CFU (CUMULATED FLUXES) AND XFU (INSTANTANEOUS FLUXES) IN THE CYCLE 45 OF ARPEGE/IFS.

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

June 28, 2017

Abstract:
This documentation describes some diagnostics done on the physical fluxes: CFU (cumulated fluxes) and XFU (instantaneous fluxes). Some algorithmic aspects and technical aspects (organigramme for example) are described.

Résumé:
Cette documentation décrit certains diagnostics faits sur les flux issus de la physique: les CFU (flux cumulés) et les XFU (flux instantanés). On y aborde certains aspects algorithmiques et techniques (organigramme par exemple).

Contents

1 Introduction.

This documentation has to aim to describe the cumulated fluxes diagnostics (CFU) and the instantaneous fluxes diagnostics (XFU) in the cycle 45 of ARPEGE/IFS. These diagnostics can be used only at METEO-FRANCE with a non-lagged package of physics and ARPEGE files. The post-processing of CFU and XFU is not described in this documentation but in the FULL-POS documentation (IDFPOS).

Cumulated fluxes and instantaneous fluxes are computed on model layers or interlayers when upper air ones; some fields are surface ones. They are grid-point fields and never converted to spectral ones. The CFU and XFU can be activated for fluxes, the physical parameterization which computes them is switched on; otherwise the code aborts. g is the gravity acceleration.
The list of available cumulated fluxes (CFU) can be found in module **PTRGFU**.
The list of available instantaneous fluxes (XFU) can be found in module **PTRXFU**.

2 Algorithm.

$*$ **Instantaneous fluxes:** Fluxes computed in the physics are simply stored in a grid-point buffer **GFUBUF** (**YOMGFUB**) then written on a file.

$*$ **Cumulated fluxes:** The cumulated flux $\overline{F}$ is computed from the instantaneous flux F(t) by the formula:

$$\overline{F} = \int_{t'=0}^{t'=t} F(t')dt \tag{1}$$

the discretisation of which is:

$$\overline{F} = \sum_{jstep=0}^{jstep=nstepcfu} F(t')\Delta t \tag{2}$$

where $nstepcfu$ is the number of the timestep where CFU are requested and Δt is the timestep.

$*$ **Deep layer equations:** Fluxes stored in the code are "additive" ones: extensive quantities divided by a surface which is the projection of the true horizontal surface on a layer where the radius is the mean Earth radius.

3 Files containing CFU and XFU.

- They are ARPEGE files. No code is provided for GRIB files.
- When needed (FULL-POS for example) departure files are read on logical unit **NINISH**, **NINMSH**, **NFGISH**, **NINIGG**, **NPPPSH**, **NULUSR1** (CFU only), **NULUSR2** (XFU only) according to configuration (see **YOMLUN** and **SULUN** for definition of these logical units).
- If the variable **LFBDAP** is .T. in **NAMCT0** the CFU and XFU are written on the historic files (unit **NTRJSH**); otherwise the CFU are written on unit **NULUSR1** and the XFU are written on unit **NULUSR2**.

4 Organigramme.

4.1 Setup routines and call tree above STEPO.

∗ **General architecture under CNT0:** Only features concerning CFU and XFU are mentioned.

```
CNT0 ->
* SU0YOMA ->
  - SUCT0
* SU0YOMB ->
  - SUIOS
  - SUCFU -> SUFPCFU
  - SUXFU -> SUFPXFU
* CNT1 ->
  - SU1YOM ->
    * SUINIF ->
      - SUGRCFU -> (organigramme not detailed)
      - SUGRXFU -> (organigramme not detailed)
    * SUCT1
  - CNT2 -> CNT3 -> CNT4 -> STEPO (see below call tree under STEPO).
```

4.2 Grid-point routines and output routines, call tree under STEPO.

∗ **General architecture under STEPO:** Only features concerning CFU and XFU are mentioned.

```
STEPO ->
* Management of file reading/writing: IOPACK (see below call tree under IOPACK).
* Inverse spectral transforms: TRANSINVH
* Grid point computations SCAN2M -> GP_MODEL_HEAP or GP_MODEL_STACK -> GP_MODEL ->
  (see below call tree under GP_MODEL).
* Direct spectral transforms: TRANSDIRH
* Spectral computations: SPCM.
```

Transforms and spectral calculations are not involved for CFU and XFU.

∗ **Architecture under IOPACK, WRFU, WRXFU:** Part 5 of **IOPACK** concerns partly CFU and XFU.

```
IOPACK ->
* [ WRMLPP if global model -> ] WRMLPPA ->
  - WRGRIDALL if LUSEWRGRIDALL=T -> (organigramme not detailed)
  - WRXFU if LUSEWRGRIDALL=F -> (organigramme not detailed)
  - WRFU if LUSEWRGRIDALL=F -> (organigramme not detailed)
* some other routines not used for CFU and XFU
```

This call tree is called when CFU and XFU are written on historic files.

∗ **General architecture of SCAN2M:** **SCAN2M** →:

- Some memory transfers and pointer computations before grid-point computations.
- Comparison with observations (non-lagged part, then information communication between processors, then lagged part).
- Model grid-point computations (non-lagged part, then information communication between processors, then lagged part).
- Grid-point computations for analysis.

∗ **General architecture concerning CFU and XFU in SCAN2M:**

```
SCAN2M -> GP_MODEL_HEAP or GP_MODEL_STACK -> GP_MODEL ->
* CPG_DRV -> CPG ->
  - non-lagged physics (MF_PHYS -> call tree not detailed)
  - CPG_DIA -> CPCFU and CPXFU (organigramme not detailed)
```

4.3 Action and brief description of each routine.

- Expression “full level” is synonym of “middle of layer”.
- Expression “half level” is synonym of “interlayer”.
- For meaning of [L5] see section “Sequences of calls of post-processing”.

∗ Grid-point and spectral routines of directory “adiab”:

- **CPG_DRV**: driver for non lagged part of grid-point calculations.
- **CPG**: non lagged part of grid-point calculations, including model dynamics, non lagged physics, diagnostics.
- **CPG_DIA**: part of the non lagged part grid-point calculations managing the diagnostics.

∗ Control routines of directory “control”:

- **CNT0**: controls integration job at level 0.
- **CNT1**: controls integration job at level 1.
- **CNT2**: controls integration job at level 2.
- **CNT3**: controls integration job at level 3.
- **CNT4**: controls integration job at level 4.
- **SCAN2M**: interface for grid-point computations.
- **GP_MODEL**: part of the grid-point computations for model and some diagnostics (CFU,XFU,DDH).
- **SPCM**: interface for spectral computations.
- **STEPO**: control routine for one time integration step.

∗ Routines of directory “dia”.

- **CPCFU**: grid point calculations of the CFU fields.
- **CPXFU**: grid point calculations of the XFU fields.
- **WRFU**: routine to write CFU fields on an ARPEGE file.
- **WRGRIDALL**: interface routine to write grid-point fields, called if **LUSEWRGRIDALL**=T.
- **WRMLPP** and **WRMLPPA**: interface routines to write model-layer fields on an historic file.
- **WRXFU**: routine to write XFU fields on an ARPEGE file.

∗ Distributed memory environment routines (directory “parallel”): See documentation (IDDM) about distributed memory features.

∗ Set-up routines of directory “fullpos”:

- **SUFPCFU**: initialises cumulated fluxes switches for FULL-POS: corrects the values of the **YOMCFU** variables in order to compute the CFU which are post-processed.
- **SUFPXFU**: initialises instantaneous fluxes switches for FULL-POS: corrects the values of the **YOMXFU** variables in order to compute the XFU which are post-processed.

∗ Set-up routines of directory “setup”:

- **SUCFU**: initialises the control of cumulated fluxes: reads **NAMCFU**, sets-up variables of **YOMCFU**, computes the **PTRGFU** pointers.
- **SUCT0**: routine to initialise level 0 control module.
- **SUCT1**: sets-up **YOMCT1**.
- **SUGRCFU**: reads the cumulated fluxes on ARPEGE files.
- **SUGRXFU**: reads the instantaneous fluxes on ARPEGE files.
- **SUINIF**: interface routine for reading the departure files.
- **SUIOS**: sets-up **YOMIOS**.
- **SUXFU**: initialises the control of instantaneous fluxes: reads **NAMXFU**, sets-up variables of **YOMXFU**, computes the **PTRXFU** pointers.
- **SU0YOMA**: 0-level interface routine for set-up: first part.
- **SU0YOMB**: 0-level interface routine for set-up: second part.
- **SU1YOM**: 1-level interface routine for set-up.

∗ Routines of directory “transform”. For more details see documentation (IDTS) about spectral transforms.

- **TRANSDIRH**: interface routine for direct spectral transforms.
- **TRANSINVH**: interface routine for inverse spectral transforms.

∗ Routines of directory “utility”.

- **IOPACK**: interface for writing data on ARPEGE, ALADIN or GRIB files.

5 Sequences of calls.

Cumulated fluxes and instantaneous fluxes are computed on model layers or interlayers when upper air ones; some fields are surface ones. CFU and XFU grid-point part calls a special sequence for **STEPO**. A sequence is defined by nine letters (or zeros) [L1][L2][L3][L4][L5][L6][L7][L8][L9] (variable **CLCONF** in routine **control/CNT4** and **CDCONF** in routine **control/STEPO**).

- L1 controls the file reading/writing.
- L2+L3 controls the inverse transforms.
- L4 controls the grid-point computations for dynamics and physics.
- L5 controls the grid-point computations for some diagnostics.
- L6 controls the grid-point computations for assimilation.
- L7 controls the coupling in LAM models.
- L8 controls the direct transforms.
- L9 controls the spectral computations.

For example a model integration time-step is defined by the sequence [L1]AAA00AAA. Additional sequences can be performed by calls to **SCAN2M** under routines other than **STEPO**. The sequence called for CFU and XFU is [L1]AAX00000.

6 Some distributed memory features.

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**. The total number of processors is **NPROC=NPRGPNS*NPRGPEW**.

One 2D model field has **NGPTOTG** points divided into **NPROC** sets of **NGPTOT** points treated by each processor. **NGPTOT** may be processor-dependent (with very small variations): the maximum value of **NGPTOT** is **NGPTOTMX**.

For one given processor, the **NGPTOT** points are divided into packets of length **NPROMA** (the useful number of values in each packet is lower or equal than **NPROMA**). **NPROMA** is identical for all processors. There are **NGPBLKS** blocks of **NPROMA** packets:

$$\mathbf{NGPBLKS} = int[(\mathbf{NGPTOT} + \mathbf{NPROMA} - 1)/\mathbf{NPROMA}]$$

A **NPROMA**-packet does not always contain a set of complete latitudes.

7 Pointer, module and namelist variables to be known.

These modules are auto-documented so description of each variable is provided in the code source. We can recall here the most important variables to know for each module:

- **PTRGFU** (contains the pointers relative to the CFU).
- **PTRXFU** (contains the pointers relative to the XFU).
- **YOMCFU** (contains variables relative to the CFU, in particular contains keys switching on cumulated fluxes). Some of them are in namelist **NAMCFU**.
- **YOMXFU** (contains variables relative to the XFU, in particular contains keys switching on instantaneous fluxes). Some of them are in namelist **NAMXFU**.
- **YOMGFUB** (contains buffer GFUBUF for CFU).
- **YOMXFUB** (contains buffer XFUBUF for XFU).
- **YOMCT0** (0-level control).
- **YOMCT1** (1-level control). In particular N1CFU and N1XFU. Some of them are in namelist **NAMCT1**.
- **YOMDIM**, **YOMDIMV** and **YOMDIMF** (dimensioning): most of variables. Some of these variables are in namelist **NAMDIM**.
- **YOMLUN** (logical units).
- **YOMMP0** and **YOMMP** (distributed memory environment, see documentation (IDDM) for more details).
- **YOMOPH0**. In particular LINC and LTIMEP0. Some of them are in namelist **NAMOPH**.
- **TYPE_FLUXES** (defines descriptors of families of model fluxes).
- **PARFPOS**, **YOM4FPOS**, **YOMAFN**, **YOMFP4**, **YOMFPC** for applications doing post-processing on CFU and XFU (see FULL-POS documentation (IDFPOS)).

8 References.

- (TDECDYN) 2016: IFS technical documentation (CY41R2). Part III: dynamics and numerical procedures. Available at "https://software.ecmwf.int/wiki/display/IFS/Official+IFS+Documentation".
- (TDECTEC) 2016: IFS technical documentation (CY41R2). Part VI: technical and computational procedures. Available at "https://software.ecmwf.int/wiki/display/IFS/Official+IFS+Documentation".
- (IDBAS) Yessad, K., 2017: Basics about ARPEGE/IFS, ALADIN and AROME in the cycle 45 of ARPEGE/IFS (internal note).
- (IDFPOS) Yessad, K., 2017: FULL-POS in the cycle 45 of ARPEGE/IFS (internal note).
- (IDDM) Yessad, K., 2017: Distributed memory features in the cycle 45 of ARPEGE/IFS (internal note).
- (IDEUL) Yessad, K., 2017: Integration of the model equations, and Eulerian dynamics, in the cycle 45 of ARPEGE/IFS (internal note).