EULERIAN AND SEMI-LAGRANGIAN DATA FLOW : DIRECT, TL, AD CODES ; TRAJECTORY HANDLING ; PHYSICS INTERFACE (CY36T2).

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

EULERIAN AND SEMI-LAGRANGIAN FORMULATIONS.

- Eulerian formulation :

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

  (A = NL + linear adiabatic terms, F = physics).
  Stability condition = CFL criterion.
  Always discretised as a leap-frog scheme.

- Semi-Lagrangian formulation :

$$\frac{dX}{dt} = \mathcal{A} + \mathcal{F}$$

  Stability condition = Lipschitz criterion, less stringent (the trajectories O - F must not cross each other). Physics often impose a slightly more stringent stability condition.
  Can be discretised as a leap-frog (three-time level) SL scheme or as a two-time level SL scheme (cheaper).

EULERIAN AND SEMI-LAGRANGIAN FORMULATIONS.

- Eulerian formulation :

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

  (A = NL + linear adiabatic terms, F = physics).
  Stability condition = CFL criterion.
  Always discretised as a leap-frog scheme.

- Semi-Lagrangian formulation :

$$\frac{dX}{dt} = \mathcal{A} + \mathcal{F}$$

  Stability condition = Lipschitz criterion, less stringent (the trajectories O - F must not cross each other). Physics often impose a slightly more stringent stability condition.
  Can be discretised as a leap-frog (three-time level) SL scheme or as a two-time level SL scheme (cheaper).

EULERIAN AND SEMI-LAGRANGIAN FORMULATIONS.

- Eulerian formulation :

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

(A = NL + linear adiabatic terms, F = physics).
Stability condition = CFL criterion.
Always discretised as a leap-frog scheme.

- Semi-Lagrangian formulation :

$$\frac{dX}{dt} = \mathcal{A} + \mathcal{F}$$

Stability condition = Lipschitz criterion, less stringent (the trajectories O - F must not cross each other). Physics often impose a slightly more stringent stability condition.
Can be discretised as a leap-frog (three-time level) SL scheme or as a two-time level SL scheme (cheaper).

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS.

- Eulerian discretisation :

$$X(t+\Delta t) - \Delta t\beta\mathcal{B}(t+\Delta t) = X(t-\Delta t)$$

$$-2\Delta t\left[\vec{V}\vec{\nabla}X\right](t) - 2\Delta t\left[\dot{\eta}\frac{\partial X}{\partial\eta}\right](t) + 2\Delta t[\mathcal{A}(t) - \beta\mathcal{B}(t)] + \Delta t\beta\mathcal{B}(t-\Delta t) + 2\Delta t\mathcal{F}(t-\Delta t)$$

  $\mathcal{B}$: linear terms.
  All terms are evaluated at the same model grid-point F.

- LSETTLS-type two-time level semi-Lagrangian discretisation without uncentering :

$$X(t+\Delta t, F) - 0.5\Delta t\beta\mathcal{B}(t+\Delta t, F) = X(t, O) + \{[0.5\Delta t\mathcal{A}(t) - 0.5\Delta t\beta\mathcal{B}(t)]\}_F$$

$$+\{[\Delta t\mathcal{A}(t) - \Delta t\beta\mathcal{B}(t)] - [0.5\Delta t\mathcal{A}(t-\Delta t) - 0.5\Delta t\beta\mathcal{B}(t-\Delta t)] + [0.5\Delta t\beta\mathcal{B}(t) + \Delta t\mathcal{F}(t)]\}_O$$

  Requires the calculation of an origin point O and interpolations at this point.

- Trajectories are great circles on the geographical sphere in ARPEGE, and straight lines on the projection plane in ALADIN.
  The computation of the origin point O is performed by an iterative method (2 or 3 iter) described by Robert (1981) and adapted to the sphere by M. Rochas.
  In ALADIN, O bounded inside C+I except for the analytical calculation of the Coriolis term.

- Interpolations : generally 32 points or trilinear interpolations, but possible choice of quasi-monotonic interpolations, SLHD interpolations, spline cubic interpolations.

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS.

- Eulerian discretisation :

$$X(t+\Delta t) - \Delta t \beta \mathcal{B}(t+\Delta t) = X(t-\Delta t)$$

$$-2\Delta t \left[\vec{V}\vec{\nabla} X\right](t) - 2\Delta t \left[\dot{\eta}\frac{\partial X}{\partial \eta}\right](t) + 2\Delta t[\mathcal{A}(t) - \beta\mathcal{B}(t)] + \Delta t \beta \mathcal{B}(t-\Delta t) + 2\Delta t \mathcal{F}(t-\Delta t)$$

  $\mathcal{B}$: linear terms.
  All terms are evaluated at the same model grid-point F.

- LSETTLS-type two-time level semi-Lagrangian discretisation without uncentering :

$$X(t+\Delta t, F) - 0.5\Delta t \beta \mathcal{B}(t+\Delta t, F) = X(t, O) + \{[0.5\Delta t \mathcal{A}(t) - 0.5\Delta t \beta \mathcal{B}(t)]\}_F$$

$$+\{[\Delta t \mathcal{A}(t) - \Delta t \beta \mathcal{B}(t)] - [0.5\Delta t \mathcal{A}(t-\Delta t) - 0.5\Delta t \beta \mathcal{B}(t-\Delta t)] + [0.5\Delta t \beta \mathcal{B}(t) + \Delta t \mathcal{F}(t)]\}_O$$

  Requires the calculation of an origin point O and interpolations at this point.

- Trajectories are great circles on the geographical sphere in ARPEGE, and straight lines on the projection plane in ALADIN.
  The computation of the origin point O is performed by an iterative method (2 or 3 iter) described by Robert (1981) and adapted to the sphere by M. Rochas.
  In ALADIN, O bounded inside C+I except for the analytical calculation of the Coriolis term.

- Interpolations : generally 32 points or trilinear interpolations, but possible choice of quasi-monotonic interpolations, SLHD interpolations, spline cubic interpolations.

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS.

- Eulerian discretisation :

$$X(t+\Delta t) - \Delta t \beta \mathcal{B}(t+\Delta t) = X(t-\Delta t)$$

$$-2\Delta t \left[\vec{V}\vec{\nabla}X\right](t) - 2\Delta t \left[\dot{\eta}\frac{\partial X}{\partial \eta}\right](t) + 2\Delta t[\mathcal{A}(t) - \beta\mathcal{B}(t)] + \Delta t \beta \mathcal{B}(t-\Delta t) + 2\Delta t \mathcal{F}(t-\Delta t)$$

  $\mathcal{B}$: linear terms.
  All terms are evaluated at the same model grid-point F.

- LSETTLS-type two-time level semi-Lagrangian discretisation without uncentering :

$$X(t+\Delta t, F) - 0.5\Delta t \beta \mathcal{B}(t+\Delta t, F) = X(t, O) + \{[0.5\Delta t \mathcal{A}(t) - 0.5\Delta t \beta \mathcal{B}(t)]\}_F$$

$$+\{[\Delta t \mathcal{A}(t) - \Delta t \beta \mathcal{B}(t)] - [0.5\Delta t \mathcal{A}(t-\Delta t) - 0.5\Delta t \beta \mathcal{B}(t-\Delta t)] + [0.5\Delta t \beta \mathcal{B}(t) + \Delta t \mathcal{F}(t)]\}_O$$

  Requires the calculation of an origin point O and interpolations at this point.

- Trajectories are great circles on the geographical sphere in ARPEGE, and straight lines on the projection plane in ALADIN.
  The computation of the origin point O is performed by an iterative method (2 or 3 iter) described by Robert (1981) and adapted to the sphere by M. Rochas.
  In ALADIN, O bounded inside C+I except for the analytical calculation of the Coriolis term.

- Interpolations : generally 32 points or trilinear interpolations, but possible choice of quasi-monotonic interpolations, SLHD interpolations, spline cubic interpolations.

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS.

- Eulerian discretisation :

$$X(t+\Delta t) - \Delta t\beta\mathcal{B}(t+\Delta t) = X(t-\Delta t)$$

$$-2\Delta t\left[\vec{V}\vec{\nabla}X\right](t) - 2\Delta t\left[\dot{\eta}\frac{\partial X}{\partial\eta}\right](t) + 2\Delta t[\mathcal{A}(t) - \beta\mathcal{B}(t)] + \Delta t\beta\mathcal{B}(t-\Delta t) + 2\Delta t\mathcal{F}(t-\Delta t)$$

  $\mathcal{B}$: linear terms.
  All terms are evaluated at the same model grid-point F.

- LSETTLS-type two-time level semi-Lagrangian discretisation without uncentering :

$$X(t+\Delta t, F) - 0.5\Delta t\beta\mathcal{B}(t+\Delta t, F) = X(t, O) + \{[0.5\Delta t\mathcal{A}(t) - 0.5\Delta t\beta\mathcal{B}(t)]\}_F$$

$$+\{[\Delta t\mathcal{A}(t) - \Delta t\beta\mathcal{B}(t)] - [0.5\Delta t\mathcal{A}(t-\Delta t) - 0.5\Delta t\beta\mathcal{B}(t-\Delta t)] + [0.5\Delta t\beta\mathcal{B}(t) + \Delta t\mathcal{F}(t)]\}_O$$

  Requires the calculation of an origin point O and interpolations at this point.

- Trajectories are great circles on the geographical sphere in ARPEGE, and straight lines on the projection plane in ALADIN.
  The computation of the origin point O is performed by an iterative method (2 or 3 iter) described by Robert (1981) and adapted to the sphere by M. Rochas.
  In ALADIN, O bounded inside C+I except for the analytical calculation of the Coriolis term.

- Interpolations : generally 32 points or trilinear interpolations, but possible choice of quasi-monotonic interpolations, SLHD interpolations, spline cubic interpolations.

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS.

- Eulerian discretisation :

$$X(t+\Delta t) - \Delta t\beta\mathcal{B}(t+\Delta t) = X(t-\Delta t)$$

$$-2\Delta t\left[\vec{V}\vec{\nabla}X\right](t) - 2\Delta t\left[\dot{\eta}\frac{\partial X}{\partial \eta}\right](t) + 2\Delta t[\mathcal{A}(t) - \beta\mathcal{B}(t)] + \Delta t\beta\mathcal{B}(t-\Delta t) + 2\Delta t\mathcal{F}(t-\Delta t)$$

  $\mathcal{B}$: linear terms.
  All terms are evaluated at the same model grid-point F.

- LSETTLS-type two-time level semi-Lagrangian discretisation without uncentering :

$$X(t+\Delta t, F) - 0.5\Delta t\beta\mathcal{B}(t+\Delta t, F) = X(t, O) + \{[0.5\Delta t\mathcal{A}(t) - 0.5\Delta t\beta\mathcal{B}(t)]\}_F$$

$$+\{[\Delta t\mathcal{A}(t) - \Delta t\beta\mathcal{B}(t)] - [0.5\Delta t\mathcal{A}(t-\Delta t) - 0.5\Delta t\beta\mathcal{B}(t-\Delta t)] + [0.5\Delta t\beta\mathcal{B}(t) + \Delta t\mathcal{F}(t)]\}_O$$

  Requires the calculation of an origin point O and interpolations at this point.

- Trajectories are great circles on the geographical sphere in ARPEGE, and straight lines on the projection plane in ALADIN.
  The computation of the origin point O is performed by an iterative method (2 or 3 iter) described by Robert (1981) and adapted to the sphere by M. Rochas.
  In ALADIN, O bounded inside C+I except for the analytical calculation of the Coriolis term.

- Interpolations : generally 32 points or trilinear interpolations, but possible choice of quasi-monotonic interpolations, SLHD interpolations, spline cubic interpolations.

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS (CONT'D).

- Horizontal discretisation : Spectral + grid-point calculations. Horizontal derivatives are evaluated during the inverse spectral transforms.
- Vertical discretisation :
  $*$ Finite difference discretisation (prognostic variables at full levels, some intermediate variables at half levels).
  $*$ Finite element discretisation (LVERTFE=T, all variables at full levels).

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS (CONT'D).

- Horizontal discretisation : Spectral + grid-point calculations. Horizontal derivatives are evaluated during the inverse spectral transforms.
- Vertical discretisation :
  $*$ Finite difference discretisation (prognostic variables at full levels, some intermediate variables at half levels).
  $*$ Finite element discretisation (LVERTFE=T, all variables at full levels).

EULERIAN AND SEMI-LAGRANGIAN DISCRETISATIONS (CONT'D).

- Horizontal discretisation : Spectral + grid-point calculations. Horizontal derivatives are evaluated during the inverse spectral transforms.
- Vertical discretisation :
  $*$ Finite difference discretisation (prognostic variables at full levels, some intermediate variables at half levels).
  $*$ Finite element discretisation (LVERTFE=T, all variables at full levels).

GROUPS OF PROGNOSTIC VARIABLES.

- Upper-air quantities :
  - GMV (3D) variables ($\mathcal{A}$ and $\mathcal{B}$ are non-zero). Example : wind components (VOR/DIV in spectral calculations), temperature, additional NH variables. The GMV variables other than the wind components or divergence/vorticity are the "thermodynamical variables" (there are NFTHER thermodynamical variables in the model).
  - GMVS (2D) variables ($\mathcal{A}$ and $\mathcal{B}$ are non-zero). Example : logarithm of surface pressure.
  - GFL (3D) variables ($\mathcal{A}$ and $\mathcal{B}$ are zero). Example : humidity, liquid water, ice, TKE, ozone, etc...

    This list also contains some pseudo-historic variables (ex CPF = convective precipitation flux).
- Surface prognostic quantities : buffers SP_.. of the surface dataflow. Examples : temperature and water content of the soil reservoirs.

GROUPS OF PROGNOSTIC VARIABLES.

- Upper-air quantities :
  - GMV (3D) variables ($\mathcal{A}$ and $\mathcal{B}$ are non-zero). Example : wind components (VOR/DIV in spectral calculations), temperature, additional NH variables. The GMV variables other than the wind components or divergence/vorticity are the "thermodynamical variables" (there are NFTHER thermodynamical variables in the model).
  - GMVS (2D) variables ($\mathcal{A}$ and $\mathcal{B}$ are non-zero). Example : logarithm of surface pressure.
  - GFL (3D) variables ($\mathcal{A}$ and $\mathcal{B}$ are zero). Example : humidity, liquid water, ice, TKE, ozone, etc...
    This list also contains some pseudo-historic variables (ex CPF = convective precipitation flux).
- Surface prognostic quantities : buffers SP_.. of the surface dataflow. Examples : temperature and water content of the soil reservoirs.

GROUPS OF PROGNOSTIC VARIABLES.

- Upper-air quantities :
  - GMV (3D) variables ($\mathcal{A}$ and $\mathcal{B}$ are non-zero). Example : wind components (VOR/DIV in spectral calculations), temperature, additional NH variables. The GMV variables other than the wind components or divergence/vorticity are the "thermodynamical variables" (there are NFTHER thermodynamical variables in the model).
  - GMVS (2D) variables ($\mathcal{A}$ and $\mathcal{B}$ are non-zero). Example : logarithm of surface pressure.
  - GFL (3D) variables ($\mathcal{A}$ and $\mathcal{B}$ are zero). Example : humidity, liquid water, ice, TKE, ozone, etc...
    This list also contains some pseudo-historic variables (ex CPF = convective precipitation flux).
- Surface prognostic quantities : buffers SP_.. of the surface dataflow. Examples : temperature and water content of the soil reservoirs.

ORGANIGRAMME UNDER STEPO.

STEPO = management of one timestep :

- $X(t)$ available as spectral variable.
- Inverse transforms + compute horizontal derivatives [(E)TRANSINVH].
- Grid-point calculations [GP_MODEL] (explicit dynamics, physics, SL interpolations).
- Coupling (ALADIN only) [ECOUPL1].
- Direct transforms [(E)TRANSDIRH].
  Remark for spectral transforms : Fourier + Legendre in ARPEGE (code in the TFL library), double Fourier in ALADIN (code in the TAL library).
- Spectral calculations [(E)SPCH] (SI scheme, horizontal diffusion).
- Provides $X(t + \Delta t)$, which becomes $X(t)$ at the following timestep.

ORGANIGRAMME UNDER STEPO.

STEPO = management of one timestep :

- $X(t)$ available as spectral variable.
- Inverse transforms + compute horizontal derivatives [(E)TRANSINVH].
- Grid-point calculations [GP_MODEL] (explicit dynamics, physics, SL interpolations).
- Coupling (ALADIN only) [ECOUPL1].
- Direct transforms [(E)TRANSDIRH].
  Remark for spectral transforms : Fourier + Legendre in ARPEGE (code in the TFL library), double Fourier in ALADIN (code in the TAL library).
- Spectral calculations [(E)SPCH] (SI scheme, horizontal diffusion).
- Provides $X(t + \Delta t)$, which becomes $X(t)$ at the following timestep.

SPECTRAL VARIABLES.

- GMV : SP[X], [X]=VOR,DIV,T,SPD,SVD.
  SPHV gathers all the thermodynamic variables.
- GMVS : SP[X], [X]=SP. Additional array SPOR for spectral orography.
- GFL : SP[X], [X]=Q,L,I,A,O3, etc... [some of them currently mis-named].
  SPGFL gathers all the spectral GFL variables.
- SPA3 : gathers all the 3D GMV+GFL variables.
- SPA2 : gathers all the 2D GMV variables (+ the spectral orography).
- SPA1 : mean wind, in ALADIN only.

SPECTRAL VARIABLES.

- GMV : SP[X], [X]=VOR,DIV,T,SPD,SVD.
  SPHV gathers all the thermodynamic variables.
- GMVS : SP[X], [X]=SP. Additional array SPOR for spectral orography.
- GFL : SP[X], [X]=Q,L,I,A,O3, etc... [some of them currently mis-named].
  SPGFL gathers all the spectral GFL variables.
- SPA3 : gathers all the 3D GMV+GFL variables.
- SPA2 : gathers all the 2D GMV variables (+ the spectral orography).
- SPA1 : mean wind, in ALADIN only.

SPECTRAL VARIABLES.

- GMV : SP[X], [X]=VOR,DIV,T,SPD,SVD.
  SPHV gathers all the thermodynamic variables.
- GMVS : SP[X], [X]=SP. Additional array SPOR for spectral orography.
- GFL : SP[X], [X]=Q,L,I,A,O3, etc... [some of them currently mis-named].
  SPGFL gathers all the spectral GFL variables.
- SPA3 : gathers all the 3D GMV+GFL variables.
- SPA2 : gathers all the 2D GMV variables (+ the spectral orography).
- SPA1 : mean wind, in ALADIN only.

GRID-POINT VARIABLES.

- GMV : GMV gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1 gathers the $t + \Delta t$ variables.
- GMVS : GMVS gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1S gathers the $t + \Delta t$ variables.
- GFL : GFL gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GFLT1 gathers the $t + \Delta t$ variables.
- SP_[group] : surface dataflow. In particular contains 2D surface variables used in the physics.
- Individual variables (for example under CPG) :
  P[X]T0 : X at t; (P[X]T0L,P[X]T0M) : grad(X) at t.
  P[X]T9 : X at t; (P[X]T9L,P[X]T9M) : grad(X) at $t - \Delta t$.
  P[X]T1 : X at $t + \Delta t$.
  Sometimes suffix F for full level, H for half level.
- Semi-Lagrangian buffer for interpolations : SLBUF1 or PB1. Stores the quantities to be interpolated (SL scheme). Inside some LA... routines one finds some individual P[X]SL arrays (subset of PB1) : example PGMVSL, PGFLSL.
- PB2 stores some quantities to be communicated between the different g.p. calculations, which cannot be stored in the GMV+GFL arrays.

GRID-POINT VARIABLES.

- GMV : GMV gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1 gathers the $t + \Delta t$ variables.
- GMVS : GMVS gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1S gathers the $t + \Delta t$ variables.
- GFL : GFL gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GFLT1 gathers the $t + \Delta t$ variables.
- SP_[group] : surface dataflow. In particular contains 2D surface variables used in the physics.
- Individual variables (for example under CPG) :
  P[X]T0 : X at t; (P[X]T0L,P[X]T0M) : grad(X) at t.
  P[X]T9 : X at t; (P[X]T9L,P[X]T9M) : grad(X) at $t - \Delta t$.
  P[X]T1 : X at $t + \Delta t$.
  Sometimes suffix F for full level, H for half level.
- Semi-Lagrangian buffer for interpolations : SLBUF1 or PB1. Stores the quantities to be interpolated (SL scheme). Inside some LA... routines one finds some individual P[X]SL arrays (subset of PB1) : example PGMVSL, PGFLSL.
- PB2 stores some quantities to be communicated between the different g.p. calculations, which cannot be stored in the GMV+GFL arrays.

GRID-POINT VARIABLES.

- GMV : GMV gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1 gathers the $t + \Delta t$ variables.
- GMVS : GMVS gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1S gathers the $t + \Delta t$ variables.
- GFL : GFL gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GFLT1 gathers the $t + \Delta t$ variables.
- SP_[group] : surface dataflow. In particular contains 2D surface variables used in the physics.
- Individual variables (for example under CPG) :
  P[X]T0 : X at t ; (P[X]T0L,P[X]T0M) : grad(X) at t.
  P[X]T9 : X at t ; (P[X]T9L,P[X]T9M) : grad(X) at $t - \Delta t$.
  P[X]T1 : X at $t + \Delta t$.
  Sometimes suffix F for full level, H for half level.
- Semi-Lagrangian buffer for interpolations : SLBUF1 or PB1. Stores the quantities to be interpolated (SL scheme). Inside some LA... routines one finds some individual P[X]SL arrays (subset of PB1) : example PGMVSL, PGFLSL.
- PB2 stores some quantities to be communicated between the different g.p. calculations, which cannot be stored in the GMV+GFL arrays.

GRID-POINT VARIABLES.

- GMV : GMV gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1 gathers the $t + \Delta t$ variables.
- GMVS : GMVS gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1S gathers the $t + \Delta t$ variables.
- GFL : GFL gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GFLT1 gathers the $t + \Delta t$ variables.
- SP_[group] : surface dataflow. In particular contains 2D surface variables used in the physics.
- Individual variables (for example under CPG) :
  P[X]T0 : X at t; (P[X]T0L,P[X]T0M) : grad(X) at t.
  P[X]T9 : X at t; (P[X]T9L,P[X]T9M) : grad(X) at $t - \Delta t$.
  P[X]T1 : X at $t + \Delta t$.
  Sometimes suffix F for full level, H for half level.
- Semi-Lagrangian buffer for interpolations : SLBUF1 or PB1. Stores the quantities to be interpolated (SL scheme). Inside some LA... routines one finds some individual P[X]SL arrays (subset of PB1) : example PGMVSL, PGFLSL.
- PB2 stores some quantities to be communicated between the different g.p. calculations, which cannot be stored in the GMV+GFL arrays.

GRID-POINT VARIABLES.

- GMV : GMV gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1 gathers the $t + \Delta t$ variables.
- GMVS : GMVS gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GMVT1S gathers the $t + \Delta t$ variables.
- GFL : GFL gathers the $t - \Delta t$ and t variables (including horizontal derivatives), GFLT1 gathers the $t + \Delta t$ variables.
- SP_[group] : surface dataflow. In particular contains 2D surface variables used in the physics.
- Individual variables (for example under CPG) :
  P[X]T0 : X at t ; (P[X]T0L,P[X]T0M) : grad(X) at t.
  P[X]T9 : X at t ; (P[X]T9L,P[X]T9M) : grad(X) at $t - \Delta t$.
  P[X]T1 : X at $t + \Delta t$.
  Sometimes suffix F for full level, H for half level.
- Semi-Lagrangian buffer for interpolations : SLBUF1 or PB1. Stores the quantities to be interpolated (SL scheme). Inside some LA... routines one finds some individual P[X]SL arrays (subset of PB1) : example PGMVSL, PGFLSL.
- PB2 stores some quantities to be communicated between the different g.p. calculations, which cannot be stored in the GMV+GFL arrays.

ORGANIGRAMME UNDER GP_MODEL FOR EULERIAN 3D MODEL.

Under STEPO $->$ SCAN2H $->$ SCAN2M $->$ GP_MODEL :

- CPG (unlagged dynamics, unlagged MF physics)
- EC_PHYS (lagged ECMWF physics)
- CPGLAG (lagged dynamics)

ORGANIGRAMME UNDER GP_MODEL FOR EULERIAN 3D MODEL.

Under STEPO $->$ SCAN2H $->$ SCAN2M $->$ GP_MODEL :

- CPG (unlagged dynamics, unlagged MF physics)
- EC_PHYS (lagged ECMWF physics)
- CPGLAG (lagged dynamics)

```
control/STEPO -> control/SCAN2H -> control/SCAN2M -> control/GP_MODEL ->
  * adiab/CPG ->
    - adiab/CPG_GP ->
      * adiab/GPTF2 (Asselin filter, second part)
      * adiab/GPMPFC and adiab/GPMPFC_GMVS (multiply by M or M**2)
      * adiab/GP_SPV (retrieve "prehyd" and its gradient)
      * utility/SC2RDG (buffer reading)
      * some adiab/GP.. and adiab/GNH.. routines
        computing intermediate grid-point quantities at t and t-dt.
        For ex. the pressure gradient term.
      * adiab/GPINISLB (sets-up the content of PB2)
    - dia/GPINIDDH (for DDH package)
    - phys_dmn/MF_PHYS_PREP and MF_PHYS (MF unlagged physics or AROME physics)
    - adiab/CPG_DIA -> (routines for some diagnostics: DDH, CFU, XFU)
    - adiab/CPG_DYN ->
      * adiab/CPEULDYN (computes the explicit part of the RHS of equations,
        add them to GFLT1,GMVT1,GMVT1S) ->
        - some adiab/GP.. routines computing intermediate g.p. quantities.
        - some adiab/SI.. routines computing some linear terms used in SI scheme.
      * sinvect/VDIFLCZ (Buizza simplified physics)
    - adiab/CPG_END ->
      * var/WRPHTRSF (buffer writing)
      * adiab/GPMPFC and adiab/GPMPFC_GMVS (divide by M or M**2)
      * utility/SC2WRG (buffer writing)
  * phys_ec/RADDRV (ECMWF lagged radiation scheme used at ECMWF)
  * phys_ec/EC_PHYS_DRV (ECMWF lagged physics)
  * adiab/CPGLAG ->
    - adiab/GPTF1 (Asselin filter, first part)
    - adiab/GPENDTR (end calculations and memory transfers)
```

ORGANIGRAMME UNDER GP_MODEL FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPO $->$ SCAN2H $->$ SCAN2M $->$ GP_MODEL :

- CPG (unlagged dynamics, unlagged MF physics)
- CALL_SL (lagged SL dynamics, traj research, interpolations)
- EC_PHYS (lagged ECMWF physics)
- CPGLAG (remaining lagged dynamics)

ORGANIGRAMME UNDER GP_MODEL FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPO $->$ SCAN2H $->$ SCAN2M $->$ GP_MODEL :

- CPG (unlagged dynamics, unlagged MF physics)
- CALL_SL (lagged SL dynamics, traj research, interpolations)
- EC_PHYS (lagged ECMWF physics)
- CPGLAG (remaining lagged dynamics)

```
control/STEPO -> control/SCAN2H -> control/SCAN2M -> control/GP_MODEL ->
  * adiab/CPG ->
    - adiab/CPG_GP (cf. Eulerian)
    - dia/GPINIDDH (for DDH package)
    - phys_dmn/MF_PHYS (MF unlagged physics or AROME physics)
    - adiab/CPG_DIA -> (routines for some diagnostics: DDH, CFU, XFU)
    - adiab/CPG_DYN ->
      * adiab/LACDYN (computes the explicit part of the RHS of equations,
        update GFLT1,GMVT1,GMVT1S with quantities at F, fills buffer SLBUF1) ->
        - adiab/LASURE (initialisations)
        - adiab/LANHSI (NH) or adiab/LASSIE (HYD): linear terms for SI scheme.
        - adiab/LAVENT: fills PB1 for trajectory wind.
        - adiab/LATTEX: fills PB1 + updates (GFLT1,GMVT1) for 3D variables.
        - adiab/LATTES: fills PB1 + updates (GMVT1S) for 2D variables.
        - adiab/LAVABO: upper and lower boundary condition in PB1.
        - additional adiab/LA... routines for particuliar configurations.
      * sinvect/VDIFLCZ (Buizza simplified physics)
    - adiab/CPG_END (cf. Eulerian)
  * phys_ec/RADDRV (ECMWF lagged radiation scheme used at ECMWF)
  * adiab/CALL_SL ->
    - some parallel environment routines spread in the code (SLCOMM.., (E)SLEXTPOL.. routines).
    - adiab/LAPINEA ->
      * adiab/LARMES (adiab/ELARMES in ALADIN) (Trajectory research) ->
        - adiab/LARCINA ->
          - adiab/LARCHE + adiab/LASCAW (adiab/ELARCHE + adiab/ELASCAW in ALADIN)
            (LARCHE: geographical -> apparent coordinates of O; (E)LASCAW: weights for interpolations)
          - adiab/LAITLI (trilinear interpolations)
      * adiab/LARCINA ->
        - adiab/LARCHE + adiab/LASCAW (adiab/ELARCHE + adiab/ELASCAW in ALADIN)
      * adiab/LARCINHA ->
        - adiab/LARCHE + adiab/LASCAW (adiab/ELASCAW in ALADIN)
    - adiab/LAPINEB (manages the interpolations, updates GFLT1,GMVT1,GMVT1S with the interpolated values) ->
      * adiab/LARCINB (full level variables) ->
        - adiab/LAITRE_GMV (interpolations for GMV)
        - adiab/LAITRE_GFL (interpolations for GFL)
        - some specific interpolations routines (LAI...)
      * adiab/LARCINHB (half level quantities) ->
        - adiab/LAITRE_GMV
        - adiab/LAITLI
      * some adiab/GP... and adiab/GNH... routines.
      * adiab/LARCHE, adiab/LACONE, adiab/GNHGW2SVD (specific applications)
  * phys_ec/EC_PHYS_DRV (ECMWF lagged physics)
  * adiab/CPGLAG (cf. Eulerian)
```

ORGANIGRAMME UNDER STEPOTL.

STEPOTL = management of one timestep, TL model :

- Inverse transforms + compute horizontal derivatives [(E)TRANSINVH].
- Grid-point calculations [GP_MODEL_TL] (explicit dynamics, simplified physics, SL interpolations).
- Coupling (ALADIN only) [ECOUPL1].
- Direct transforms [(E)TRANSDIRH].
- Spectral calculations [(E)SPCH] (SI scheme, horizontal diffusion).

Remarks :

- Spectral transforms and calculations = linear algebra : TL = DIRECT.
- Coupling = linear algebra : TL = DIRECT.
- Grid-point calculations = requires a TL code. The structure of the TL code generally matches the structure of the direct code, with additional trajectory calculations.
- TL of NH with d3 or d4 not yet coded.

ORGANIGRAMME UNDER STEPOTL.

STEPOTL = management of one timestep, TL model :

- Inverse transforms + compute horizontal derivatives [(E)TRANSINVH].
- Grid-point calculations [GP_MODEL_TL] (explicit dynamics, simplified physics, SL interpolations).
- Coupling (ALADIN only) [ECOUPL1].
- Direct transforms [(E)TRANSDIRH].
- Spectral calculations [(E)SPCH] (SI scheme, horizontal diffusion).

Remarks :

- Spectral transforms and calculations = linear algebra : TL = DIRECT.
- Coupling = linear algebra : TL = DIRECT.
- Grid-point calculations = requires a TL code. The structure of the TL code generally matches the structure of the direct code, with additional trajectory calculations.
- TL of NH with d3 or d4 not yet coded.

ORGANIGRAMME UNDER STEPOTL.

STEPOTL = management of one timestep, TL model :

- Inverse transforms + compute horizontal derivatives [(E)TRANSINVH].
- Grid-point calculations [GP_MODEL_TL] (explicit dynamics, simplified physics, SL interpolations).
- Coupling (ALADIN only) [ECOUPL1].
- Direct transforms [(E)TRANSDIRH].
- Spectral calculations [(E)SPCH] (SI scheme, horizontal diffusion).

Remarks :

- Spectral transforms and calculations = linear algebra : TL = DIRECT.
- Coupling = linear algebra : TL = DIRECT.
- Grid-point calculations = requires a TL code. The structure of the TL code generally matches the structure of the direct code, with additional trajectory calculations.
- TL of NH with d3 or d4 not yet coded.

ORGANIGRAMME UNDER GP_MODEL_TL FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPOTL $->$ SCAN2HTL $->$ SCAN2MTL $->$ GP_MODEL_TL :

- CPGTL (unlagged dynamics, unlagged MF physics)
- CALL_SL_TL (lagged SL dynamics, traj research, interpolations)
- EC_PHYS_TL (lagged ECMWF physics)
- CPGLAGTL (remaining lagged dynamics)

ORGANIGRAMME UNDER GP_MODEL_TL FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPOTL $->$ SCAN2HTL $->$ SCAN2MTL $->$ GP_MODEL_TL :

- CPGTL (unlagged dynamics, unlagged MF physics)
- CALL_SL_TL (lagged SL dynamics, traj research, interpolations)
- EC_PHYS_TL (lagged ECMWF physics)
- CPGLAGTL (remaining lagged dynamics)

```
control/STEPOTL -> control/SCAN2HTL -> control/SCAN2MTL -> control/GP_MODEL_TL ->
  * adiab/CPGTL ->
    - adiab/CPG5_GP (Trajectory version of CPG_GP)
    - adiab/CPG_GP_TL (TL version of CPG_GP)
    - phys_dmn/MF_PHYSTL (TL of MF simplified physics)
    - adiab/CPG_DYN_TL ->
      * adiab/LACDYNTL (computes the explicit part of the RHS of equations,
        update GFLT1,GMVT1,GMVT1S with quantities at F, fills buffer SLBUF1) ->
        - adiab/LASURE (initialisations)
        - [adiab/LANHSITL (NH)] or adiab/LASSIETL (HYD):
          linear terms for SI scheme (trajectory + TL).
        - adiab/LAVENTTL: fills PB1+PB15 for trajectory wind.
        - adiab/LATTEXTL: fills PB1+PB15
          + updates (GFLT1,GMVT1,GFLT15,GMVT15) for 3D variables.
        - adiab/LATTESTL: fills PB1+PB15
          + updates (GMVT1S+GMVT15S) for 2D variables.
        - adiab/LAVABOTL: upper and lower boundary condition in PB1+PB15.
      * sinvect/VDIFLCZTL (TL of Buizza simplified physics)
    - adiab/CPG_END_TL (TL version of CPG_END)
  * adiab/CALL_SL_TL ->
    - some parallel environment routines spread in the code (SLCOMM.., (E)SLEXTPOL.. routines).
    - adiab/LAPINEA5 (trajectory version of LAPINEA, manages the SL trajectory research for the TL-trajectory).
    - adiab/LAPINEATL ->
      * adiab/LARMESTL (future adiab/ELARMESTL in ALADIN) (Trajectory research for TL) ->
        - adiab/LARCINATL ->
          - adiab/LARCHETL + adiab/LASCAWTL (future adiab/ELARCHETL + adiab/ELASCAWTL in ALADIN)
          - adiab/LAITLITL (TL of trilinear interpolations)
      * adiab/LARCINATL -> (see above)
    - adiab/LAPINEBTL (manages the interpolations, updates GFLT1,GMVT1,GMVT1S with the interpolated values) ->
      * adiab/LARCINBTL (full level variables) ->
        - adiab/LAITRE_GMV_TL (currently some LAI...TL for GMV, also used for GFL)
        - some specific interpolations routines (LAI...TL)
      * adiab/LARCHETL, adiab/LACONETL (specific applications)
  * phys_ec/EC_PHYS_TL (TL of ECMWF lagged simplified physics)
  * adiab/CPGLAGTL (TL of CPGLAG)
```

ADDITIONAL REMARKS ABOUT TL CODE.

- The structure of the TL code does not completely match the direct code because some cleanings or developments have been done on the direct code and not yet on the TL code. Examples :
  TL code for NH (d3,d4) not yet coded ;
  level of modularisation different in direct and TL.
  => necessity to develop if possible the TL code at the same time as the direct code to keep the consistency.
- Some differences between the TL and the direct code are also due to the trajectory code (spread in the code, but specific ..5 routines appear, for ex. LAPINEA5).
- LAITLI : linear interpolations but not linear code in the SL interpolator (contrary to the OBS interpolator) ! ! ! (Quantity to be interpolated is a variable, the weights also so $Y = W * X1 + (1 - W) * X2$ is not a linear code).
  => needs to have a TL routine LAITLITL.

ADDITIONAL REMARKS ABOUT TL CODE.

- The structure of the TL code does not completely match the direct code because some cleanings or developments have been done on the direct code and not yet on the TL code. Examples :
  TL code for NH (d3,d4) not yet coded ;
  level of modularisation different in direct and TL.
  => necessity to develop if possible the TL code at the same time as the direct code to keep the consistency.
- Some differences between the TL and the direct code are also due to the trajectory code (spread in the code, but specific ..5 routines appear, for ex. LAPINEA5).
- LAITLI : linear interpolations but not linear code in the SL interpolator (contrary to the OBS interpolator) ! ! ! (Quantity to be interpolated is a variable, the weights also so $Y = W * X1 + (1 - W) * X2$ is not a linear code).
  => needs to have a TL routine LAITLITL.

ADDITIONAL REMARKS ABOUT TL CODE.

- The structure of the TL code does not completely match the direct code because some cleanings or developments have been done on the direct code and not yet on the TL code. Examples :
  TL code for NH (d3,d4) not yet coded ;
  level of modularisation different in direct and TL.
  => necessity to develop if possible the TL code at the same time as the direct code to keep the consistency.
- Some differences between the TL and the direct code are also due to the trajectory code (spread in the code, but specific ..5 routines appear, for ex. LAPINEA5).
- LAITLI : linear interpolations but not linear code in the SL interpolator (contrary to the OBS interpolator) ! ! ! (Quantity to be interpolated is a variable, the weights also so $Y = W * X1 + (1 - W) * X2$ is not a linear code).
  => needs to have a TL routine LAITLITL.

ADDITIONAL REMARKS ABOUT TL CODE.

- The structure of the TL code does not completely match the direct code because some cleanings or developments have been done on the direct code and not yet on the TL code. Examples :
  TL code for NH (d3,d4) not yet coded ;
  level of modularisation different in direct and TL.
  => necessity to develop if possible the TL code at the same time as the direct code to keep the consistency.
- Some differences between the TL and the direct code are also due to the trajectory code (spread in the code, but specific ..5 routines appear, for ex. LAPINEA5).
- LAITLI : linear interpolations but not linear code in the SL interpolator (contrary to the OBS interpolator) ! ! ! (Quantity to be interpolated is a variable, the weights also so $Y = W * X1 + (1 - W) * X2$ is not a linear code).
  => needs to have a TL routine LAITLITL.

ORGANIGRAMME UNDER STEPOAD.

STEPOAD = management of one timestep, AD model :

- AD of spectral calculations [(E)SPCHAD] (SI scheme, horizontal diffusion).
- AD of direct transforms [(E)TRANSDIRHAD].
- AD of coupling (ALADIN only) [ECOUPL1AD].
- AD of grid-point calculations [GP_MODEL_AD] (explicit dynamics, simplified physics, SL interpolations).
- AD of "Inverse transforms + compute horizontal derivatives" [(E)TRANSINVHAD].

Remarks :

- Inversion of the order of calls compared to the TL code.
- AD of NH with d3 or d4 not yet coded.

ORGANIGRAMME UNDER STEPOAD.

STEPOAD = management of one timestep, AD model :

- AD of spectral calculations [(E)SPCHAD] (SI scheme, horizontal diffusion).
- AD of direct transforms [(E)TRANSDIRHAD].
- AD of coupling (ALADIN only) [ECOUPL1AD].
- AD of grid-point calculations [GP_MODEL_AD] (explicit dynamics, simplified physics, SL interpolations).
- AD of "Inverse transforms + compute horizontal derivatives" [(E)TRANSINVHAD].

Remarks :

- Inversion of the order of calls compared to the TL code.
- AD of NH with d3 or d4 not yet coded.

ORGANIGRAMME UNDER STEPOAD.

STEPOAD = management of one timestep, AD model :

- AD of spectral calculations [(E)SPCHAD] (SI scheme, horizontal diffusion).
- AD of direct transforms [(E)TRANSDIRHAD].
- AD of coupling (ALADIN only) [ECOUPL1AD].
- AD of grid-point calculations [GP_MODEL_AD] (explicit dynamics, simplified physics, SL interpolations).
- AD of "Inverse transforms + compute horizontal derivatives" [(E)TRANSINVHAD].

Remarks :

- Inversion of the order of calls compared to the TL code.
- AD of NH with d3 or d4 not yet coded.

ORGANIGRAMME UNDER GP_MODEL_AD FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPOAD $->$ SCAN2HAD $->$ SCAN2MAD $->$ GP_MODEL_AD :

- CPG5 (trajectory unlagged dynamics)
- CPGLAGAD (remaining lagged dynamics)
- EC_PHYS_AD (lagged ECMWF physics)
- CALL_SL_AD (lagged SL dynamics, traj research, interpolations)
- CPGAD (unlagged dynamics, unlagged MF physics)
- Remark : the structure of the AD code does not completely match the TL code (ex : call to CPG5 in the AD code).

ORGANIGRAMME UNDER GP_MODEL_AD FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPOAD $- >$ SCAN2HAD $- >$ SCAN2MAD $- >$ GP_MODEL_AD :

- CPG5 (trajectory unlagged dynamics)
- CPGLAGAD (remaining lagged dynamics)
- EC_PHYS_AD (lagged ECMWF physics)
- CALL_SL_AD (lagged SL dynamics, traj research, interpolations)
- CPGAD (unlagged dynamics, unlagged MF physics)
- Remark : the structure of the AD code does not completely match the TL code (ex : call to CPG5 in the AD code).

ORGANIGRAMME UNDER GP_MODEL_AD FOR SEMI-LAGRANGIAN 3D MODEL.

Under STEPOAD $->$ SCAN2HAD $->$ SCAN2MAD $->$ GP_MODEL_AD :

- CPG5 (trajectory unlagged dynamics)
- CPGLAGAD (remaining lagged dynamics)
- EC_PHYS_AD (lagged ECMWF physics)
- CALL_SL_AD (lagged SL dynamics, traj research, interpolations)
- CPGAD (unlagged dynamics, unlagged MF physics)
- Remark : the structure of the AD code does not completely match the TL code (ex : call to CPG5 in the AD code).

```
control/STEPOAD -> control/SCAN2HAD -> control/SCAN2MAD -> control/GP_MODEL_AD ->
  * adiab/CPG5 (trajectory unlagged dynamics)
  * adiab/CPGLAGAD (AD of CPGLAG)
  * phys_ec/EC_PHYS_AD (AD of ECMWF lagged simplified physics)
  * adiab/CALL_SL_AD ->
    - some parallel environment routines spread in the code (SLCOMM.., (E)SLEXTPOL.. routines).
    - adiab/LAPINEA5 (trajectory version of LAPINEA, manages the SL trajectory research for the AD-trajectory).
    - adiab/LAPINEBAD (manages the interpolations, updates GFLT1,GMVT1,GMVT1S with the interpolated values) ->
      * adiab/LARCINB5 (trajectory for LARCINBAD, does interpolations)
      * adiab/LARCHEAD, adiab/LACONEAD (specific applications)
      * adiab/LARCINBAD (full level variables) ->
        - adiab/LAITRE_GMV_AD (currently some LAI...AD for GMV, also used for GFL)
        - some specific interpolations routines (LAI...AD)
    - adiab/LAPINEAAD ->
      * adiab/LARCINAAD -> (see above)
      * adiab/LARMESAD (future adiab/ELARMESAD in ALADIN) (Trajectory research for AD) ->
        - adiab/LARCINAAD ->
          - adiab/LAITLIAD (AD of trilinear interpolations)
          - adiab/LARCHEAD + adiab/LASCAWAD (future adiab/ELARCHEAD + adiab/ELASCAWAD in ALADIN)
  * adiab/CPGAD ->
    - adiab/CPG5_GP (Trajectory version of CPG_GP)
    - adiab/CPG_END_AD (AD version of CPG_END)
    - adiab/CPG_ZERO_AD (sets local arrays to zero)
    - adiab/CPG_DYN_AD ->
      * sinvect/VDIFLCZAD (AD of Buizza simplified physics)
      * adiab/LACDYNAD (computes the explicit part of the RHS of equations,
        update GFLT1,GMVT1,GMVT1S with quantities at F, fills buffer SLBUF1) ->
        - adiab/LASURE (initialisations)
        - adiab/LATTESAD: fills PB1+PB15
          + updates (GMVT1S+GMVT15S) for 2D variables.
        - adiab/LATTEXAD: fills PB1+PB15
          + updates (GFLT1,GMVT1,GFLT15,GMVT15) for 3D variables.
        - adiab/LAVENTAD: fills PB1+PB15 for trajectory wind.
        - [adiab/LANHSIAD (NH)] or adiab/LASSIEAD (HYD):
          linear terms for SI scheme (trajectory + AD).
    - phys_dmn/MF_PHYSAD (AD of MF simplified physics)
    - adiab/CPG_GP_AD (AD version of CPG_GP)
```

MANAGEMENT OF INTERNAL TRAJECTORY.

- Job with TL or AD code : needs to have the trajectory when running the TL or AD code (direct : Z=XY ; TL : dZ = X dY + Y dX).
- Runs first the direct code, stores the trajectory at least for prognostic variables in buffers TRAJ_CST, TRAJ_SFC, TRAJ_SPEC, TRAJ_GMV, TRAJ_GMVS, TRAJ_GFL (call some STORE_TRAJ_... routines).
- According to LTRAJGP, data stored only in spectral (LTRAJGP=F) or both spectral+grid-point (LTRAJGP=T).
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).
  If LTRAJGP=F additional spectral transforms are required to recover the grid-point trajectory.
- Additionally some trajectory calculations must be done in the TL/AD codes (spread code or specific SUBR..5 routines). Ex : LARMES5.
- In the TL/AD code the trajectory variables have a name generally ending by '5'. Ex : trajectory version of X0 (resp. X9) is X5 (resp. X95).

MANAGEMENT OF INTERNAL TRAJECTORY.

- Job with TL or AD code : needs to have the trajectory when running the TL or AD code (direct : Z=XY ; TL : dZ = X dY + Y dX).
- Runs first the direct code, stores the trajectory at least for prognostic variables in buffers TRAJ_CST, TRAJ_SFC, TRAJ_SPEC, TRAJ_GMV, TRAJ_GMVS, TRAJ_GFL (call some STORE_TRAJ_... routines).
- According to LTRAJGP, data stored only in spectral (LTRAJGP=F) or both spectral+grid-point (LTRAJGP=T).
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).
  If LTRAJGP=F additional spectral transforms are required to recover the grid-point trajectory.
- Additionally some trajectory calculations must be done in the TL/AD codes (spread code or specific SUBR..5 routines). Ex : LARMES5.
- In the TL/AD code the trajectory variables have a name generally ending by '5'. Ex : trajectory version of X0 (resp. X9) is X5 (resp. X95).

MANAGEMENT OF INTERNAL TRAJECTORY.

- Job with TL or AD code : needs to have the trajectory when running the TL or AD code (direct : Z=XY ; TL : dZ = X dY + Y dX).
- Runs first the direct code, stores the trajectory at least for prognostic variables in buffers TRAJ_CST, TRAJ_SFC, TRAJ_SPEC, TRAJ_GMV, TRAJ_GMVS, TRAJ_GFL (call some STORE_TRAJ_... routines).
- According to LTRAJGP, data stored only in spectral (LTRAJGP=F) or both spectral+grid-point (LTRAJGP=T).
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).
  If LTRAJGP=F additional spectral transforms are required to recover the grid-point trajectory.
- Additionally some trajectory calculations must be done in the TL/AD codes (spread code or specific SUBR..5 routines). Ex : LARMES5.
- In the TL/AD code the trajectory variables have a name generally ending by '5'. Ex : trajectory version of X0 (resp. X9) is X5 (resp. X95).

MANAGEMENT OF INTERNAL TRAJECTORY.

- Job with TL or AD code : needs to have the trajectory when running the TL or AD code (direct : Z=XY ; TL : dZ = X dY + Y dX).
- Runs first the direct code, stores the trajectory at least for prognostic variables in buffers TRAJ_CST, TRAJ_SFC, TRAJ_SPEC, TRAJ_GMV, TRAJ_GMVS, TRAJ_GFL (call some STORE_TRAJ_... routines).
- According to LTRAJGP, data stored only in spectral (LTRAJGP=F) or both spectral+grid-point (LTRAJGP=T).
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).
  If LTRAJGP=F additional spectral transforms are required to recover the grid-point trajectory.
- Additionally some trajectory calculations must be done in the TL/AD codes (spread code or specific SUBR..5 routines). Ex : LARMES5.
- In the TL/AD code the trajectory variables have a name generally ending by '5'. Ex : trajectory version of X0 (resp. X9) is X5 (resp. X95).

MANAGEMENT OF INTERNAL TRAJECTORY.

- Job with TL or AD code : needs to have the trajectory when running the TL or AD code (direct : $Z=XY$; TL : $dZ = X\,dY + Y\,dX$).
- Runs first the direct code, stores the trajectory at least for prognostic variables in buffers TRAJ_CST, TRAJ_SFC, TRAJ_SPEC, TRAJ_GMV, TRAJ_GMVS, TRAJ_GFL (call some STORE_TRAJ_... routines).
- According to LTRAJGP, data stored only in spectral (LTRAJGP=F) or both spectral+grid-point (LTRAJGP=T).
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).
  If LTRAJGP=F additional spectral transforms are required to recover the grid-point trajectory.
- Additionally some trajectory calculations must be done in the TL/AD codes (spread code or specific SUBR..5 routines). Ex : LARMES5.
- In the TL/AD code the trajectory variables have a name generally ending by '5'. Ex : trajectory version of X0 (resp. X9) is X5 (resp. X95).

MANAGEMENT OF EXTERNAL TRAJECTORY.

- Application :
  Currently, run the direct code with resol N1 and the TL and AD code with resol N2<N1 in the same job NOT POSSIBLE
  => use of an external high resolution trajectory in a low resolution TL/AD simulation (ex = minimisation).
- Runs first the direct code, reads the high resolution trajectory transformed to low resolution from a file (routines READ_TRAJ_...). Stores these data read in the TRAJ_... buffers.
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).

MANAGEMENT OF EXTERNAL TRAJECTORY.

- Application :
  Currently, run the direct code with resol N1 and the TL and AD code with resol N2<N1 in the same job NOT POSSIBLE
  => use of an external high resolution trajectory in a low resolution TL/AD simulation (ex = minimisation).
- Runs first the direct code, reads the high resolution trajectory transformed to low resolution from a file (routines READ_TRAJ_...). Stores these data read in the TRAJ_... buffers.
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).

MANAGEMENT OF EXTERNAL TRAJECTORY.

- Application :
  Currently, run the direct code with resol N1 and the TL and AD code with resol $N2<N1$ in the same job NOT POSSIBLE
  $=>$ use of an external high resolution trajectory in a low resolution TL/AD simulation (ex $=$ minimisation).
- Runs first the direct code, reads the high resolution trajectory transformed to low resolution from a file (routines READ_TRAJ_...). Stores these data read in the TRAJ_... buffers.
- Runs the TL/AD code $=>$ reads the trajectory from the above buffers (call some GET_TRAJ_... routines).

MANAGEMENT OF EXTERNAL TRAJECTORY.

- Application :
  Currently, run the direct code with resol N1 and the TL and AD code with resol N2<N1 in the same job NOT POSSIBLE
  => use of an external high resolution trajectory in a low resolution TL/AD simulation (ex = minimisation).
- Runs first the direct code, reads the high resolution trajectory transformed to low resolution from a file (routines READ_TRAJ_...). Stores these data read in the TRAJ_... buffers.
- Runs the TL/AD code => reads the trajectory from the above buffers (call some GET_TRAJ_... routines).

PHYSICS PROCESSES.

ARPEGE, ALADIN, IFS :

- Radiation.
- Convection, and convective precipitations.
- Stratiform precipitations.
- Vertical diffusion.
- Orographic gravity wave drag.
- Surface processes.

AROME :

- Radiation.
- Convection.
- Microphysics of warm clouds.
- Microphysics for atmospheric ice.
- Sub-grid condensation.
- Turbulence.
- Surface processes (SURFEX package).

PHYSICS PROCESSES.

ARPEGE, ALADIN, IFS :

- Radiation.
- Convection, and convective precipitations.
- Stratiform precipitations.
- Vertical diffusion.
- Orographic gravity wave drag.
- Surface processes.

AROME :

- Radiation.
- Convection.
- Microphysics of warm clouds.
- Microphysics for atmospheric ice.
- Sub-grid condensation.
- Turbulence.
- Surface processes (SURFEX package).

PHYSICS PROCESSES.

ARPEGE, ALADIN, IFS :

- Radiation.
- Convection, and convective precipitations.
- Stratiform precipitations.
- Vertical diffusion.
- Orographic gravity wave drag.
- Surface processes.

AROME :

- Radiation.
- Convection.
- Microphysics of warm clouds.
- Microphysics for atmospheric ice.
- Sub-grid condensation.
- Turbulence.
- Surface processes (SURFEX package).

ORGANIGRAMME FOR PHYSICS INTERFACE.

Organigramme :

Under STEPO $->$ SCAN2H $->$ SCAN2M $->$ GP_MODEL :

- CPG $->$ MF_PHYS $->$ APLPAR (unlagged MF ARPEGE/ALADIN physics).
- CPG $->$ MF_PHYS $->$ HL_APLPAR (unlagged HIRLAM physics).
- CPG $->$ MF_PHYS $->$ APL_AROME (unlagged MF AROME physics).
- RADDRV (lagged ECMWF radiation).
- EC_PHYS_DRV $->$ ... $->$ EC_PHYS $->$ CALLPAR (lagged ECMWF physics).

Unlagged/lagged :

- Physics never temporally centered (numerically unstable).
- Physics can be evaluated at the initial instant (unlagged) : METEO-FRANCE, HIRLAM applications.
- Physics can be evaluated at the final instant (lagged), or a mix of initial and final instants : ECMWF.
- Parallel call of physical processes (MF, HIRLAM) or mixed sequential/parallel (ECMWF).

ORGANIGRAMME FOR PHYSICS INTERFACE.

Organigramme :

Under STEPO – > SCAN2H – > SCAN2M – > GP_MODEL :

- CPG – > MF_PHYS – > APLPAR (unlagged MF ARPEGE/ALADIN physics).
- CPG – > MF_PHYS – > HL_APLPAR (unlagged HIRLAM physics).
- CPG – > MF_PHYS – > APL_AROME (unlagged MF AROME physics).
- RADDRV (lagged ECMWF radiation).
- EC_PHYS_DRV – > ... – > EC_PHYS – > CALLPAR (lagged ECMWF physics).

Unlagged/lagged :

- Physics never temporally centered (numerically unstable).
- Physics can be evaluated at the initial instant (unlagged) : METEO-FRANCE, HIRLAM applications.
- Physics can be evaluated at the final instant (lagged), or a mix of initial and final instants : ECMWF.
- Parallel call of physical processes (MF, HIRLAM) or mixed sequential/parallel (ECMWF).

ORGANIGRAMME FOR PHYSICS INTERFACE.

Organigramme :

Under STEPO − > SCAN2H − > SCAN2M − > GP_MODEL :

- CPG − > MF_PHYS − > APLPAR (unlagged MF ARPEGE/ALADIN physics).
- CPG − > MF_PHYS − > HL_APLPAR (unlagged HIRLAM physics).
- CPG − > MF_PHYS − > APL_AROME (unlagged MF AROME physics).
- RADDRV (lagged ECMWF radiation).
- EC_PHYS_DRV − > ... − > EC_PHYS − > CALLPAR (lagged ECMWF physics).

Unlagged/lagged :

- Physics never temporally centered (numerically unstable).
- Physics can be evaluated at the initial instant (unlagged) : METEO-FRANCE, HIRLAM applications.
- Physics can be evaluated at the final instant (lagged), or a mix of initial and final instants : ECMWF.
- Parallel call of physical processes (MF, HIRLAM) or mixed sequential/parallel (ECMWF).

THANK YOU / MERCI.