator in the left hand side.
5. Semi-implicit spectral computations and predictor-corrector schemes

* Spectral computations to solve system of equations (63) to (65).
Algorithm works zonal wave number by zonal wave number \( m \) for a given zonal wave number \( m \) and the truncation \( N_m \).
- Inversion of Helmholtz equation for the case “reduced divergence” (case \( \text{LESIDG} = \text{F.} \) and \( \text{LESIDG} = \text{F.} \)) and method via a diagonalisation in the eigenmodes space.
  - First the diagonalisation of \( B \) is used: \( B = Q^{-1} A Q \), where \( A \) is a diagonal \( L \times L \) matrix, the diagonal coefficients \( a_I \) of which are stored in the array \( \text{SIVP} \). \( Q \) is a \( L \times L \) matrix stored in the array \( \text{SIMI} \). \( Q^{-1} \) is stored in the array \( \text{SIMI} \). Note that the vertical operators \( \nu, \mu, \tau, \gamma \), \( B \), \( Q \) commute with the horizontal operator \( \nabla^2 \).
  - Helmholtz equation (64) becomes, for each eigenmode \( l \):
    \[
    (1 - \beta^2 \Delta t^2 a_{l} \nabla^2 M^2) Q D_{l+\Delta t} = Q(D^* - \beta \Delta t \nabla^2 (\gamma T^* + \mu P^*))
    \]
    (68)
    - For each eigenmode \( l \) and each zonal wave number \( m \): \( (1 - \beta^2 \Delta t^2 a_{l} \nabla^2 M^2) \) is a diagonal matricial operator \((N_m + 1 - | m | \) \times \((N_m + 1 - | m |)\): spectral coefficients of the right-hand side member of \( (68) \) are simply divided by the diagonal coefficients of this matrix. The result is then multiplied by \( Q \).
- Inversion of Helmholtz equation for the case “unreduced divergence” (case \( \text{LESIDG} = \text{T.} \) in ARPEGE, \( \text{LESIDG} = \text{T.} \) in ALADIN): in this case \( M^2 \) is symmetric matricial operator containing values of the divergence for all levels and five total wave numbers \((n - 2 \) to \( n + 2 \)).
  - First the diagonalisation of \( B \) is used: \( B = Q^{-1} A Q \), where \( A \) is a diagonal \( L \times L \) matrix, the diagonal coefficients \( a_I \) of which are stored in the array \( \text{SIVP} \). \( Q \) is a \( L \times L \) matrix stored in the array \( \text{SIMI} \). \( Q^{-1} \) is stored in the array \( \text{SIMI} \). Note that the vertical operators \( \nu, \mu, \tau, \gamma \), \( B \), \( Q \) commute with the horizontal operators \( \nabla^2 \) and \( M^2 \).
  - Helmholtz equation (64) (resp. (67)) becomes, for each eigenmode \( l \):
    \[
    (1 - \beta^2 \Delta t^2 a_{l} \nabla^2 M^2) Q D_{l+\Delta t} = Q(D^* - \beta \Delta t \nabla^2 (\gamma T^* + \mu P^*))
    \]
    (69)
    resp.:
    \[
    (\nabla^2 - \beta^2 \Delta t^2 a_{l} M^2) Q D_{l+\Delta t} = Q(\nabla^2 D^* - \beta \Delta t (\gamma T^* + \mu P^*))
    \]
    (70)
    - For each eigenmode \( l \) and each zonal wave number \( m \): \((\nabla^2 - \beta^2 \Delta t^2 a_{l} M^2) \) is a symmetric pentadiagonal matricial operator \((N_m + 1 - | m | \times (N_m + 1 - | m |)\).
  - The factorisation \( LU \) of this matrix is computed, where \( L \) is a lower triangular tridiagonal matrix, \( U \) is an upper triangular tridiagonal matrix. All useful coefficients of \( L \), \( U \) are computed in the set-up routine \( \text{SUHIEG} \) and stored in the array \( \text{SUHIEG} \).
  - The right-hand side member of (70) is computed, then multiplied by the inverse of the symmetric pentadiagonal operator \((\nabla^2 - \beta^2 \Delta t^2 M^2 a_{l}) \) (resolution of two tridiagonal triangular systems by routine \( \text{MXTURS} \)). That yields \( Q L D_{l+\Delta t}^* \). Multiplying by \( Q^{-1} \) one obtains \( D_{l+\Delta t}^* \).
5. Semi-implicit spectral computations and predictor-corrector schemes

- For the zonal wave number \( m = 0 \) equation (69) is used rather than (70) because, for the total wave number \( n = 0 \), \( \nabla'^2 \) is equivalent to a multiplication by 0 and \( \nabla'^{-2} \) is equivalent to a division by 0. The only difference is that the pentadiagonal but non-symmetric operator \((I - \beta^2 \Delta t^2 a_0 \nabla'^2 M^2)\) is factorised and inverted. All useful coefficients of L, U are computed in the set-up routine SUHEG and stored in the arrays SIHEG and SIHEG2.

- Once known \( \dot{D}'_{t+\Delta t} \) equation (63) provides \( \log(\Pi_t)_{t+\Delta t} \) and equation (65) provides \( T'_{t+\Delta t} \). For the case \( \text{LSIDG} = \text{T.} \) only (resp. \( \text{LESIDG} = \text{T.} \) in ALADIN), spectral multiplications by \( M^2 \) are performed by the product of a symmetric pentadiagonal matrix of dimensions \((N_s + 1 - |m|) \times (N_s + 1 - |m|)\) (useful coefficients computed in routine SUSMAP (resp. SUESMAP in ALADIN) and stored in the array SCGMAP (resp. ESCGMAP in ALADIN)) by a vector containing spectral coefficients \((m,n)\) for \( n \) varying from \(|m|\) to \( N_s \).

- Semi-implicit scheme ends by a final memory transfer.

5.2 3D NH-PDVD model.

The code described is valid for options \( \text{NPDVAR=2} \) and \( \text{NVDVAR=3} \). For \( \text{NPDVAR=4} \) the linear terms are the same as for \( \text{NPDVAR=3} \).

* Expression of the linear term \( B \) for GMV and GMVS variables:

- Continuity equation \((X = \log(\Pi_t))\):
  \[ B = -\nu(M^2 D') \] (71)

- Divergence equation \((X = D')\):
  \[ B = -\nabla'^2[\gamma T - T^*(\gamma \hat{Q}) + \mu \log(\Pi_t) + R_d T^* \hat{Q}] \] (72)

- Vorticity equation \((X = \zeta')\):
  \[ B = 0 \] (73)

- Temperature equation \((X = T)\):
  \[ B = -\frac{R_d T^*}{c_{e,d}}[M^2 D' + d] \] (74)

- Pressure departure variable equation \((X = \hat{Q})\):
  \[ B = -\left[\frac{c_{p,d}}{c_{v,d}}(M^2 D' + d) - \frac{c_{p,d}}{R_d T^*}(M^2 D')\right] \] (75)

- Vertical divergence equation \((X = d)\):
  \[ B = -\frac{g^2}{R_d T^*}(L^* \hat{Q}) \] (76)

* System to be solved: \( \) Equations are written for \( \log(\Pi_t) \) as a prognostic variable for continuity equation.

\[ \log(\Pi_{t})_{t+\Delta t} + \beta \Delta t \nu(M^2 \dot{D}'_{t+\Delta t}) = P^\ast \] (77)

\[ \dot{D}'_{t+\Delta t} + \beta \Delta t \nabla'^2[\gamma T_{t+\Delta t} - T^*(\gamma \dot{Q}_{t+\Delta t}) + \mu \log(\Pi_{t})_{t+\Delta t} + R_d T^* \dot{Q}_{t+\Delta t}] = D'^\ast \] (78)
5. Semi-implicit spectral computations and predictor-corrector schemes

\[
\dot{Q}_{t+\Delta t} + \beta \Delta t \left[ \frac{c_{pd}}{c_{vd}} \left( \frac{M^2}{R_dT_s} D'_{t+\Delta t} + d_{t+\Delta t} \right) - \frac{c_{pd}}{R_dT_s} \gamma \left( \frac{M^2}{R_dT_s} \dot{D}'_{t+\Delta t} \right) \right] = \dot{Q}^* \tag{79}
\]

\[
d_{t+\Delta t} + \beta \Delta t \frac{\beta^2}{R_dT_s} (L^* \dot{Q}_{t+\Delta t}) = \tilde{D}^* \tag{80}
\]

\[
T_{t+\Delta t} + \beta \Delta t \frac{R_dT_s}{c_{vd}} \left( \frac{M^2}{R_dT_s} D'_{t+\Delta t} + d_{t+\Delta t} \right) = T^* \tag{81}
\]

$P^*$, $\tilde{D}^*$, $Q^*$, $\tilde{D}^*$, $T^*$ correspond to $X^*$ defined in equation 17 and are available in spectral arrays (SPSP, SPDIV, SPSPD, SPSVD, SPT) at the beginning of spectral computations.

* Elimination of variables: The following calculations are valid for both $M$ horizontally constant and $\bar{M}$ horizontally variable. All the constants and the vertical operators commute with $\nabla^2$ and $\bar{M}$.

- Elimination of $T$, $\dot{Q}$ and $\log(\Pi)$ between equations 17, 18, 19 and 21 leads to equation 55:

\[
\begin{align*}
\dot{D}'_{t+\Delta t} - (\beta \Delta t)^2 \nabla^2 \left[ \{R_dT^* \left( \frac{\gamma}{R_d} - 1 \right) \left( \frac{c_{pd}}{c_{vd}} \frac{R_dT^*}{R_d} \gamma + \frac{R_dT^*}{R_d} \nu \right) M^2 D'_{t+\Delta t} \\
+ \left\{ -R_dT^* \frac{c_{pd}}{c_{vd}} \frac{R_dT^*}{R_d} (\frac{R_dT^*}{R_d} - 1) \right\} d_{t+\Delta t} \right]
\end{align*}
\]

\[
= \tilde{D}^* + \beta \Delta t \nabla^2 \left[ \left( R_dT^* \left( \frac{\gamma}{R_d} - 1 \right) \dot{Q}^* - \gamma T^* - R_dT^* P^* \right) \right] \tag{82}
\]

\[
\tilde{D}^{**} \text{ is defined by equation 53:} 
\]

\[
\tilde{D}^{**} = \tilde{D}^* + \beta \Delta t \nabla^2 \left[ \left( R_dT^* \left( \frac{\gamma}{R_d} - 1 \right) \dot{Q}^* - \gamma T^* - R_dT^* P^* \right) \right] \tag{83}
\]

We use the relationship $c_{pd} - c_{vd} = R_d$, the definition of $C$ and we isolate the term $COR$ (see equation 58): this equation can be rewritten:

\[
\left[ - (\beta \Delta t)^2 \nabla^2 \left( C^2 - T^* \gamma \right) \right] d_{t+\Delta t} + \left[ - (\beta \Delta t)^2 \nabla^2 \left( C^2 (1 + COR) \bar{M}^2 \right) \right] D'_{t+\Delta t} = \tilde{D}^{**} \tag{84}
\]

Quantity $COR$ is zero when the constraint “C1” is fulfilled. This is the case for the finite-difference vertical discretisation with $NDLNP=1$. In the other cases, $COR$ is generally non-zero but weak ($COR << 1$).

- Elimination of $\dot{Q}$ between equations 19 and 20 leads to equation 80:

\[
\begin{align*}
d_{t+\Delta t} - (\beta \Delta t)^2 \frac{T^*}{T_s} \left[ \frac{L^*}{H^2} \left( -c_{pd} + C^2 \bar{M}^2 D'_{t+\Delta t} + C^2 \bar{M}^2 \bar{L}^* d_{t+\Delta t} \right) \right] = \tilde{D}^{**} \tag{85}
\end{align*}
\]

where $\tilde{D}^{**}$ is defined by:

\[
\tilde{D}^{**} = \tilde{D}^* + \beta \Delta t \left[ \frac{L^*}{H^2} \left( -c_{pd} + C^2 \bar{M}^2 \bar{L}^* \dot{Q}^* \right) \right] \tag{86}
\]

- When $COR = 0$, elimination of $D'$ between equations 21 and 80 leads to Helmholtz equation 59:

\[
\begin{align*}
\left[ 1 - \beta^2 \Delta t^2 C^2 (\bar{M}^2 \nabla^2 + \frac{L^*}{H^2} \frac{L^*}{H^2}) - \beta^4 \Delta t^4 N^2 C^2 \bar{M}^2 \nabla^2 \frac{T^*}{T_s} T^* \right] d_{t+\Delta t}
\end{align*}
\]

\[
= \left( 1 - \beta^2 \Delta t^2 C^2 \bar{M}^2 \nabla^2 \right) \tilde{D}^{**} + \beta \Delta t^2 \frac{T^*}{T_s} \left( -c_{pd} + C^2 \bar{M}^2 \bar{D}^{**} \right) \tag{87}
\]
5. Semi-implicit spectral computations and predictor-corrector schemes 195

which can be rewritten:
\[
\left( I - \beta^2 \Delta t^2 C^2 \frac{L_c^*}{L_c} \right) - \left( \beta^2 \Delta t^2 C^2 + \beta^4 \Delta t^4 N^2 C^2 \frac{T^*}{T^2_c} \right) \hat{M}^2 \nabla^2 \right) dt + \Delta t
\]

or:
\[
\left( I - \beta^2 \Delta t^2 C^2 \left( I - \beta^2 \Delta t^2 C^2 \frac{L_c^*}{L_c} \right)^{-1} \left( I + \beta^2 \Delta t^4 N^2 \frac{T^*}{T^2_c} \right) \hat{M}^2 \nabla^2 \right) dt + \Delta t
\]

or:
\[
\left( I - \beta^2 \Delta t^2 C^2 \frac{L_c^*}{L_c} \right)^{-1} \left( I - \beta^2 \Delta t^2 C^2 \frac{T^*}{T^2_c} \right) \frac{\nabla^2}{\hat{M}^2 \nabla^2} \hat{D}^{**} + \beta^2 \Delta t^2 \frac{L_c^*}{L_c} \left( -c_{p,t}^2 + C^2 \hat{M}^2 \hat{D}^{**} \right) \]

\[
\text{or: }
\left( I - \beta^2 \Delta t^2 C^2 \left( I - \beta^2 \Delta t^2 C^2 \frac{L_c^*}{L_c} \right)^{-1} \left( I + \beta^2 \Delta t^4 N^2 \frac{T^*}{T^2_c} \right) \hat{M}^2 \nabla^2 \right) dt + \Delta t
\]

- In the LHS of equation \([89]\), we denote by \( B \) the following matrix, by analogy with the hydrostatic model:
\[
B = C^2 \left( I - \beta^2 \Delta t^2 C^2 \frac{T^*}{T^2_c} \right)^{-1} \left( I + \beta^2 \Delta t^4 N^2 \frac{T^*}{T^2_c} \right)
\]

The LHS of equation \([89]\) can be rewritten:
\[
\left( I - \beta^2 \Delta t^2 \hat{D}^{**} \hat{M}^2 \nabla^2 \right) dt + \Delta t
\]

If we compare with the LHS of the Helmholtz equation in the hydrostatic case we can notice three things:

- The order of \( \hat{M}^2 \) and \( \nabla^2 \) is inverted (these two operators do not commute if \( \hat{M} \) is not constant).
- \( B \) now depends on \( \Delta t \), it must be recomputed each time the timestep is changed.
- \( B \) is now a tridiagonal matrix (at least in the LVERTFE=.F. discretisation).

- When \( \hat{M} = M \) (case \( \text{LSIDG}=.T. \) in ARPEGE, \( \text{LESIDG}=.T. \) in ALADIN) it is more convenient to rewrite equation \([89]\) as:
\[
\left( \nabla^2 - \beta^2 \Delta t^2 C^2 \left( I - \beta^2 \Delta t^2 C^2 \frac{L_c^*}{L_c} \right)^{-1} \left( I + \beta^2 \Delta t^4 N^2 \frac{T^*}{T^2_c} \right) \hat{M}^2 \right) \left[ \nabla^2 dt + \Delta t \right]
\]

which shows a symmetric matricial operator in the left hand side.

* Spectral computations to solve system of equations \([77]\) to \([81]\).

The algorithm has some similarities with what is done in the hydrostatic model. Algorithm works zonal wave number by zonal wave number \( m \) ( \( m \) varies between 0 and the truncation \( N_z \)) and performed in the routine \( \text{SPNHSI} \) before all horizontal diffusion schemes. For a given zonal wave number \( m \):

- After a preliminary memory transfer the right-hand side member of equation \([80]\) is computed for all total wave numbers \( n \) between \( m \) and \( N_z \). We can do some remarks:

  - During the elimination of variables between the equations \([77]\) to \([81]\) we have to perform matricial multiplications by \( \left( I - \beta^2 \Delta t^2 C^2 \hat{M} \nabla^2 \right) \) and \( \hat{M}^2 \). When \( \hat{M} \) is horizontally constant (case \( \text{LSIDG}=.F. \), \( \text{LESIDG}=.F. \)) these operators are purely diagonal; when \( \hat{M} = M \) these operators are pentadiagonal and matricial products require additional calls to routine \( \text{MXPTMA} \).
196 5. Semi-implicit spectral computations and predictor-corrector schemes

- The matricial operator

\[
\begin{pmatrix}
1 - \beta^2 \Delta t^2 C^2 \frac{T^*}{T^2} \frac{L^*}{H^2}
\end{pmatrix}
\]

which appears in the RHS of (88) and also in B is pre-computed in routine SUNHSI and is stored in the array SIFACI. Its inverse is stored in the array SIFACI.

- Inversion of Helmholtz equation for the case "constant $\overline{M}$" (case LSIDG=-F., LESIDG=-F.) and method via a diagonalisation in the eigenmodes space.

- First the diagonalisation of B is used: \( B = Q^{-1} AQ \), where \( A \) is a diagonal \( L \times L \) matrix, the diagonal coefficients \( a_i \) of which are stored in the array SIVP. \( Q \) is a \( L \times L \) matrix stored in the array SIMI, \( Q^{-1} \) is stored in the array SIMO. Note that the vertical operators \( \nu \), \( \tau \), \( \gamma \), \( B \), \( Q \), \( L^* \), \( T^* \) commute with the horizontal operator \( \nabla^2 \).

- Helmholtz equation (88) becomes, for each eigenmode \( l \):

\[
(I - \beta^2 \Delta t^2 a_i \overline{M}^2 \nabla^2) d_{t+\Delta t} = Q \left( I - \beta^2 \Delta t^2 C^2 \frac{T^*}{T^2} \frac{L^*}{H^2} \right)^{-1} \left( I - \beta^2 \Delta t^2 C^2 \overline{M}^2 \nabla^2 \right) D^* + \beta^2 \Delta t^2 \frac{T^*}{T^2} \frac{L^*}{H^2} (-c_{pd} \tau + C^2 \overline{M}^2 D^*)
\]

- For each eigenmode \( l \) and each zonal wave number \( m \): \( I - \beta^2 \Delta t^2 a_i \overline{M}^2 \nabla^2 \) is a diagonal matricial operator \((N_s + 1 - |m|) \times (N_s + 1 - |m|)\): spectral coefficients of the right-hand side member of Helmholtz equation contains values of \( d \) for all levels and five total wave numbers \((n - 2) \times n + 2\). Of course \( M^2 \) is a symmetric pentadiagonal matrix, for a given zonal wave number \( m \). Pay attention to the fact that \( M^2 \) does not commute with the diagonal operator \( \nabla^2 \).

- First the diagonalisation of B is used: \( B = Q^{-1} AQ \), where \( A \) is a diagonal \( L \times L \) matrix, the diagonal coefficients \( a_i \) of which are stored in the array SIVP. \( Q \) is a \( L \times L \) matrix stored in the array SIMI, \( Q^{-1} \) is stored in the array SIMO. Note that the vertical operators \( \nu \), \( \tau \), \( \gamma \), \( B \), \( Q \), \( L^* \), \( T^* \) commute with the horizontal operators \( \nabla^2 \) and \( M^2 \).

- Helmholtz equation (88) (resp. 89) becomes, for each eigenmode \( l \):

\[
(I - \beta^2 \Delta t^2 a_i \overline{M}^2 \nabla^2) d_{t+\Delta t} = Q \left( I - \beta^2 \Delta t^2 C^2 \frac{T^*}{T^2} \frac{L^*}{H^2} \right)^{-1} \left( I - \beta^2 \Delta t^2 C^2 \overline{M}^2 \nabla^2 \right) D^* + \beta^2 \Delta t^2 \frac{T^*}{T^2} \frac{L^*}{H^2} (-c_{pd} \tau + C^2 \overline{M}^2 D^*)
\]

resp.:

\[
(I - \beta^2 \Delta t^2 a_i \overline{M}^2) \nabla^2 d_{t+\Delta t} = Q \left( I - \beta^2 \Delta t^2 C^2 \frac{T^*}{T^2} \frac{L^*}{H^2} \right)^{-1} \left( I - \beta^2 \Delta t^2 C^2 \overline{M}^2 \nabla^2 \right) D^* + \beta^2 \Delta t^2 \frac{T^*}{T^2} \frac{L^*}{H^2} (-c_{pd} \tau + C^2 \overline{M}^2 D^*)
\]

- For each eigenmode \( l \) and each zonal wave number \( m \): \( \nabla^2 = -\beta^2 \Delta t^2 a_i M^2 \) is a symmetric pentadiagonal matricial operator \((N_s + 1 - |m|) \times (N_s + 1 - |m|)\). The factorisation LU of this matrix is computed, where L is a lower triangular tridiagonal matrix, U is a upper triangular tridiagonal matrix with coefficients equal to 1 on the main diagonal. All the useful coefficients of L, U are computed in the set-up routine SUNHHEG and stored in the array SIHEG.
5. Semi-implicit spectral computations and predictor-corrector schemes

- The right-hand side member of (94) is computed, then multiplied by the inverse of the symmetric pentadiagonal operator \( (\nabla^2 - \beta^2 M^2) \) (resolution of two tridiagonal triangular systems by routine \textsc{MXTURS})). That yields \( Q[\nabla^2 d_{i+\Delta t}] \). Multiplying by \( Q^{-1} \) then \( \nabla^2 \) one obtains \( d_{i+\Delta t} \).

- For the zonal wave number \( m = 0 \) equation (94) is used rather than (93) because, for the total wave number \( n \), \( \nabla^2 \) is equivalent to a multiplication by 0 and \( \nabla^2 \) is equivalent to a division by 0. The only difference is that the pentadiagonal but non-symmetric operator \( (I - \beta \Delta^2 a_2 M^2 \nabla^2) \) is factorised and inverted. All useful coefficients of \( L \), \( U \) are computed in the set-up routine \textsc{SUNHHEG} and stored in the arrays \textsc{SIHEG} and \textsc{SIHEG2}.

- Once known \( d_{i+\Delta t} \) equation (81), which can be rewritten as:
  \[
  D_{i+\Delta t} = [I - (\beta \Delta t)^2 \nabla^2 C^2 M^2]^{-1}D^* + (\beta \Delta t)^2 \nabla^2 (-T^* \gamma + C^2)d_{i+\Delta t}
  \]
  yields \( D_{i+\Delta t} \). Note that when \( M \) is constant, \( [I - (\beta \Delta t)^2 \nabla^2 C^2 M^2] \) is a purely diagonal operator, but when \( M = M \) \( [I - (\beta \Delta t)^2 \nabla^2 C^2 M^2] \) is a pentadiagonal operator which has to be inverted by a LU method (all useful coefficients of \( L \), \( U \) are computed in the set-up routine \textsc{SUNHHEG} and are stored in \textsc{SIHEGB} and \textsc{SIHEGB2}) and that introduces additional calls to routines \textsc{MXTURS} or \textsc{MXTURS}. For \( m > 0 \) we prefer to invert the symmetric operator \( \nabla^2 - (\beta \Delta t)^2 C^2 M^2 \).

- After calculation of \( M^2 D_{i+\Delta t} \) (which requires an additional call to routine \textsc{MXTURS} \textsc{if} \( M = M \)), equation (81) yields \( T_{i+\Delta t} \), equation (79) yields \( Q_{i+\Delta t} \), and equation (77) yields \( \log D_{i+\Delta t} \). For the case \textsc{LSIDG=}\( \text{.=T.} \) only (resp. \textsc{LSIDG=}\( \text{.T.} \) in \textsc{ALADIN}), spectral multiplications by \( M^2 \) are performed by the product of a symmetric pentadiagonal matrix of dimensions \( (N_{s} + 1 - |m|) \times (N_{s} + 1 - |m|) \) (useful coefficients computed in routine \textsc{SUSMAP} (resp. \textsc{SUESMAP} in \textsc{ALADIN}) and stored in the array \textsc{SCGMAP} (resp. \textsc{ESCGBMAP} in \textsc{ALADIN})) by a vector containing spectral coefficients \( (m, n) \) for \( n \) varying from \( |m| \) to \( N_{s} \).

- Semi-implicit scheme ends by a final memory transfer.

* Case where the constraint C1 is not matched (non zero \textsc{COR}):
An iterative algorithm has been implemented in \textsc{ARPEGE}, \textsc{IFS} and \textsc{ALADIN}, which can briefly described as follows:

- The total number of iterations is \textsc{NITERHELM}.
- For the predictor step, we replace \textsc{COR} by 0 and we do the eliminations and the inversion of the Helmholtz equation like previously described.
- For the corrector step, the term containing \textsc{COR} is put in the RHS, it is multiplied by \( D_{i+\Delta t} \) at the previous iteration. The LHS is unchanged, so the elimination and the Helmholtz solving can be done like in the predictor step. We rather take as unknowns the increments between the current iteration and the predictor step, that allows to simplify the calculation of the RHS for the corrector step (some terms become 0).

Some preliminary tests done with this iterative algorithm unfortunately show that it does not converge (\textsc{COR} is too big, especially at the top of the model, and ill-conditioned).

5.3 3D NH-GEOGW model.

The code described is valid for NH prognostic variables \( \Phi \) and \( gw \) but linear terms do not change if taking \( \Phi - B \Phi \), or \( gw - Bgw \).
Expression of the linear term $B$ for GMV and GMVS variables:

- Continuity equation ($X = \log(\Pi_s)$):
  \[ B = -\nu (\mathcal{M}^2 D') \]  
  \[ (96) \]

- Divergence equation ($X = D'$):
  \[ B = -\nabla^2 \left[ R_d T + R_d T^* \log(\Pi_s) + (\partial^* + 1)\Phi \right] \]  
  \[ (97) \]

- Vorticity equation ($X = \zeta'$):
  \[ B = 0 \]  
  \[ (98) \]

- Temperature equation ($X = T$):
  \[ B = -\frac{R_d T^*}{c_{wd}} \left[ \mathcal{M}^2 D' - \frac{1}{R_d T^*_a} \partial^*(gw) \right] \]  
  \[ (99) \]

- Geopotential equation ($X = \Phi$):
  \[ B = \left[ \frac{R_d T^*}{c_{wd}} \left( \nu - \frac{c_{pd}}{R_d T^*_a} \right) (\mathcal{M}^2 D') - (gw) \right] \]  
  \[ (100) \]

- Vertical velocity equation ($X = gw$):
  \[ B = -\frac{g^2}{R_d T^*_a} \left[ -R_d (\partial^* + 1) T - (\partial^* + 1)(\partial^* \Phi) \right] \]  
  \[ (101) \]

System to be solved: Equations are written for $\log(\Pi_s)$ as a prognostic variable for continuity equation.

\[ \log(\Pi_s)_{t+\Delta t} + \beta \Delta t \nu (\mathcal{M}^2 D'_{t+\Delta t}) = \mathcal{P}^* \]  
\[ (102) \]

\[ D'_{t+\Delta t} + \beta \Delta t \nabla^2 \left[ R_d T_{t+\Delta t} + R_d T^* \log(\Pi_s)_{t+\Delta t} + (\partial^* + 1)\Phi_{t+\Delta t} \right] = D'^* \]  
\[ (103) \]

\[ T_{t+\Delta t} + \beta \Delta t \frac{R_d T^*}{c_{wd}} \left[ \mathcal{M}^2 D'_{t+\Delta t} - \frac{1}{R_d T^*_a} \partial^*(gw_{t+\Delta t}) \right] = T^* \]  
\[ (104) \]

\[ \Phi_{t+\Delta t} + \beta \Delta t \left[ -(gw_{t+\Delta t}) - R_d T^* (\nu - \frac{c_{pd}}{R_d T^*_a} \tau) (\mathcal{M}^2 D'_{t+\Delta t}) \right] = \Phi^* \]  
\[ (105) \]

\[ gw_{t+\Delta t} + \beta \Delta t \frac{g^2}{R_d T^*_a} \left[ -R_d (\partial^* + 1) T_{t+\Delta t} - (\partial^* + 1)(\partial^* \Phi_{t+\Delta t}) \right] = gW^* \]  
\[ (106) \]

$\mathcal{P}^*$, $D'^*$, $T^*$, $\Phi^*$, $gW^*$, correspond to $\mathcal{X}^*$ defined in equation (17) and are available in spectral arrays (SPSP, SPDIV, SPT, SPSPD, SPSVD) at the beginning of spectral computations.

We can notice that this linear system contains vertical integrals and vertical derivatives; equation (106) uses something looking like the Laplacian operator $L^*$. 
5. Semi-implicit spectral computations and predictor-corrector schemes 199

* Elimination of variables: The following calculations are valid for both $M$ horizontally constant and $\overline{M}$ horizontally variable. All the constants and the vertical operators commute with $\nabla^2$ and $\overline{M}$.

We introduce the following denotations:

$$\mathcal{D}^{**} = \mathcal{D}' - \beta \Delta t \nabla^2 \left[ R_d T^* P^* + R_d T^* + (\vartheta^* + 1) \vartheta^* \right]$$  \hspace{1cm} (107)

$$\mathcal{W}^{**} = \mathcal{W}' + \beta \Delta t \frac{g}{R_d T_a} \left[ R_d (\vartheta^* + 1) T^* + (\vartheta^* + 1) (\vartheta^* + 1) \right]$$  \hspace{1cm} (108)

$$B_1 = - \left[ \frac{R_d^2 T^*}{c_{vd} T_a} + R_d T^* \nu \right] + \left[ R_d T^* (\vartheta^* + 1) (\nu - \frac{c_{vd}}{R_d T_a} \tau) \right]$$  \hspace{1cm} (109)

$$B_2 = \frac{R_d}{c_{vd} T_a} \vartheta^* + (\vartheta^* + 1)$$  \hspace{1cm} (110)

$$B_3 = g^2 \frac{R_d T^*}{c_{vd} T_a} (\vartheta^* + 1) - g^2 \frac{T^*}{T_a} (\vartheta^* + 1) (\nu - \frac{c_{vd}}{R_d T_a} \tau))$$  \hspace{1cm} (111)

$$B_4 = - \frac{g^2 T^*}{c_{vd} T_a} (\vartheta^* + 1) \vartheta^* - \frac{g^2}{R_d T_a} (\vartheta^* + 1) \vartheta^*$$  \hspace{1cm} (112)

- Elimination of $T$, $\Phi$ and log($\Pi$) between equations (102), (104), (105), and (103) leads to equation (113):

$$|I + (\beta \Delta t)^2 B_1 \nabla^2 \overline{M}^2| D'_{t+\Delta t} + [(\beta \Delta t)^2 B_2 \nabla^2]^2 g w_{t+\Delta t} = \mathcal{D}^{**}$$  \hspace{1cm} (113)

- Elimination of $T$ and $\Phi$ between equations (104), (105) and (106) leads to equation (114):

$$[(\beta \Delta t)^2 B_3 \overline{M}^2] D'_{t+\Delta t} + [I + (\beta \Delta t)^2 B_4] g w_{t+\Delta t} = g \mathcal{W}^{**}$$  \hspace{1cm} (114)

The elimination which requires the least constraints is the one providing an Helmholtz equation with the unknown $D'_{t+\Delta t}$. In this case the commutativity between $(\vartheta^* + 1)$ and $\vartheta^*$ is a sufficient condition: it is ensured if we take exactly the same operator $\vartheta^*$ everywhere. Elimination of $gw_{t+\Delta t}$ leads to Helmholtz equation (110):

$$[I - (\beta \Delta t)^2 B \nabla^2 \overline{M}^2] D'_{t+\Delta t} = \mathcal{D}^{**} - [I + (\beta \Delta t)^2 B_4]^{-1} [(\beta \Delta t)^2 B_2 \nabla^2]^2 g \mathcal{W}^{**}$$  \hspace{1cm} (115)

where:

$$B = [I + (\beta \Delta t)^2 B_4]^{-1} [-B_1 - (\beta \Delta t)^2 (B_2 B_1 - B_2 B_3)]$$  \hspace{1cm} (116)

When $\overline{M} = M$ (case LSIDG=T or LESIDG=..T.) it is more convenient to rewrite equation (115) as:

$$\nabla^2 \vartheta - (\beta \Delta t)^2 B \overline{M}^2 D'_{t+\Delta t} = \nabla^2 \vartheta \mathcal{D}^{**} - [I + (\beta \Delta t)^2 B_4]^{-1} [(\beta \Delta t)^2 B_2 \nabla^2]^2 g \mathcal{W}^{**}$$  \hspace{1cm} (117)

which shows a symmetric matricial operator in the left hand side.

We can do some remarks:
Spectral computations to solve system of equations (102) to (106). The algorithm has some similitudes with what is done in the hydrostatic model and the NH-PDVD model. Algorithm works zonal wave number by zonal wave number before all horizontal diffusion schemes. For a given zonal wave number \( m \):

- After a preliminary memory transfer the right-hand side member of equation (115) is computed for all total wave numbers \( n \) between \( m \) and \( N_s \). We can do some remarks:
  
  - The matricial operator \( [I + (\beta \Delta t)^2 B_{\mu}] \) which appears in the RHS of (115) and also in \( B \) is pre-computed in routine SUNHSI and is stored in the array SIFAC. Its inverse is stored in the array SIFACI.
  
  - Inversion of Helmholtz equation for the case “reduced divergence” (case LSIDG = F. and LESIDG = F.) and method via a diagonalisation in the eigenmodes space.
    
    - First the diagonalisation of \( B \) is used: \( B = Q^{-1} A Q \), where \( A \) is a diagonal \( L \times L \) matrix, the diagonal coefficients \( a_l \) of which are stored in the array SIVP, \( Q \) is a \( L \times L \) matrix stored in the array SIMI, \( Q^{-1} \) is stored in the array SIMO. Note that the vertical operators \( \nu, \tau, \gamma, \partial^2 \), \( B \), \( Q \) commute with the horizontal operator \( \nabla^2 \).
    
    - Helmholtz equation (115) becomes, for each eigenmode \( l \):
      \[
      (I - \beta^2 \Delta t^2 a_l \nabla^2 M^2) \Omega_l = Q \Omega^{-1} [I + (\beta \Delta t)^2 B_{\mu}]^{-1} (\beta \Delta t)^2 B_{\mu} \nabla^2 \Omega_l \Omega^{-1} \Omega^{-1}
      \]
    
    - For each eigenmode \( l \) and each zonal wave number \( m \): \( (I - \beta^2 \Delta t^2 a_l \nabla^2 M^2) \) is a diagonal matricial operator \( (N_s + 1 - | m |) \times (N_s + 1 - | m |) \); spectral coefficients of the right-hand side member of (118) are simply divided by the diagonal coefficients of this matrix. The result is then multiplied by \( Q \).

- Inversion of Helmholtz equation for the case “unreduced divergence” (case LSIDG = T. and LESIDG = T. in ARPEGE, LESIDG = F. in ALADIN): in this case \( \tilde{M} = M \). Inversion of Helmholtz equation is more complicated than in the case of semi-implicit scheme with reduced divergence because the left-hand side member of Helmholtz equation contains values of the divergence for all levels and five total wave numbers \( n \) to \( n + 2 \). Of course \( M^2 \) is a symmetric pentadiagonal matrix, for a given zonal wave number \( m \). Pay attention to the fact that \( M^2 \) does not commute with the diagonal operator \( \nabla^2 \).
    
    - First the diagonalisation of \( B \) is used: \( B = Q^{-1} A Q \), where \( A \) is a diagonal \( L \times L \) matrix, the diagonal coefficients \( a_l \) of which are stored in the array SIVP, \( Q \) is a \( L \times L \) matrix stored in the array SIMI, \( Q^{-1} \) is stored in the array SIMO. Note that the vertical operators \( \nu, \tau, \gamma, \partial^2 \), \( B \), \( Q \) commute with the horizontal operators \( \nabla^2 \) and \( M^2 \).
Upper and lower boundary conditions for $\partial^*$. This point is not easy to solve in the linear model because these boundary conditions are not always well known. There is a possibility to ignore them: in this case they are hidden in the operator $R_{\text{deri}}$, but we do not guarantee that they are always proper ones, especially at the bottom boundary. There is also possibility to specify them explicitly (use a $R_{\text{deri}}$ operator which takes account of them). Here is currently the status of what is done:

- All first-order vertical derivatives are assumed to be 0 at the top and bottom: in particular, $\partial^*Z$ is assumed to be 0 at the top and bottom in $(\partial^* + 1)(\partial^*Z)$.
- $(\nu - \frac{\nu_0}{R_{\text{deri}}^T})$: top condition equal to $\nu$, bottom condition equal to 0.
- $\partial^*$ is assumed to match $\partial^*\nu = 0$ (this is currently the case).
- For other quantities, top condition is assumed constant.
- Bottom condition is assumed constant for $T$.
- Bottom condition is not convenient to specify for $\Phi$ or $gw$ for calculations done in spectral space, because we are not assumed to have surface values (actually we have it for $\Phi$ but not for $gw$). This is one of the reasons (but not the only one) why it seems better to take the prognostic variables $\Phi - B\Phi$, and $gw - Bgw$: bottom conditions are 0 (but the assumption that the first-order vertical derivative is zero is not very good for this case!).

- Boundary conditions are assumed constant for $\nu$ and $\Phi$.

- Semi-implicit scheme ends by a final memory transfer.
5.4 2D shallow-water model.

* Expression of the linear term $B$:

- **Continuity equation ($X = \Phi$):**
  \[
  B = -\Phi^* M^2 D' \tag{121}
  \]

- **Divergence equation ($X = D'$):**
  \[
  B = -\nabla'^2(\Phi) \tag{122}
  \]

- **Vorticity equation ($X = \zeta'$):**
  \[
  B = 0 \tag{123}
  \]

* System to be solved:

\[
\Phi_{t+\Delta t} + \beta \Delta t M^2 \Phi^* D'_{t+\Delta t} = \mathcal{H}^* \tag{124}
\]

\[
D'_{t+\Delta t} + \beta \Delta t \nabla'^2(\Phi_{t+\Delta t}) = D^* \tag{125}
\]

$\mathcal{H}^*$, $D^*$ correspond to $\mathcal{X}^*$ defined in equations (17) and are available in spectral arrays (SPSP, SPDIV) at the beginning of the spectral computations. Equations (124) and (125) yield (Helmholtz equation):

\[
(1 - \beta^2 \Delta t^2 \Phi^* \nabla'^2 M^2)D'_{t+\Delta t} = D^* - \beta \Delta t \nabla'^2(\mathcal{H}^*) \tag{126}
\]

When $M = M$ (case LSIDG = .T.) it is more convenient to rewrite equation (126) as:

\[
(\nabla'^{-2} - \beta^2 \Delta t^2 \Phi^* M^2)D'_{t+\Delta t} = \nabla'^{-2} D^* - \beta \Delta t \mathcal{H}^* \tag{127}
\]

which shows a symmetric matricial operator in the left hand side.

* Spectral computations to solve system of equations (124) and (125). Algorithm works zonal wave number by zonal wave number $m$ ($|m|$ varies between 0 and the truncation $N_s$) and performed in the routine SPC2 before all horizontal diffusion schemes. For a given zonal wave number $m$:

- After a preliminary memory transfer the right-hand side member of equation (124) is computed for all total wave numbers $n$ between $m$ and $N_s$.
- Inversion of Helmholtz equation for the case "reduced divergence" (case LSIDG = .F.):
  For each zonal wave number $m$ ($|m|$ varies between 0 and the truncation $N_s$) $\mathcal{H}_{t+\Delta t}$ is a diagonal matricial operator ($N_s + 1 - |m|$) $\times$ ($N_s + 1 - |m|$). Spectral coefficients of the right-hand side member of (124) are simply divided by the diagonal coefficients of this matrix.
- Inversion of Helmholtz equation for the case "unreduced divergence" (case LSIDG = .T.):
  Inversion of Helmholtz equation is more complicated than in the case of semi-implicit scheme with reduced divergence because the left-hand side member of Helmholtz equation contains values of the divergence for all levels and five total wave numbers ($n - 2$ to $n + 2$).
5. Semi-implicit spectral computations and predictor-corrector schemes

- For each zonal wave number $m$: $(\nabla^2 - \beta^2 \Delta t^2 M^2 \Phi^*)$ is a symmetric pentadiagonal matricial operator $(N_m + 1 - |m|) \times (N_m + 1 - |m|)$. The factorisation $LU$ of this matrix is computed, where $L$ is a lower triangular tridiagonal matrix, $U$ is an upper triangular tridiagonal matrix with coefficients equal to 1 on the main diagonal. All useful coefficients of $L$, $U$ are computed in the set-up routine SUHEG and stored in the array SIHEG.
- The right-hand side member of (127) is multiplied by the inverse of the symmetric pentadiagonal operator $(\nabla^2 - \beta^2 \Delta t^2 \Phi^* M^2)$ (resolution of two tridiagonal triangular systems by routine MXTURS). That yields $D_t^{t+\Delta t}$.
- For the zonal wave number $m = 0$ equation (128) is used preferably than (127) because, for the total wave number $n = 0$, $\nabla^2$ is equivalent to a multiplication by 0 and $\nabla^2$ is equivalent to a division by 0. The only difference is that the pentadiagonal but non-symmetric operator $(1 - \beta^2 \Delta t^2 \Phi^* \nabla^2 M^2)$ is factorised and inverted. All useful coefficients of $L$, $U$ are computed in the set-up routine SUHEG and stored in the arrays SIHEG and SIHEG2.

- Once known $D_t^{t+\Delta t}$ equation (124) provides $\Phi_t^{t+\Delta t}$. For the case “unreduced divergence” (case LSIDG=.T.) only, spectral multiplications by $M^2$ are performed by the product of a symmetric pentadiagonal matrix of dimensions $(N_m + 1 - |m| \times (N_m + 1 - |m|)$ (useful coefficients computed in routine SUSMAP and stored in the array SCGMAP) by a vector containing spectral coefficients $(m,n)$ for $n$ varying from $|m|$ to $N_m$.
- Semi-implicit scheme ends by a final memory transfer.

5.5 Plane geometry.

For 3D models, semi-implicit calculations are done in ESPCSI, ESPNHSI, ESPNHSI_GEOGW instead of SPCS, SPNHSI and SPNHSI_GEOGW.

Option LSIDG=.T. has an equivalent LESIDG=.T. in the cycle AL37T1 of ALADIN, coded only for a tilted-rotated Mercator projection.

5.6 Shortcomings of the formulation of the semi-implicit scheme with “reduced divergence” (LSIDG=.F.) in case of stretching.

* ARPEGE/IFS: In the grid points computations for some equations (for example temperature and continuity equation in the 3D hydrostatic model), the linear term $B$ contains the reduced quantity $(M^2 D^t)$. This quantity is added to geographical quantities. That is no problem near the high resolution pole. This reduced quantity becomes very large near the low resolution pole: if the stretching coefficient is $c$, $\frac{M^2}{\sqrt{c}} = c^4$ at the low resolution pole, which is equal to 33.2 if $c = 2.4$. Thus the order of magnitude of the semi-implicit correction tendency becomes too high and physically absurd in the low resolution zone (gravity waves are no longer treated implicitly). That leads to instabilities in regions of the low resolution zone with high orography, in adiabatic eulerian runs, or in semi-Lagrangian runs with time-steps above the limit imposed by the Courant-Friedrich-Levy condition. In Eulerian runs with physics, the combination of physics and small time-steps inhibits this instability (at least in the hydrostatic model), but scores are degraded, especially far from the high resolution pole. In order to avoid this instability, we have implemented a new formulation of the semi-implicit scheme which allows to avoid mixing of reduced and geographical quantities in the grid-point computations, and which gives an implicit treatment of the gravity waves everywhere on the sphere and not only near the high resolution pole. This formulation is a formulation with unreduced divergence (simply by replacing the quantity $M^2$ by the mapping factor $M$).
Remark: in the deep layer equations, the implicit treatment involves in this case the quantity $M^2 D'$ and not $D$. The small residual term $D - M^2 D' = (a/r - 1) M^2 D'$ has an explicit treatment.

* **ALADIN:** The previous point is not an issue in most applications of ALADIN where the mapping factor $M$ has small variations in the forecast domain (which is not too large). It can become an issue for large domains (use of large domains with a tilted-rotated Mercator projection), and option **LESIDG** becomes useful in this case.
5. Semi-implicit spectral computations and predictor-corrector schemes

6 Inclusion of Coriolis term in the semi-implicit scheme.

Although this term is non linear, it is added to linear terms in the semi-implicit scheme. Equations are written for a leap-frog scheme (Eulerian scheme of three-time level semi-Lagrangian scheme). For a two-time level semi-Lagrangian scheme replace $\Delta t$ by $0.5\Delta t$. In ARPEGE/IFS this option is coded for both Eulerian and semi-Lagrangian schemes (3D model and shallow-water model). This option is available only in unstretched untilted spherical geometry, in setting $\text{LIMPF} = .T.$ in namelist $\text{NAMDYN}$. The reason to use such option is accuracy for long range forecasts (but not stability issues: $\text{LIMPF} = .F.$ is as stable as $\text{LIMPF} = .T.$ but may lead to slightly degraded scores for long-range forecasts).

6.1 Semi-implicit scheme including Coriolis term in the 3D hydrostatic model.

Thin layer equations.

* Expression of the linear term $\mathcal{B}$: Equations (60) and (61) become respectively:

- Divergence equation ($X = D'$):
  \[ \mathcal{B} = -\nabla''^2(\gamma T + \mu \log \Pi_s) - 2\nabla' (\Omega \wedge V) \]  
  (128)

- Vorticity equation ($X = \zeta'$):
  \[ \mathcal{B} = -2k.\{\nabla' \wedge (\Omega \wedge V)\} \]  
  (129)

* System to be solved: Equations (63) and (65) are unchanged. Equation (64) is replaced by the two following equations, for divergence and vorticity:

\[
\begin{align*}
D'_{t+\Delta t} + \beta \Delta t \nabla''^2(\gamma T_{t+\Delta t} + \mu \log (\Pi_s)_{t+\Delta t}) + \beta \Delta t (2\nabla' (\Omega \wedge V)) &= D^* \\
\zeta'_{t+\Delta t} + \beta \Delta t (2k.\{\nabla' \wedge (\Omega \wedge V)\}) &= \zeta^*
\end{align*}
\]  
(130)

$D^*$, $\zeta^*$ correspond to $\mathcal{X}^*$ defined in equation (17) and are available in spectral arrays (SPDIV, SPVOR) at the beginning of the spectral computations.

* Restriction to not stretched and not tilted model: Inclusion of Coriolis term in the semi-implicit scheme will be treated only in not stretched and not tilted geometry for different reasons, including the following considerations:

- In not stretched and not tilted geometry, Coriolis parameter $f = 2\Omega \sin \theta$ writes as a first degree polynomial of the sinus of computational sphere latitude. This property leads to invert pentadiagonal matrices in the algorithm which will be described.

- In stretched and not tilted geometry, Coriolis parameter $f = 2\Omega \sin \theta$ writes an homographical function of the sinus of computational sphere latitude. This property leads to invert full matrices in the algorithm which will be described (which in our case is very expensive in memory).

- In stretched and tilted geometry, Coriolis parameter $f = 2\Omega \sin \theta$ depends on the computational sphere longitude and latitude. This property leads to a coupling between all spectral coefficients and obliges to solve the semi-implicit system in the spectral space of geographical sphere, which is very expensive in memory and cost.
There is another way to treat implicitly Coriolis term in the semi-Lagrangian scheme which works as well in the two-time level semi-Lagrangian scheme, replacing the prognostic variable $V$ by $V + 2\Omega \times r$, where $r$ is the vertical vector from Earth centre to computational point ($r = ak$), and keeping the semi-implicit scheme unchanged. Contrary to inclusion of Coriolis term in the semi-implicit scheme, this method does not increase difficulty in stretched or tilted geometry.

In not stretched and not tilted geometry, equations (130) and (131) become:

\[ D_t + \beta \Delta t \nabla^2 (\gamma T_t + \mu \log(\Pi_s) + \beta \Delta t (2\nabla(\Omega \cdot V)) = D^* \]

\[ \zeta_t + \beta \Delta t (2k_\perp \nabla \times (\Omega \times V)) = \zeta^* \]

* **Divergence and vorticity in spherical geometry:** For a vector $Y$ of horizontal components $Y_x$ and $Y_y$, the divergence and vertical component of vorticity write as:

\[ \nabla Y = \frac{1}{a \cos \theta} \left( \frac{\partial Y_x}{\partial \lambda} + \frac{\partial (Y_y \cos \theta)}{\partial \theta} \right) \]

\[ k_\perp (\nabla \times Y) = \frac{1}{a \cos \theta} \left( \frac{\partial Y_y}{\partial \lambda} - \frac{\partial (Y_x \cos \theta)}{\partial \theta} \right) \]

* **Helmholtz equation:** Using relations (134) and (135) in equations (132) and (133) lead to the following equations:

\[ D_t + \beta \Delta t \nabla^2 (\gamma T_t + \mu \log(\Pi_s) + \beta \Delta t (-2\Omega \sin \theta) \zeta + 2\Omega \cos \theta) U_t = D^* \]

\[ \zeta_t + \beta \Delta t (2\Omega \sin \theta) D_t + \beta \Delta t \left( \frac{2\Omega \cos \theta}{a} \right) V_t = \zeta^* \]

The four following expressions are used, for each complex spectral coefficient:

- Relationship between divergence and velocity potential $\chi$:
  \[ D = \nabla^2 \chi \]

- Relationship between vorticity and stream function $\psi$:
  \[ \zeta = \nabla^2 \psi \]

- (135) and (139) can be rewritten, for each complex spectral component:
  \[ D_{(m,n)} = -\frac{n(n+1)}{a^2} \chi_{(m,n)} \]
  \[ \zeta_{(m,n)} = -\frac{n(n+1)}{a^2} \psi_{(m,n)} \]

- Relationship between $U$, $\psi$ and $\chi$:
  \[ (Ua \cos \theta)_{(m,n)} = im \chi_{(m,n)} + (n-1)\psi_{(m,n-1)} - (n+2)\psi_{(m,n+1)} \]
5. Semi-implicit spectral computations and predictor-corrector schemes

- Relationship between $V$, $\psi$ and $\chi$:

\[
(V a \cos \theta)_{(m,n)} = im\psi_{(m,n)} - (n-1)e_{(m,n)}\chi_{(m,n-1)} + (n+2)e_{(m,n+1)}\chi_{(m,n+1)}
\]  

(143)

where $e_{(0,0)} = 0$ and:

\[
e_{(m,n)} = \sqrt{\frac{n^2 - m^2}{4n^2 - 1}}
\]

(144)

Equations (140) to (143) allow to eliminate $U_{t+\Delta t}$ and $V_{t+\Delta t}$ in equations (146) and (147).

\[
[1 - i\frac{2\Delta t m}{n(n+1)}] D_{(m,n), t+\Delta t} + \beta \Delta t \nabla^2 (\gamma T_{(m,n), t+\Delta t} + \mu \log(\Pi_x))_{(m,n), t+\Delta t} = \frac{2\Delta t (\sin \theta)}{n+1} \mu \frac{\partial \log(\Pi_x)}{\partial \nabla}\nabla^2 (\gamma T_{(m,n), t+\Delta t} + \mu \log(\Pi_x))_{(m,n), t+\Delta t} = \frac{2\Delta t (\sin \theta)}{n+1} \mu \frac{\partial \log(\Pi_x)}{\partial \nabla}
\]

(145)

\[
[1 - i\frac{2\Delta t n}{n(n+1)}] \zeta_{(m,n), t+\Delta t} + \beta \Delta t (2\Omega \sin \theta) D_{(m,n), t+\Delta t} + \beta \Delta t \frac{2\Delta t (\sin \theta)}{n+1} \zeta_{(m,n+1), t+\Delta t} = \frac{2\Delta t (\sin \theta)}{n+1} \mu \frac{\partial \log(\Pi_x)}{\partial \nabla}
\]

(146)

Multiplication by $\sin \theta$ is eliminated by using the following relationship valid for any variable $X$, in not stretched and not tilted geometry:

\[
[(\sin \theta) X]_{(m,n)} = e_{(m,n)}X_{(m,n-1)} + e_{(m,n+1)}X_{(m,n+1)}
\]

(147)

Equations (145) and (146) become:

\[
[1 - i\frac{2\Delta t m}{n(n+1)}] D_{(m,n), t+\Delta t} + \beta \Delta t \nabla^2 (\gamma T_{(m,n), t+\Delta t} + \mu \log(\Pi_x))_{(m,n), t+\Delta t} = \frac{2\Delta t (\sin \theta)}{n+1} \mu \frac{\partial \log(\Pi_x)}{\partial \nabla}\nabla^2 (\gamma T_{(m,n), t+\Delta t} + \mu \log(\Pi_x))_{(m,n), t+\Delta t} = \frac{2\Delta t (\sin \theta)}{n+1} \mu \frac{\partial \log(\Pi_x)}{\partial \nabla}
\]

(148)

\[
[1 - i\frac{2\Delta t n}{n(n+1)}] \zeta_{(m,n), t+\Delta t} + \beta \Delta t \frac{2\Delta t (\sin \theta)}{n+1} \zeta_{(m,n), t+\Delta t} = \frac{2\Delta t (\sin \theta)}{n+1} \mu \frac{\partial \log(\Pi_x)}{\partial \nabla}
\]

(149)

$\zeta$ is eliminated in equation (138) by using (149) in replacing $n$ by $n - 1$ then by $n + 1$. Equation (148) becomes:

\[
\left[ I - i\frac{2\Delta t m}{n(n+1)} + \frac{(2\Delta t \Omega)^2}{n(n+1)} \right] D_{(m,n), t+\Delta t} + \frac{(2\Delta t \Omega)^2}{n(n+1)} e_{(m,n)} e_{(m,n-1)} (\frac{1}{n(n+1)})\nabla^2 (\gamma T_{(m,n), t+\Delta t}) + \frac{(2\Delta t \Omega)^2}{n(n+1)} e_{(m,n)} e_{(m,n+1)} (\frac{1}{n(n+1)})\nabla^2 (\gamma T_{(m,n), t+\Delta t}) + \frac{(2\Delta t \Omega)^2}{n(n+1)} e_{(m,n)} e_{(m,n+1)} (\frac{1}{n(n+1)})\nabla^2 (\gamma T_{(m,n), t+\Delta t}) + \frac{(2\Delta t \Omega)^2}{n(n+1)} e_{(m,n)} e_{(m,n+1)} (\frac{1}{n(n+1)})\nabla^2 (\gamma T_{(m,n), t+\Delta t}) = D^*_{(m,n)} + \frac{(2\Delta t \Omega)^2}{n(n+1)} e_{(m,n)} e_{(m,n-1)} (\frac{1}{n(n+1)})\nabla^2 (\gamma T_{(m,n), t+\Delta t}) + \frac{(2\Delta t \Omega)^2}{n(n+1)} e_{(m,n)} e_{(m,n+1)} (\frac{1}{n(n+1)})\nabla^2 (\gamma T_{(m,n), t+\Delta t})
\]

(150)
Helmholtz equation (151):
\[
\log(\Pi^{\star})_{m,n}.
\]
Then equation (63) is used to compute \( \zeta \):
\[
(\text{Equations (128) to (131) remain valid. Only the horizontal part of the Coriolis term } \n\text{remains explicit. Equations (132) to (151) remain valid, replacing } D \text{ by } (rs/a)D, \zeta \text{ by } (rs/a)\zeta, \n\text{by } (rs/a)\nabla. \text{The code of spectral computations is unchanged.})
\]
\[
(2) (\text{Done in routine SIMPLICO, for each zonal wave number } m. \text{All computations are currently in cycle 3T1 done in SPCSI. For } m = 0 \text{ complex operators become real operators.})
\]
\[
\ast \text{ Determination of other quantities at } t + \Delta_t: \text{Equation (150) is used to compute } \zeta_{t+\Delta_t} \text{(for each zonal wave number } m, \text{multiplication by a complex tridiagonal matrix). Then equation (65) is used to compute } \log(\Pi_{t+\Delta_t}) \text{ and equation (65) is used to compute } T_{t+\Delta_t}.
\]

Deep layer equations (according to White and Bromley, 1995).

Equations (128) to (131) remain valid. Only the horizontal part of the Coriolis term \((-2\Omega \wedge V)\) can be included in the semi-implicit scheme. The term \((-2\Omega \wedge Wk)\) remains explicit. Equations (132) to (151) remain valid, replacing \( D \) by \((rs/a)D, \zeta \text{ by } (rs/a)\zeta, \nabla \text{ by } (rs/a)\nabla. \text{ The code of spectral computations is unchanged.}\)

6.2 Semi-implicit scheme including Coriolis term in the 3D NH-PDVD model.

Extending such an algorithm to the 3D NH-PDVD model in not stretched and not tilted geometry is possible but not straightforward. Most difficulties come from the fact that the unknown in the Helmholtz equation is \( d \) and not \( D \).

Thin layer equations.

**Expression of the linear term \( \mathcal{B} \):** Equations (22) and (33) become respectively:

- **Divergence equation \((X = D)\):**
  \[
  \mathcal{B} = -\nabla^2 [\gamma T - T^\star (\gamma \hat{Q}) + \mu \log(\Pi_\ell) + R_\ell T^\star \hat{Q}] - 2\nabla^\star (\Omega \wedge V)
  \]

- **Vorticity equation \((X = \zeta^\star)\):**
  \[
  \mathcal{B} = -2k. [\nabla^\star (\Omega \wedge V)]
  \]
5. Semi-implicit spectral computations and predictor-corrector schemes 209

* Helmholtz equation: Elimination of $T$, $\dot{Q}$ and log($LL_t$) between equations (67), (73), (81), and the modified (78) is done like for the case LIMPF - $F$. We have now a 3-equations system with the unknowns $d$ or $d_4$, $D'$ and $\zeta'$ (equation of $\zeta'$ is identical to the hydrostatic one). Since LIMPF can work only with a not stretched not tilted spherical geometry, we omit the mapping factor and we replace $D'$ and $\zeta'$ by $D$ and $\zeta$ everywhere.

The similitudes with the hydrostatic model are the following ones: (79), (81), and the modified (78) is done like for the case LIMPF hydrostatic one). Since $U$ in the same manner as in the hydrostatic model. Using these relationships allows to replace the occurrences of $U$ and $V$ exactly as it is done in the hydrostatic model (calculations are not detailed) and to have only the spectral components of $d$ (or $d_4$), $D'$ and $\zeta'$.

To simplify, we adopt the following denotations (consistent with the ones used in the NH-GEOGW model):

- $B_1 = -C^2$
- $BC_1 = -C^2 + COR$
- $B_2 = -C^2 + T^* \gamma$
- $B_3 = -\frac{T^*}{\tau^*} L^* (-c_{pq} T + C^2)$
- $B_4 = -\frac{T^*}{\tau^*} C^2 L^*$

Equations for $d$ (or $d_4$), $D$ and $\zeta$ become:

\[
\begin{align*}
\left[ I - i \frac{2\Delta t \Delta m_{m,n}}{n(n+1)} + (\beta \Delta t)^2 \nabla^2 (B_4 + BC_1) \right] D_{(m,n),t+\Delta t} + \left[ (\beta \Delta t)^2 \nabla^2 B_2 \right] d_{(m,n),t+\Delta t} - \beta \Delta t \frac{2\Delta \epsilon_{m,n+1} n}{n+1} \zeta_{(m,n+1),t+\Delta t} = D^*_{(m,n)}(154) \\
(\beta \Delta t)^2 B_4 D_{(m,n),t+\Delta t} [ I + (\beta \Delta t)^2 B_4 ] d_{(m,n),t+\Delta t} = \tilde{D}^*_{(m,n)}(155) \\
I - i \frac{2\Delta t \Delta m_{m,n}}{n(n+1)} \zeta_{(m,n),t+\Delta t} + \beta \Delta t \frac{2\Delta \epsilon_{m,n} (n+1)}{n} D_{(m,n-1),t+\Delta t} + \beta \Delta t \frac{2\Delta \epsilon_{m,n+1} n}{n+1} D_{(m,n+1),t+\Delta t} = \zeta^*_{(m,n)}(156)
\end{align*}
\]

$\zeta$ is eliminated in equation (154) by using (156) in replacing $n$ by $n - 1$ then by $n + 1$ (exactly like we do in the hydrostatic model). Equation (154) becomes:

\[
\begin{align*}
\left[ I - i \frac{2\Delta t \Delta m_{m,n}}{n(n+1)} + (\beta \Delta t)^2 \epsilon_{m,n} (n+1)(n-1) \nabla^2 + \left( \frac{2\Delta t \Delta t}{1 - i \frac{2\Delta t \Delta m_{m,n}}{n(n+1)}} \right) \nabla^2 B_2 \right] D_{(m,n),t+\Delta t} \\
+ \left[ (\beta \Delta t)^2 \nabla^2 B_2 \right] d_{(m,n),t+\Delta t} = \tilde{D}^*_{(m,n)} + \left[ (\beta \Delta t)^2 \nabla^2 B_2 \right] d_{(m,n),t+\Delta t} \\
+ \left[ (\beta \Delta t)^2 \nabla^2 B_2 \right] d_{(m,n),t+\Delta t} = \tilde{D}^*_{(m,n)} + \left[ (\beta \Delta t)^2 \nabla^2 B_2 \right] d_{(m,n),t+\Delta t}
\end{align*}
\]

The similitudes with the hydrostatic model are the following ones:
210 5. Semi-implicit spectral computations and predictor-corrector schemes

- The $\zeta^*$ terms added to $D_{(m,n)}^{**}$ are the same ones than those added to $D_{(m,n)}^{*}$ in the hydrostatic Helmholtz equation. We introduce the following quantity:

$$D_{(m,n)}^{***} = D_{(m,n)}^{**} + \frac{(2\beta \Delta t \Omega)}{n} \frac{(n+1)}{n} \zeta_{(m,n-1)}^{**} + \frac{(2\beta \Delta t \Omega)}{n} \frac{(n+1)}{n} \zeta_{(m,n+1)}^{**} + \frac{(2\beta \Delta t \Omega)}{n} \frac{(n+1)}{n} \zeta_{(m,n+1)}^{**},$$

(158)

- In the LHS, the additional terms containing $\Omega$ (in factor of $D_{(m,n-2),t+\Delta t}$, $D_{(m,n),t+\Delta t}$ and $D_{(m,n+2),t+\Delta t}$) are the same ones as in the hydrostatic model.

That means that some pieces of code present in the hydrostatic code (the call to SIMPLICO and most of the input dummy arguments of SIMPLICO) can be re-used with no significant change in the NH model.

The main difference with the hydrostatic model, which will provide an additional difficulty, is the elimination between the $D$ equation and the $d$ equation. We first assume that $COR = 0$ (constraint C1) and, like in the case LIMPF = F, we do the elimination in order to have an Helmholtz equation with $d$ as unknown.

Equation (155) is used three times, for the total wavenumbers $n - 2$, $n$ and $n + 2$ to do the elimination of $D_{(m,n-2),t+\Delta t}$, $D_{(m,n),t+\Delta t}$ and $D_{(m,n+2),t+\Delta t}$. After this elimination, a left multiplication of the LHS and of the RHS by $[1 + (\beta \Delta t)^2 B_4]$ (which commutes with all the coefficients containing $\Omega$) is performed. We obtain an Helmholtz equation containing $d_{(m,n-2),t+\Delta t}$, $d_{(m,n),t+\Delta t}$ and $d_{(m,n+2),t+\Delta t}$ in the LHS:

$$[I - (\beta \Delta t)^2 B_4 \nabla^2] - \frac{(2\beta \Delta t \Omega)}{n(n+1)} e_{(m,n)}^{(n+1)} \frac{(n+1)}{n} e_{(m,n-1)}^{(n+1)} \frac{(n+1)}{n} d_{(m,n-2),t+\Delta t} + \frac{(2\beta \Delta t \Omega)^2}{n} e_{(m,n-1)}^{(n+1)} e_{(m,n+2)}^{(n+2)} d_{(m,n+2),t+\Delta t} = [I + (\beta \Delta t)^2 B_4]^{-1} \left(- (\beta \Delta t)^2 B_4 D_{(m,n)}^{**} + \hat{D}_{(m,n)}^{***}\right),$$

(159)

where

$$\hat{D}_{(m,n)}^{***} =$$

$$[I + (\beta \Delta t)^2 B_4 \nabla^2] - \frac{(2\beta \Delta t \Omega)}{n(n+1)} e_{(m,n)}^{(n+1)} \frac{(n+1)}{n} e_{(m,n-1)}^{(n+1)} \frac{(n+1)}{n} D_{(m,n-2)}^{**} + \frac{(2\beta \Delta t \Omega)^2}{n} e_{(m,n-1)}^{(n+1)} e_{(m,n+2)}^{(n+2)} d_{(m,n+2),t+\Delta t} = [I + (\beta \Delta t)^2 B_4]^{-1} \left(- (\beta \Delta t)^2 B_4 D_{(m,n)}^{**} + \hat{D}_{(m,n)}^{***}\right),$$

(160)

We can first remark that the RHS of the Helmholtz equation requires an additional multiplication by a complex pentadiagonal matrix which is not present in the hydrostatic model. Such a multiplication is done in routine SI_MXPTCO which re-uses some input dummy arguments of SIMPLICO. This multiplication computes a quantity which must be added to the $D_{(m,n)}^{**}$ term, before applying the inverse of $[I + (\beta \Delta t)^2 B_4]$.

Once computed its RHS, equation (159) is solved in eigenmodes space. Inversion of Helmholtz equation is equivalent to invert a pentadiagonal complex matrix (done in routine SIMPLICO), for each zonal wave number $m$; the coefficients are the same ones as in the hydrostatic model, the only difference being the content of $B$ (filled in the set-up), so this is transparent in routine SPNHSI. All computations are currently (in cycle 3T1) done in SPNHSI, by calling routine SIMPLICO. For $m = 0$ complex operators become real operators.
5. Semi-implicit spectral computations and predictor-corrector schemes 211

Once computed \(d_{(m,n),t+\Delta t}\), equation (157) is used to retrieve \(D_{(m,n),t+\Delta t}\). Equation (157) contains \(D_{(m,n-1),t+\Delta t}\), \(D_{(m,n),t+\Delta t}\) and \(D_{(m,n+1),t+\Delta t}\), that means that we need a second complex matrix inversion (there was only one in the hydrostatic model). But we can notice that the coefficients in the LHS look like those present in the LHS of the Helmholtz equation:

- All the coefficients containing \(\Omega\) are exactly the same ones of the Helmholtz equation.
- The coefficient \((\beta \Delta t)^2 B \nabla^2\) of the Helmholtz equation is replaced by \(- (\beta \Delta t)^2 B_1 \nabla^2\), where \(B_1\) is already a (constant coefficients) diagonal matrix.

That means that, for this complex matrix inversion, routine SIMPLICO can be re-used without any change (the only input coefficient which changes is the one containing the eigenvalues).

Once computed \(D_{t+\Delta t}\), equation (156) allows to retrieve \(\zeta_{t+\Delta t}\), and this piece of calculations is identical to what is done in the hydrostatic model.

* Determination of other quantities at \(t+\Delta t\): Retrieval of \(T\), \(\hat{Q}\) and \(\log(\Pi_s)\) at \(t+\Delta t\) is done exactly like in the LIMPF=F case.

* Combination with the NITERHELM algorithm when the constraint C1 is not matched: This is possible and now implemented.

Deep layer equations (according to Wood and Staniforth, 2003).

Equations (152) to (156) remain valid. Only the horizontal part of the Coriolis term \((-2\Omega \land V)\) can be included in the semi-implicit scheme. The term \((-2\Omega \land \omega_k)\) remains explicit. Equations (157) to (159) remain valid, replacing \(D\) by \((r/a)D\), \(\zeta\) by \((r/a)\zeta\), \(\nabla\) by \((r/a)\nabla\). The code of spectral computations is unchanged.

6.3 Semi-implicit scheme including Coriolis term in the 3D NH-GEOGW model.

Thin layer equations.

* Expression of the linear term \(B\): Equations (97) and (98) become respectively:

- Divergence equation \((X = D')\):
  \[B = -\nabla^2[R_dT + R_dT^* \log(\Pi_t) + (\partial \phi + 1)\phi] - 2\nabla'(\Omega \land V)\] (161)

- Vorticity equation \((X = \zeta')\):
  \[B = -2k[\nabla' \land (\Omega \land V)]\] (162)

The other equations are unchanged.

* Restriction to not stretched and not tilted model: cf. what has been said for the hydrostatic model.
*Helmholtz equation:* Elimination of $T$, $\Phi$ and $\log(II)$ between equations (104), (105), (106), and the modified (103) is done like for the case LIMPF-$F$. Elimination of $T$ and $\Phi$ in the (106) equation is done like for the case LIMPF-$P$. We have now a 3-equations system with the unknowns $gw$, $D'$ and $\zeta'$ (equation of $\zeta'$ is identical to the hydrostatic one). Since LIMPF can work only with a not stretched not tilted spherical geometry, we omit the mapping factor and we replace $D'$ and $\zeta'$ by $D$ and $\zeta$ everywhere.

The relationships (138), (139), (138), (141), (140), (142), (143), (144), (147) are still used, in the same manner as in the hydrostatic model. Using these relationships allows to replace the occurrences of $U$ and $V$ exactly as it is done in the hydrostatic model (calculations are not detailed) and to have only the spectral components of $gw$, $D'$ and $\zeta'$.

At this stage the (106) equation is unchanged compared to its LIMPF-$F$ form. Additional terms present in the divergence and vorticity equations (terms containing $\Omega$) are the same as the ones in the hydrostatic model: elimination of $\zeta$ terms into the divergence equation is done exactly as in the hydrostatic model.

We use the matrix denotations $B_1$, $B_2$, $B_3$, $B_4$ defined by formulae (109) to (112). After some long calculations often identical of those of the hydrostatic model (elimination of $\zeta$ terms, see above), we obtain the following two-equations system, with unknowns $D'$ and $gw$:

\[
\begin{align*}
\left[ I - i \frac{2\Delta t \Delta m}{n(n+1)} + \frac{(2\Delta\Omega)^2}{1-1/n(n-1)n} \frac{\gamma^2}{m,n} \frac{(n-1)(n+1)}{n^2} + \frac{(2\Delta\Omega)^2}{1-1/(n+1)(n+2)} \frac{\gamma^2}{m,n+1} \left( \frac{n+1}{n+2} \right)^2 + (\beta\Delta t)^2 \text{B}_1 \nabla^2 \right] D(m,n),t+\Delta t
\end{align*}
\]

\[
\begin{align*}
&+ \frac{(2\Delta\Omega)^2}{1-1/n(n-1)n} \frac{\gamma^2}{m,n} \frac{(n+1)}{n} \frac{\epsilon^2}{(m,n-1)} + \frac{(2\Delta\Omega)^2}{1-1/(n+1)(n+2)} \frac{\gamma^2}{m+2} \frac{n}{n+2} \frac{\epsilon^2}{m+2,n+2} D(m+2,n+2),t+\Delta t
\end{align*}
\]

\[
= D_{m,n}^{*,*} + \frac{(2\Delta\Omega)^2}{1-1/n(n-1)n} \frac{\gamma^2}{m,n} \frac{(n+1)}{n} \frac{\epsilon^2}{(m,n-1)} + \frac{(2\Delta\Omega)^2}{1-1/(n+1)(n+2)} \frac{\gamma^2}{m+2} \frac{n}{n+2} \frac{\epsilon^2}{m+2,n+2} D_{m,n}^{*,*} + \frac{(\beta\Delta t)^2 B_4_g}{g \nabla^2} [D_{m,n},t+\Delta t = g \nabla^2 m,n] \tag{164}
\]

After elimination of $g \nabla^2 m,n,t+\Delta t$ between both equations we obtain the following Helmholtz equation:

\[
\begin{align*}
\left[ I - i \frac{2\Delta t \Delta m}{n(n+1)} + \frac{(2\Delta\Omega)^2}{1-1/n(n-1)n} \frac{\gamma^2}{m,n} \frac{(n-1)(n+1)}{n^2} + \frac{(2\Delta\Omega)^2}{1-1/(n+1)(n+2)} \frac{\gamma^2}{m,n+1} \left( \frac{n+1}{n+2} \right)^2 - \beta^2 \Delta t^2 \text{B}_1 \nabla^2 \right] D(m,n),t+\Delta t
\end{align*}
\]

\[
\begin{align*}
&+ \frac{(2\Delta\Omega)^2}{1-1/n(n-1)n} \frac{\gamma^2}{m,n} \frac{(n+1)}{n} \frac{\epsilon^2}{(m,n-1)} D(m,n-2),t+\Delta t + \frac{(2\Delta\Omega)^2}{1-1/(n+1)(n+2)} \frac{\gamma^2}{m+2} \frac{n}{n+2} \frac{\epsilon^2}{m+2,n+2} D(m+2,n+2),t+\Delta t
\end{align*}
\]

\[
= D_{m,n}^{*,*} - \left[ I + (\beta\Delta t)^2 B_4 \right]^{-1} [(\beta\Delta t)^2 B_4 \nabla^2] g \nabla^2 m,n \tag{165}
\]

Equation (165) is solved in eigenmodes space. The right-hand side member needs a multiplication by a tridiagonal complex matrix, for each zonal wave number $m$. Inversion of Helmholtz equation is equivalent to invert a pentadiagonal complex matrix (done in routine SIMPLICO), for each zonal wave number $m$: this calculation is similar to the one done in the hydrostatic model. All computations are currently (in cycle 37T1) done in SPNHSI_ GEOGW. For $m = 0$ complex operators become real operators.
5. Semi-implicit spectral computations and predictor-corrector schemes

Once computed $D_{t + \Delta t}$, retrieval of $g w_{t + \Delta t}$, then retrieval of $T_{t + \Delta t}$, $\Phi_{t + \Delta t}$, and $\log(\Pi_s)_{t + \Delta t}$ is done exactly like in the case LIMPF = F.

Retrieval of $\zeta$ uses the following equation (for each zonal wave number $m$, multiplication by a complex tridiagonal matrix, like in the hydrostatic model).

$$1 - \left( \frac{2i\beta \Delta t}{n(n + 1)} \right) \frac{\Delta t}{n} D_{(m,n+1),t+\Delta t} + \beta \Delta t \frac{2\Delta t}{n+1} D_{(m,n),t+\Delta t} = \zeta_{(m,n)}$$

(166)

Deep layer equations (according to Wood and Staniforth, 2003).

Not coded.

6.4 Semi-implicit scheme including Coriolis term in the 2D shallow-water model.

Such an algorithm is also coded in the 2D shallow-water model in not stretched and not tilted geometry.

- Replace $(\gamma T + \mu \log(\Pi))$ by $\Phi$ in equation (125).
- Replace $(\gamma T_{t+\Delta t} + \mu \log(\Pi))_{t+\Delta t}$ by $\Phi_{t+\Delta t}$ in equations (130), (132) and (136).
- Replace $(\gamma T_{t_{(m,n)},t+\Delta t} + \mu \log(\Pi))_{(m,n),t+\Delta t}$ by $\Phi_{(m,n),t+\Delta t}$ in equations (145), (148) and (150).
- Replace $(\gamma T^*_{(m,n)} + \mu P^*_{(m,n)})$ by $\mathcal{H}_{(m,n)}$ in equation (151).
- Once computed $D'_{t+\Delta t}$ equation (124) provides $\Phi_{t+\Delta t}$.

6.5 Conclusion.

The simplest configuration to treat the case LIMPF = F is the case where the unknown is the horizontal divergence in the Helmholtz equation: this is the case in the hydrostatic model, the shallow-water model and in the NH-GEOGW model, and treatment of implicit Coriolis term is similar in all these cases, with only one call to SIMPLICO.

In NH models where this is $w$ or $d$ which is the unknown in the Helmholtz equation, treatment of implicit Coriolis term is more tricky, because we must solve a modified Helmholtz equation with additional $\Omega$ terms, then retrieve the horizontal divergence via a modified equation, then retrieve the horizontal vorticity. In such configurations we can expect two calls of SIMPLICO, and one call to SI_MXPTCO. This is the case in the NH-PDVD model.
7 Spectral multiplications by polynomial expressions of the mapping factor.

* Expression of mapping factor $M$ in spectral space in ARPEGE.

Let us denote by:

- $a_c = 0.5(c + \frac{1}{2})$
- $b_c = 0.5(c - \frac{1}{2})$
- $\epsilon(0,0) = 0$
- $\epsilon(m,n) = \sqrt{\frac{n^2-m^2}{4n^2-1}}$

Expression of $M$ is:

$$M = a_c + b_c \xi$$

(167)

where $\xi$ is the sinus of computational sphere latitude. Expression of $[MX](m,n)$ is:

$$[MX](m,n) = b_c \epsilon(m,n) X(m,n-1) + a_c X(m,n) + b_c \epsilon(m,n+1) X(m,n+1)$$

(168)

It is easy from (168) to retrieve the coefficients of spectral multiplication by any first degree polynomial of $M$. This is equivalent to a multiplication by a tridiagonal symmetric matrix in spectral space.

* Expression of $M^2$ in spectral space in ARPEGE.

$$[M^2X](m,n) = b_c^2 \epsilon(m,n) \epsilon(m,n-1) X(m,n-2) + 2a_c b_c \epsilon(m,n) X(m,n-1) + (a_c^2 + b_c^2(\epsilon^2(m,n) + \epsilon^2(m,n+1))) X(m,n) + 2a_c b_c \epsilon(m,n+1) X(m,n+1) + b_c^2 \epsilon(m,n+1) \epsilon(m,n+2) X(m,n+2)$$

(169)

This is equivalent to a multiplication by a pentadiagonal symmetric matrix in spectral space.

* Expression of $M$ and $M^2$ in spectral space in ALADIN. This formula is only valid on a tilted-rotated Mercator projection, and it assumes that the reference latitude of the projection is at the middle of sub-domain C+I.

If we assume that the ALADIN plane coordinates will be not slanted relatively to the longitudes and latitudes of the Mercator projection, the mapping factor $M$ always depends only on the $y$ coordinate and never vary along the $x$ coordinate.

$$M = \frac{1}{\cos \theta} = \frac{1}{\sqrt{1 - \mu^2}} = \cosh(y/a)$$

where $\mu = \sin \theta$, the $y$ coordinate (this is an distance measured on the plane projection) assumes that $y = 0$ at the apparent equator, and $a$ is the mean Earth radius. $M$ is not a low order polynomial function of $y$ so even in this case it needs to be approximated. The approximation used is, for a Fourier decomposition of $M^2$, to have only two harmonics.
8 Order of spectral computations.

* **ARPEGE/IFS:** Spectral computations are done in the 3D model in the following order:
  - Mass corrector computations ([SPCMASCOR](#)).
  - Semi-implicit computations in the hydrostatic model ([SPCSI](#)).
  - Semi-implicit computations in the NH-PDVD model ([SPNHSI](#)).
  - Semi-implicit computations in the NH-GEOGW model ([SPNHSI_GEOGW](#)).
  - Main horizontal diffusion scheme ([SPCHOR](#)).
  - Nudging (i.e. linear relaxation towards a pre-defined state) ([SPCHOR](#)).
  - Filtering of post-processed fields ([SPOS](#) for FULL-POS).

In the 2D model the semi-implicit computations and the main horizontal diffusion scheme are done in [SPC2](#).

* **ALADIN:** Spectral computations are done in the 3D model in the following order:
  - Semi-implicit computations in the hydrostatic model ([ESPCSI](#)).
  - Semi-implicit computations in the NH-PDVD model ([ESPNHSI](#)).
  - Semi-implicit computations in the NH-GEOGW model ([ESPNHSI_GEOGW](#)).
  - Spectral nudging (coupling) from a coarser model ([ESPSC2R](#) for temporal interpolation providing coupler at the current timestep, [ESPCL](#) for spectral relaxation).
  - Main horizontal diffusion scheme ([ESPCHOR](#)).
  - Filtering of post-processed fields ([SPOS](#) for FULL-POS).

* **Case of iterative centred-implicit model (option LPC_FULL=.T.):** The inversion of Helmholtz equation is done at all iterations and horizontal diffusion can be done at the last iteration or at all iterations. In particular:
  - Spectral calculations when horizontal diffusion is done at the last iteration only: `CDCONF(9:9)='T'` for \( i = 0 \), `CDCONF(9:9)='S'` for \( i = 1 \) to \( i = N_{\text{iter}} - 1 \), `CDCONF(9:9)='A'` for \( i = N_{\text{iter}} \). `CDCONF(9:9)='T'`, `CDCONF(9:9)='S'` and `CDCONF(9:9)='A'` activate the inversion of Helmholtz equation. Only `CDCONF(9:9)='A'` activates the horizontal diffusion.
  - Spectral calculations when horizontal diffusion is done at all iterations: `CDCONF(9:9)='A'` for all values of \( i \).
  - Grid-point calculations: `CDCONF(4:4)='A'`. 

5. Semi-implicit spectral computations and predictor-corrector schemes

9 Organigramme of the spectral part of the semi-implicit computations.

9.1 Set-up and control routines until STEPO.

CNT0 ->
  * SUYOYOMA ->
    - SUCT0
    - SUDYNA
    - SUALLD -> SUALDYN
  * SUYOYOMB ->
    - SUDYN ->
      * SUALDYNB (ARP) or SUELHYNB (ALD).
      * SUSMAP -> SUGMRE (ARP)
      * SUESMAP (ALD)
    * SUSI (hydrostatic model) ->
      - GPPREH, GPXYB, GPPREF
      - SUBMAT -> SITNU and SIGAM
      - EIGSOL
      - SCORDO
      - MINV
      - SUHEG -> SUHER and SUHES (LSIDG=T only)
    * SUHEG -> SUHER and SUHES (LESIDG=T only)
  * SUNEHSI (NH models) ->
    - GPPREH, GPXYB, GPPREF
    - SUNHEMAT -> SISEVE and MINV (NH-PDVD model)
    - SUNHEMAT_GEDGW -> SITNU, SIVDERI and MINV (NH-GEDGW model)
    - EIGSOL
    - SCORDO
    - MINV
    - SUENHEG -> SUHER and SUHES (LSIDG=T only, NH-PDVD model)
    - SUENHHSI (NH model) -> (see above)
    - SUENHEG -> SUHER and SUHES (LESIDG=T only, NH-PDVD model)
    - SUENHEG -> SUHER and SUHES (LSIDG=T only, NH-GEDGW model)
    * CNT1 -> CNT2 -> CNT3 -> CNT4 ->
      - SUHEG -> SUHER and SUHES (LSIDG=T only)
      - SUENHEG -> SUHER and SUHES (LESIDG=T only)
      - SUNHSI (NH model) -> (see above)
      - SUENHSI (NH model) -> (see above)
      - STEPO -> (organigramme not detailed)

* Adjoint code: For adjoint code STEPO is replaced by STEPOAD, organigramme is slightly different between CNT0 and STEPOAD. For example CNT3 is replaced by CNT3AD, CNT4 is replaced by CNT4AD.

* Tangent linear code: For tangent linear code STEPO is replaced by STEPOTL, organigramme is slightly different between CNT0 and STEPOTL. For example CNT3 is replaced by CNT3TL, CNT4 is replaced by CNT4TL.

9.2 Direct code under STEPO or tangent linear code under STEPOTL.

STEPO or STEPOTL ->
  * SPCH ->
    - SPCIMPFINIT (LIMPFF=T only)
    - TRMTDS (transposition routine for distributed memory)
5. Semi-implicit spectral computations and predictor-corrector schemes

- **SPCSI (hydrostatic model)** ->
  - SITU -> VERINT
  - SIGAM -> VERINT
  - SPCIMPSOLVE (LIMPF=T only) -> TRSTDM, SIMPLICO, TRMTOS
  - MXXMOP
  - MXTURS and MXTURE (LSIDG=T or LESIDG=T only)
  - MXPTMA (LSIDG=T or LESIDG=T only)
- **SPNHSI (NH-PDVD model)** ->
  - SIDD -> SIGAM and SISEVE
  - SI_CCCOR
  - MXPTMA (LSIDG=T or LESIDG=T only)
  - SISEVE -> VERDER
  - SITU -> VERINT
  - SIMPLICO and SI_MXPTCO (LIMPF=T only)
  - MXXMOP
  - MXTURS and MXTURE (LSIDG=T or LESIDG=T only)
  - SIGAM -> VERINT
- **SPNHSI_GEOGW (NH-GEOGW model)** ->
  - SITU -> VERINT
  - SIVDERI -> VERDER
  - SIMPLICO (LIMPF=T only)
  - MXPTMA (LSIDG=T or LESIDG=T only)
  - MXXMOP
  - MXTURS and MXTURE (LSIDG=T or LESIDG=T only)
  - TRSTDM (transposition routine for distributed memory)
  - SPCIMPFPOST (LIMPF=T only)
  - SPCIMFPPOST (LIMPF=T only)
- **SPCHOR (horizontal diffusion)**, **SPOS (FULL-POS)**:
  these routines are not detailed in this documentation.
  - **SPC2M -> SPCZ** ->
    - SIMPLICO (LIMPF=T only)
    - MXTURS (LSIDG=T or LESIDG=T only)
    - MXTURE (LSIDG=T or LESIDG=T only)
    - MXPTMA (LSIDG=T or LESIDG=T only)
    - BALADSM

In ALADIN:

- **ESPCM, ESPCSI, ESNHSI, ESNHSI_GEOGW, ESPCHOR** are called
  instead of **SPCM, SPCSI, SPNHSI, SPNHSI_GEOGW, SPCHOR**.

9.3 Adjoint code under STEPOAD.

**STEPOAD ->**
- **SPCMAD ->**
  - BAPTOB -> PE2SET
  - SPCHORAD (horizontal diffusion):
    this routine is not detailed in this documentation.
    - SPCIMPFPOSTAD (LIMPF=T only)
    - TRMTOD (transposition routine for distributed memory)
  - SPCSADIAD (hydrostatic model) ->
    - SITUAD -> VERINTAD
    - SIGAMAD -> VERINTAD
    - SPCIMPSOLVEAD (LIMPF=T only) -> TRSTDM, SIMPLICOAD, TRMTOS
    - MXXMOP
    - MXTURS and MXTURE (LSIDG=T or LESIDG=T only)
    - MXPTMA (LSIDG=T or LESIDG=T only)
  - [ SPNHSIADIAD (NH-PDVD model, not yet coded) ] ->
    - SIDDAD -> SIGAMAD and SISEVEAD
    - MXPTMA (LSIDG=T or LESIDG=T only)
5. Semi-implicit spectral computations and predictor-corrector schemes

- SISEVEAD
- SITNUAD -> VERINTAD
- MMAD
- MXMAOP
- MXURS and MXURE (LSIDG=T or LESIDG=T only)
- SIGAD -> VERINTAD
- TRSTOM (transposition routine for distributed memory)
- SPCMFINITAD (LMPF=T only)

In ALADIN:

- ESPCMAD, ESPCSIAD, ESPNHSIAD and ESPCHORAD are called instead of SPCMAD, SPCSIAD, SPNHSIAD and SPCHORAD.

9.4 Action done by these routines.

* Set-up routines:

- SUCT0: computes 0-level control variables, and also some variables linked to dynamics.
- SUDYNA: set-up for dynamics, part A.
- SUDYN: set-up for dynamics, part B.
- SUSI: set-up for hydrostatic SI scheme.
- SUNHSI: set-up for non-hydrostatic SI schemes.
- GPPREH, GPXYB, GPPREF: compute some reference vertical-dependent hydrostatic pressure quantities.
- SCORDO: reorders eigenvalues and eigenvectors in order to have ascending values of eigenvectors.
- SUBMAT, SUNHBMAT, SUNHBMAT_GEOGW: computes array SIB containing operator B respectively for 3D hydrostatic model, 3D NH-PDVD model, 3D NH-GEOGW model.
- SUSMAP: computes array SCGMAP containing coefficients for spectral multiplication by $M^2$. SUSMAP calls intermediate routine SUGMRE.
- SUESMAP: computes array ESCGMAP containing coefficients for spectral multiplication by $M^2$ (ALADIN, tilted-rotated Mercator projection).
- SUHEG (SUEHEG in ALADIN): computes the LU factorisation of Helmholtz operator in case of semi-implicit scheme with unreduced divergence. Routine SUHEG (resp. SUEHEG) is called only if LSIDG=’T’. (resp. LESIDG=’T’), for 3D hydrostatic model, 3D NH-GEOGW model and 2D shallow-water model (and more generally when the elimination gives an Helmholtz equation with $D’$ as unknown).
- SUNHHEG (resp. SUENHHHEG in ALADIN): the same as SUHEG (resp. SUEHEG in ALADIN) but for the NH-PDVD model.
- SUALDYNB (ARPEGE) or SUELDYNB (ALADIN): allocation of some arrays used in the semi-implicit scheme.

* Control routines: CNT0, CNT1, CNT2, CNT3, CNT4, CNT3AD, CNT4AD, CNT3TL, CNT4TL, STEPO, STEPOAD, STEPOTL are control routines. STEPO (resp. STEPOAD, STEPOTL) manages one direct (resp. adjoint, tangent linear) integration timestep.
5. Semi-implicit spectral computations and predictor-corrector schemes 219

* Routines under STEPO, STEPOAD or STEPOTL:
  
  - **SPCSI**: semi-implicit scheme spectral computations in the hydrostatic model.
  - **SPNHSI** (resp. **SPNHSI_GEOGW**): semi-implicit scheme spectral computations in the NH-PDVD (resp. NH-GEOGW) model.
  - **SPC2**: spectral space computations, including the semi-implicit scheme and all horizontal diffusion schemes in the 2D model.
  - **SPCM**: distributed memory interface for **SPCSI** and **SPNHSI**.
  - **SPC2M**: distributed memory interface for **SPC2**.
  - Their ALADIN counterparts have names **ESPC..** instead of **SPC..**.
  - These routines have adjoints (same names + “AD”).
  - **SPCIMPFINIT**, **SPCIMPFPOST**, **SPCIMPFSOLVE**: contain code for **LIMPF**) case.
  - **SI_CCCOR**: computes **COR** (NH-PDVD model).
  - **SITNU**: computes a linear application by operator $\nu$ (see equations (31) and (30)).
  - **SIGAM**: computes a linear application by operator $\gamma$ (see equations (27) and (26)).
  - **SISEVE**: computes a linear application by operator $L^*$ in the NH-PDVD model (see equation (34)).
  - **SIDD**: performs the elimination of $T$, $\log I_\pi$ and $\hat{Q}$ in the linear system of NH-PDVD equations (in order to compute the RHS of the Helmholtz equation).
  - **SIVDERI**: computes a linear application by operator $\partial^*$. 
  - **VERINT**: does vertical integrations (vertical finite element scheme).
  - **VERDER**: does vertical derivations (vertical finite element scheme).
  - **BALADSM**: solve linear balance equation in spectral space to convert vorticity into geopotential (used for vorticity 2D equation only).

* Linear algebra routines (project XLA/ALGOR):

  - **EIGSOL**: finds eigenvalues and eigenvectors of a matrix.
  - **MINV**: inverts a matrix.
  - **MXTURE**: inverts a set of tridiagonal (lower or upper) triangular matrices.
  - **MXTURS**: combinaison of two calls to **MXTURE** to invert a set of symmetric pentadiagonal matrices, the decomposition $LU$ of which is known.
  - **MXPTMA**: products of a set of pentadiagonal matrices by a set of matrices or vectors.
  - **MXMAOP**: matrix by matrix product.
  - **SIMPLICO**: solves a set of set of complex pentadiagonal systems (in practical called when **LIMPF**) = T).
  - **SI_MXPTCO**: complex multiplications by a penta-diagonal matrix (in practical called when **LIMPF**) = T).
  - **SUHER**: performs the $LU$ factorisations of a set of non symmetric pentadiagonal matrices.
  - **SUHES**: performs the $LU$ factorisations of a set of symmetric pentadiagonal matrices.
* Distributed memory routines (for ex. transposition routines)
doing communications between processors: the list of transposition and
communication routines used in horizontal diffusion scheme is the following: TRSTOM,
TRMTOS, BRPTOB and PE2SET; see documentation (IDDM) about distributed
memory features for the action of these routines.
10 Other remarks.

* Adjoint and tangent linear codes: These codes have been updated for semi-implicit computations in cycle 37T1, for 3D hydrostatic model and 2D model. Use of both options LSIDG=.F. and LSIDG=.T. is possible for adjoint and tangent linear codes of semi-implicit scheme in cycle 37T1 (except for non-hydrostatic model). Notice that tangent linear of SPCSI is SPCSI itself.

* Place of calculation of linear terms in grid-point space:

In the 3D model, we find such calculations in CPEULDYN for the Eulerian advection and in LACDYN (calls LASSIE, LANHSI and LANHSI_GEOGW) for the semi-Lagrangian advection. Grid-point coupling also needs to compute these terms to add them to couplers (routine ESEIMPLS called by ECOUPL1).

In the 2D model, we find such calculations in CPG2 for the Eulerian advection and in LACDYNSHW for the semi-Lagrangian advection.

Linear term calculations are done according to the following pattern:

- **3D hydrostatic model:**
  - one call to SITNU.
  - two calls to SIGAM.
  - if LSPRT=T, a conversion for the T-equation linear term.
  - if LIMPF=T, add Coriolis term to wind-equation linear term.

- **3D NH-PDVD model:**
  - one call to SIPTP.
  - two calls to SIDD.
  - one call to SISEVE.
  - if LSPRT=T, a conversion for the T-equation linear term.
  - if LIMPF=T, add Coriolis term to wind-equation linear term.

- **3D NH-GEOGW model:**
  - one call to SITNU.
  - several calls to SIVDER1.
  - additional in-lined calculations.
  - if LSPRT=T, a conversion for the T-equation linear term.
  - if LIMPF=T, add Coriolis term to wind-equation linear term.

- **timestep of calculation (given for predictor step):**
  - Eulerian advection: linear terms for \(X(t - \Delta t) - 2X(t)\).
  - 3TLSL advection: linear terms for \(X(t)\) and \(X(t - \Delta t)\).
  - 2TLSL advection: linear terms for \(X(t)\).
  - coupler: linear terms for coupler instant.
Some distributed memory features:

- The total number of processors involved in the A-level parallelisation is $N_{P_{TRW}}$.
- The total number of processors involved in the B-level parallelisation is $N_{P_{TRN}}$.
- One processor treats only a subset of zonal wave numbers.
- If $LSIDG=.T.$ or $LIMPF=.T.$ spectral part of the semi-implicit scheme is done zonal wave number by zonal wave number. A call to $SPCSI$ currently treats only one zonal wave number (case $LLONEM=.T.$).
- In the other cases a call to $SPCSI$ can treat several wave numbers: currently all the zonal wave numbers treated by the current processor (case $LLONEM=.F.$).
- All the $NFLEVG$ layers are treated together, there is no subdivision into packets of $NFLEVL$ layers when the second level of parallelisation is activated contrary to the horizontal diffusion. That means that additional transpositions ($TRSTOM$ in the direct code) are necessary between the semi-implicit calculations of $SPCSI$ and the horizontal diffusion calculations of $SPCHOR$ to convert the fields from the $NFLEVG$ structure required in the semi-implicit calculations to the $NFLEVL$ structure required in the horizontal diffusion calculations.
- In ALADIN the way of distributing $ESPCSI$, $ESPNHSI$ and $ESPNHSI\_GEOGW$ is the same one as in $SPCSI$, $SPNHSI$ and $SPNHSI\_GEOGW$. 

222 5. Semi-implicit spectral computations and predictor-corrector schemes
11 Precomputed module and namelist quantities.

11.1 YEMDYN.

- **ESCGMAP**: cf. SCGMAP in YOMDYN but for ALADIN.
- **LESIDG**: cf. LSIDG in YOMDYN but for ALADIN. Is implemented only for tilted-rotated Mercator projection.

11.2 YOEPHY.

The following variable is also present in namelist **NAEPHY**.

- **NEPHYS_PCFULL**: way of doing the lagged ECMWF physics when LPC_FULL=.T.
  - 0: physics called in both predictor and all corrector steps (expensive case).
  - 1: physics from previous timestep used in predictor and intermediate corrector steps, and physics called in last corrector step; available if split physics only.
  - 2: physics only called in predictor, stored using CPG_PT and used in corrector steps.
  - 3: predictor and intermediate corrector steps are adiabatic and physics is called in last corrector step only (this is actually the only case which numerically works).

11.3 YOMCT0.

The following variables are also present in namelist **NAMCT0**. These variables are DM-global.

- **NCONF**: configuration. Default value is 1.
- **LPC_FULL**: if .T., full iterative centred-implicit scheme (with reiterations of trajectories).
- **LPC_CHEAP**: if .T., cheap version of full iterative centred-implicit scheme (without reiterations of trajectories).
- **LPC_NESC**: if .T., non-extrapolating SL2TL SI scheme; \( X(t + dt/2) = X(t) \) during predictor (RHS of equations).
- **LPC_NESCT**: cf. LPC_NESC but for SL trajectory.
- **LVERCOR**: .T./.F.: deep/thin layer equations according to White and Bromley.

* **Namelist NAMCT0**: variables listed above are in namelist NAMCT0.

* **Distributed memory environment**: The following variables (some of them are in namelist NAMPAR0) can be useful when running distributed memory jobs: NPRGPN, NPRGPEW, NPRTRW, NPRTRV, NPROC, NPRTRN, NPRTRNS, LMPOFF, LMPDIAG, NOUTPUT. See documentation (IDDM) for more details about these variables.
11.4 YOMCVER.

The following variables are also present in namelist NAMDYN.

! * Variables related to vertical discretisation in finite elements:
! LRNHC1 : .T.: in the NH model, ensure constraint C1 by using (FD,NDLNPR=1)
! in the linear terms (SI scheme) even if other vertical
! discretisations (ex VFE) are activated in the non-linear model.
! LVFE_LAPL : VFE for vertical Laplacian term (NH model)
! LVFE_LAPL_BC : VFE for boundary cond. in vertical Laplacian term (NH model)
! : if inner domain is purely in VFE manner
! LVFE_Z_TERM : VFE Z-term (w on full levels in NH model)
! LVFE_GW : VFE for vertical velocity (NH model); in this case
! vertical velocity is at full levels.
! LVFE_DELNHPRE : VFE to compute [Delta pre] at full levels.
! LVFE_GWMPA : VFE for AROME physics vertical velocity
! (NH model with AROME physics).
! LVFE_DBCS : If VFE for derivatives in non-advective terms, use RDERI.
! LVFE_DBCT : If VFE for derivatives in non-advective terms, use RDERI
! and correct top boundary condition.

The following variables are also present in namelist NAMCT0.

! * Variables related to vertical discretisation in finite elements:
! NVSCH : type of basis if the finite element vertical discretisation is used.
! 1: linear functions.
! 3: Hermite cubic functions.

Some other variables (not in a namelist):

! * Variables related to vertical discretisation in finite elements:
! RINTE : matricial operator for vertical integrations in the
! finite element vertical discretisation.
! RDERI : matricial operator for vertical first-order derivatives in the
! finite element vertical discretisation.
! RDERB : version of RDERI taking account of top and bottom boundary conditions.
! RDERAI : matricial operator for vertical second-order derivatives in the
! finite element vertical discretisation (NH model only).

11.5 YOMDIM.

Contains dimensioning variables. The following variables are useful for spectral computations.

- **NFLEVG**: total number of model layers. DM-global.
- **NFLEVL**: partial number of model layers used in the B-set processor in Legendre, Fourier space and for some spectral calculations. (A-set DM-global, B-set DM-local).
- **NFLEVLMX**: the maximum possible value of NFLEVL. DM-global.
5. Semi-implicit spectral computations and predictor-corrector schemes

- **NFLSUR**: odd number surdimension for NFLEVL.
- **NSMAX**: model main truncation. DM-global.
- **NMSMAX**: truncation order in longitude, only used in ALADIN. DM-global.
- **NSEFRE**: number of degrees of freedom in the spectral space. DM-global.
- **NSPEC**: number of complex spectral coefficients. DM-global.
- **NSPEC2**: is equal to $2 \times $NSPECG and is used to dimension DM-global spectral arrays.
- **NSPEC2**: number of complex spectral coefficients treated by the current processor, DM-local version of NSPECG.
- **NSPEC2MX**: the maximum possible value of NSPEC2. DM-global.
- **NUMP**: number of spectral waves handled by this processor. DM-local.

* Namelist NAMDIM: The following variables of YOMDIM is in namelist NAMDIM: NFLEVG, NSMAX, NMSMAX.

11.6 YOMDYNA.

* Module variables: all variables are DM-global.
  - **NPDVAR**: formulation for pressure departure equation.
  - **NVDVAR**: formulation for vertical divergence equation.
  - **ND4SYS**: switch for the way of treatment of term NHX in the NH-PDVD $d_4$ equation.
  - **LNH_PDVD**: NH-PDVD model if .T.
  - **LNH_GEOGW**: NH-GEOGW model if .T.
  - **LGWADV**: If .T., explicit model with gw but linear model with $d$ or $d_4$ in the NH-PDVD model (semi-Lagrangian only).
  - **NGWADVSI**: when LGWADV=T and extrapolating PC schemes, alternate treatments for linear terms.
  - **LRWSDLW, LRWSDLR, LRWSDLR2, LRWSDLG**: keys for Wood and Staniforth deep-layer equations.

* Namelist NAMDYNA: most of these variables are in namelist NAMDYNA.

11.7 YOMDYN.

* Scalar quantities also present in namelist NAMDYN: all variables are DM-global; see in SUDYN for default values.
  - **BETADT**: variable $\beta$.
  - **SIPR**: reference pressure $\Pi^*$.
  - **SITR**: reference temperature $T^*$.
  - **SITRA**: reference temperature $T^*_a$.
  - **VESL**: uncentering factor in semi-Lagrangian model.
  - **XIDT**: uncentering factor in semi-Lagrangian model if pseudo second-order uncentering is used for linear terms (also called "alternate averaging of linear terms").
5. Semi-implicit spectral computations and predictor-corrector schemes

- **LIMPF**: if .F., no Coriolis term in semi-implicit scheme; if .T. Coriolis term in semi-implicit scheme (possible only for untilted unstretched global geometry).
- **NSITER**: number of iterations of the iterative centred-implicit scheme **LPC_FULL**.
- **NITERHELM**: number of iterations of the iterative algorithm of inversion of the Helmholtz equation: NH-PDVD model, case where constraint C1 is not matched.
- **LRHDI_LASTITERPC**: .T.: horizontal diffusion done at the last corrector step; .F.: horizontal diffusion done at all predictor/corrector steps.

* Other scalar quantities in YOMDYN (computed in SUDYN): all variables are DM-global.

- **LSIDG**: if .FALSE., semi-implicit computations with constant $\overline{M}$ (use of reduced horizontal divergence in the hydrostatic model); if .TRUE., semi-implicit computations with $\overline{M} = M$ (use of unreduced divergence in the hydrostatic model).
- **SIRPRG**: $R_i T^*$.
- **SIRPRN**: 1.
- **SITIME**: $2\Delta t$ from the second time-step in leap-frog scheme, $\Delta t$ in other cases.
- **NCURRENT_ITER**: current iteration for iterative centred-implicit scheme.
- **LSLINLC1**: separate linear terms from non-linear terms in continuity equation (case $\text{LRNHC1}=\text{T}$, NH-PDVD model with SL scheme).
- **LSLINLC2**: separate linear terms from non-linear terms in continuity equation (case $(\text{LGW ADV, LSETTLES})=\text{T}$, NH-PDVD model with SL scheme).
- **LSLINL**: separate linear terms from non-linear terms in other equations (case $(\text{LGW ADV, LSETTLES})=\text{T}$, NH-PDVD model with SL scheme).

* Arrays in YOMDYN computed in SUDYN: all variables are DM-global.

- **SIDELP**: $\Delta \Pi^*$.
- **SIRDEL**: $1/\Delta \Pi^*$.
- **SILNPR** and **SIALPH**: $\delta^*$ and $\alpha^*$.
- **SITLAF** and **SITLAH**: respectively full-level and half-level reference hydrostatic pressure.
- **SIDPHI**: $\Delta \Phi^*$ at full levels.
- **SIB**: matrix $B$ in Helmholtz equation. $B$ does not represent the same quantity in hydrostatic and non-hydrostatic models.
- **SIBI**: matrix $B^{-1}$.
- **SIVP**: eigenvalues $a_i$ of $B$. In 2D shallow-water model, $\text{SIVP}(1)=\Phi^*$.
- **SIMI**: Q (eigenvectors of $B$). **SIMI** is used only in the 3D models.
- **SIMO**: $Q^{-1}$.
- **SIFAC**: matricial operator (NH model only)

$$
\left( 1 - \beta^2 \Delta t^2 C \frac{T^*}{T^2} L^* \right)
$$

- **SIFACI**: inverse of **SIFAC**.
5. Semi-implicit spectral computations and predictor-corrector schemes

* Arrays in YOMDYN computed in SUHEG and SUNHHEG:

- **SIHEG:**
  - Hydrostatic model, NH-GEOGW model or 2D shallow-water model: for zonal wave numbers different from zero, contains the non-zero diagonals of \( LU \) decomposition of:
    \[
    (\nabla'^2 - \beta^2 \Delta t^2 a_l M^2)
    \]
    for zonal wave number equal to 0, contains the non-zero diagonals of \( L \) of the \( LU \) decomposition of:
    \[
    (1 - \beta^2 \Delta t^2 a_l \nabla'^2 M^2)
    \]
  - NH-PDVD 3D model: for zonal wave numbers different from zero, contains the non-zero diagonals of \( LU \) decomposition of:
    \[
    (\nabla'^2 - \beta^2 \Delta t^2 a_l M^2)
    \]
    for zonal wave number equal to 0, contains the non-zero diagonals of \( L \) of the \( LU \) decomposition of:
    \[
    (1 - \beta^2 \Delta t^2 a_l M^2 \nabla'^2)
    \]

- **SIHEG2:**
  - Hydrostatic model, NH-GEOGW model or 2D shallow-water model: for zonal wave number equal to zero, contains the non-zero diagonals of \( U \) of the \( LU \) decomposition of:
    \[
    (1 - \beta^2 \Delta t^2 a_l \nabla'^2 M^2)
    \]
  - NH-PDVD 3D model: for zonal wave number equal to zero, contains the non-zero diagonals of \( U \) of the \( LU \) decomposition of:
    \[
    (1 - \beta^2 \Delta t^2 a_l M^2 \nabla'^2)
    \]

- **SIHEGB** for NH-PDVD 3D model: for zonal wave numbers different from zero, contains the non-zero diagonals of \( LU \) decomposition of:
    \[
    (\nabla'^2 - \beta^2 \Delta t^2 CM^2)
    \]
    for zonal wave number equal to 0, contains the non-zero diagonals of \( L \) of the \( LU \) decomposition of:
    \[
    (1 - \beta^2 \Delta t^2 \nabla'^2 CM^2)
    \]

- **SIHEGB2** for NH-PDVD 3D model: for zonal wave number equal to zero, contains the non-zero diagonals of \( U \) of the \( LU \) decomposition of:
    \[
    (1 - \beta^2 \Delta t^2 \nabla'^2 CM^2)
    \]

11.8 YOMMP.

Variables specific to the distributed memory environment. For more details see documentation (IDDM) about distributed memory features.

11.9 YOMMPG.

Variables specific to the distributed memory environment. DM-global variables mainly used to handle reading and writing of grib data. For more details see documentation (IDDM) about distributed memory features.
11.10 YOMSP.

* SPDIV, SPVOR, SPHV, SPT, SPSPD, SPSVD, SPNHX, SPSP (in YOMSP): Spectral arrays containing respectively divergence, vorticity, GMV thermodynamic variables, temperature, NH variables, logarithm of surface hydrostatic pressure. SPT, (and SPSPD, SPSVD in the NH model) is associated with the same piece of memory as SPHV. These arrays are DM-local, they contain all the NFLEVG layers but only the NSPEC2 spectral coefficients treated by the current processor. No variable in namelist.
12 References.

12.1 Publications.


12.2 Internal notes and documentation.

- (TDECTEC) 2010: IFS technical documentation (CY36R1). Part VI: technical and computational procedures. Available at "http://www.ecmwf.int/research/ifsdocs/".
• (IDPC) Bénard, P., 2002: Incremental versus non-incremental predictor-corrector schemes. Appendix A of the previous internal note.


• (IDBAS) Yessad, K., 2011: Basics about ARPEGE/IFS, ALADIN and AROME in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrn.meteo.fr/gmapdoc/*”).

• (IDDH) Yessad, K., 2011: Horizontal diffusion in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server "http://www.cnrn.meteo.fr/gmapdoc/*").

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server "http://www.cnrn.meteo.fr/gmapdoc/*").

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server "http://www.cnrn.meteo.fr/gmapdoc/*").

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server "http://www.cnrn.meteo.fr/gmapdoc/*").

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911 (internal note, available on the intranet server “http://www.cnrn.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrn.meteo.fr/gmapdoc/*”).

• (IDEUL) Yessad, K., 2011: Integration of the model equations, and Eulerian dynamics, in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrn.meteo.fr/gmapdoc/*”).

• (IDESIDG) Yessad, K., 2006: Implementation of option LESIDG in ALADIN (internal note, 13pp).

This chapter recalls some basics about spectral horizontal diffusion schemes, and describes how it is implemented in ARPEGE/IFS. The ECMWF way of implementing it (which is slightly different from the METEO-FRANCE one) is not described in detail. There is a main horizontal diffusion scheme and a Rayleigh friction. An organigramme is provided.

1 Introduction.

* General considerations for ARPEGE/IFS: In the code of ARPEGE/IFS horizontal diffusion computations are generally spectral computations. There is a main horizontal diffusion scheme in spectral space. In case of constant resolution, the main horizontal diffusion scheme is discretised by a purely diagonal operator in spectral space. The main horizontal diffusion scheme is implicit in order to remain stable even with high diffusion coefficients and are called after the semi-implicit scheme. Horizontal diffusion is called after inversion of the semi-implicit scheme and before nudging.

* Horizontal diffusion in ALADIN: Horizontal diffusion computations are still spectral computations. There is a main horizontal diffusion scheme only. For the main horizontal diffusion scheme, the formulation looks like the unstretched ARPEGE one. There are specific routines for spectral calculations.

* Rayleigh friction: there is a Rayleigh friction in the grid-point space computations which acts only in upper atmosphere for the $U$-component of the horizontal wind (properly coded only for untitled spherical geometry or untitled plane geometry).

* Two-time level semi-Lagrangian scheme: equations are written for a leap-frog scheme. The algorithm remains the same for a two-time level semi-Lagrangian scheme.

* Deep layer equations: For conveniency, equations are written with the geographical horizontal gradient operator $\nabla$, which is used in the horizontal diffusion scheme for thin layer equations. For deep layer equations, the operator actually used is $\left[ \frac{1}{r} \nabla \right]$. 
where $r$ is the actual radius, to simulate a “pseudo-geographical” diffusion, instead of $\nabla$: this approximation is reasonable and allows to avoid any tricky modification in the horizontal diffusion spectral computations; use of the true geographical gradient $\nabla$ sticks to the grid-point calculations.

* **Distributed memory:** some distributed memory features are now introduced in the code and will be briefly described. For convenience one uses some generic appellations.

- Expression “DM-local” for a quantity means “local to the couple of processors (proca,procb)”: each processor has its own value for the quantity. Expression “DM-local computations” means that the computations are done independently in each processor on “DM-local” quantities, leading to results internal to each processor, which can be different from a processor to another one.
- Expression “DM-global” for a quantity means that it has a unique value available in all the processors. Expression “DM-global computations” means that the computations are either done in one processor, then the results are dispatched in all the processors, or the same computations are done in all the processors, leading to the same results in all the processors.
- In a routine description the mention “For distributed memory computations are DM-local” means that all calculations done by this routine are DM-local; the mention “For distributed memory computations are DM-global” means that all calculations done by this routine are DM-global; when no information is provided it means that a part of calculations are DM-local and the other part is DM-global.
- Expression “main” processor currently refers to the processor number 1: (proca,procb)=(1,1).

* **Modifications since cycle 37:**
- Minor changes on the organigramme (for example call (E)SPCM directly from STEPO).
- Adaptation of default value of SDRED.

**2 Formulation of the horizontal diffusion schemes.**

**2.1 Main horizontal diffusion scheme.**

* **General formulation:** The main horizontal diffusion formulation is close to:

$$\frac{\partial X}{\partial t} = -K'^'X M\nabla'\nabla'X$$

where $K'^'$ is a vertically dependent and horizontally constant coefficient. $K'^'$ is generally complex: $\exp(0.5\pi ir)$ multiplied by a real positive coefficient. $r$ is the power of the horizontal diffusion scheme. $M$ is the mapping factor.

In ARPEGE:

$$K'^' = \exp(-0.5\pi ir) \left[ \sqrt{\frac{N_r(N_r+1)}{a^2}} \right]^{-r} \Omega \ h_X \ g$$

($a$ is the mean Earth radius, $N_r$ is the truncation). $g$ is horizontally constant, vertically dependent. $\Omega$ is the angular velocity of the Earth rotation ($0.0007292115 \text{ s}^{-1}$). $h_X$
6. Horizontal diffusion

is a constant coefficient for each prognostic variable. There are seven constants, one for vorticity \((h_\zeta)\), one for divergence \((h_D)\), one for temperature \((h_T)\), one for humidity \((h_q)\), one for ozone \((h_{O_3})\), one for the extra GFL variables \((h_{EXT})\), one for the surface hydrostatic pressure \((h_{SP})\) which is also used for the equivalent height in the 2D model. There are additional constants for non-hydrostatic variables in the non-hydrostatic model.

For divergence expression of \(h_D\) matches:

\[
\frac{1}{\Omega h_D} = \frac{2\pi a}{N_{\text{dmon}}} \frac{(1 + 0.5 r_{\text{algine}})^{2.5}}{r_{\text{dtau}} r_{\text{dampdiv}}}
\]

\(r_{\text{algine}}\) is the increment from linear to quadratic grid \((r_{\text{algine}} = 0\) for a linear grid, 1 for a quadratic grid); \(r_{\text{dtau}}\) is a tuning parameter; \(N_{\text{dmon}}\) is the number of grid-points on a latitude situated near the equator of the computational sphere.

In ALADIN:

\[
K''_X = \exp(-0.5\pi ir) \left[ (2\pi) \sqrt{\frac{1}{N_{\text{ms}}} + \frac{N_{\text{ms}}^2}{N_{\text{ms}}}} \right]^{-r} \Omega h_X g
\]

\((a\) is the mean Earth radius, \(L_x\) and \(L_y\) are respectively the zonal and the meridian lengths of the ALADIN domain, these lengths being taken on a surface situated at a distance \(a\) of the Earth center, \(N_{\text{ms}}\) is the zonal truncation, \(N_{\text{s}}\) is the meridian truncation).

\[
\frac{1}{\Omega h_D} = \left[ \Delta X_{\text{gp}} \right] \frac{(1 + 0.5 r_{\text{algine}})^{2.5}}{r_{\text{dtau}} r_{\text{dampdiv}}}
\]

where:

\[
\left[ \Delta X_{\text{gp}, \text{zonal}} \right] = \sqrt{0.5 \left( \left[ \Delta X_{\text{gp}, \text{zonal}} \right]^2 + \left[ \Delta X_{\text{gp}, \text{merid}} \right]^2 \right)}
\]

\(\left[ \Delta X_{\text{gp}, \text{zonal}} \right]\) and \(\left[ \Delta X_{\text{gp}, \text{merid}} \right]\) are respectively the zonal and meridian grid-point mesh-sizes taken at a location where the mapping factor is equal to 1.

In both ARPEGE and ALADIN, for the other 3D upper air fields:

\[
\frac{1}{\Omega h_X} = \frac{r_{\text{dampX}}}{r_{\text{dampdiv}} \Omega h_D}
\]

which can be rewritten:

\[
(\Omega h_X) = \frac{r_{\text{dampdiv}}}{r_{\text{dampX}}} (\Omega h_D)
\]

* Modifications brought to this scheme if the SLHD (semi-Lagrangian horizontal diffusion) interpolations are done in the semi-Lagrangian scheme:  
   When the advection scheme is a semi-Lagrangian one, it is possible to activate more diffusive interpolations in the semi-Lagrangian scheme ("SLHD" diffusion), and in this case the horizontal diffusion scheme must be modified:

- Less diffusion in the above formulation (generally by reducing the value of function \(g(l)\)).
• Adding of a second horizontal diffusion scheme for divergence and vorticity (and also vertical divergence in the NH model): this scheme has the same formulation as the previous one, but the following variables are changed:
  - \( h_X \) becomes \( h_d h_X \) (currently \( \frac{1}{r_{\text{damp}} h_X} = \frac{1}{r_{\text{damp}} h_d h_X} \), i.e. \( \Omega h_d h_X \) = \( \Omega h_{d h} \).
  - \( K''_X \) becomes \( K_{d h} h_X \).
  - the power \( r \) becomes \( s \) (\( s \) is generally above \( r \)).

The "SLHD" diffusion can now be activated on a subset of prognostic variables and the above modifications are applied only to this subset of variables.

* Additional remarks: In the unstretched version of ARPEGE, \( M = 1 \). In AL-ADIN, the horizontal variations of \( M \) are generally neglected and \( M \) is replaced by a constant hidden in the coefficients \( K''_X \) and \( K_{d h} h_X \). So the ALADIN diffusion looks like the unstretched ARPEGE one. This approximation may become not very appropriate for large domains (used particularly with the rotated-tilted Mercator projection).

2.2 Rayleigh friction.

The following scheme is applied on the \( U \)-component of the momentum equation (i.e. the zonal component on the Gaussian grid).

\[
\left( \frac{\partial U}{\partial t} \right)_{\text{fric}} = -K_{\text{fric}} U
\]

\( -K_{\text{fric}} U \) is a Rayleigh friction which is activated if \( LRFRIC = \text{T.} \) in NAMCT0. \( K_{\text{fric}} \) is a vertical dependent coefficient which is non-zero only for a standard pressure lower than a threshold stored in variable \( \text{RRFPLM} \).

2.3 Nudging.

This scheme called after horizontal diffusion is a linear relaxation of prognostic variables towards pre-defined fields and is not properly saying a diffusion scheme, so it will not be described in detail in this documentation. Pre-defined fields (for example climatological fields) can be read on a file. Nudging coefficients can be different for each variable. Nudging coefficients are not vertically dependent, except for the upper atmosphere where they can be set to zero. Nudging is applied to all variables present in spectral space. There is also nudging for grid-point variables in grid-point space, such surface temperature, surface moisture, deep moisture and snow depth. Nudging is controlled by \( \text{LNUDG} \) in namelist NAMNUD.

3 Discretisation of the horizontal diffusion schemes.

3.1 Main horizontal diffusion scheme.

Main horizontal diffusion scheme in unstretched ARPEGE/IFS.

* Discretisation: Calculations are done in spectral space. The discretised equation which is coded is:

\[
X^+_{(m,n)} - X^-_{(m,n)} = -\Omega h_X g(l)f(n, N, \eta_0(X), x_0, r) Dt X^+_{(m,n)}
\]

(9)
where the superscripts + and − indicate respectively variables after horizontal diffusion and variables before horizontal diffusion. $Dt$ is the time step in the first integration step and twice the time step later (leap frog scheme), $n$ is the total wave number (between 0 and the truncation $N_s$), $m$ is the zonal wave number (between $−n$ and $n$ in a triangular truncation), $r$ is the order of the horizontal diffusion operator, $N$ is a reference wave number, $n_0(X)$ is a threshold depending on variable $X$ (generally zero except for vorticity where it is 2 at METEO-FRANCE and 0 at ECMWF), $x_0$ is a threshold between 0 and 1. $\nabla'$ is the first order horizontal reduced derivative operator (i.e in the grid point space, $\nabla = M\nabla'$, where $M$ is the mapping factor). This horizontal diffusion scheme is an implicit horizontal diffusion scheme (in the right-hand side member of the equation there is $X^+(m,n)$ and not $X^-(m,n)$).

Equation (9) becomes (10):

$$X^+(m,n) = \frac{1}{1 + K_X(n)Dt}X^-(m,n)$$

where:

$$K_X(n) = \Omega h_X g(l)f(n, N_s, n_0(X), x_0, r)$$

Expression of $f(n, N, n_0(X), x_0, r)$ is currently:

$$f(n, N, n_0(X), x_0, r) = \max(0, \min(1, \left(1 - \frac{\max(0, n(n+1)-n_0)}{\max(0, N(N+1)-n_0)} - x_0 \right)^r))$$

An exact discretisation of equation (11) would give:

$$f(n, N, n_0(X), x_0, r) = f(n, N_s, 0, 0, r)$$

One can notice that this discretisation is equivalent to a purely diagonal operator in spectral space.

* Vertical dependency of $g$: $g(l)$ only depends on the altitude. For the layer number $l$, expression of $g(l)$ is:

$$g(l) = \min\left(y_0 \frac{\Pi_{ref}}{\Pi_{ST}(l)} \frac{1}{y_3}\right)$$

where $\Pi_{ST}(l)$ is the standard atmosphere pressure for the layer number $l$, $\Pi_{ref}$ is a reference pressure (sea level pressure for the standard atmosphere: 1013.25 hPa). $y_0$ is between 0 and 1, $g(l) = 1$ if $\Pi_{ST}(l)/\Pi_{ref}$ is above $y_0$ and $g(l)$ is above 1 if $\Pi_{ST}(l)/\Pi_{ref}$ is below $y_0$. $y_3$ is between 0 and $y_0$, in practical it is significantly lower than $y_0$ and it is used to avoid too much diffusion in the high stratosphere and in the mesosphere. In the 2D model, $g(l)$ can only take one value equal to 1. Definition of $g(l)$ is different in ECMWF configuration (LECMWF =... T.).
Main horizontal diffusion scheme in ALADIN.

Compared to ARPEGE, the main differences are:

- $K_X$ also depends on the zonal wavenumber $m$.
- Function $f$ also depends on $m$ and on the zonal truncation $N_{ms}$, and has a different expression.
- $n_0$ is replaced by zero in the code (for conveniency one will let $n_0$ in the following formulae).
- Equation (9) becomes:
  \[ X^+_{(m,n)} - X^-_{(m,n)} = -\Omega h_X g(l) f(m, n, N_m, N, n_0(X), x_0, r) \, dt \, X^0_{(m,n)} \quad (15) \]
  where $N_m$ is a reference zonal wavenumber.
- Equation (10) becomes:
  \[ X^+_{(m,n)} = 1 + K_X(m, n) \frac{dt}{\Delta t} X^0_{(m,n)} \quad (16) \]
- Equation (11) becomes:
  \[ K_X(m, n) = \Omega h_X g(l) f(m, n, N_{ms}, N_s, n_0(X), x_0, r) \quad (17) \]
- Expression of $f(m, n, N_m, N, n_0(X), x_0, r)$ is currently:
  \[ f(m, n, N_m, N, n_0(X), x_0, r) = \max \left( 0, \min \left( 1, \left( \frac{n_0^2 + m_0^2}{N_{ms}^2 N_s^2} \frac{1}{1 - x_0} \right)^\frac{1}{2} - x_0 \right) \right) \quad (18) \]
- An exact discretisation of equation (11) would give:
  \[ f(m, n, N_m, N, n_0(X), x_0, r) = f(m, n, N_{ms}, N_s, 0, 0, r) \quad (19) \]

Main horizontal diffusion scheme in stretched ARPEGE/IFS.

* Expression of mapping factor in spectral space. Let us denote by:
  
  - $a_c = 0.5 \left( c + \frac{1}{c} \right)$
  - $b_c = 0.5 \left( c - \frac{1}{c} \right)$
  - $e_{(0,0)} = 0$
  - $e_{(m,n)} = \sqrt{\frac{n_0^2 - m_0^2}{4n_s^2 - 1}}$

Expression of $M$ is:
  \[ M = a_c + b_c \mu \quad (20) \]
where $\mu$ is the sinus of computational sphere latitude. Expression of $[MX]_{(m,n)}$ is:
  \[ [MX]_{(m,n)} = b_c e_{(m,n)} X_{(m,n-1)} + a_c X_{(m,n)} + b_c e_{(m,n+1)} X_{(m,n+1)} \quad (21) \]
A multiplication by $M$ in spectral space is equivalent to a multiplication by a symmetric tridiagonal matrix.
6. Horizontal diffusion

* Discretisation: Calculations are done in spectral space. The discretised equation which is coded is:

\[
X_{(m,n)}^- = +b_c e_{(m,n)} \Omega h X \ g(l) Dtf(n - 1, N, n_0(X), x_0, r) X_{(m,n-1)}^+ \\
+ (1 + a_c \Omega h X \ g(l) Dtf(n, N, n_0(X), x_0, r)) X_{(m,n)}^- \\
+ b_c e_{(m,n+1)} \Omega h X \ g(l) Dtf(n + 1, N, n_0(X), x_0, r) X_{(m,n+1)}^+ \tag{22}
\]

Equation (22) is equivalent to invert a tridiagonal matrix for each zonal wave number \( m \). The computation of this matrix and a decomposition into a product of two triangular bidiagonal matrixes are performed in the set-up routine SUHDU. At each time step these two triangular bidiagonal matrixes are inverted in order to compute the \( X_{(m,n)}^- \).

3.2 Modification of the horizontal diffusion scheme if “SLHD” interpolations are done in the semi-Lagrangian scheme.

* Modified expression of \( g(l) \): Expression of \( g(l) \) is now modified; new expression of \( g(l) \) is:

\[
g(l) = \min \left( \frac{\Pi_{ref}}{\Pi_{ST}(l)} \cdot \frac{1}{y_3} \right) - s_{dred} \tag{23}
\]

* Unstretched ARPEGE/IFS: A double diffusion is applied on vorticity and divergence (and also the vertical divergence variable in the NH model). It writes:

\[
X_{(m,n)}^+ = \left[ \frac{1}{1 + K_{hds X} (n) Dt} \right] \left[ \frac{1}{1 + K_X (n) Dt} \right] X_{(m,n)}^-
\]

where:

\[
K_{hds X} (n) = \Omega h_{hds X} g_{hds} (l) f(n, N_s, n_0(X), x_0, s) \tag{25}
\]

* ALADIN: A double diffusion is applied on vorticity and divergence (and also the vertical divergence variable in the NH model). It writes:

\[
X_{(m,n)}^+ = \left[ \frac{1}{1 + K_{hds X} (m,n) Dt} \right] \left[ \frac{1}{1 + K_X (m,n) Dt} \right] X_{(m,n)}^-
\]

where:

\[
K_{hds X} (m,n) = \Omega h_{hds X} g_{hds} (l) f(m, N_s, N_a, n_0(X), x_0, s) \tag{27}
\]
Stretched ARPEGE/IFS: A double diffusion is applied on vorticity and divergence. It writes:

\[ X_{(m,n)} = \]
\[ + b_c e_{(m,n)} \Omega h_X g(l) D(t)(n - 1, N, n_0(X), x_0, r) X_{(m,n-1)}^{++} \]
\[ + (1 + a_c \Omega h_X g(l) D(t)(n, N, n_0(X), x_0, r)) X_{(m,n)}^{++} \]
\[ + b_c e_{(m,n+1)} \Omega h_X g(l) D(t)(n + 1, N, n_0(X), x_0, r) X_{(m,n+1)}^{++} \]  \( (28) \)

then:

\[ X_{(m,n)}^{++} = \]
\[ + b_c e_{(m,n)} \Omega h_{hds} X g_{hds}(l) D(t)(n - 1, N, n_0(X), x_0, s) X_{(m,n-1)}^{++} \]
\[ + (1 + a_c \Omega h_{hds} X g_{hds}(l) D(t)(n, N, n_0(X), x_0, s)) X_{(m,n)}^{++} \]
\[ + b_c e_{(m,n+1)} \Omega h_{hds} X g_{hds}(l) D(t)(n + 1, N, n_0(X), x_0, s) X_{(m,n+1)}^{++} \]  \( (29) \)

This discretisation requires to invert two tridiagonal symmetric matrices in spectral space.

**Expression of** \( g_{hds}(l) \): Expression of \( g_{hds}(l) \) is:

\[ g_{hds}(l) = \min \left( y \frac{\Pi_{ref}}{\Pi_{ST}(l)}, \frac{1}{y_3} \right) \]  \( (30) \)

### 3.3 Particular use of a second reference truncation \( N_2 \) for standard pressure levels \( \Pi_{ST}(l) \) below \( y_0 * \Pi_{ref} \).

In ARPEGE, it is possible in the code to modify \( g(l) \) in order to simulate a replacing of \( f(n, N, n_0(X), x_0, r) \) by \( f(n, N_2, n_0(X), x_0, r) \), where \( N_2 < N_s \) for standard pressures \( \Pi_{ST}(l) < y_1 * \Pi_{ref} \), where \( y_1 < y_0 \). In this case \( g(l) \) also depends on \( n \) for standard pressures below \( y_0 * \Pi_{ref} \).

If \( \Pi_{ST}(l)/\Pi_{ref} \) is below \( y_1 \) (\( y_3 \) is assumed to be lower than \( y_1 \)):

\[ g(l, n) = \min \left( y_0 \frac{\Pi_{ref}}{\Pi_{ST}(l)}, \frac{1}{y_3} \right) \frac{f(n, N_2, n_0(X), x_0, r)}{f(n, N_s, n_0(X), x_0, r)} \]  \( (31) \)

If \( \Pi_{ST}(l)/\Pi_{ref} \) is over \( y_1 \) and below \( y_0 \):

\[ g(l, n) = \frac{y_0 * \Pi_{ref} - b(n)}{\Pi_{ST}(l) - b(n)} \]  \( (32) \)

where:

\[ b(n) = \frac{(y_0 * \Pi_{ref} + \Pi_{ref})(f(n, N_2, n_0(X), x_0, r) - f(n, N_s, n_0(X), x_0, r))}{(y_0 + \Pi_{ref}) f(n, N_2, n_0(X), x_0, r) - (y_1 + \Pi_{ref}) f(n, N_s, n_0(X), x_0, r)} \]  \( (33) \)

One can check that \( (29) \) and \( (30) \) yield \( g(l, n) = 1 \) for \( \Pi_{ST} = y_0 * \Pi_{ref} \) and that \( (31) \) and \( (32) \) are continuous for \( \Pi_{ST} = y_1 * \Pi_{ref} \). When taking \( N_2 = N_s \), one retrieves formula \( (14) \).

Choice of \( N_2 < N_s \) is useful in climate modelisation.

The same type of modification can be extended to \( g_{hds} \).

This option is not coded in ALADIN.
3.4 Application of diffusion to each variable.

Application to GMV variables and to GFL variables in 3D models.

For GMV variables, horizontal diffusion is applied to:

- “reduced” vorticity \( \zeta' \) with \( n_0(\zeta) = 2 \) at METEO-FRANCE, 0 at ECMWF.
- “reduced” divergence \( D' \) with \( n_0(D) = 0 \).
- For temperature, a variable which looks like \( T - \alpha \log(\Pi) \) (see part 3.4), with \( n_0(T) = 0 \).
- In the NH model, the two additional NH variables, with \( n_0 = 0 \) (and also the auxiliary variable NHX if \( \text{NVDVAR}=4 \)).

About NH variables, some recommendations must be taken into account.

- Vertical divergence and horizontal divergence (and NHX if \( \text{NVDVAR}=4 \)) must be diffused with the same intensity.
- It is recommended not to diffuse pressure departure.
- For NH models with \( \Phi \) and \( gw \) prognostic variables, the way to diffuse \( \Phi \) and \( gw \) remains to be studied.
- \( \log(\Pi) \) should not be diffused (even if some diffusion code is implemented for this variable).

For GFL variables, horizontal diffusion is applied to:

- Specific humidity \( q \) if spectral one, with \( n_0(q) = 0 \).
- Ozone \( O_3 \) if spectral one, with \( n_0(O_3) = 0 \).
- Extra GFL variables if spectral ones, with \( n_0 = 0 \).

For the other advectable GFL variables and for the non-advectable GFL variables, no diffusion is currently coded.

Application to temperature.

Diffusion has to be done on a variable less sensitive to the orography than \( T \), so one chooses a variable looking like \( T - \alpha \log(\Pi) \) with a “good” formulation of \( \alpha \). \( T - \alpha \log(\Pi) \) is diffused instead of \( T \), where \( \alpha \) is a coefficient depending on the altitude and using parameters related to standard atmosphere. Let us denote by \( T_{sST} \) the surface temperature in a standard atmosphere (288.15 K), \( R \) the air constant, \( g \) the acceleration due to gravity, \( \frac{dT}{dz}_{ST} \) the tropospheric temperature vertical gradient in a standard atmosphere (-0.0065 K/m), \( T_{tST} \) the tropopause temperature in a standard atmosphere (217.15 K), and \( \Pi_{ref} \) a reference temperature equal to 1013.25 hPa. If:

\[
(T_{sST})(\Pi/\Pi_{ref})^{(R/g)[dT/dz]_{ST}} > (T_{tST})
\]

the expression of \( \alpha_l \) for the layer number \( l \) is:

\[
\alpha_l = -B_l(R/g) \left( \frac{dT}{dz} \right)_{ST} (T_{sST})(\Pi/\Pi_{ref})^{-(R/g)[dT/dz]_{ST}+1}
\]

where \( B_l \) is used in the definition of pressure on layers and interlayers (hybrid vertical coordinate). In the other cases:

\[
\alpha_l = 0
\]
Temperature horizontal diffusion writes as:

\[
(T^+_{(m,n)} - \alpha \log(\Pi_s)^-_{(m,n)}) = \frac{1}{1 + K_T(n)Dt}(T^-_{(m,n)} - \alpha \log(\Pi_s)^-_{(m,n)})
\]  
(36)

Equation (36) yields (37):

\[
T^+_{(m,n)} = (T^-_{(m,n)} + K_T(n)Dt\alpha \log(\Pi_s)^-_{(m,n)})/(1 + K_T(n)Dt)
\]  
(37)

For temperature \(n_0(T) = 0\).

Application to GMV variables in 2D models.

Horizontal diffusion is performed for vorticity, divergence and equivalent height.

3.5 Other spectral calculations.

* Numerical horizontal diffusion for post-processed fields by FULL-POS. See documentation (IDFPOS) about FULL-POS. This diffusion is done in routine SPOS. Adjoint code is not yet coded in the cycle 37T1.

* Rayleigh friction (ARPEGE/IFS only). The following scheme is discretised in grid-point space for the \(U\)-component of the momentum equation (i.e. the zonal component on the Gaussian grid). In equation \(K_{fric} U\) is a Rayleigh friction which is activated if \(LRFRIC=T\) in NAMCT0. \(K_{fric}\) is a vertical dependent coefficient which is non-zero only for a standard pressure lower than \(\Pi_{fric}\). \(K_{fric}\) is computed in SURAYFRIC according the following formula for a layer \(l\) of standard pressure \(\Pi_{ST}(l)\) lower than \(\Pi_{fric}\):

\[
K_{fric}(l) = \frac{1 - \frac{1}{7} \tanh \left( R_{\text{fr}1} - 7 \log \left( 100000/\Pi_{\text{ST}}(l) \right) \right)}{3 \times 86400}
\]  
(38)

\(R_{\text{fr}1}\) is stored in variable RRFZ1 in YOMDYN. \(K_{fric}\) is stored in variable RKRF in YOMDYN. This scheme has never been used at METEO-FRANCE, but only at ECMWF. It is suited for models having layers in the high stratosphere, with an untilted geometry (i.e. NSTTYP=1).

4 Organigramme of the horizontal diffusion computations.

4.1 Order of spectral computations.

* ARPEGE/IFS: Spectral computations are done in the 3D model in the following order:
  
  - Mass corrector computations (SPCMASCOR).
  - Semi-implicit computations in the hydrostatic model (SPCSI).
  - Semi-implicit computations in the NH model (SPNHSI or SPNHSI GEOGW).
  - Main horizontal diffusion scheme (SPCHOR).
  - Nudging (i.e. linear relaxation towards a pre-defined state) (SPCHOR).
  - Filtering of post-processed fields (SPOS for FULL-POS).

In the 2D model the semi-implicit computations and the main horizontal diffusion scheme are done in SPC2.
6. Horizontal diffusion

* **ALADIN:** Spectral computations are done in the 3D model in the following order:
  
  - Semi-implicit computations in the hydrostatic model (**ESPCSI**).
  - Semi-implicit computations in the NH model (**ESPHSI** or **ESPHSI_GEOGW**).
  - Spectral nudging (coupling) from a coarser model (**ESPS2R** for temporal interpolation providing coupler at the current timestep, **ESPCPL** for spectral relaxation).
  - Main horizontal diffusion scheme (**ESPCHOR**).
  - Filtering of post-processed fields (**SPOS** for FULL-POS).

* **Case of iterative centred-implicit schemes:** For **LPC_FULL**, horizontal diffusion can be called at the last iteration of the iterative algorithm only, or at all iterations (option by default which is recommended).

4.2 Set-up and control routines until STEPO.

**CNT0 ->
* SUYOYMA -->
  - SUCTO
  - SUDYNA
  - SUALLO -> SUALDYN
* SUYOYMB -->
  - SULAP
  - SUDYN -->
    * SUNDIR
    * SUGMRD and SUGMRE [Main diffusion scheme in stretched ARPEGE only]
    * SUHDF (ARPEGE) or SUERDF (ALADIN) [Main diffusion scheme only]
    * SUHDU -> SUHER [Main diffusion scheme in stretched ARPEGE only]
    * SURCORDI
    * SURAYFRIC
* CNT1 -> CNT2 -> CNT3 -> CNT4 -->
  - SUHDU -> SUHER [Main diffusion scheme in stretched ARPEGE only]
  - STEPO --> (organigramme not detailed)

* **Adjoint code:** For adjoint code **STEPO** is replaced by **STEPOAD**, organigramme is slightly different between **CNT0** and **STEPOAD**. For example **CNT3** is replaced by **CNT3AD**, **CNT4** is replaced by **CNT4AD**.

* **Tangent linear code:** For tangent linear code **STEPO** is replaced by **STEPOTL**, organigramme is slightly different between **CNT0** and **STEPOTL**. For example **CNT3** is replaced by **CNT3TL**, **CNT4** is replaced by **CNT4TL**.

* **DFI initialisation:** The sequence **SUHDU → SUHER** is also called by **DFI3** (digital filtering initialisation).

4.3 Direct code under STEPO or tangent linear code under STEPOTL for spectral calculations.

**STEPO or STEPOTL →
  - SPCM (ARPEGE/IFS only) →
    - SPOS (FULL-POS).**
– **TRMTOS** (transposition routine for distributed memory).
– **SPCMASCOR** (mass corrector).
– **SPCSI** (semi-implicit scheme in the hydrostatic model).
– **SPNHSI** or **SPNHSI_GEOGW** (semi-implicit scheme in the NH model).
– **TRSTOM** (transposition routine for distributed memory).
– **SPCHOR** → **MXTURHD** (horizontal diffusion and nudging).

**ESPCM** (ALADIN only) →
– **SPOS** (FULL-POS).
– **TRMTOS** (transposition routine for distributed memory).
– **ESPCSI** (semi-implicit scheme in the hydrostatic model).
– **ESPNSI** or **ESPNSI_GEOGW** (semi-implicit scheme in the NH model).
– **ESPSC2R** and **ESPCPL** (spectral nudging (coupling) from a coarser model).
– **ESPCHOR** → **MXTURHD** (horizontal diffusion).
– **TRSTOM** (transposition routine for distributed memory).

**SPC2M** (ARPEGE/IFS only) → **SPC2** →
– **MXTURS**
– **MXTURE**
– **MXPTMA**
– **BALADSM**

**4.4 Adjoint code under STEPOAD for spectral calculations.**

**STEPOAD** →

**SPCMAD** (ARPEGE/IFS only) →
– **BRPTOB** (communication routine for distributed memory) → **PE2SET**
– **SPCHORAD** → **MXTURHD** and **GATHER** (horizontal diffusion).
– **TRMTOS** (transposition routine for distributed memory).
– **SPCSIAD** (semi-implicit scheme for the hydrostatic model).
– **SPNHSIAD** (semi-implicit scheme for the NH model).
– **TRSTOM** (transposition routine for distributed memory)

**ESPCMAD** (ALADIN only) →
– **TRMTOS** (transposition routine for distributed memory).
– **ESPCHORAD** → **MXTURHD** and **GATHER** (horizontal diffusion).
– **ESPCSIAD** (semi-implicit scheme for the hydrostatic model).
– **ESPNSIAD** (semi-implicit scheme for the NH model).
– **TRSTOM** (transposition routine for distributed memory).

**SPC2MAD** (ARPEGE/IFS only) → **SPC2AD** →
– **MXTURS**
– **MXTURE**
– **MXPTMA**
4.5 Rayleigh friction done in grid-point space (ARPEGE only).

Rayleigh friction is done in the grid-point calculations (CPEULDYN, LATTEX) if LEGWWMS=F. There is some Rayleigh diffusion code (done in Fourier space) under the arborescence

LTINV -> ASRE1 -> ASRE1B -> FSPGL_INT -> adiab/FSPGLH

activated only if LEGWWMS=T. Adjoint code is not yet coded.

4.6 Action done by set-up routines.

- **SUHDIR**: computes the coefficients HDIR..., HRDIR..., RDAMP....
- **SUHDF**: computes the arrays RDI..., RDS....
- **SUEHDF**: for ALADIN; computes the arrays RDI...E, RDS...E.
- **SURCORDI**: computes the array RCORDIT.
- **SURAYFRIC**: computes the setup of the Rayleigh friction.
- **SUGMRD**: computes the coefficients \(a_0\) and \(a_1\) of the first degree polynomial defining \(M\).
- **SUGMRE**: computes coefficients of the matrix defining \(M\) and stores them in array GMR of YOMDYN.
- **SUHDU**: fills RDHI of YOMDYN. Array RDHI contains the non-zero diagonals of the two triangular bidiagonal matrices, the product of which defines the inverse of horizontal diffusion operator (right-hand side member of equation (22)).

4.7 Control routines.

CNT0, CNT1, CNT2, CNT3, CNT4, CNT3AD, CNT4AD, CNT3TL, CNT4TL, STEPO, STEPOAD, STEPOTL are control routines. STEPO (resp. STEPOAD, STEPOTL) manages one direct (resp. adjoint, tangent linear) integration timestep.

4.8 Action done by routines under STEPO, STEPOAD or STEPOTL.

* Algorithmic routines:

- **SPCHOR**: spectral horizontal diffusion schemes (except Rayleigh friction) and nudging in the 3D model.
- **SPC2**: spectral space computations, including the semi-implicit scheme and all horizontal diffusion schemes in the 2D model.
- **SPCM**: interface routine for SPCHOR.
- **SPC2M**: interface routine for SPC2.
- **ESPCM, ESPC2M, ESPCHOR, ESPC2 are called instead of SPCM, SPC2M, SPCHOR, SPC2 in ALADIN.
- These routines have adjoints (same names + “AD”). Horizontal diffusion on post-processed fields and nudging are not yet coded in SPCHORAD.
- **BALADSM**: solve linear balance equation in spectral space to convert vorticity into geopotential (used for vorticity 2D equation only).
6. Horizontal diffusion

* Linear algebra routines (project XLA/ALGOR):
  
  - **MXTURE**: called by SPC2 and inverts bidiagonal or tridiagonal triangular linear systems.
  - **MXTURHD**: a version of MXTURE well suited for horizontal diffusion.
  - **MXTURS**: combination of two calls to MXTURE to invert a set of symmetric tridiagonal or pentadiagonal matrices, the decomposition $LU$ of which is known. Called by SPC2.
  - **MXPTMA**: products of a set of tridiagonal or pentadiagonal matrices by a set of matrices or vectors. Called by SPC2.
  - **SUHER**: performs the $LU$ factorisations of a set of non-symmetric tridiagonal matrices.

* Distributed memory routines (for ex. transposition routines) doing communications between processors:
  
  The list of transposition and communication routines used in horizontal diffusion scheme is the following: TRSTOM, TRMTOS, BRPTOB, MPL_ALLREDUCE, GATHER_T, PE2SET and SET2PE; see documentation (IDDM) about distributed memory features for the action of these routines.

5 Some distributed memory features.

  - The total number of processors involved in the A-level parallelisation is NPRTRW.
  - The total number of processors involved in the B-level parallelisation is NPRTRV.
  - One processor treats only a subset of zonal wave numbers.
  - A call to SPCHOR can treat several wave numbers (currently all the zonal wave numbers treated by the current processor). For computations needing matricial operations (geographical component of unified diffusion scheme for example) calculations are split zonal wave number by zonal wave number inside SPCHOR.
  - **SPOS**: same rules as in SPCHOR.
  - For 2D model a call to SPC2 treats only one zonal wave number.
  - For the 3D model there is a splitting into NFLEV layers packets when NPRTRV>1. One B-set processor treats NFLEV layers; an exception is done in the adjoint code SPCHORAD when diffusing variables using the temperature array, some calculations on all the NFLEV layers are necessary to compute a vertical integral which does not exist in the direct code. That means that additional transpositions (TRSTOM in the direct code) are necessary between the semi-implicit calculations of SPCSI and the horizontal diffusion calculations of SPCHOR to convert the fields from the NFLEV structure required in the semi-implicit calculations to the NFLEV structure required in the horizontal diffusion calculations.
  - **ALADIN** does not completely follow ARPEGE for the second level of distribution: the NFLEV structure is not yet implemented in ESPCHOR, and ESPCHOR works like ESPCSI. That explains why the order of calls is SPCSI, TRSTOM, SPCHOR in ARPEGE, but ESPCSI, ESPCHOR, TRSTOM in ALADIN.

6 Module and namelist variables.

YEM... and YOM... are modules in cycle 37T1.
6. Horizontal diffusion

6.1 YEMDYN.

RDIVORE, RDIIDIVE, RDITE, RDIEGFL, RDISPE, RDIPDE, RDIVDE: the same as RDIVOR, RDIIDIV, RDIT, RDIGFL, RDISP, RDIPD, RDIVD but for ALADIN.

RDSVORE, RDSDIV, RDSVD: the same as RDSVOR, RDSDIV, RDSVD: but for ALADIN.

6.2 YEMGE0.

* Defining the geometry in ALADIN:

  - ELX and ELY: zonal and meridian lengths of the ALADIN domain ($L_x$ and $L_y$).
  - EDELX (resp. EDELY): zonal (resp. meridian) grid-point meshsize at the center of the ALADIN domain (is different from $\Delta X_{gp, zonal}$ (resp. $\Delta X_{gp, merid}$) if $M$ is different from 1 at the center of the ALADIN domain).

6.3 YOMCT0.

* General environment variables:

  - NCONF: configuration.

* Horizontal diffusion:


* Namelist NAMCT0: Variables NCONF, LECMWF and LRFRIC are in namelist NAMCT0.

* Distributed memory environment: The following variables (some of them are in namelist NAMPAR0) can be useful when running distributed memory jobs: NPRGPNs, NPRGPEW, NPRTRW, NPRTRV, NPROC, NPRTRN, NPRTRNS, LMPOFF, LMPDIAG, NOUTPUT. See documentation (IDDM) for more details about these variables.

6.4 YOMDIM.

Contains dimensioning variables. The following variables are useful for spectral computations:

  - NFLEVG: total number of model layers. DM-global.
  - NFLEVLMX: the maximum possible value of NFLEVL. DM-global.
  - NFLSUR: odd number surdimension for NFLEVL.
• **NSMAX**: model main truncation. DM-global.
• **NMSMAX**: truncation order in longitude, only used in ALADIN. DM-global.
• **NSEFRE**: number of degrees of freedom in the spectral space. DM-global.
• **NSPECG**: number of complex spectral coefficients. DM-global.
• **NSPEC2G**: is equal to 2*NSPECG and is used to dimension DM-global spectral arrays.
• **NSPEC**: number of complex spectral coefficients treated by the current processor, DM-local version of NSPECG.
• **NSPEC2**: is equal to 2*NSPEC and is used to dimension DM-local spectral arrays.
• **NSPEC2MX**: the maximum possible value of NSPEC2. DM-global.
• **NUMP**: number of spectral waves handled by this processor. DM-local.

* Namelist NAMDIM: The following variables of YOMDIM are in namelist NAMDIM: NFLEVG,NSMAX,NMSMAX.

6.5 YOMDYNA.

* **SLHD** horizontal diffusion scheme in ARPEGE or ALADIN for GMV and GMVS variables:
  * **LSLHD**: Internal model switch for semi-Lagrangian diffusion computation.
  * **LSLHD_W**: switch for SLHD of horizontal wind.
  * **LSLHD_T**: switch for SLHD of temperature.
  * **LSLHD_SPD**: switch for SLHD of (NH) pressure departure.
  * **LSLHD_SVD**: switch for SLHD of (NH) vertical divergence.
  * **LSLHD_GLD**: use old SLHD interpolator (cubic Lagrange mixed with linear).
  * **LSLHD_STATIC**: do not diagnose kappa from horizontal flow deformation, use static value instead: SLHDKMIN for undiffused fields; SLHDKMAX for diffused fields
  * **LSLHDQUAD**: internal model switch indicating need to precompute quadratic weights.
  * **SLHDKMIN**: minimum value for the kappa function (kappa_min).
  * **SLHDKMAX**: maximum value for the kappa function (kappa_max).
  * **SLHDEPH**: dimensionless strength of horizontal Laplacian smoothing (C_slhdepsh).
  * **SLHDEPSV**: dimensionless strength of vertical Laplacian smoothing.

All these variables (excepted LSLHD and LSLHDQUAD) are in NAM Dyna.

6.6 YOMDYN.

* Main horizontal diffusion scheme in ARPEGE or ALADIN:
  • **HDIR[X]**: 1/(Ω h_X), horizontal diffusion constant for variable X, in seconds (DM-global). Currently existing variables: HDIRVOR (vorticity), HDIRDIV (divergence), HDIRT (temperature), HDIRQ (humidity), HDIRO3 (ozone), HDIRSV (extra GFL variables), HDIRSP (logarithm of hydrostatic surface pressure), HDIRPD (NH pressure departure variable), HDIRVD (NH vertical divergence variable).
  • **HRDIR[X]**: (Ω h_X) (i.e. 1/HDIR[X]).
  • **FRANDH**: x_0, threshold in formula defining f(n, N, n_0(X), x_0, r) (equation 12) between 0 and 1 (default value = 0). DM-global.
  • **REXPDPD**: r, order of the horizontal diffusion operator (default value = 4). DM-global.
6. Horizontal diffusion

- **SLEVDH**: \( y_0 \), threshold in formula defining \( g(l) \) (equation (14)) (default value = 1). DM-global.
- **SLEVDH2**: \( y_1 \), second threshold in formula defining \( g(l) \) (equations (31), (32) and (33)) (default value = 0.01). DM-global.
- **SLEVDH3**: \( y_3 \), threshold in formula defining \( g(l) \) (equation (14)) (default value = 1/1013.25). DM-global.
- **NSREFDH**: \( N_2 \), second reference wave number (in equations (31), (32) and (33)) (default value = \( NSMAX \)). DM-global.
- **RDI[X]**: Contains \( K_X \) (see equation (11)). A-set DM-global, B-set DM-local (except for \( RDITG \) which is DM-global). Currently existing variables: \( RDIVOR \) (vorticity), \( RDISSD \) (divergence), \( RDITG \) (temperature), \( RDIGFL \) (GFL variables), \( RDISP \) (logarithm of hydrostatic surface pressure), \( RDIPD \) (NH pressure departure variable), \( RDIVD \) (NH vertical divergence variable). They are used in ARPEGE only.
- **RCORDIT**: Contains \( \alpha \) (equations (34) and (36)). DM-global.
- Arrays \( RDI... \), \( RCORDIT \) are computed in subroutine \textsc{SURCORDI}.
- **RRDXTAU**: Ratio “Horizontal grid size/e-folding time at the smallest resolved scale” (variable \( r_{\text{dxtau}} \)). Default value is 123. at METEO-FRANCE.
- **RDAMP[X]**: local enhancing coefficient for diffusion of variable \( [X] \) (variable \( r_{\text{damphds}} \)).
- **LSTRHD**: If \( LSTRHD=F \) the mapping factor \( M \) is not taken in account in the main horizontal diffusion scheme (well suited for unstretched ARPEGE or for ALADIN). If \( LSTRHD=T \) the mapping factor \( M \) is taken in account in the main horizontal diffusion scheme (well suited for stretched ARPEGE). Default value is \( T \) in the stretched version of ARPEGE, and \( F \) in the other cases. DM-global.
- **RDHI**: Contains non-zero diagonals of the factorisation of the matricial operator used in the stretched geometry horizontal diffusion scheme. Array \( RDHI \) is computed in subroutine \textsc{SUHDU}. DM-local.
- **GMR**: Contains the matrix for spectral multiplication by \( M \). \( GMR \) is computed in subroutine \textsc{SUGMRE}. DM-local.

* Additional quantities used if SLHD scheme is activated:
- **HDSR[X]**: \( 1/(\Omega h_{hds}X) \), horizontal diffusion constant for variable \( X \), in seconds (DM-global). Currently existing variables: \( HDSRVOR \) (vorticity), \( HDSRDIV \) (divergence), \( HDSRVD \) (NH vertical divergence variable).
- **HRDSR[X]**: \( (\Omega h_{hds}X) \).
- **REXPDH**: \( s \), order of the additional horizontal diffusion operator (default value = 6). DM-global.
- **SLEVDHS**: \( y_{\text{hds}0} \) threshold in formula defining \( g_{\text{hds}}(l) \) (equation (30)). DM-global.
- **RDS[X]**: Contains \( K_{hds}X \) (see equation (29)). A-set DM-global, B-set DM-local. Currently existing variables: \( RDSVOR \) (vorticity), \( RDSDIV \) (divergence). They are used in ARPEGE only.
- Arrays \( RDS... \) are computed in subroutine \textsc{SUHDF}.
- **RDAMPHD**: coefficient \( r_{\text{damp}\text{hds}} \) (ratio \( HRDSRDIV/HRDIRDIV \)). Default value is 1.5 at METEO-FRANCE.
- **RDAMP[X]**: local enhancing coefficient for diffusion of variable \( [X] \).
- **SDRED**: coefficient \( s_{\text{dred}} \) between 0 and 1 reducing \( g(l) \) in formula (23) in case of SLHD diffusion.
- **RDHS**: cf. \( RDHI \) but for the SLHD diffusion in stretched ARPEGE. Array \( RDHS \) is computed in subroutine \textsc{SUHDU}. DM-local.
6. Horizontal diffusion

* Rayleigh friction:

  - RKRF : coefficient of Rayleigh friction (K_ray).
  - NMAKLEVRF : maximum level for which Rayleigh friction is applied. If no
    Rayleigh friction is applied, we set NMAKLEVRF=0
  - RRFZ1 : reference value R_fz1 for height of profile
  - RRFPFLM : pressure limit Pi_rfric; no Rayleigh friction for Pi > RRFPFLM

* Namelist NAMDYN: Some of the above YOMDYN variables are in namelist NAMDYN.

6.7 YOMGEM and namelist NAMGEM.

  - RNLGINC: factor of linear/quadratic Gaussian grid rnlgin (0 for linear, 1 for
    quadratic). DM-global.

6.8 YOMGFL.

* SLHD horizontal diffusion scheme in ARPEGE or ALADIN for
  GFL variables: Y[X]%LSLHD in YOMGFL, Y[X]_NL%LSLHD in NAMGFL
  activates the SLHD diffusion for the GFL variable [X].

6.9 YOMMP.

Variables specific to the distributed memory environment. For more details see documenta-

6.10 YOMMPG.

Variables specific to the distributed memory environment. DM-global variables mainly
  used to handle reading and writing of grib data. For more details see documentation
  (IDDM) about distributed memory features.

6.11 YOMNUD and namelist NAMNUD.

Variables involved in nudging, not described in detail.


6.12 YOMSP.

* SPDIV, SPVOR, SPHV, SPT, SPSPD, SPSVD, SPNHX, SPGFL, SPQ, SPO3, SPL, SPI, SPSP (in YOMSP):
  Spectral arrays containing respectively divergence, vorticity, GMV thermodynamic variables, temperature, pressure
  departure variable, vertical divergence variable, X-term variable for d4, GFL variables,
  humidity, ozone, extra GFL variables (passive scalars), logarithm of surface hydrostatic
  pressure. SPT is associated with the same piece of memory as SPHV. SPQ, SPO3,
  SPL and SPI are associated with the same piece of memory as SPGFL. These arrays
  are DM-local, they contain all the NFLEVG layers but only the NSPEC2 spectral
  coefficients treated by the current processor. No variable in namelist.
6. Horizontal diffusion

6.13 Functions of SUHDF and FCHDIF.

- **PDISPE**: Function $f(n, N, 0, x_0, r)$ for $x_0 = \text{FRANDH}$, $r$ order (equation 12).
- **PDISPVOR**: Function $f(n, N, 2, x_0, r)$ for $x_0 = \text{FRANDH}$, $r$ order (equation 12).
- **PDILEV**: Function $g(l)$ used at ECMWF (LECMWF = .T.) and also in ALADIN; in the configurations of ARPEGE used at METEO-FRANCE the same information is put in $\text{ZDILEV}$ (routine SUHDF).
- **PDISPEL**, **PDISPEE** and **PDISPEX** are used only at ECMWF (not detailed, see SUHDF).
- **BDISPE**: Function $f(m, n, N, 0, x_0, r)$ for $x_0 = \text{FRANDH}$, $r$ order (equation 18). This function is defined in module FCHDIF.

7 References.

7.1 Publications:


7.2 Internal notes:

- *(TDECTEC) 2010: IFS technical documentation (CY36R1). Part VI: technical and computational procedures. Available at “http://www.ecmwf.int/research/ifsdocs/".*
- *(IDEUL) Yessad, K., 2011: Integration of the model equations, and Eulerian dynamics, in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/").*
• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911. (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).


• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911. (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).


• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911. (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).


• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911. (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).


• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911. (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).


• (IDSI) Yessad, K., 2011: Semi-implicit spectral computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDTS) Yessad, K., 2011: Spectral transforms in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDSL) Yessad, K., 2011: Semi-Lagrangian computations in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDFPOS) Yessad, K., 2011: FULL-POS in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDRD) Yessad, K., 2011: Sphere to sphere transforms in spectral space in the cycle 37T1 of ARPEGE/IFS: configuration 911. (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

• (IDDM) Yessad, K., 2011: Distributed memory features in the cycle 37T1 of ARPEGE/IFS (internal note, available on the intranet server “http://www.cnrm.meteo.fr/gmapdoc/*”).

Gravity wave drag

1 Parametrization of orographic gravity wave drag

1.1 Calculation at the surface

The momentum flux due to a wave excited by the displacement of a large-scale flow, vertically, on a small scale mountain, is given by:

\[ \vec{\tau}_s = \rho_s N_s \vec{v}_s K_g h_s \]  

(1)

This vector corresponds in the code to PSTRDU and PSTRDV at level KLEV. It represents the drag exerted by the lowest atmospheric layer on surface. In the right-hand side:

- \( \rho_s \) is the density at surface (PGWDCS)
- \( \vec{v}_s \) is the effective wind at surface (see section 2.1)
- \( N_s \) is the effective Brunt-Väisälä frequency at surface (see section 2.1)
- \( h_s \) is the standard deviation of unresolved orography
- \( K_g \) is a dimensionless coefficient for the adjustment of the parametrization (GWDSE)

1.2 Calculation at a given level

The momentum flux of a wave of amplitude \( a \) and wavelength \( \lambda \) is given by:

\[ \tau = \rho N U \frac{a^2}{\lambda} \]

where \( U \) is the projection of wind \( \vec{v} \) on the effective wind at surface \( \vec{v}_s \):

\[ U = \frac{\vec{v} \cdot \vec{v}_s}{\|\vec{v}_s\|} \]

As long as the wave remains linear (low amplitude compared to the wavelength), there is no interaction with the large-scale flow (surge or saturation) and \( \tau \) remains constant on the vertical. It is admitted moreover that:
7. Gravity wave drag

- the wave breaks at the surface.
- the condition of saturation is: $\lambda = CU/N$ (Lindzen criterion) where $C$ is a constant on the vertical.

Thus, posing:

$$\Gamma = \frac{\rho_s}{\rho_s} \left( \frac{U}{U_s} \right)^3 \frac{N_s}{N} \quad \text{(ZRAPP)}$$

there will be saturation if $\tau = \Gamma \tau_s$. One thus has $\tau_L = \min(\Gamma_s, \tau_{l+1})$. In fact, one defines a factor of proportionality to $\tau_s$. One takes at surface $\Gamma_L = 1$, and one goes up while posing $\Gamma_l = \min(\Gamma, \Gamma_{l+1})$. As soon as $\Gamma_l \leq 0$, one poses $\Gamma_l = 0$ and one sets it to zero above (there is no more flux because the wave has been completely absorbed). That occurs as soon as wind $\vec{v}_l$ forms an angle higher than 90° with the surface wind, or, at last, at the top of the atmosphere ($\rho = 0$).

2 Refinements for orographic gravity wave drag

2.1 Effective surface wind

The formulas above utilize the surface wind $\vec{v}_s$. In the model, the surface wind is zero, and using the lowest model level wind would introduce a dependence of the scheme with the vertical discretization. It is supposed that the wave generating orography induces a boundary layer thickness $H = K_h h_s$ where $K_h$ is an empirical dimensionless coefficient (HOBST). One calculates the average parameters in this mountainous boundary layer:

$$\vec{v}_s = \vec{v} = \frac{1}{H} \int_0^H \vec{v} dz \quad \text{(ZSUMU,ZSUMV)}$$

in the same way for $< N_s^2/(\rho, g)^2 > \quad \text{(ZSUMF)}$. To maintain the continuity of $\Gamma_l$, one modifies the vertical profile of wind in the mountainous boundary layer:

$$\vec{v}_l = \vec{v}_s (1 - \Delta p_l/\Delta p_{\text{obst}}) + \vec{v}_l \Delta p_l/\Delta p_{\text{obst}} \quad \text{(ZU,ZV)}$$

where $\Delta p_l$ is the thickness in pressure between the surface and level $l$, and $\Delta p_{\text{obst}}$ the thickness between the surface and the top of the mountainous boundary layer. The same way is applied for the standardized Brunt-Väisälä frequency (ZNFNO).

2.2 Mountain anisotropy

The theory supporting this parametrization makes the assumption that the flow is two-dimensional. In fact the sub-grid scale orography can be represented by the anisotropic coefficient $\gamma^2$ (PVRLAN) which is the ratio of the two eigenvalues of the tensor of variance of unresolved orography, and angle $\psi$ (PVRLDI) which are the direction (null in the direction of increasing pseudo-longitudes) of the first eigenvector of this tensor. The surface stress is not co-linear to the surface wind. One calculates a fictitious surface wind...
Gravity wave drag

\( (u_fs, v_fs) \), in the code (ZUSUR, ZVSUR), which would give in the isotropic case same flux as the one obtained with \( (u_s, v_s) \):

\[
\begin{align*}
u_{fs} &= Au_s + D(u_s \cos 2\psi + v_s \sin 2\psi) \\
u_{fs} &= Av_s + D(u_s \sin 2\psi - v_s \cos 2\psi)
\end{align*}
\]

\[
A = \gamma^2 + \frac{1}{2\pi} (4(1 - \gamma)(1 + \alpha_1 \gamma) - \gamma(1 + \alpha_2) \ln \gamma)
\]

\[
D = \frac{1}{2\pi} (4(1 - \gamma)(1 + \delta_1 \gamma) + 3\gamma(1 + \delta_2) \ln \gamma)
\]

Coefficients \( \alpha_1, \alpha_2, \delta_1 \), and \( \delta_2 \) are calculated from elliptic integrals and are about 1.44, 0.22, 0.67 and 0.44 respectively (more accurate values are used in the code, based on asymptotic approximations). One thus uses \( \vec{v}_{fs} \) to calculate the surface flux and the projection of the wind in the free atmosphere in the place of \( \vec{v}_s \) in Equations (1) and (2).

2.3 Resonance

This theory assumes the wave does not deposit all its momentum on the saturation level, but is reflected downwards, and can, after a certain number of reflections, produce a resonance. The theory has been developed by Clark and Peltier (1977). They showed from very simplified considerations, and also with results from simulations with a mesoscale model, that resonant amplification takes place when altitude \( z \) of the critical level checks:

\[
N \frac{U}{z} \equiv \frac{3\pi}{2} [2\pi]
\]

One defines a function of resonance \( f \), periodic and of average value 1, as a reference for the parametrization:

\[
f(\theta) = |1 - iK_a e^{i\theta} - K_a^2 e^{2i\theta} + \cdots| = (1 + 2K_a \sin \theta + K_a^2)^{-\frac{1}{2}}
\]

where \( K_a \) is an empirical dimensionless parameter (GWDAMP). This function reaches its maximum \( 1/(1 - K_a) \) for \( \theta = 3\pi/2 \) (amplification) and its minimum \( 1/(1 + K_a) \) for \( \theta = \pi/2 \) (destruction).

In the parametrization of the friction of the orographic gravity waves, one includes the assumption of resonance as formulated above. One locates \( z_{crit} \) the first critical level reached (\( \Gamma_l \) lower than 1, for the first time), where the wave is likely to be reflected in phase with itself; one evaluates amplification by resonance by calculating:

\[
\theta = \int_0^{z_{crit}} \frac{N}{U} \, dz
\]

One deduces \( f(\theta) \) from it. One defines \( \Gamma_{crit} \) as \( \min(1, f(\theta)) \), and one modifies factor \( \Gamma_l \):

- between the surface and the critical level, one takes a linear profile with pressure, with \( f(\theta) \) at the surface and \( \Gamma_{crit} \) at the critical level.
- above the critical level, one takes \( \min(\Gamma_l, \Gamma_{crit}) \).

One does not take into account the phenomena of reflection on the levels above the first critical level.
2.4 Trapping

In the case of a neutral or unstable stratification of the atmosphere, the Brunt-Väisälä frequency goes to zero as well as the drag due to the orographic gravity waves. One seeks the first level starting from bottom where \( N \) is zero or changes sign. If this level is below the critical level of resonance, one does not apply the modification of the profile by resonance (by imposing \( f(\theta) = 1 \)). Profile \( \Gamma_l \) is modified:

- above level \( N = 0 \) by setting it to zero.
- below level \( N = 0 \), by cutting off a linear function of pressure in order not to modify the surface stress and to set it to zero at level \( N = 0 \).

2.5 Sub-grid orography

The effect of blocking of flow by the succession of the mountains and the valleys was taken into account in certain models by envelope orography, method which consisted in replacing the average orography by an increase of a fraction of the standard deviation of the unresolved fluctuations. Lott and Miller (1995) using the results of the PYREX experiment, proposed to better define the local impact of the unresolved mountains by means of an increase in friction by gravity waves only in the model levels occupied by the actual orography. The adaptation to ARPEGE consists in multiplying coefficient \( \Gamma_l \) (and consequently tension \( \vec{\tau} \)) by:

\[
1 + a_d \sqrt{\frac{1 - \frac{z}{b_d H}}{1 + \frac{z}{H}}} ^3
\]

on the levels whose altitude \( z \) is lower than \( b_d H \), where \( H \) is the thickness of boundary layer calculated above. From projection \( U_s \) of the effective surface wind \( \vec{v}_s \) onto the fictitious wind (which takes into account the anisotropy of orography) \( \vec{v}_{fs} \), one introduces a standardized thickness \( H_N \):

\[
H_N = H \frac{N_{s} \| \vec{v}_{fs} \|}{U_s^2}
\]

Two empirical parameters are introduced: a factor of friction at the surface \( K_d \) (GWDCD) and a critical value for standardized thickness \( H_{NC} \) (1/GWDBC). When standardized thickness \( H_N \) is higher than the critical value, coefficients \( a_d \) (ZAA) and \( b_d \) (ZBB) are given by:

\[
a_d = K_d \frac{\| \vec{v}_{fs} \|^2}{U_s^2} \frac{H_N - H_{NC}}{H_N^2} \quad \text{and} \quad b_d = \frac{H_N - H_{NC}}{H_N}
\]

In the opposite case, \( a_d \) and \( b_d \) are zero and the stress is not modified.
2.6 The lift effect

Here one takes into account the transverse force exerted on the wind attacking an un-resolved mountain due to the effect of terrestrial rotation combined with the deviation of the flow which circumvents the obstacle (in an asymmetrical way). In theory it would be necessary that this force is exerted transversely to a hypothetical flow undisturbed by the obstacle (geostrophic wind for example). But this calculation would be difficult to carry out in ARPEGE: the force defined by this way would produce a work, a property contrary with the principle of the effect of rising (hereafter called lift), and whose conjugation with the effect of form-drag of the sub-grid orography (see section 2.5) would be unexpected. One could also have constrained this force to be exactly orthogonal with the forces due to orographic friction previously defined in this chapter, but this would have answered only the second of the two criticisms formulated above. As a simplification here, we just have defined a (positive) complement to Coriolis force, with an intensity proportional to the volume of the sub-grid mountain.

This force of lift is exerted on the volume of atmosphere of the model between the ground and \( H \) and varies according to \( (1 - z/H)/(1 + z/H) \) to remain in agreement with the choice of the shape of the sub-grid mountain implicitly made in section 2.5. There is a multiplying coefficient \( L_t \) whose intensity corresponds to the vertical integral of the effects layer by layer (and thus with the total stress applied): \( GWDLT \). For a value 1 of this coefficient, one obtains on average a doubling of the “Coriolis” effect on depth \( H \). From the phenomenological point of view, this amounts (when \( GWDLT = 1 \)) to increasing the rotational effects of the mountain solved by what would give an effect of envelope orography with coefficient \( K_h \), without having now the effects of vertical displacement and blocking associated with the modification with the large-scale orography seen by the model.

The effect of lift is calculated with gravity waves by convenience and because the effects of the sub-grid orography intervene there in a way close to what is made in the section 2.5. But as this additional force physically does not have anything common with the remainder of calculations previously described, it will be added in an absolutely independent way. In particular it will not undergo to the numerical securities described hereafter. By applying it without caution to a wind which is perhaps not the average of the vector wind over the duration of the time step, this force would consume or produce energy, even marginally, and this could cause problem for very long integrations. A trapezoidal split-implicit calculation is thus carried out so that the wind vector conserves its module, when this effect is the only one taken into account during the time step.

All this leads to the following formulations. For \( z < H \):

\[
f^* = L_t f \left( \frac{1 - z/H}{1 + z/H} \right) \frac{K_h}{2 \ln(2) - 1}
\]

\[
\left( \frac{\partial u}{\partial t} \right)_{lift} = \frac{f^*}{1 + (\Delta t f^*/2)^2} (u - u \Delta t f^*/2)
\]

\[
\left( \frac{\partial v}{\partial t} \right)_{lift} = -\frac{f^*}{1 + (\Delta t f^*/2)^2} (u + v \Delta t f^*/2)
\]

For \( z > H \) there is no lift force.
3 Numerical securities in orography waves

3.1 Implicit formulation versus surface wind

All fluxes are proportional to projection $U_s$ of wind $\vec{v}_s$ on $\vec{v}_{fs}$, therefore the tendency of $U_s$ also:

$$\frac{\partial U_s}{\partial t} = -KU_s$$

However, for large time steps the system can become unstable. Consequently the following relations are used ($U_s$ indicating the value with $t$ and $U_s^+$ with $t + \Delta t$):

$$\frac{U_s^+ - U_s}{\Delta t} = -KU_s^+$$

$$\frac{U_s^+ - U_s}{\Delta t} = -\dot{K}U_s$$

One replaces $K$ by $\dot{K} = \frac{K}{1 + \frac{K}{\Delta t}}$ (split-implicit scheme). This is equivalent to multiplying all relations of proportionality between flux at a given level and $U_s$ by:

$$\frac{1}{1 - \Delta t \frac{\partial U_s}{\partial t}}$$

$\frac{\partial U_s}{\partial t}$ is calculated by integrating the vertical derivative of flux inside the mountainous boundary layer defined in the section 2.1.

3.2 Implicit advective formulation

One has just seen how to implicitly take into account the dependence of the tendency of $\vec{v}$ on $\vec{v}_s$; in fact, one has treated only component of the wind in the direction of $\vec{v}_{fs}$, but the tendency of wind is zero in the direction orthogonal to $\vec{v}_{fs}$. However one does not take into account the dependence on $\vec{v}$ itself, since it does not appear in a linear way in the equations. One wants to discretize:

$$\frac{\partial U}{\partial t} = -g \frac{\partial \tau}{\partial p}$$

where $U$ is the projection of $\vec{v}$ on $\vec{v}_{fs}$. Here $\tau = \Gamma \tau_s$ and $\Gamma$ depends on $U$ (Equation (2) and corrections). One poses:

$$\tau = \frac{1}{g} U \Gamma_l$$

If one regards the phenomenon as a vertical advection of moment, $\Gamma_l$ plays the role a vertical velocity (or of a mass flux, by analogy with the convection). One introduces the dimensionless quantity, similar to a vertical CFL criterion:

$$A_l = \frac{\Gamma_l \Delta t}{\delta p_l}$$
One discretizes the pseudo-advection with an upstream implicit scheme:

\[ U^+_t - U_t = -(A_t U^+_t - A_{t-1} U^+_{t-1}) \]

which one can write as:

\[(1 + A_t)(U^+_t - U_t) = -A_t U_t + A_{t-1} U_{t-1} + A_{t-1}(U^+_{t-1} - U_{t-1}) \]

If one knows the implicit tendency on level \( l - 1 \), \( U^+_{l-1} - U_{l-1} \), and the explicit tendency calculated by parametrization on level \( l \), \( -A_t U_t + A_{t-1} U_{t-1} \), one deduces from it the implicit tendency on level \( l \). One starts at level \( l = 1 \) since the tendency is null at the top and one carries out a downward loop. One will replace stress \( \tau \) by a stress \( \tilde{\tau} \) such as:

\[ \frac{U^+ - U}{\Delta t} = -g \frac{\partial \tilde{\tau}}{\partial p} \]

It comes:

\[ \tilde{\tau}_0 = 0 \]

\[ \tilde{\tau}_l = \tilde{\tau}_{l-1} - \frac{1}{1 + A_t}(\tilde{\tau}_{l-1} - \tau_l) \]

In the code, \( \frac{1}{1 + A_t} \) is called ZALPHA.

4 Parametrization of convective gravity wave drag

This parametrization of impact of tropospheric convection on the stratospheric jets was introduced by Bossuet et al. (1998).

4.1 Calculations at the top of the cloud

In a way similar to the flow over a mountain, the convective zones produce vertical small-scale velocities which generate gravity waves in the stable atmosphere layers located above. The parametrization described here supposes that the intensity of these waves is a function of the precipitation flux at the base of the cloud (as an index of the intensity of the convection). It is supposed that these waves move vertically in the reference frame related to the base of the cloud, and that, similarly with the waves of orographic origin, the interaction with the large-scale flow occurs only in the direction of the absolute wind at the top of the cloud, level of the source of the gravity waves.

The momentum flux due to a wave excited by convective motions can be empirically represented by:

\[ \tilde{\tau}_s = K_c P_{\text{con}} \frac{\vec{v}_s}{\|\vec{v}_s\|} \]  

(3)

This vector corresponds in the code to ZSTUS and ZSTVS. It also corresponds to vector PSTRCU and PSTRCV at level KIUPC (stress at the top of the cloud). This vector represents the stress exerted by a layer on the layer below. In the right-hand side:

- \( K_c \) is a tuning parameter
- \( P_{\text{con}} \) is convective precipitation (rain+snow)
- \( \vec{v}_s \) is wind at the top of the cloud
4.2 Calculations above the cloud

As long as the wave remains linear (low amplitude compared with the wavelength), there is no interaction with the large-scale flow (surge or saturation) and $\tau$ remains constant on the vertical. By using Lindzen criterion for the surge, one introduces, in the same way as for the orographic waves, profile $\Gamma$:

$$\Gamma = \frac{U}{U_s} \left( \frac{U}{U_s} \right)^3 N_s \frac{N}{N_s} \quad \text{(ZRAPP)}$$

where $U$ is projection on the wind at the top of the cloud of the relative wind in the reference frame related to the base of the cloud:

$$U = \left( \vec{v} - \vec{v}_b \right) \cdot \vec{v}_s \left\| \vec{v}_s \right\|$$

There will be saturation if $\tau = \Gamma \tau_s$. One thus has $\tau_l = \min(\Gamma \tau_s, \tau_{l+1})$. In fact, one defines a factor of proportionality to $\tau_s$. One takes at the top of cloud $\Gamma_s = 1$, and one goes up while posing $\Gamma_l = \min(\Gamma, \Gamma_{l+1})$. As soon as $\Gamma_l \leq 0$, one poses $\Gamma_l = 0$ and one sets it to zero above (flux is zero because the wave has been completely absorbed). That occurs as soon as the wind at level $l$ forms an angle greater than 90° with the wind at the top of the cloud (in the mobile reference frame), or, at last, at the top of atmosphere ($\rho = 0$).

4.3 Calculations inside the cloud

In the case of the waves of orographic origin, the momentum borrowed from the atmosphere is yielded to the earth, according to the principle of conservation. In the case here, there is no exchange of momentum with the ground, and the quantity must be yielded to the lower atmospheric layers. It is reasonable to think that the stable layers located under the base of the cloud are not affected either. One thus takes a linear variation of flow between value $\tau_s$ at the top and zero at the base.

In the above calculation, one considers that if there are several convective layers disjointed, one takes the bottom of the lowest layer and the top of the highest layer.

5 Numerical securities for convective waves

The temporal evolution of the wind at level $l$ depends, via $U_l$ on the wind itself. As a consequence, an explicit formulation of the temporal advance can lead to numerical instabilities. It would be too complicated to solve the full system with an implicit scheme because it is non-linear (with cubic terms and thresholds) and multidimensional. In fact one wants to discretize:

$$\frac{\partial U}{\partial t} = -g \frac{\partial \tau}{\partial p}$$

where $U$ is the projection of the relative wind. We have $\tau = \Gamma_l \tau_s$, and $\Gamma_l$ depends on $U$ (Equation (1)). One poses:

$$\tau_l = \frac{1}{g} U_l M_l$$
7. Gravity wave drag

If one considers the phenomenon as a vertical advection of momentum, $M_l$ plays the role of a vertical velocity (or of a mass flux, by analogy with the convection). One introduces the dimensionless quantity:

$$A_l = \frac{M_l \Delta t}{\delta p}$$

One discretizes the pseudo-advection with an upstream implicit scheme:

$$U_l^+ - U_l = -(A_l U_l^+ - A_{l-1} U_{l-1}^+)$$

which can be written as:

$$(1 + A_l)(U_l^+ - U_l) = -A_l U_l + A_{l-1} U_{l-1} + A_{l-1}(U_{l-1}^+ - U_{l-1})$$

If one has calculated the implicit tendency on level $l-1$, $U_{l-1}^+ - U_{l-1}$, and the explicit tendency calculated by parametrization on level $l$, $-A_l U_l + A_{l-1} U_{l-1}$, one deduces from it the implicit tendency at level $l$. One starts at level $l = 1$ since the tendency is null at the top and one carries out a downward loop. One will replace stress $\tau$ by a stress $\dot{\tau}$ such as:

$$\frac{U^+ - U}{\Delta t} = -g \frac{\partial \dot{\tau}}{\partial p}$$

Then:

$$\dot{\tau}_0 = 0$$

$$\dot{\tau}_l = \dot{\tau}_{l-1} - \frac{1}{1 + A_l}(\dot{\tau}_{l-1} - \tau_l)$$

This is the same procedure as in section 3.2.

6 Parametrization of non-orographic gravity wave drag

This parametrization of non-orographic gravity wave drag is mainly described in Lott et al. (2012) and is treated by subroutine ACNORGWD.

6.1 Formalism

A broadband spectrum of gravity waves (GWs) is represented via the superposition of a large ensemble of statistically independent monochromatic ones. To produce this large ensemble at a reasonable numerical cost, one launches at each model time step $\Delta t$ a finite number $M$ (typically 8) of waves with characteristics chosen randomly and compute the tendencies due to them. As they are independent realizations the averaged tendency they produce (the gravity wave drag $(\partial_t u)_{gw}$) is the average of these $M$ tendencies. One takes
into account that the lifetime of these waves (typically $\Delta t \sim 1$ day) exceeds the model time step (typically $\delta t < 1$ h).

We then redistribute this averaged tendency over a longer time scale $\Delta t$ by first rescaling it by $\delta t/\Delta t$ and second by using a lag-one auto-regressive (AR-1) relation between the GW tendencies at two successive time steps:

$$
(\partial_t u)^{t+\delta t}_{gw} = \frac{\Delta t - \delta t}{\Delta t} (\partial_t u)^{t}_{gw} + \frac{\delta t}{\Delta t} \frac{1}{M} \sum_{n=1}^{M} \frac{\partial F_n}{\partial z}
$$

In other words, and at each time step, $M$ new waves are emitted, and ones reduces the probabilities of all the others by the multiplicative factor $(\Delta t - \delta t)/\Delta t$. A few hundred waves are then active at each model step, which gives an excellent spectral resolution at a reasonable computational cost.

At each time step, the GWs horizontal wave number is chosen randomly within the interval $k_{min} \leq k \leq k_{max}$, with $k_{max}$ and $k_{min}$ are related to the grid dimensions. The intrinsic phase speed is also randomly chosen from a Gaussian distribution of mean 0 and standard deviation $c_{max}$.

6.2 Launch spectrum

The Eliassen-Palm (EP) momentum flux carried by each wave is specified at a given launching altitude $z_l$ by

$$
F(z_l) = \rho_r G_p \left( \frac{RL_w}{\rho_r H c_p} \right)^2 \frac{k^2 e^{-(m \Delta z_p)^2}}{N \Omega^3} \frac{\Delta z_f}{4f} \int_{z_l}^{z_{top}} \rho_0(z) N(z)^2 \zeta^2(z) e^{-\pi \sqrt{J(z)} \Delta z_p} dz
$$

The first term in the right-hand side of equation (6) represents the contribution from convective sources. The second term in the right-hand side represents the contribution from the background flux and accounts for GWs from nonconvective (and nonorographic) sources.

To relate the gravity waves to the convective forcing, the surface precipitation $P$ is used. And to translate precipitation into diabatic heating, the latent heat flux produced by precipitation is distributed into the vertical over a Gaussian distribution with standard deviation $\Delta z_p$, that characterizes the heating depth (see Lott et al. (2013) for mathematical derivation). In (4), $\rho_r = 1$ kg/m$^3$ is the reference density, $L_w$ is the latent heat of condensation, $c_p$ is the heat capacity at constant pressure, $H$ is the characteristic height of the atmosphere, $N$ is the Brunt-Vaisala frequency, $k = ||k||$ is the horizontal wavenumber, $\Omega$ is the intrinsic frequency $\Omega = \omega - k \cdot u$ and $m$ is the vertical wavenumber $m = N k / \Omega$. $G_p$ is a tuning parameter of order 1.

To relate the gravity waves to frontal sources, the potential vorticity $\zeta$ is used (see De la Camara et al. (2015) for details). In (4), $f$ is the Coriolis parameter, $z_{top}$ is the model top altitude, $\rho_0 = \rho_r e^{-z/H}$ is the background density, $J = N^2 / \Lambda^2$ is the Richardson number ($\Lambda$ is the vertical shear of the horizontal wind), $\Delta z_f$ is the tunable depth of the frontal source, and $G_f$ is also a tunable parameter of order 1 that controls the amplitude of the EP flux.
6.3 Vertical propagation

First, the momentum flux carried by gravity waves is set to zero where the waves encounter a critical level. In the absence of critical levels, the momentum flux remains nearly constant except for a small dissipation $\mu/\rho_0$ in order to guarantee that the waves are finally erased over the last few model layers. The GW momentum flux is also limited by that produced by a saturated monochromatic wave. The passage of EP flux from one level $z$ to the next $z + \delta z$ can be written as

$$ F(z + \delta z) = \frac{k}{k} \Theta(\Omega(z)\Omega(z + \delta z)) \min \left( |F(z)|e^{-2\frac{\omega^3}{k^2} \delta z}, \rho_0 S_c \frac{\Omega^3 \delta z}{k_{\min}^2} \right) $$

(7)

where the Heaviside function $\Theta$ handles critical levels and $S_c$ is a tunable parameter controlling the saturated momentum flux.

6.4 Computation aspects

Subroutine ACNORGWD is active under LNORGWD logical key and calculates the gravity wave drag $(PTEND\_U, PTEND\_V) = ((\partial_t u)_{gw}, (\partial_t v)_{gw})$ from the vertical profiles of temperature PTT, zonal wind PUU, meridional wind PVV and potential vorticity PVOVO. The total (stratiform plus convective) precipitation PPREC is also used as a input of the subroutine.

The parameters of the scheme are defined in the module YOMNORGWD.

- NORGWD_RDISS is the dissipation coefficient ($\mu$).
- NORGWD_SAT is the saturation parameter ($S_c$).
- NORGWD_KMIN is the minimal horizontal wavenumber ($k_{\min}$).
- NORGWD_KMAX is the maximal horizontal wavenumber ($k_{\max}$).
- NORGWD_CMAX is the 'maximal' intrinsic phase velocity ($c_{\max}$).
- NORGWD_DELTAT is the time scale ($\Delta t$) of the life cycle of the waves parameterized.
- NORGWD_DZ is the characteristic depth of the convective sources ($\Delta z_p$).
- NORGWD RUWMAX is the tunable parameter that controls the amplitude of the EP flux from convective GWs ($G_p$).
- NORGWD_DZFRON is the characteristic depth of the frontal sources ($\Delta z_f$).
- NORGWD_GFRON is the tunable parameter that controls the amplitude of the EP flux emitted by frontal GWs ($G_f$).
7. Gravity wave drag
References

References


References


