# The variational computations inside ARPEGE/ALADIN : cycle CY32

Claude Fischer [1]
with contributions by Ludovic Auger and Bernard Chapnik
Météo-France, CNRM/GMAP

2 février 2007

[1]corresponding author address : 42, avenue Gustave Coriolis, 31057 TOULOUSE CEDEX (France)

# Table des matières

**1 Acknowledgements** **3**

**2 Introduction** **4**
2.1 History of the Arpège variational assimilation . . . . . 4
2.2 Genesis of the Aladin variational analysis . . . . . 4
2.3 Goal of this Arpège and Aladin documentation . . . . . 5

**3 Formalism of incremental variational analysis** **6**
3.1 3dvar . . . . . 6
3.2 Background error covariance matrix $B$ . . . . . 7
3.2.1 Expression of $B$ and signification of $B^{-1/2}$ . . . . . 7
3.2.2 The sequential projection on decorrelated components . . . . . 8
3.2.3 Cross-covariances . . . . . 8
3.2.4 Horizontal auto-covariances . . . . . 8
3.2.5 Vertical covariances . . . . . 9
3.2.6 Compactly supported correlations - stressing the LAM case . . . . . 9
3.2.7 Design of the horizontal balance $\mathcal{H}$ in Aladin . . . . . 9
3.2.8 Design of the horizontal balance $\mathcal{H}$ in Arpège . . . . . 10
3.3 Preconditioning . . . . . 10
3.4 4dvar . . . . . 11
3.4.1 Formulation . . . . . 11
3.4.2 Simplified physics . . . . . 11
3.4.3 Multi-incremental approach . . . . . 11
3.5 3dvar/fgat . . . . . 11

**4 Data flow and code organisation** **12**
4.1 Screening . . . . . 12
4.1.1 Specific geometry settings and setup of observations . . . . . 12
4.1.2 Computing the obs-guess departures . . . . . 13
4.1.3 Obs/physics wedding for “wet radiances” . . . . . 15
4.1.4 Screening, thinning and quality control . . . . . 15
4.2 3dvar minimisation (focussing on Aladin) . . . . . 22
4.2.1 The different parameters within the control variable . . . . . 24
4.2.2 Setup of $B$ . . . . . 24
4.2.3 Control level . . . . . 26
4.2.4 Simulator . . . . . 26
4.2.5 Change of variables and canonical injections . . . . . 29
4.2.6 TL observation operators . . . . . 31
4.2.7 AD observation operators . . . . . 38
4.3 4dvar minimisation . . . . . 47
4.3.1 Trajectory handling . . . . . 47
4.3.2 A very short overview of simplified physics . . . . . 47
4.3.3 Specific multi-incremental features . . . . . 47
4.4 conjugate gradient . . . . . 47
4.5 3dvar/fgat . . . . . 47

**5 User's guide : scripts and namelists** **48**
5.1 Sequences of the execution and Input/Output . . . . . 48
5.1.1 Creation of a limited area observational database in ODB format (Aladin only) . . . . . 48
5.1.2 Screening of the data . . . . . 48
5.1.3 3dvar/4dvar/FGAT minimizations . . . . . 49
5.2 Most important namelist parameters . . . . . 51
5.2.1 LAMFLAG (Aladin only) . . . . . 51
5.2.2 Screening . . . . . 51
5.2.3 3dvar/4dvar/FGAT minimizations . . . . . 51
5.3 How to check for the execution ? . . . . . 57
5.4 Frequently encountered ... crashes . . . . . 58

**A Configurations of the Arpège/Aladin executables** **59**

**B Domain definitions in Aladin** **60**

**C Communications between observation and model spaces in DM mode** **62**

**D Aspects of the data distribution on MPP machines** **63**

# Chapitre 1

# Acknowledgements

The author appreciated many general and detailed discussions with the core developpers of the Arpège assimilation system at CNRM/GMAP : Jean-Noël Thépaut, Vincent Cassé (for animated discussions about lateral boundaries and spectral covariance computations) and Marta Janiskova (for TL/AD coding and extra stress such as "whatever you do, don't touch *my* simplified physics"). I would like to especially thank Patrick Moll, who proved to be a reliable contact person for Aladinists over the years, to help solving probably just as many problems as they are lines of code in the variational algorithm.

The initial reviewers Thibault Montmerle and Patrick Moll provided interesting guidelines to improve the content and the readibility of the document.

# Chapitre 2

# Introduction

## 2.1 History of the Arpège variational assimilation

## 2.2 Genesis of the Aladin variational analysis

The decision to derive a variational analysis tool for the limited area model ALADIN dates back to the early 90ies, and was at that time driven by the pioneering spirit of András Horányi, Liviu Dragulanescu and a few others. The first model configurations at work concerned the tangent linear and adjoint propagation models : test of the tangent linear model (configuration 501[1]), test of the adjoint (401), computations of gradients with respect to the initial conditions (801). These models ran in 1995/96 in shared memory on the Cray C90 at Météo-France. A kick-off workshop for the validation of the "observational" part of the variational code, that is quality control by first-guess check (configuration 001/002 "screening") and 3dvar minimisation (131), took place in Toulouse in spring 1997. This workshop gathered almost 15 people. However, part of the developments still were shaky at that time : a first screening was performed, able to compute an observational cost function, but the minimisation was not yet validated. In 1998 and 1999, an important work was sustained to port the variational code to distributed memory architecture (András Horányi, Maria Široka, Claude Fischer). In june/july 99, a small workshop finally permitted to validate for the first time the screening and the 3dvar minimisation on the VPP700 of Météo-France (at that time, Cornel Soci and Wafaa Sadiki joined the group). Furthermore, the computation of singular vectors was achieved as well.

In parallel, the computation and the evaluation of a first $J_b$ matrix was obtained, based on the NMC method (Loïk Berre, Maria Monteiro, Maria Široka). This tool allowed first scientific publications (Berre, 2000, Sadiki etal., 2000, Široka etal., 2003).

The early years 2000 have seen a many-fold evolution of the "ALADIN/VAR" projects :

- the actual difficulties of maintaining such complex configurations, using both observational databases, model environment and ALADIN specificities, in the frame of the ever evolving ARPEGE/IFS corset. Maintenance of ALADIN/3dvar demands a consequent subset of knowledge about the ARPEGE/4dvar environment.
- the practical apparition of the new ODB dataset format.
- the necessity for the ALADIN community to gain knowledge about observations : their physical properties (values, errors, observation operators), their availability (transmission, observation stations, remote sensing, storage), their impact on the analysis.
- local installations of 3dvar : Prague (summer 99 for the $J_b$, early 2000 for the configurations 002/131), Budapest (summer 2000), Casablanca (late 2001).

These points are just but a few among the many challenges the ALADIN variational community will have to face in the coming years. Others would be more general : maintain the scientific knowledge and the quality of work in variational applications, train new people and

[1] the "configuration" is the name given to any application which can be run with the model binary file plus a specific namelist. For example, the forecast is configuration 001. Every configuration of the model is identified by a 3-digit number. A list of identified configurations is given in Table A.1.

face the "human turnover". One crucial point will be to assure a correct balance between the 3 pillars of the project (science, human touch, technical maintenance).

The year 2005 may be marked as a special one, since it has seen the first operational applications of Aladin 3D-VAR : Hungary (May 05) and France (July 05, see Fischer etal., 2005).

## 2.3 Goal of this Arpège and Aladin documentation

The goal of this note is to give a quite detailed description of the algorithmic part, for the benefit of central phasing in Toulouse, remote installations and development. The reader can compare this technical section with the theoretical overview, which is recalled for completeness. A "User's Guide" chapter is added for the general users of 3dvar, who either are not interested in the technical details, or are at their beginning stage (where usually the same FAQ arise). This document is supposed to follow the code updates, though for understandable reasons, the author asks for tolerance should the documentary update not be in close phase with the cycling of ARPEGE/ALADIN. More information on the variational assimilation and data assimilation in general can be obtained by visiting the ECMWF web site, where one can find a very exhaustive on-line documentation on ARPEGE/IFS 4dvar. Especially, many items such as observation operators, data distribution, reduced rank Kalman filter, $J_c$, etc ... are described there in much more details than in this technical note : http ://www.ecmwf.int/

General information about the ARPEGE/ALADIN galaxy, or other technical tools, can be obtained from the ALADIN web server : http ://www.cnrm.meteo.fr/aladin/

# Chapitre 3

# Formalism of incremental variational analysis

This note is not a lecture course on atmospherical data assimilation. Only some basic formulae are recalled, to clarify the algorithmic and user's guides. The principles of data assimilation can be found in adequate text books such as Daley (1991) or on-line documents, for example Gérald Desroziers's lecture notes from the Gourdon Alatnet seminar (download from http ://intra.cnrm.meteo.fr/gmap/index.html, "Documentation Principale"/"Cours et tutoriaux" in the menu).

## 3.1 3dvar

The formalism of the variational analysis follows closely the work in ARPEGE/IFS as published by Courtier etal. (1994) or Courtier etal. (1998). 3dvar minimizes the cost function $J$ :

$$J = J_b + J_o + J_c$$

which consists of the normalized distances of any model state towards the background field (usually taken as a 6 hour forecast), the observations and a filtered model state, respectively. In ALADIN, we do not have so far any filtered constraint, so $J_c = 0$. Possibilities in 3dvar to use normal modes were never considered (and so far never felt mandatory). For an Arpège 3dvar, $J_{cNMI}$ may be activated. In later 4dvar, a digital filter constraint might be considered ($J_{cDFI}$, as presented in Gauthier and Thépaut, 2001).

The first two cost functions read, in incremental version :

$$J_b = \frac{1}{2}\delta x^t B^{-1} \delta x$$

$$J_o = \frac{1}{2}(H\delta x - d)^t R^{-1}(H\delta x - d)$$

where $\delta x$ is the increment, $B$ and $R$ stand for the error covariance matrices of the background and the observations, resp., and $H$ is the linearized observation operator. The innovation $d$ reads :

$$d = y - \mathcal{H}(x^b)$$

$d$ represents the departures between the observations and the background model state, interpolated to the observation space. $\mathcal{H}$ contains both spatial vertical and horizontal interpolations, and physical inversions or conversions if needed. For example, the radiative transfer equation is part of the $\mathcal{H}$ operator for radiance data. $\mathcal{H}$ can be strongly non-linear, and even possess some dependence with the background field (for example, a background temperature profile). $H$ is then the linear approximation of $\mathcal{H}$ in the vicinity of the background $x^b$. In a spatially incremental algorithm, such as the ARPEGE multi-incremental 4dvar, $H$ is a low-resolution, linear approximation.

The incremental formulation of 3dvar consists now in solving for $\delta x$ the inverse problem defined by the direct observation operator $\mathcal{H}$, given the innovation vector $d$ and the background constraint. The gradient of $J$ is obtained by differentiation with respect to $\delta x$ :

$$\nabla_{\delta x} J = (B^{-1} + H^t R^{-1} H)\delta x - H^t R^{-1} d$$

At the minimum, the gradient vanishes, so that we retrieve the classical relation for the analysis final increment :

$$(B^{-1} + H^t R^{-1} H)\delta x^a = H^t R^{-1} d$$

which satisfies also the Optimal Interpolation (OI) equation, provided the very same observation operators are considered (thus, assuming mainly linear operators) :

$$\delta x^a = x^a - x^b = BH^t(HBH^t + R)^{-1} d$$

Most OI applications depend on statistical models for $HBH^t$ and $BH^t$ (see Lönnberg and Hollingsworth, 1986, for example). Variational algorithms use directly the observation operators. Thus, the variational approach is better suited for complex data such as satellite radiances. However, in many real situations, the practical implementation is done in several steps, starting first with a 1dvar inversion of the raw data to provide "observed" model quantities, and followed only later by the actual use of a genuine observation operator (radiances for sure, precipitations ?, radar reflectivities ?). Finally, statistical models for the $B$ and $R$ matrices still have to be provided. The description of the ALADIN $B$ matrix is given in sections 3.2 and 4.2.2.

## 3.2 Background error covariance matrix $B$

### 3.2.1 Expression of $B$ and signification of $B^{-1/2}$

The background error covariance matrix is written as the product of several sparse matrices :

$$B = KFSF^{-1}DVWW^tV^tD^tF^{-t}S^tF^tK^t$$

In order to explain this long and apparantly complex sequence of operators, one may note that it relies on the following expression for the square-root of the covariance matrix $B^{1/2}$, which is such as $B = B^{1/2}B^{t/2}$ :

$$B^{1/2} = KFSF^{-1}DVW$$

Explaining the design of $B^{-1/2}$ (the inverse of $B^{1/2}$) is itself easier. One can note indeed that $B^{-1/2}$ is an operator $U$ that transforms $\epsilon$, the background error vector in the model space, into another variable $\hat{\epsilon} = U\epsilon$, whose covariance matrix is diagonal and equal to the identity matrix :

$$\overline{\hat{\epsilon}\hat{\epsilon}^t} = \overline{U\epsilon\epsilon^t U^t} = U\overline{\epsilon\epsilon^t}U^t = UBU^t = B^{-1/2}B^{1/2}B^{t/2}B^{-t/2} = I$$

In other words, designing $B^{-1/2}$ amounts defining a sequence of operators that projects $\epsilon$ into a space where its components are decorrelated.

In the following, we will describe the scientific meaning of the sequence of $B$-matrix decomposition. The description of the setup calling tree can be found in §4.2.2, and the calling tree for the application of $B^{-1/2}$ is found in §4.2.5, always for the Aladin case.

### 3.2.2 The sequential projection on decorrelated components

There are three types of covariances involved for the errors $\epsilon$ in the model space (which consist in errors on the following variables : vorticity $\zeta$, divergence $\eta$, temperature $T$, logarithm of surface pressure $P_s$, and specific humidity $q$) :
- Cross-covariances between the different model variables.
- Horizontal auto-covariances for each model variable.
- Vertical auto-covariances for each model variable.

These three kinds of covariances are accounted for successively, and some specific tranforms are used at each step, to obtain a new variable with the corresponding decorrelation. The complete sequence of operators that defines $B^{-1/2}$ is the following one, in which three main steps can be distinguished by some parenthesis :

$$B^{-1/2} = W^{-1}V^{-1}D^{-1}FS^{-1}F^{-1}K^{-1} = (W^{-1}V^{-1})\ (D^{-1}FS^{-1}F^{-1})\ (K^{-1})$$

For a detailed evaluation of the principles that underly this decomposition, we refer to Derber and Bouttier (1999), Berre (2000), Sadiki etal. (2000) or Široka etal. (2003).

### 3.2.3 Cross-covariances

Accounting for cross-covariances, and obtaining variables whose cross-correlations are zero, can be done by using the following set of multiple linear regressions :

$$\begin{array}{rcl} \zeta & = & \zeta \\ \eta & = & \mathcal{M}\mathcal{H}\zeta + \eta_u \\ (T, P_s) & = & \mathcal{N}\mathcal{H}\zeta + \mathcal{P}\eta_u + (T, P_s)_u \\ q & = & \mathcal{Q}\mathcal{H}\zeta + \mathcal{R}\eta_u + \mathcal{S}(T, P_s)_u + q_u \end{array}$$

where $H$ is a geostrophic operator, and the matrices $\mathcal{M}, \mathcal{N}, \mathcal{P}, \mathcal{Q}, \mathcal{R}, \mathcal{S}$ contain the regression coefficients (they are called the "balance operators") ; they are deduced from the cross- and auto-covariance matrices. For instance, $\mathcal{M} = \overline{\eta P_b^t}(\overline{P_b P_b^t})^{-1}$ with $P_b = H\zeta$. The subscript $u$ is used for the residual of the regressions, and $x_u$ is sometimes called the "unbalanced" part of the variable $x$.

An important property of the linear regressions is that the residuals are decorrelated with the predictors. This can be illustrated by looking e.g. at the covariance between $\eta_u$ and $P_b = H\zeta$ :

$$\begin{array}{rcl} \overline{\eta_u P_b^t} & = & \overline{(\eta - \mathcal{M}P_b)P_b^t} \\ & = & \overline{\eta P_b^t} - \mathcal{M}\overline{P_b P_b^t} \\ & = & \overline{\eta P_b^t} - \overline{\eta P_b^t}(\overline{P_b P_b^t})^{-1}\overline{P_b P_b^t} \\ & = & 0 \end{array}$$

Noting that $\zeta = H^{-1}P_b$, it can be verified that this translates into $\overline{\eta_u \zeta} = 0$.

This can be written as : $\epsilon = K\epsilon_u$, with $\epsilon = \begin{pmatrix} \zeta \\ \eta \\ TP_s \\ q \end{pmatrix}$ and $\epsilon_u = \begin{pmatrix} \zeta \\ \eta_u \\ (TP_s)_u \\ q_u \end{pmatrix}$,

or equivalently $\epsilon_u = K^{-1}\epsilon$.

The use of linear regressions ensures that the four components of $\epsilon_u$ are decorrelated.

### 3.2.4 Horizontal auto-covariances

If spatial covariances are horizontally homogeneous, spherical harmonics (Arpège) or bi-Fourier (Aladin) spectral modes that differ are decorrelated. In other words, each three-dimensional auto-covariance matrix is block-diagonal in spectral space in this case (with one block of vertical correlations for each wave number). Being in spectral space rather than in gridpoint space therefore allows to have some horizontal decorrelation.

On the other hand, it is easier to represent the horizontal variations of variances in gridpoint space than in spectral space. Therefore a transformation to physical space is used, followed by a division by the gridpoint standard-deviations, and by a return to spectral space. This is represented by the sequence of operators $FS^{-1}F^{-1}$ where $F$ is the Fourier tranform, and $S$ is the diagonal matrix that contains the gridpoint standard-deviations. This facility to use gridpoint $\sigma_b$'s is presently only provided in Arpège/IFS (key LSPFCE=.FALSE.), and was so far never tested nor implemented in Aladin. A diversion of this method however was introduced in order to define a gridpoint mask for $\sigma_b$'s in a rim zone around an inner border of C+I (with keys LSPFCE=.TRUE. and LEVARGP=.TRUE., see routine SUEVARGP).

In order to get closer to the identity matrix, a division by the spectral standard-deviations is then applied : this corresponds to the diagonal matrix $D^{-1}$. These spectral standard-deviations describe what is the contribution of each wave number to the total variance, and they determine the shape of the horizontal correlations.

### 3.2.5 Vertical covariances

The remaining covariances that need to be accounted for are the vertical ones. The projection on the eigen-vectors of the vertical auto-correlation matrix (of each variable and of each wave number), which is represented by the matrix $V^{-1}$, allows to obtain a variable whose elements are completely decorrelated.

The corresponding covariance matrix is thus diagonal. The last step is therefore to normalize by the square-root of the eigen-values (which is represented by the matrix $W^{-1}$) : this provides a final variable $\hat{\epsilon}$, whose covariance matrix is the identity matrix.

### 3.2.6 Compactly supported correlations - stressing the LAM case

In the single observation analysis experiments, it can be seen that the raw 3D-VAR increments possess a slobbery aspect, with a tail of increment able to span over the full computational domain, though the $J_b$ lengthscales are very short (of the order of 30 to 100 km). In addition, the increments can wrap around the domain, and reappear on the opposite side due to their biperiodic nature. To circumvent these shortcomings, or at least to partly diminish these effects, the isotropic Aladin structure functions are forced over compact supports. Compact supports force the structure functions to be zero after a predefined distance from one single observation (Gaspari and Cohn, 1999).

Taking advantage of the isotropic assumption, each 1D correlation function of the Aladin $B$-matrix can be injected into an elliptic 2D spectral function, which is transformed to gridpoint, multiplied by a gridpoint mask and eventually transformed back to spectral and reinjected into a 1D profile. The result is a spectral, isotropic 1D correlation profile which ensures zero correlation for distances larger than the radius defined in the gridpoint mask. The toroidal geometry of Aladin forces one to use bi-Fourier transforms, instead of the single Legendre transform performed in Arpège (Courtier etal., 1998, F. Bouttier - personal communication -). The operations are concentrated in the setup routine SUEJBCOSU for Aladin and SUJBCOSU for Arpège.

Tests have shown some improvement with compact support, when bands of observations are analyzed over a sub-part of the domain. This is typically the case when only a band of satellite data is present. When the whole domain is covered by observations, compact supports do not show any significant difference compared with the usual $B$ (V. Guidard, personal communication).

### 3.2.7 Design of the horizontal balance $\mathcal{H}$ in Aladin

As noted in the section 3.2.3, the operator $\mathcal{H}$ transforms vorticity $\zeta$ into the so-called balanced geopotential $P_b = \mathcal{H}\zeta$. In this section, we describe the two possible expressions for this geostrophic balance operator $\mathcal{H}$.

A first possible option is to use a linear regression between the spectral coefficients of vorticity $\hat{\zeta}$ and those of the linearized geopotential $\hat{P}_t$ :

$$\hat{P}_b(m,n) = \beta(k^*)\ \hat{\zeta}(m,n)$$

where $\beta(k^*)$ is a vertical and isotropic average of the linear regression coefficients $\beta'$ : $\beta'(m,n) = \frac{cov(\hat{P}t(m,n),\hat{\zeta}(m,n))}{var(\hat{\zeta}(m,n))}$

This is very close to the analytical $f$-plane balance equation :

$$\hat{P}_b(m,n) = \Delta(m,n)^{-1} f_0\ \hat{\zeta}(m,n)$$

A second possible option (El Ouaraini and Berre, 2004) is to use something close to a $\beta$-plane balance equation, by setting the namelist variable LJBBALBETA equal to TRUE. This allows to represent most of the horizontal variations of $f$ in the following equation :

$$P_b = \Delta^{-1}(f\zeta)$$

It can be shown that this can be done in the space of the bi-Fourier complex coefficients using a multi-diagonal approach, for the multiplication between $f$ and $\zeta$ :

$$(\hat{f\zeta})(m,n) = \sum_{p=-P}^{+P} \hat{f}(0,p)\hat{\zeta}(m,n-p) \tag{3.1}$$

where $P$ is a suitable truncation parameter, which corresponds to the number of wavenumbers that allows to represent most of the horizontal variations of $f$ (a usual order of magnitude of $P$ is around 15). The corresponding namelist variable is called $NPSMAX$.

This is close to the following $\beta$-plane balance equation : $P_b = \Delta^{-1}((f_0 + \beta)\zeta)$, although the coefficient $\beta$ is not explicitly involved in the equation (3.1).

Note moreover that the equation (3.1) is valid for complex bi-Fourier coefficients. It can be shown that this equation in complex space is much more simple than the four corresponding expressions for the usual ALADIN real quadruplets. It has appeared therefore appropriate (and with no increase of cost) to code the balance equation in the space of the complex bi-Fourier coefficients (or more precisely, in the space of their real and imaginary parts), and to use the direct and inverse transformations between the two involved spaces (namely the complex space and the space of real quadruplets).

The equations of these transformations can be found in Joly (1992), except that the correct equation for the fourth element of the quadruplet, $Q_{mi}^{ni}$, is the following :

$$Q_{mi}^{ni} = -\frac{1}{2}(Q_{m,r}^{n} - Q_{m,r}^{-n})$$

instead of its opposite :

$$Q_{mi}^{ni} = \frac{1}{2}(Q_{m,r}^{n} - Q_{m,r}^{-n})$$

### 3.2.8 Design of the horizontal balance $\mathcal{H}$ in Arpège

## 3.3 Preconditioning

A classical result from the theory of minimisation is that the search is made more efficient if the problem is made isotropic : for a purely quadratic cost function, the search for the minimum along a spherical surface will be more efficient than to search for the same solution along a distorted elliptic surface. To achieve this preconditioning, it is convenient to normalize the space of possible solutions by an adequate norm, ideally the Hessian of the cost function :

$$\nabla\nabla_{\delta x} J = B^{-1} + H^t R^{-1} H$$

This quantity, of the same size than $B$, also corresponds exactly to the analysis error covariance matrix. In practice, this operator is not known (the usual estimation methods such as a simplified Kalman filter or derivations from conjugate gradient methods still are under study for global models). The actual renormalization is therefore done by the background error covariance matrix alone, and we define the actual control variable of the 3dvar algorithm as :

$$\chi = B^{-1/2}\delta x = L\delta x$$

Note that in the space of this renormalized control variable, the $J_b$ cost function and its gradient become trivial :

$$J_b = \chi^t \chi$$

$$\nabla_\chi J_b = 2\chi$$

On the opposite, one needs to perform forth and back conversions for the observational cost function and its gradient.

## 3.4 4dvar

### 3.4.1 Formulation

### 3.4.2 Simplified physics

Scientific documentation on the Arpège simplified physics package can be found in Janisková (1998) and in Janisková *etal.*(1999).

### 3.4.3 Multi-incremental approach

The strategy and mathematical developments for the multi-incremental 4D-VAR approach can be found in Veersé (1997) and in Veersé and Thépaut (1998).

## 3.5 3dvar/fgat

# Chapitre 4

# Data flow and code organisation

Here, we describe the general data flow, trying to stress the link with the theoretical formalism and the ALADIN specific issues where relevant. Please note that we do not intend to be exhaustive in the listing of the Aladin specificities ! The "Arpège/Aladin Tech'book" ("Implications of LAM aspects on the global model code"), co-authored by Mark Žagar and Claude Fischer, lists a more extensive number of switches for Aladin inside the Arpège code. Most of these dataflow diagrams and calling trees directly apply to the Arpège model, which is the technological backbone of all this document.

## 4.1 Screening

Screening is the last step of the pre-treatment of the data to be used in the analysis. We seek to remove the wrong data and to make the final choices inside the set of data which are found to be potentially acceptable by the control and the monitoring. The observations are controlled against the background, to verify their vertical consistency, and then thinned when their spatial density is too high compared to the resolution of the analysis (to avoid representativeness error). A further description of the several steps of screening is done in §4.1.4, and a lecture course is available on-line on the GMAP intranet page (http ://intra.cnrm.meteo.fr/gmap/index.html, section "Documentation Principale"/"Cours et tutoriaux").

Screening consists in running the direct forecast model, plus one or several calls to the observation operators, followed by the first guess check, the quality control and the thinning of the observations. One usually separates 3d-screening (where only the initial model state is read and observation operators are called once), from 4d-screening where the full forecast is actually computed and observation operators are called in the middle of each timeslot. In 3d-screening, all the observations are considered as co-located in time, while in 4d-screening, the observations are distributed into one-hour timeslots (thus far, this is the same for Aladin and Arpège). As such, screening runs through the configuration 001 control routines, with extra calls to the setup of observations, and then calls the screening monitor (SCREEN). Through command line, screening is identified as "configuration 002". However, the configuration number NCONF is quickly reset to 1 under SU0YOMA.

### 4.1.1 Specific geometry settings and setup of observations

For the limited area model, some switches have been introduced in the global model screening. They basically take into account the plane geometry of Aladin (instead of spherical) and the limited area data distribution and observation positioning. Here are the most remarkable Aladin patches :

- OBATABS : this routine deals with the location of the obs points in the model grid, and sets up obs tables. Especially, OBATABS writes the positions of the observations with respect to the transformed model grid (thus, plane geometry for Aladin) into the observation database (at ROBHDR( :,MDBTLA) and ROBHDR( :,MDBTLO)). For Aladin, we call the interface to the LAM geometry (ELALO2XY) and check for the location of the observations with respect to the C+I area (see appendix B). Originally, the Aladin code in OBATABS was designed to treat observations inside an area bigger than the C+I zone (thus, located in the E zone or further). However, Sandor Kertesz has proven the danger of this solution for maintenance, and the mandatory rule now is to enter the screening *only with observations that are surely located in the C+I zone.* Thus, the code in OBATABS should be useless, but was kept to indicate which values are outside (grep for GREPGEOM in the listing of screening, since CY29).

- ECSET : for similar reasons, specific Aladin switches used to be present in ECSET, where the sets of observations for the vertical interpolations are prepared. This code should now also be obsolete, but a rather simple test on LELAM is kept for safety.
- MKGLOBSTAB : this routine sets up tables for global message passing of observation equivalents. The gridpoints are scanned to identify the overlaps between model gridpoints and observation points (held by the same processor), and this scan uses either Aladin or Arpège (lat,lon) positions. Note that some extra unique sequence number is set, to identify the obs positions in a unique way for reproducibility in the adjoint computations (call to KEYSORT).

The general calling tree for the setup of observations is shown in Figure 4.1. Some interesting features :

- DEFRUN : a lot of initialization to control the levels of rejection during the screening : first guess check bounds, VarQC and gross error setup, thinning distances. This routine also reads in the namelists NAMSCC, NAMOBS and NAMJO, where most of the observational control is specified.
- SUVNMB : sets the database variable code numbers, which are the numbers used later in the obs operator HOP for variable type selection and filtering of specific observations (H2m, V10m, specific pressure levels etc...). These numbers are also written in the ODB base (MDBVNM).
- SURAD : a lot of parameter setup for the radiances (coefficients for RT code - RT stands for radiative transfer -, standard deviations and correlations for radiance errors) and the computation of the cost function in radiance space.
- SULIMB : setup parameters for the call to the direct and adjoint of the clear limb radiance RT model.

Some other features of interest for the debugger are the data distribution and the message passing, especially for the link between model gridpoints (the world where first guess values are distributed, and most of the computations take place) and observation points (storage of YGOM* arrays and DM-local obs treatments). The reader might find some graphical descriptions in the Appendix D on this topic.

### 4.1.2 Computing the obs-guess departures

A quite intensive description of the obs operator is done in the section about the TL obs operators (see §4.2.6). Thus, we will be rather short in this part. Please keep in mind one very important idea : there is in the code a very strict separation of the horizontal and vertical interpolations. This split allows to concentrate the geometrical specificities to the horizontal operators (especially, the Aladin domain aspects), while retaining *all* the Arpège code in the vertical part, since the vertical grids and the physical observational operators are identical for Arpège and Aladin. The main algorithm is depicted in Figures 4.2, 4.3, 4.4, 4.5 and 4.6. We recall here the main items :

- at control level (Figure 4.2), screening first computes the departures $y - \mathcal{H}(x^b)$ by a call to CNT2. Then the SCREEN monitor is activated. The final information is written as print-out ($J_o$ cost function) and to the ODB base (flags).
- the first guess is read from historical file by SUSPEC, under CSTA in Arpège and under ELSAC in Aladin. Then, the direct obs operators are activated by calling CNT4 (Figure 4.2). In 3dvar and in 3d screening, NSTOP=0 and STEPO is only called once. In 4d screening, the time loop is active and the horizontal obs operators are called at the middle iteration of each timeslot (already tested in Aladin by Vincent Guidard).
- the horizontal operators are called via STEPO/SCAN2MDM (Figure 4.3). The first guess is transformed from spectral to gridpoint space (ETRANSINVH), then COBS and COBSLAG perform the interpolations (COBSLAG : see Figure 4.15). Note that

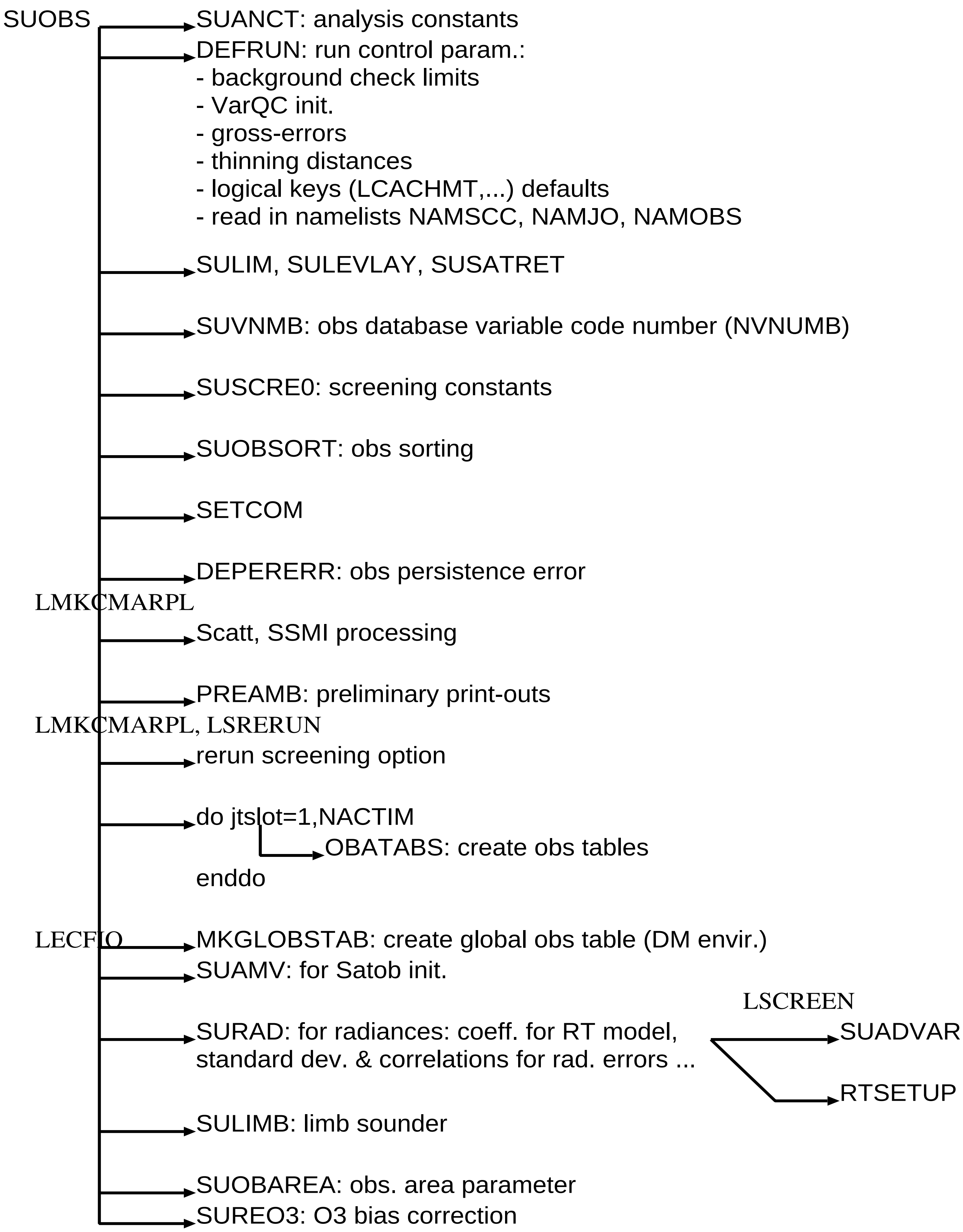



Fig. 4.1 – Calling sequence for SUOBS.

in Aladin, the interpolation weights are prepared in plane horizontal geometry (ELASCAW, as for semi-Lagrangian interpolations) instead of spherical (Arpège : LASCAW). Furthermore, bi-linear 4-point interpolations are preferred (NOBSHOR=201). bi-cubic 12-point interpolations could run, however, but are not regularly tested nor evaluated (NOBSHOR=203, as in Arpège).
– for non-uniform geometries (Arpège with stretching $c \neq 1$ or Aladin plane geometry), the wind is converted from grid North/South to geographical North/South inline in COBSLAG.
– after the horizontal interpolations, plus time evolution and timeslots in 4d screening, all the arrays are prepared for the final vertical interpolations, which are done in one go in a loop vectorized over the obs sets (Figure 4.4). The sets of observations are defined type by type, with a default length of 511 (ECSET).
– The vertical operators include all the physical operations, such as the radiative transfer (RT) equation. The monitor is HOP (see Figures 4.5 and 4.6), which is basically divided into preparation of interpolations (routines PREINT*), some specific obs setting and search in database, actual obs operators (routines PPOBSA*).
– at the end of HOP, the departures are stored in the database (words MDBOMN, MDBIFC1(NUPTRA+1), MDBOMF) by HDEPART.
– the cost function $J_o$ is computed in HJO. A diagnostic table JOT(.)%() is also filled, for final print-outs stratified by obs type, subtype, field.

### 4.1.3 Obs/physics wedding for "wet radiances"

In the case where the observation operator needs a lot of information coming from the physics parametrizations, or a number of prognostic hydrometeor fields, it is convenient to implement the call inside the main model physics. As a result, the observation operator is not called in observation space anymore, as described in the previous section, but from the model space. To achieve this, D. Salmond (ECMWF) has organized a succession of setup and conversion routines which bring dimensions and pointers of observations from observation space to model space, via SL buffers (including the halo needed for horizontal interpolations in DM mode). The general structure of this code is shown in Figure 4.7 and more details can be found in Deborah's excellent ECMWF technical note from April 13th, 2004. Note that, as this code is prepared for 1dvar retrievals, the direct sequence is followed by the adjoint one, with reverse operations. In routine GP_MODEL, John Hague has added overhead code after the calls to GP_PTR_SSMI and GP_SSMI in order to ensure load balancing for message passing MPI.

### 4.1.4 Screening, thinning and quality control

From the point of view of the code, nobody so far was brave enough to modify anything for LAM applications. Thus, whatever specific you would do in the Aladin screening, would actually be done through some namelist settings. We can separate the screening part into three steps (see Figure 4.8) :
– initial phase : setup, deallocate/allocate, convert flags from Météo-France to ECMWF format. Note that the first guess errors are read from file "errgrib" (INIFGER called from SU0YOMB) and then interpolated to the actual obs locations for the observed variables (SUFGER).
– decision taking : routine DECIS performs all the screening decisions. Note that independent decisions are taken first (those that concern each individual datum or report), followed by dependent decisions (with cross-comparisons between data and reports) :
  – check for completeness of the reports
  – evaluate reporting practices for Synop and Temp mass observations (surface pressure and geopotential)
  – check for blacklisted reports/observations. A monitoring blacklist is applied for discarding the excessively noisy or biased observations. A further data selection is done using the blacklisted data selection information and some hardcoded rules (orographic rejection limits, land/sea rejection).

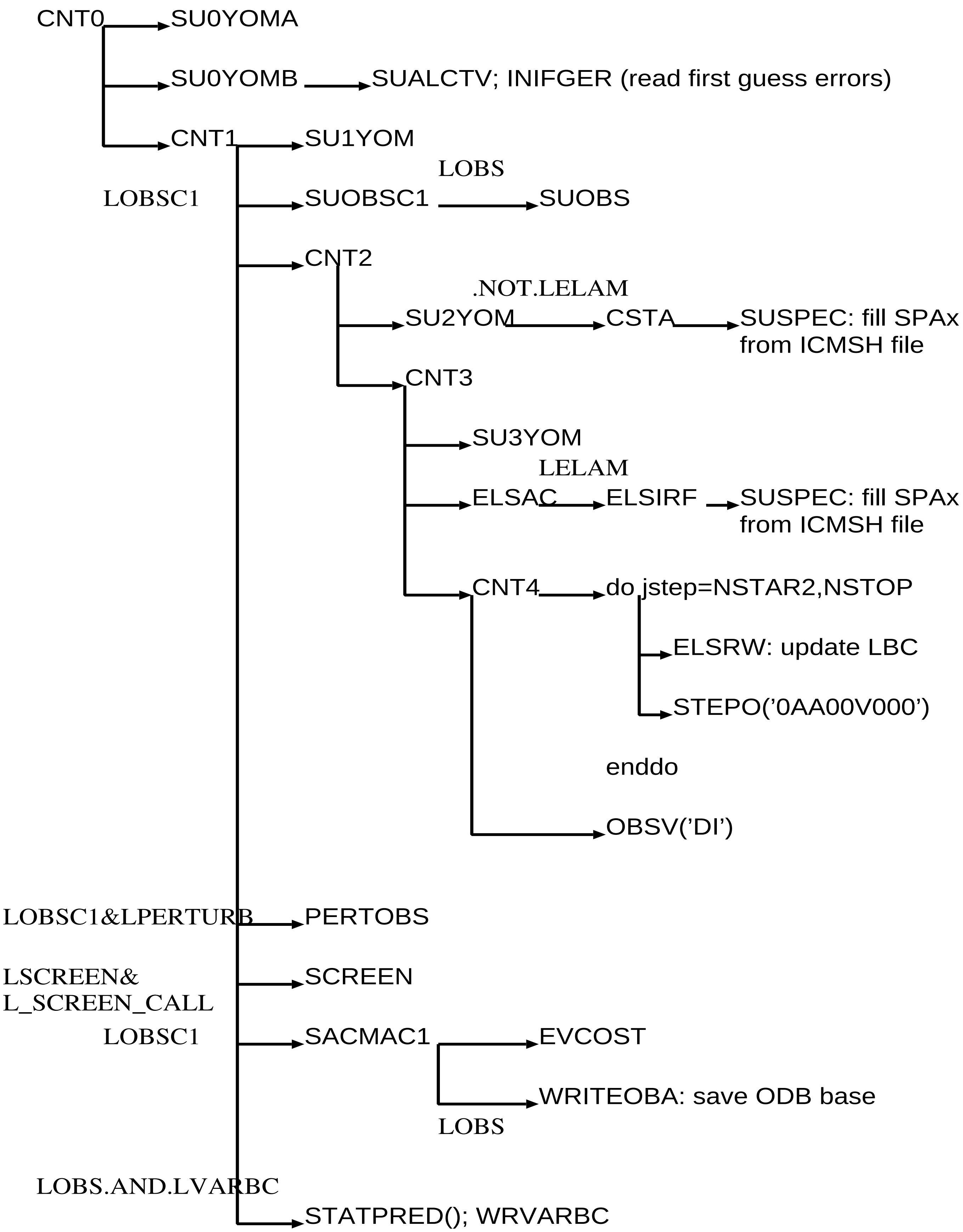



Fig. 4.2 – Calling sequence from CNT1 down to STEPO.

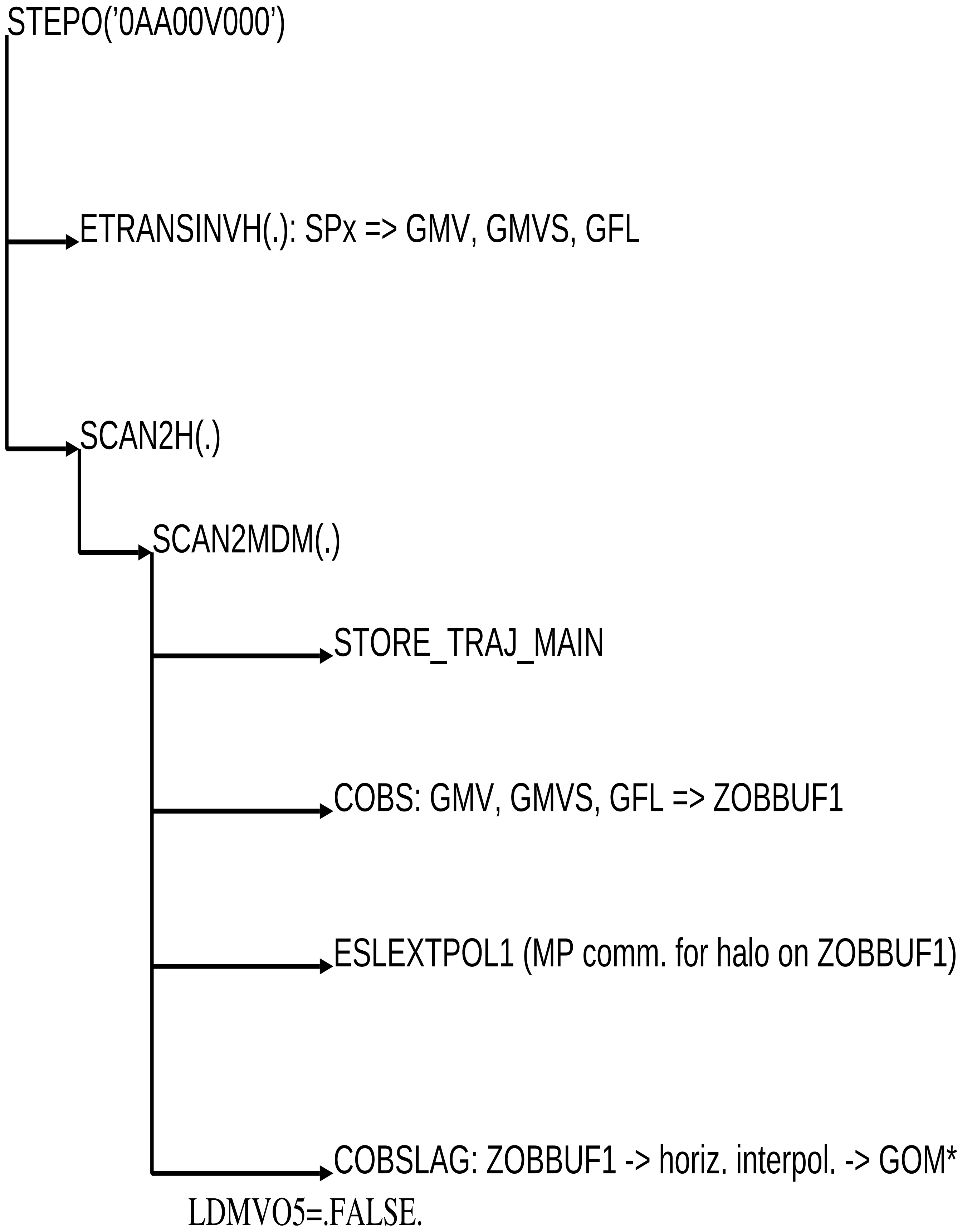



Fig. 4.3 – Calling sequence from STEPO down to COBS/COBSLAG. The calling tree for COBSLAG can be found in Figure 4.15.

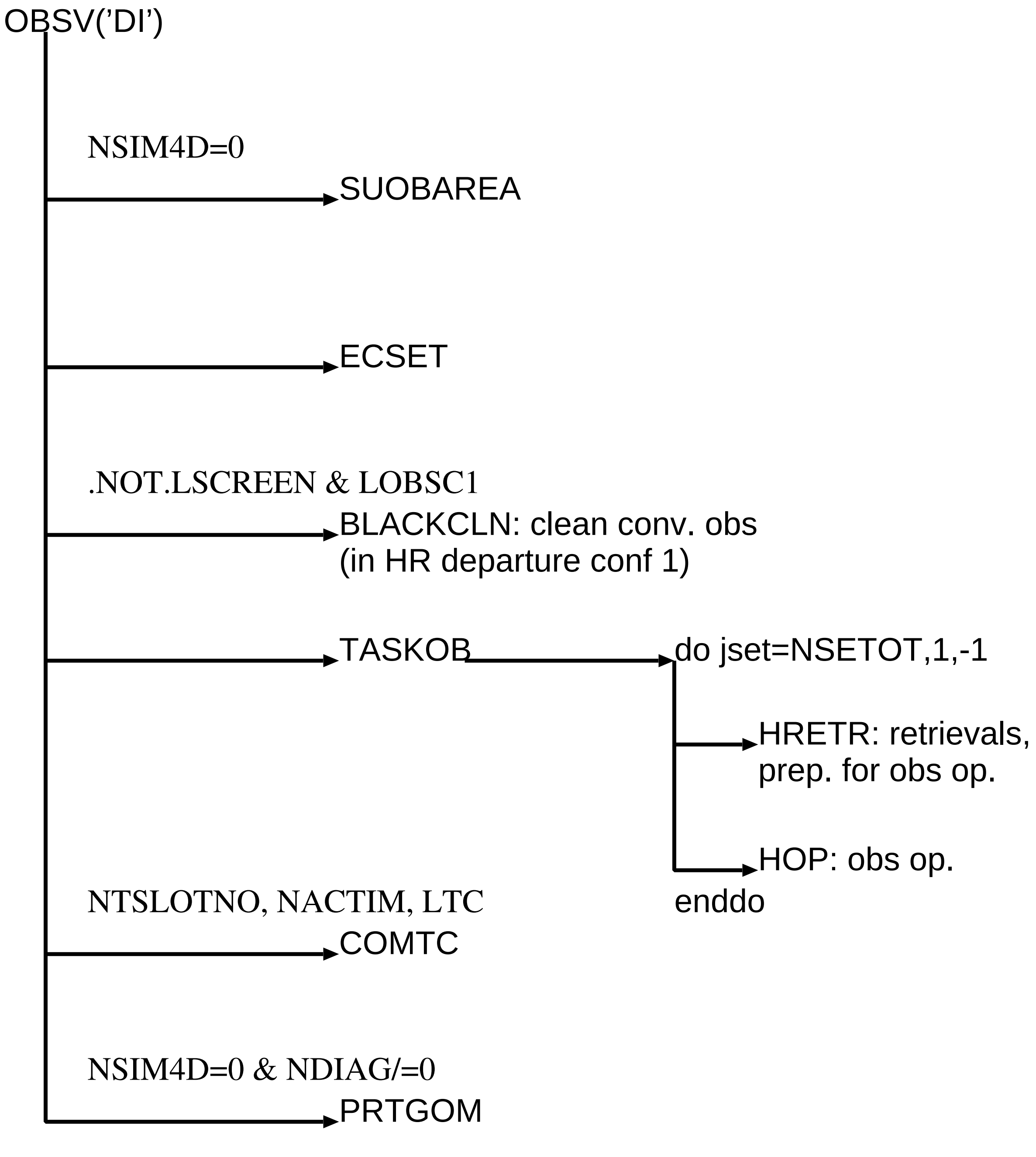



Fig. 4.4 – Calling sequence from OBSV down to the obs operator interface HOP.

HOP → HVNMTLT (link ODB|NVAR|PPOBSA var. code)

→ PREINT: interpol. of upper-air traj.; reads from GOM*

LL_SFC_OP

→ PREINTS: interpol. of surface traj.; reads from GOS*

LL_RAD_OP

→ PREINTR: interpol. of radiances (traj); reads from GSS*

LL_RAD_OP

→ QNEGLIM: make q positive

→ Search in database for all possible obs. types & nullify invalid requests (switch off a few things ...)

LL_SFC_OP

→ HINTH: surf. wind data

→ HQSCATT: treat ambiguous wind for Scatt.

→ ZHOFX=RMDI; IVAR=NMDI: preset to missing values

Fig. 4.5 – Calling sequence for HOP (first part).

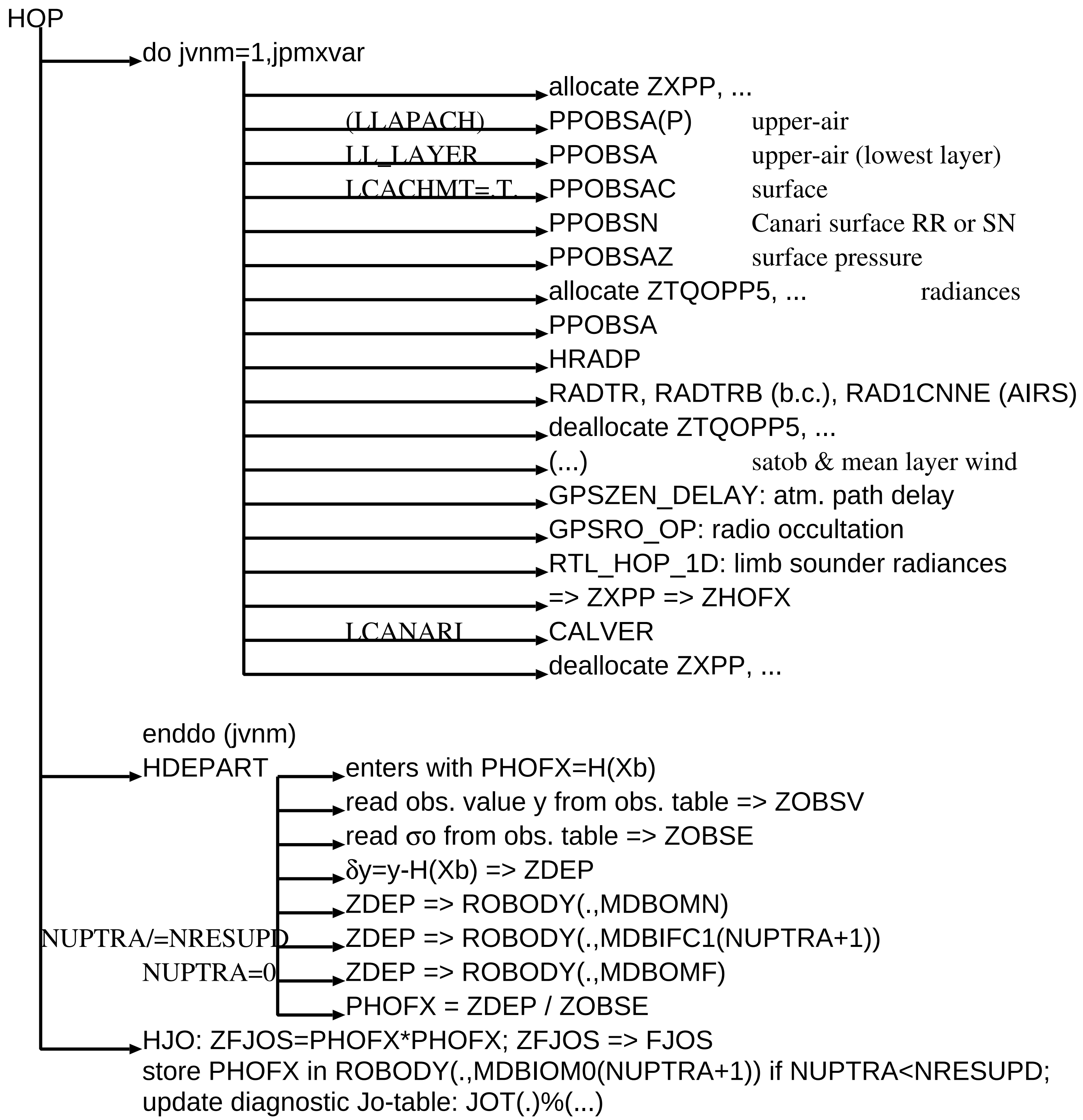



Fig. 4.6 – Calling sequence for HOP (second part).

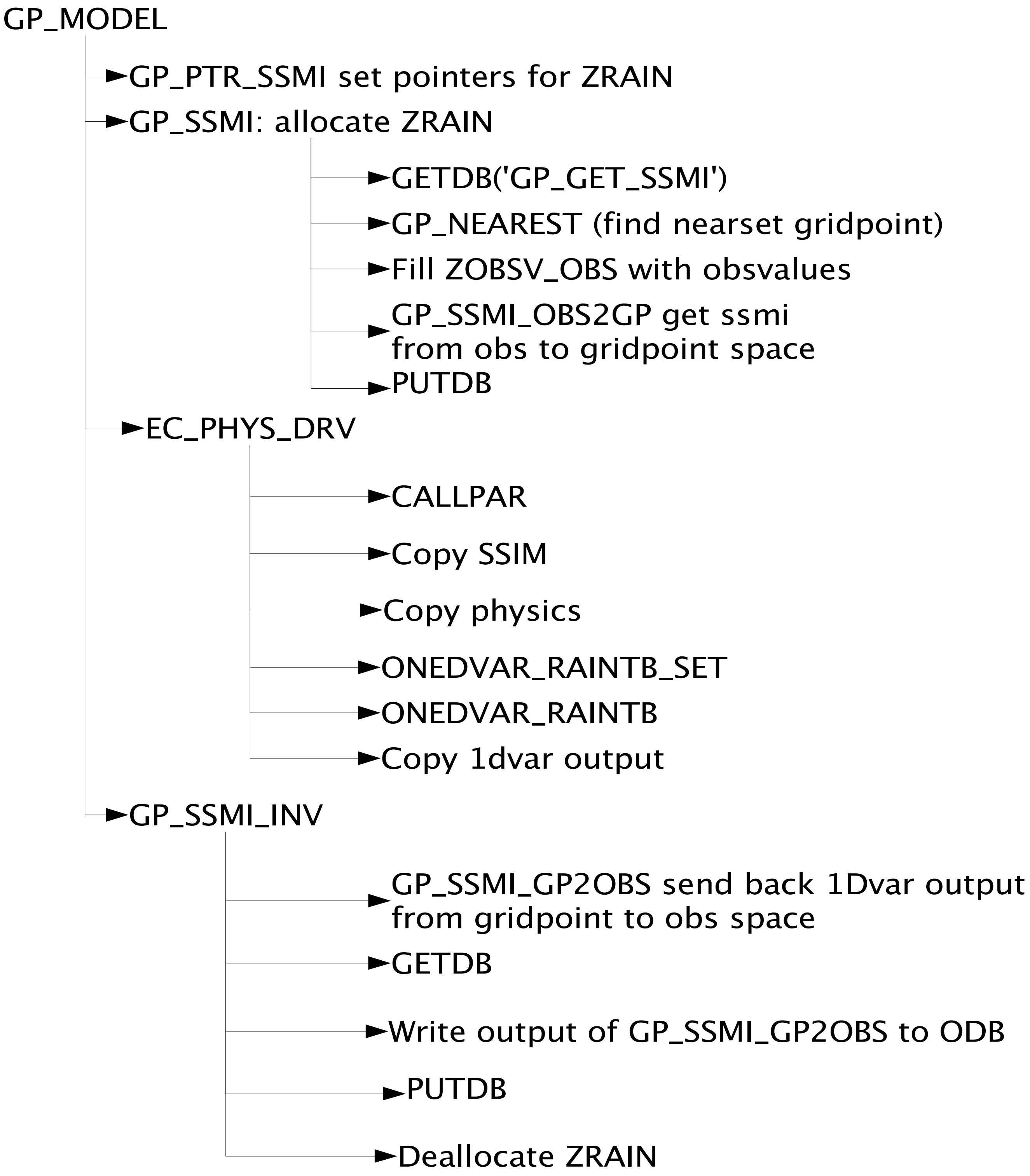



Fig. 4.7 – Calling sequence for GP_SSMI.

- background quality control. A variance for any background departure is estimated :

$$< y - \mathcal{H}(x^b), y - \mathcal{H}(x^b) > \approx \sigma_b^2 + \sigma_o^2$$

The estimate of the variance for the normalized departure is given by :

$$1 + \frac{\sigma_o^2}{\sigma_b^2} = \lambda$$

If :

$$(\frac{y - \mathcal{H}(x^b)}{\sigma_b})^2 > n \times \lambda$$

then $y$ is suspicious.
- vertical consistency of profiles. For multi-level reports, all duplicated levels are removed. If four successive layers are found to be suspicious, then these layers are rejected.
- miscellaneous
- update flags for report/datum status
- removal of duplicated reports. In the case of co-located reports of the same observation type, some or all of the content of one of the reports is rejected. For example, two Airep with the same location, the same content, but slightly different station identifier.
- check for redundant data. The redundancy check applies for all active reports which are co-located and originate from the same station. Thus :
  - For land Synop, the report closest to the centre of the screening time window and with the biggest amount of active data is retained
  - Ship Synop are considered as redundant if the moving platforms are within a circle of one degree radius
  - Temp and Pilot originating from the same station are considered at the same time in the redundancy check
  - A Synop mass observation is redundant if a Temp report is available for the same time and station, and if there are any geopotential height data that are not more than 50 hPa above the Synop mass datum, from that Temp.
- specific scatterometer quality control (QC)
- thinning of data : 3d boxes, defined by the RFIND*/RMIND* parameters in namelist NAMSCC, are defined and only one measurement per obs variable is retained in each box. A horizontal thinning is performed for Airep, Satob and TOVS (radiance) data. A vertical thinning is applied to Airep and Satob data.

- final phase : the QC flags (status) are stored in the obs database (UPECMA), and final status print-outs are performed (PRSTA)

## 4.2 3dvar minimisation (focussing on Aladin)

The general algorithm for 3dvar is common to Arpège and Aladin. Therefore, both models are treated here, though some more stress on the LAM aspects is given. This section does not include features like the propagation models (4dvar) or multi-incremental items (Arpège-specific).

The reader must be aware of one fundamental technical problem in the ALADIN 3dvar : since the variational algorithm basically works in spectral space, and since the ALADIN software carries the (2*NFLEVG) vertical profiles of mean wind as extra spectral variables in the model, many extra switches (and probably just as many potential sources of bugs) exist in the code. They generally concern extra allocations, settings, read/write operations, performed on the SPA1 array (or equivalently, SPA1JB, SP5A1 and SP7A1 which are defined later in the text). For more information about the spectral space vector in Aladin, the reader is referred to Joly (1992) or Fischer (1998).

Furthermore, it is interesting to become familiar with the main working spaces of configuration 131 :

- the model state space ($x$, $\delta x$, $\nabla_{\delta x} J$) which is carried around in the code via the SPAx arrays (SPA3, SPA2, SPA1)

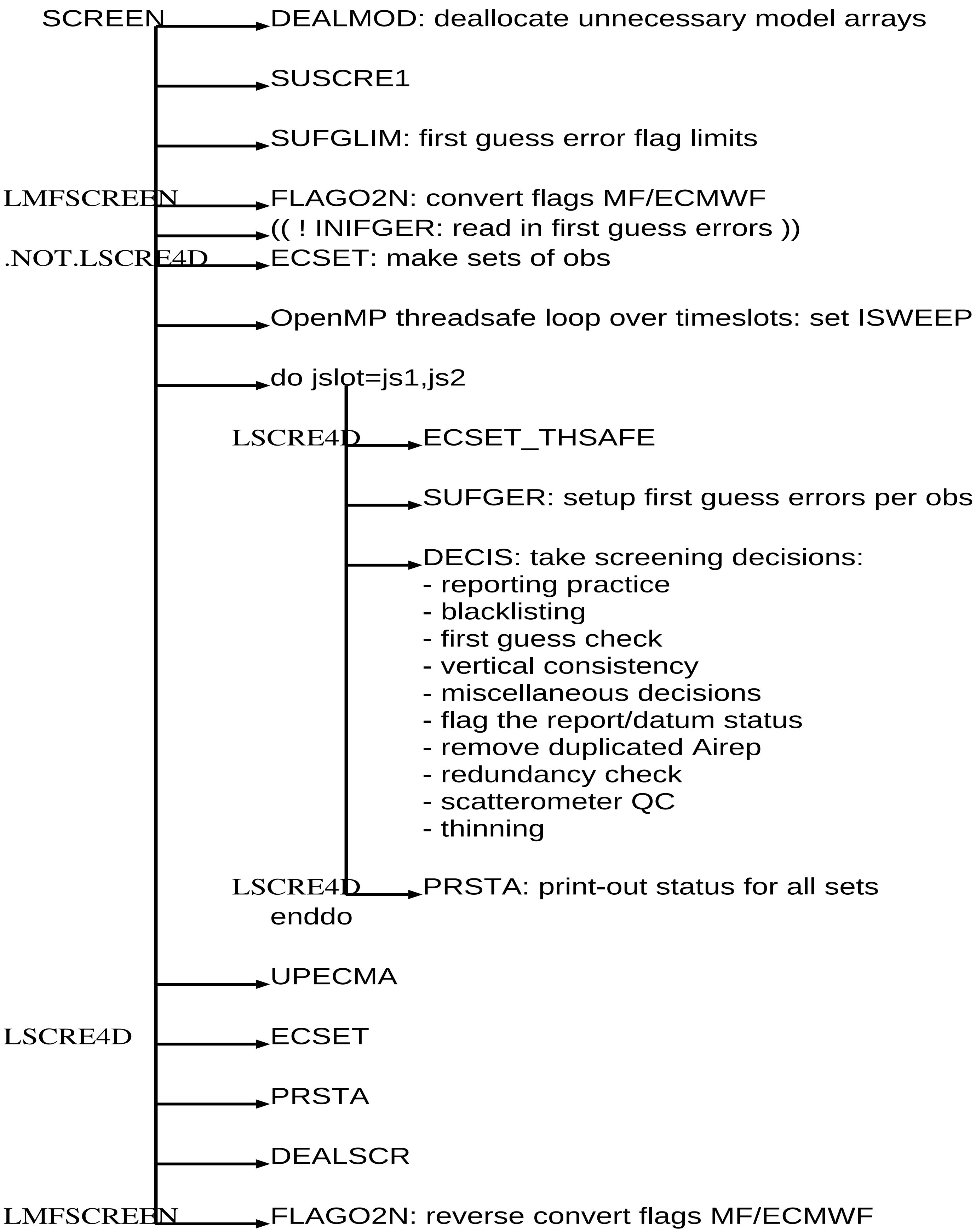



Fig. 4.8 – Calling sequence for SCREEN.

– the control variable space ($\chi$) which is carried around via the VAZX (or PVAZX) array. Note that the adjoint of the control, which corresponds to the normalized gradient of $J$, $\nabla_\chi J$, is carried around in VAZG (or PVAZG).
– the $J_b$ related spectral buffers, which map the control variable but have a model space layout : SPA3JB, SPA2JB, SPA1JB

A common rule in the code is to distinguish the perturbations in the TL/AD code (which are carried in the usual model arrays and buffers) from the trajectory data which are carried in the "twin“ arrays suffixed by '5', eg. SPA3 and SP5A3, YGOMUA and YGOMUA5, ZTS and ZTS5 and so on.

### 4.2.1 The different parameters within the control variable

An important effort has been made in order to make the control variable flexible so that many extra parameters may be added (e.g one wishes to control chemical species concentrations...) The number of parameter with a representation in SPA3JB/SPA2JB is N_SPJB_VARS (by default N_SPJB_VARS=5 for vorticity, divergence, temperature, log(Ps), and specific humidity) Each of these parameters correspnds to an entry of the SPJB_VARS_INFO array. which is an array of structure including, among others, the type of parameter (provided by a gribcode) or the localisation of the parameter related term within SPA3JB and SPA2JB.

### 4.2.2 Setup of $B$

The setup of $B$ is displayed in Figure 4.9. The initialization takes place under SU0YOMB. Note some Aladin particularities with respect to Arpège :

– the spectral diagonal blocks are valid for each total wavenumber because of the assumption of horizontally homogeneous covariances (see §3.2). However, this total wavenumber does not have the same representation as in spherical spectral space. In Arpège, the total wavenumber is readily obtained as it is $n$, from 0 to NSMAX. The zonal Fourier wavenumber $m$ follows from triangular truncation (from 0 to $n$). In the Aladin bi-Fourier representation, the total wavenumber $k^*$ defines an ellipse :

$$(k^*)^2 = NSMAX^2 \times (\frac{n^2}{NSMAX^2} + \frac{m^2}{NMSMAX^2})$$

with $n$ and $m$ the meridional and zonal wavenumbers, NSMAX and NMSMAX the meridional and zonal truncations, resp. (see also Fischer, 1998). Note the particular choice to define *hardcoded* and by convention, the total wavenumber from 0 to NSMAX. This allows to keep many do-loops and the arrays for cost functions similar to Arpège.
– in the same spirit, the systematic usage of the functional `fkstar.h` to call up for the total wavenumber corresponding to a given pair of $(m, n)$. In Arpège, the $n$ wavenumber is directly the total wavenumber, which simplifies the code somehow!
– specific humidity is included in the balance relationships, while it is not (thus, univariate) in Arpège.
– $\beta$-plane : while the original Berre formulation (Berre, 2000) considered geostrophy on a constant $f$-plane, there is now a possibility to develop the Coriolis parameter in Fourier space and compute the balance

$$\phi = \Delta^{-1}(f\zeta)$$

as a truncated convolution product in spectral space :

$$\hat{\phi}(m, n) = \Delta(m, n)^{-1} \sum_{p=-P}^{+P} \hat{f}(0, p)\hat{\zeta}(m, n - p)$$

The truncation parameter $P$ is NPSMAX in namelist NAMJG. Typically, 15 waves out of 149 are enough to recover a very acceptable Coriolis function. Note also that the expansion uses a conversion from the usual Aladin quadruplet $(\phi_r^r, \phi_r^i, \phi_i^r, \phi_i^i)$ defined over a quarter of an ellipse, into the Hirlam set $(\phi_R, \phi_I)$ defined for half an ellipse, for the sake of simplicity (see SUEJBBALBETA and EBALBETA).

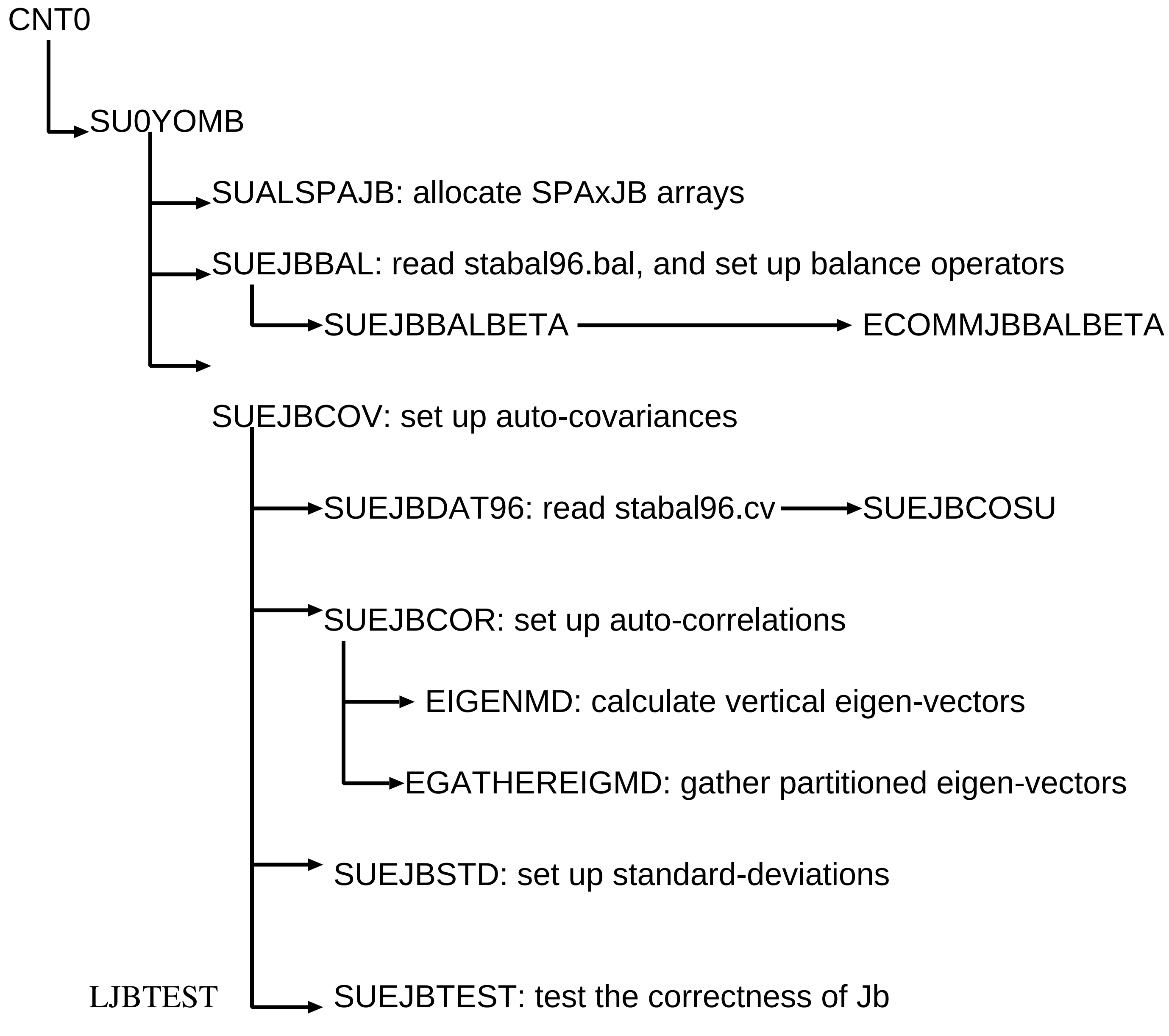



Fig. 4.9 – Calling sequence for SUEJBBAL/SUEJBCOV.

### 4.2.3 Control level

CVA1 controls the variational configuration of ARPEGE/ALADIN. Its flow diagram is given in Figure 4.10. Here are the major remarks :

- the first guess is read from input file in SUSPEC, SUGRIDU, SUGRIDF called by SUECGES. SUSPEC deals with spectral fields, SUGRIDU with upper aire gridpoint fields and SUGRIDF with the rest of the gridpoint fields. Spectral field are stored once for good in the SP7Ax arrays and copied into SPAx under SUVAZX (calling SP7TO3). In SUVAZX, the control variable vector VAZX is also set to zero at the first minimization (first outer loop of multi-incremental 4dvar). The background field is stored incore, during the trajectory computation (the call to CNT2). This storage is done by calling STORE_TRAJ_MAIN in SCAN2MDM. In 3dvar, only one storage is done in SCAN2MDM (because we only enter once the temporal loop in CNT4), and one expects to save there exactly the same fields than those stored in SP7Ax. Note that the background trajectory will be read later on, in SUBFGS (called by SIM4D), and then stored in the SP5Ax arrays.
- the observations are read in SUOBS, with additional observational setup.
- the SP5Ax arrays are allocated in SUALLT, including the ALADIN specific SP5A1 for the mean wind. SUALLT also allocates the incore trajectory, handled by the TRAJECTORY related modules. The old routine SUELLT which was ALADIN-specific has disappeared since AL27.
- the trajectory around which the linearization will be considered is computed through the call to CNT2. In spatially incremental variational analysis, one would compute here the low resolution trajectory. In ALADIN 3dvar, however, one simply recomputes the high resolution departures. In 3dfgat, one would compute in CNT2 the time evolved departures along the forecast trajectory.
- below CVA2, there are several calls to the variational simulator SIM4D. The first call is to setup the control and the gradient, then we iteratively call M1QN3[1] (Quasi-Newton minimization, see Gilbert and Lemaréchal, 1989). M1QN3 stops as soon as either the convergence criterion is fulfilled (norm of the gradient reduced by more than RCVGE, RCVGE being set in namelist NAMVAR) or the maximum number of iterations (inside M1QN3 : NITER) or the maximum number of simulations (calls to SIM4D : NSIMU) are reached. A final call to SIM4D is done in order to retrieve the final diagnostics.
- if asked (key LGRTEST in NAMVAR), the configuration 131 will run a set of iterations in order to check the quality of the gradient. This test is done by calling GRTEST. The test under consideration reads :

  $$T_1 = lim_{\delta\chi \to 0} \frac{J(\chi + \delta\chi) - J(\chi)}{< \nabla_\chi J, \delta\chi >}$$

  One expects that, for decreasing perturbation size $\delta\chi$, the $T_1$ diagnostic converges to 1. This is indeed true as long as the size of the perturbation does not become smaller than the computational zero of the computer. Shall this happen, will the test again diverge from 1. The test is otherwise similar to the one in configuration 801.

### 4.2.4 Simulator

A simulation consists of the computation of $J$ and $\nabla_\chi J$. This task is done by the routine SIM4D. Figure 4.11 shows the flow diagram of the simulator. Please note that we will concentrate on the case where the tangent linear observation operators are applied (key LOBSTL=.TRUE. in namelist NAMVRTL). In an alternative, but soon obsolete, version, one could run the finite difference approximation of $H$, which would result in calling CNT3 instead of CNT3TL. In this latter case, the cost function is not perfectly quadratic, since one considers $\mathcal{H}(x^b + \delta x) - \mathcal{H}(x^b)$ instead of $H\delta x$ in the $J_o$ term. For LOBSTL=.TRUE., the simulator works like this :

[1] Note that we do not detail here other possible minimisation tools. M1QN3 is so far the only minimizer which was tested in Aladin. Arpège/IFS does however provide other possibilities, based on conjugate gradient algorithms : N1CG1, CONGRAD (which provides an estimation of the leading directions of the Hessian by a Lanczos computation).

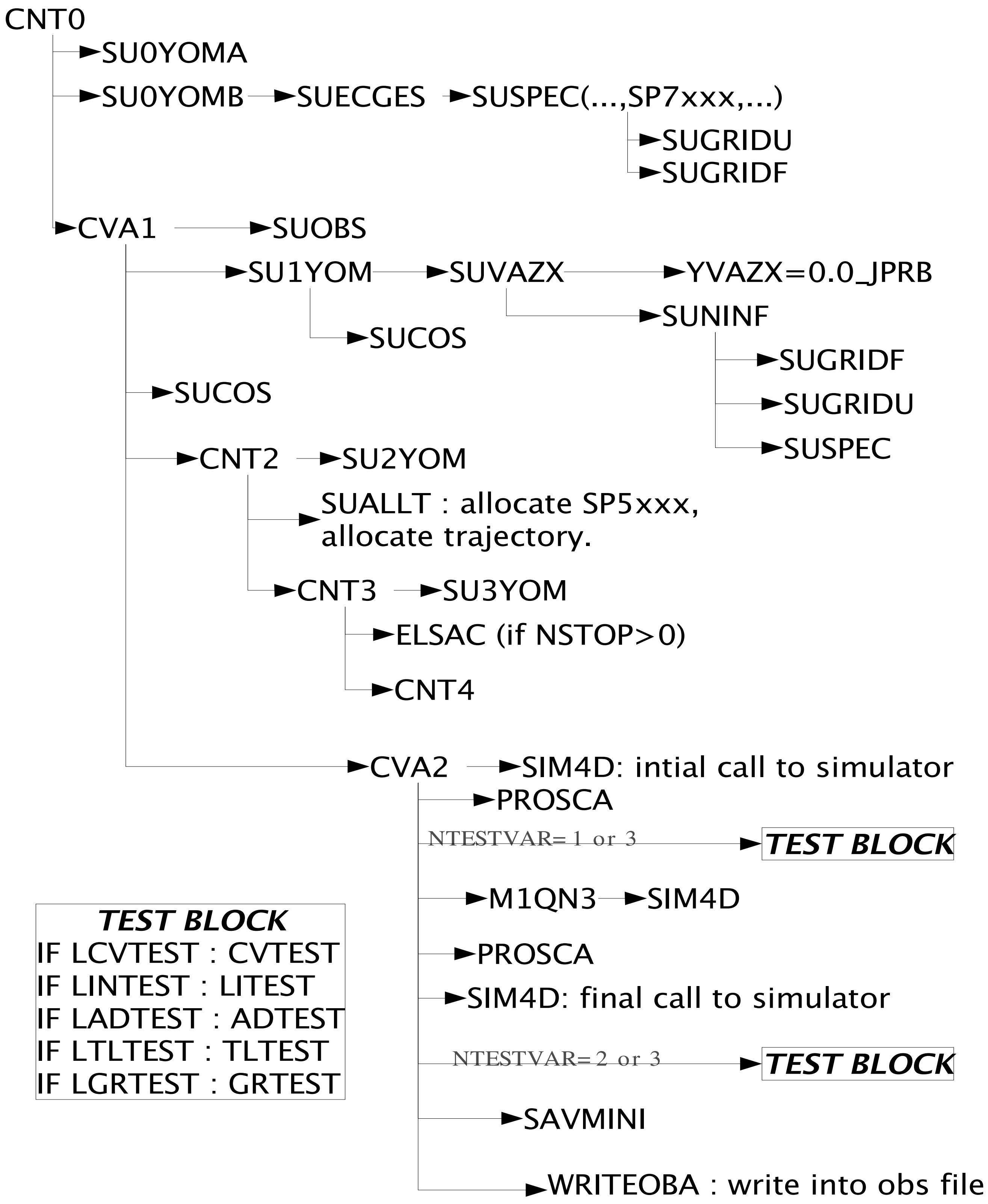



Fig. 4.10 – Calling sequence from CVA1 down to the simulator SIM4D.

– input data are the control vector coefficients from the previous inner loop (previous call to SIM4D and M1QN3). They are carried in PVAZX (VAZX at control level).
– some setup for level 2 control and cost function initialization is performed.
– $J_b$ and $\nabla_\chi J_b$ are computed in CTLVEC_SQNORM. These two terms are almost trivial in control space. However, please be aware about the coding tricks at stake !
  – Firstly, the ARPEGE control vector ignores those spectral coefficients which are always zero by definition in spherical harmonic expansion. These coefficients are the imaginary parts for zonal wavenumber $m = 0$ in triangular truncation. The coefficients however do exist in the model space, and therefore the ARPEGE computation uses a metric term to take into account the different weighting needed to handle a consistent $L^2$-norm. The metric term equals 1 or 2 in ARPEGE/IFS (see in the module procedures DOT_PRODUCT_CTLVEC(PACK_ALL_MS) and SPECTRAL_FIELDS_PARA(SPECNORM_ARRAY_LEV)). In ALADIN, every spectral coefficient is present in the control, though there exist again some imaginary parts which are always zero (those associated with mean values for $m = 0$ and $n = 0$). Thus, the metric term equals 1 everywhere. For more details, see Fischer (1998).
  – Secondly, the mean wind components are *not analyzed* in ALADIN (see Fischer, 1998, for an extensive discussion on the problematic of the mean wind coefficients in variational analysis). Thus, we actually run with NVA1D=0. Anyhow, the code in SIM4D and CTLVEC_SQNORM is uptodate to take into account the mean wind, especially to run the configuration 801 which also uses SIM4D. However, the cost functions do not contain the 2*NFLEVG values required by the mean wind. Note also that, by convention, it was decided to always treat the mean wind *after* the spectral fields (SPA2, SPA3) and the extra observational control vector (TOVSCVX and TSCV so far).
  – Thirdly, note that the $J_b$ is collected and broadcast to all processors in DOT_PRODUCT_CTLVEC. In EVCOST, every processor simply copies the $J_b$ value into the global cost function FJB.

  The gradient is finally stored in PVAZG
– convert from control space $\chi$ into model space $x : x = x^b + L^{-1}\chi$. This is done by CHAVARIN (see §4.2.5).
– for tangent linear observation operators (LOBSTL=.TRUE.), subtract the first guess again in SUBFGS (better keep LSUBBG=.TRUE. !). Note that the first guess actually is read from the incore trajectory by a call to GET_TRAJ_SPEC, using the trajectory module. Now you do have $\delta x$ in your model space vector SPAx. At the initial call to the simulator, you strongly expect these coefficients to be zero. Note that an exception was found in ARPEGE 4dvar, when the LTC key for time correlated observations was switched on (a case where apparently the background in SP7x and the first guess in SP5x did not coincide).
– call the tangent linear observation operators via CNT3TL (see §4.2.6). CNT4TL and below will be called as well, but in 3dvar the time iterations will be skipped (NSTOP=0), so only the observation operators are actually called below CNT3TL/CNT4TL. Furthermore, the cost function $J_o$ is ready at the end of CNT3TL, after the call to OBSVTL (master routine for $J_o$).
– remember : no $J_c$ in ALADIN 3dvar, so far !
– after CNT3TL, you have computed $H\delta x$. This is also luckily the input you need for the adjoint computation, to compute the gradient of $J_o$. Thus, you enter immediatly CNT3AD to call the adjoint observation operators (see §4.2.7). At the end of CNT3AD, $\nabla_{\delta x} J_o$ is present in the model arrays SPAx.
– convert $\nabla_{\delta x} J_o$, which you have in model space, back into control space by applying the adjoint change of variable : $\nabla_\chi J_o = (L^{-1})^t \nabla_{\delta x} J_o$. This is done in CHAVARINAD. Since you also used to call the canonical injection CAINAD inside CHAVARINAD, you also add here the already computed $\nabla_\chi J_b$ to the $\nabla_\chi J_o$. Thus, the output of CHAVARINAD will be the complete cost function gradient, stored in PVAZG, expressed and scaled in control space.
– the individual contributions to the cost function $J$ are gathered and added together in EVCOST. EVCOST treats $J_b$, $J_o$, $J_c$ (if present), $J_q$ (model error). $J_o$ is also stratified by observation type. EVCOST also handles the distributed memory aspect of every cost function *except* $J_b$ (already done in CTLVEC_SQNORM), and performs the requi-

red message passing. The cost function contributions are written to the output file by PRTJO.
– if asked, the control and the gradient can be written on file. Finally, the simulation counter is updated.

### 4.2.5 Change of variables and canonical injections

Small but important routines are the change of variable and the canonical injection routines : CHAVAR/IN/AD/INAD and CAIN/IN/AD/INAD. The basic task of moving the information between model space and control space is done by the canonical injections. The CAIN* routines identify those elements of the model vector that need to be transfered to the control (arrays NVA3D, NVA2D, NVA1D which act as on/off switches). CAIN/CAININ/CAINAD/CAININAD are called under the condition NCNTVAR=1, which means that we work with model variable inside control variables, whereas with NCNTVAR=2, we work with departures to background inside control variable. The adjoint routines furthermore scale the gradients with the control space inner product defined in SUESCAL/ESCALJGS (for NCNTVAR=2) and stored in (R)SCALP_DV (equals 1 in ALADIN 3dvar, which means that we work purely with the $L^2$ norm, and all spectral coefficients are retained in the control, even those that are zero for sure). The canonical injections also used to perform the distributed memory message passing between model space distributed spectral arrays (along $m$ wavenumbers) and the distributed control vector. fo SPJB array matter, and SPJB is initialized with the value of YDVAZX%initcv at the beginning of JGCORI (why not earlier ? this is weird). With Y. Trémolet's development of a DM mapping between the spectral buffer (SPJB) and the control vector (CONTROL_VECTOR type), most message passing has disappeared from the canonical injections. The control vector is held on each processor as a chunk which corresponds to the stream of those spectral coefficients held in state vector space, so that the CAIN* mainly perform a local rearrangement of coefficients. Note that the mean wind is known on every processor (as a result of the TAL transforms). The tricky part for DM message passing is for computing the (global) scalar products and the $J_b$ norm.

The change of variable routines include all the necessary steps to normalize the background departures by the $B$ matrix norm. These steps are very similar for Aladin and the old Arpège formulation based on Derber-Bouttier (1999). In the case of $\delta x = L^{-1}\chi$ illustrated in Figure 4.12, the following sequence holds :
– copy control vector data from YDVAZX into SPJB array (canonical injection).
– set mean wind elements to zero for security (these elements are not in the control)
– (neutral) transpositions from zonal number to vertical level stratified data distribution when NPROC>1 (TRMTOS)
– block-diagonal matrix multiplications by the vertical error covariances for each predictor (JGVCORI). The matrices were read from file "stabal96.cv" and diagonalized in the setup (SUEJBCOV and below)
– transpositions from vertical level to zonal number stratified data distribution when NPROC>1 and NPROCB>1 (TRSTOM). This B-level feature is neutral if NPROCB=1 in Arpège and Aladin. It was so far never tested in Aladin anyhow.
– multiplication by the horizontal auto-correlation functions for every predictor (EJGHCORI)
– multiplication by the horizontal uniform spectral standard deviations of the background (JGNRSI). There exists a possibility to perform gridpoint calculations for the standard deviations by calling STEPO. For example, ARPEGE uses a gridpoint mask for geographical dependency. In ALADIN, one can apply a rim-zone mask around the inner border of the C+I zone, to force the standard deviations to a constant, presumably small, value (LEVARGP=.TRUE. in namelist NEMVAR).
– observational elements are multiplied by pre-defined standard deviations (case of the extra TOVS brightness temperature coefficients).
– apply the horizontal balance operator between balanced mass and vorticity (takes into account geostrophy, basically) either on $f$-plane or on $\beta$-plane.
– apply the cross-correlation regression coefficients (read from file "stabal96.bal" in the setup). This operation allows to recombine all the predictors, step by step, to retrieve the full model state.

SIM4D → SU2YOM

→ SUCOS

→ CTLVEC_SQNORM: computation of Jb and Grad(Jb) using the control variable PVAZX (almost trivial ...)

LJxZERO → FJxCOST=0.0_JPRB; YDVAZG%x=0.0_JPRB

YDVAZX%nparam > 0

→ FJVARBC: compute AFJPCOST for var. b. c.

→ CHAVARIN: current control var. PVAZX transfered into SPA. (this is the current δx at the beginning of the inner loop !)

→ SUBFGS: xb + δ x -> δ x → GET_TRAJ_SPEC
SPA. = SPA. - SP5A.

L131TL
LTLINT → CNT3TL: H . δ x

→ CNT3AD: g = Grad(Jo) = Ht . R-1 . (d-H.δx)

→ CHAVARINAD: g -> (L-1)t . g
gradient transfered from SPA. into PVAZG

→ EVCOST: evaluate and print the cost functions Jb, Jo, Jc, Jq

LGRASCAL → SCAAS: evaluate and print the gradients (w/r to SCALP inner product)

→ CAIN(PVAZG)
STEPO('A00000000'): write gradient into FA file

→ NSIM4D = NSIM4D + 1

NPRTMODERR>=2

→ CTLVEC_SQNORM

Fig. 4.11 – Calling sequence for SIM4D.

– copy SPJB array into SPAx arays (jbtomodel)
– if LSUBBG=.TRUE. (default in SUJB), add the background field to SPAx arrays.

As an example, the change of variable $\delta x = L^{-1}\chi$ is done in CHAVARIN ; its adjoint counterpart $\nabla_\chi J = (L^{-1})^t \nabla_{\delta x} J$ is done in CHAVARINAD. Please check in the Appendix D for some graphical illustration of the data distribution and message passing for the control vector.

### 4.2.6 TL observation operators

In this section, we review the tangent linear observation operators in 3dvar. The calls to the model propagators (basically, the time evolution of increments, through CPGTL/CPGLAGTL) are not treated here, since they are a 4dvar feature. As seen in the simulator, the computation of the gradient requires successive calls to the tangent linear and to the adjoint observation operators. The tangent linear operators are called via CNT3TL, the usual model interface, and only the 9-character sequence CDCONF later on controls the operations (in CNT4TL). The general principle at the level of CNT3TL/CNT4TL, as shown in Figure 4.13, is to read the trajectory spectral fields, to convert them into gridpoint space (GMV5, GMV5S, GFL5), and then to apply the TL operators by calling STEPOTL. Unlike CNT4AD, every call to STEPOTL is included in the general time iterative loop. Note also that, once the increments have been computed, there is no need to transform them back into spectral space in 3dvar. The increments are simply kept in the "GOM-arrays" at observation location.

"GOM"-arrays contain the model values at observation points, but on model vertical levels. There are six main categories of arrays (actually, pointers) :

– YGOMUA for any sort of upper-air data, containing vertical profiles of U, V, T, q, O3, CLW (cloud liquid water), CLI (cloud ice), CC (cloud cover). YGOMUA points towards the GOMDATUA fortran array. If the model trajectory is treated, then YGOMUA5, which points towards GOMDATUA5, is used (for more details, we refer to the module procedure GOMS and to the setup routine SUGOMS). Upper-air GOMs contain both conventional and satellite positional profiles.
– YGOMS for surface data of any type, containing surface pressure, skin temperature, soil water content, snow cover, roughness length etc ...
– YGOMUA_2D for slanted profiles of upper-air data. Here, one considers that the positions of the observations are along a slanted, 2D, path.
– YGOMS_2D for surface data along a path (which can be the projection of a slanted path in the atmosphere).
– YGOMEC for surface constant fields at observation positions (orography, albedo, emissivity etc ...).
– YGOMGFL for any new GFL variable to be projected onto observation space.

Some of these arrays are user-controlled (namelist NAMDIM). The switches LGOMx and LGSMx can be used to switched on the allocations of the corresponding fields. For LOBSTL=.TRUE., the YGOM* arrays are duplicated into YGOM*5, which then contain the trajectory fields, while the YGOM* arrays will contain the perturbations.

An interesting feature is the strong architectural choice to separate the horizontal and vertical interpolations : the horizontal interpolations (from model grid to observation position) are performed under STEPOTL, while the vertical interpolations, which include model levels to pressure levels, but also most of the physical transformations required (such as to evaluate a radiative transfer equation), are called under OBSVTL. There is an exception to this rule, for D. Salmond's development for reverting the operations for rain and cloud-affected satellite data : the vertical interpolations then take place first, on the model grid points, to allow an easier usage of the vertical model physics (this is the so-called "obs/physics marriage", see §4.1.3 and the ECMWF technical note by D. Salmond).

Finally, for memory, we recall the specific Aladin features : ELSIN0TA in CNT4TL zeroes all the increments in the coupling zone (useful actually only when the propagating models are called, in configurations e501, e401, e601, e801) ; STORE_TRAJ_MAIN also handles the mean wind trajectory (merged with Y. Trémolet's trajectory module since CY25T2).

Let us review the main steps of the algorithm :

– When STEPOTL is called (Figure 4.14), we first transform the increments (from the previous inner loop) back to gridpoint space (GMV, GMVS, GFL). Then the observation operators are called from SCAN2MTL. The calls are repeated in the first iteration of

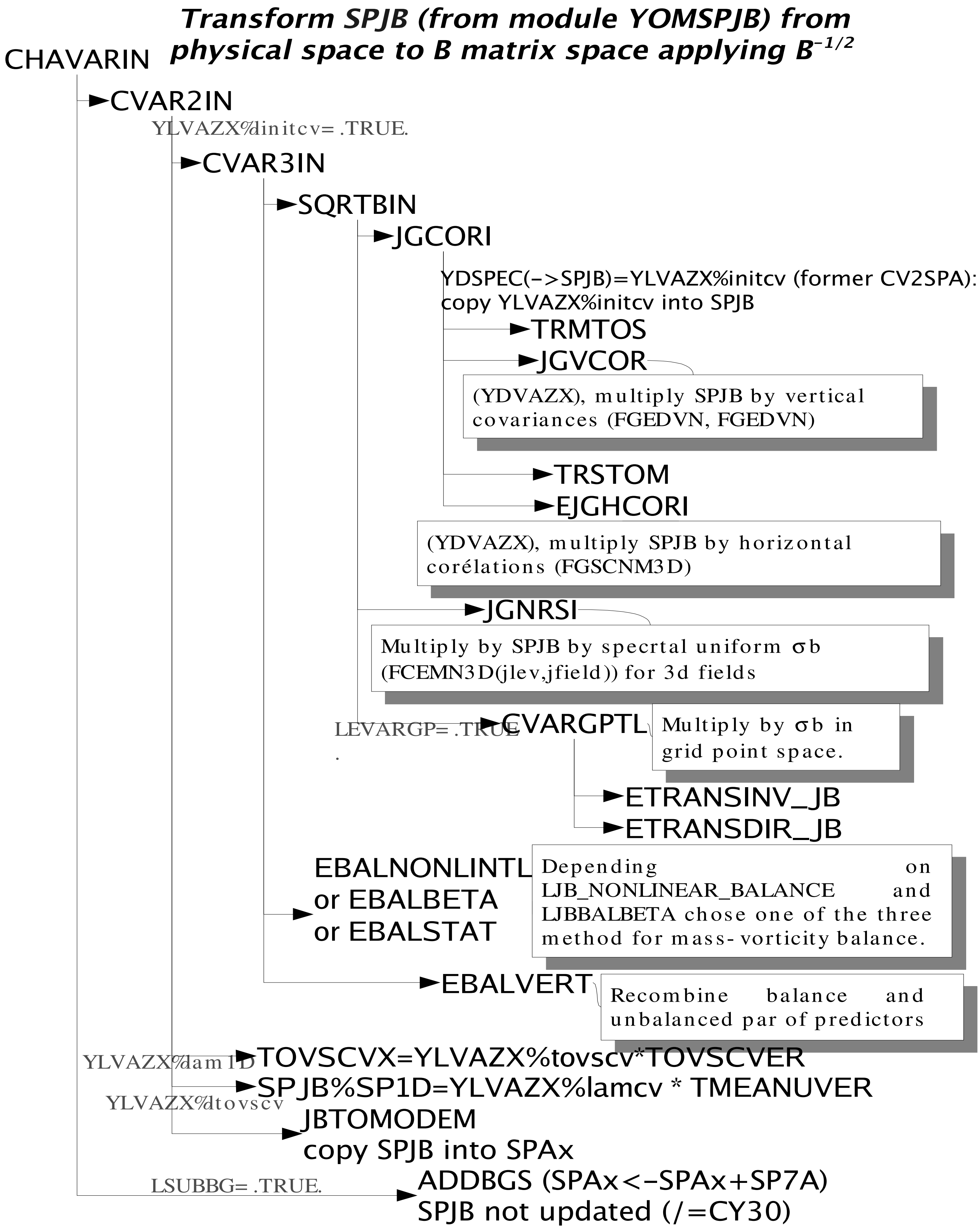



FIG. 4.12 – Calling sequence for CHAVARIN.

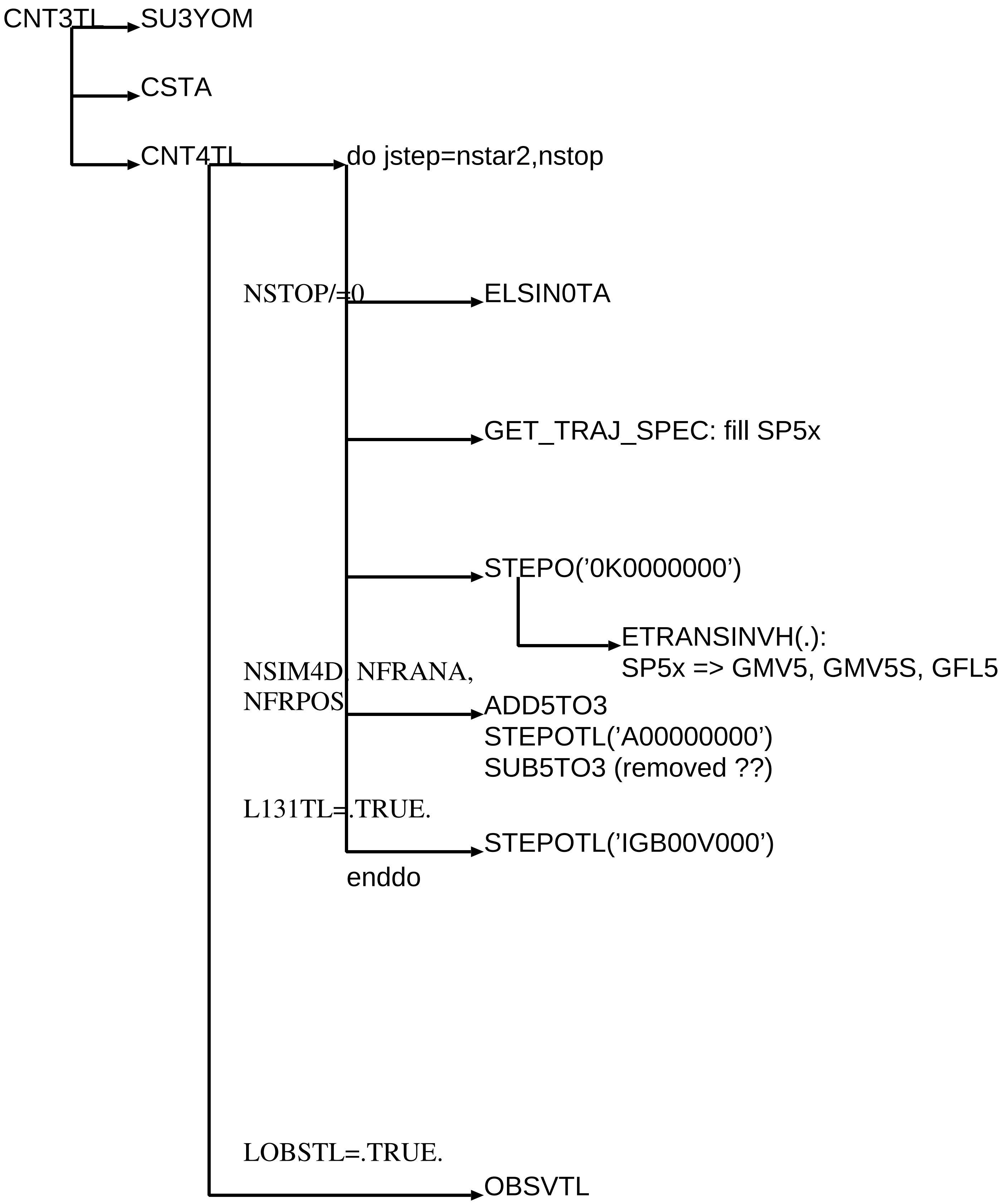



FIG. 4.13 – Calling sequence for CNT3TL/CNT4TL.

the simulator (NSIM4D=0), because we first need to recompute the elements of the trajectory, starting with the information from GMV5, GMV5S, GFL5. In the general case, only the perturbation is handled (GMV, GMVS, GFL). Note that the "old" logic of a preliminary scan (COBS), followed by a "lagged" scan (COBSLAG) is still valid, though this organisation was originally designed for efficiency of the multitasking code on the Cray platforms. COBS is actually only performing copies of the global gridpoint buffer into the "semi-Lagrangian buffer". It is followed by message passing, especially by the on-demand communications of those model gridpoints which are in the vicinity of a halo-located observation. The on-demand facility allows to communicate not all the halo, but only the needed points (Deborah Salmond, ECMWF). The actual interpolations are done below COBSLAG.

- In COBSLAG (Figure 4.15), the key LDMVO5 controls whether we call the routine for the interpolations of the trajectory fields, or for the perturbation. The very same computations are performed, anyhow. They are divided into the actual horizontal interpolations, bilinear in Aladin (NOBSHOR=201) or bicubic in Arpège (NOBSHOR=203), and the message passing communications of the final interpolated values to all processors. The interpolations are done under SLINT, where the weights are computed first (by (E)LASCAW, identical to the Semi-Lagrangian interpolations), followed by the on-demand communications of the halo (second part : all the fields are sent here), and finally the horizontal interpolations are performed. An extra step, not needed in the Arpège c=1 uniform geometry of 4dvar, is the geographical change of the wind components. A rotation is applied to the U/V wind components, taking benefit of the fact that the compas has also been evaluated at observation location in SLINT. The storage and communication of the final model values, interpolated at observation positions (so-called "observation equivalents"), are done by MPOBSEQ. MPOBSEQ message-passes the model equivalent to the processor that holds the observation. At the end of MPOBSEQ, either the YGOM* arrays are filled (for the perturbation) or the YGOM*5 arrays (trajectory). After COBSLAG, the YGOM arrays contain the model equivalents $H\delta x$, projected to the observation points, but still defined on the model vertical levels.
- The vertical interpolations, including the projection to standard pressure levels and the projection through the physical operators (hydrostatism, radiative transfer etc...), are done under OBSVTL (Figure 4.16). Note that the first operation is to define "sets" of observations (ECSET). Those sets are defined to achieve vectorization. First, the data are sorted out by data type, then sets of length NMXLEN (defined in SUDIMO, the default length is 511) are considered. The observation sets can span several timeslots in 3dfgat or 4dvar, since the input to HOPTL are the YGOM* arrays, which have been prepared for all timeslots in CNT3TL. The sets are prepared in ECSET. In former cycles than AL15, some extra Aladin code was present in order to discard the data outside the C+I zone. However, now, only a short test has remained, but should be neutral since the "lamflag" pre-processing filters these data already. Only once the sets of observations are defined, are the vertical observation operators called, in a loop over all the sets. At the end of OBSVTL, the YGOM arrays, unchanged since the beginning of this routine, are printed out by PRTGOM as root mean square values per vertical level.
- The more "magical" sequence of the vertical operations is controlled by a specific monitoring routine, HOPTL (Figures 4.17 and 4.18). It calls all the required vertical interpolations, depending on the type of observations contained in the set of observations. One mainly separates between upper-air, surface and satellite radiances. A more complicated algorithm is active for SATOB winds (which the author did not consider so far in his work!). Again, the logic is to call first the code for the trajectory, then the one for the perturbation (TL routines). The PREINT* routines read from the GOM arrays and prepare all the necessary fields for the actual interpolations. Then, the observational database is checked, to see what type of observation is present. The "body" of each observation is scanned, and the vertical coordinate along with the variable-number for each datum are stored in local tables. These tables will be used to define the exact requests to the obs operators. The latter are then performed by the PPOBSA* routines, which are called inside a loop over all possible variable-numbers. We will not go more into the detail of these routines, which involve the physics of observations. After the PPOBSA* calls, the interpolated perturbations $H\delta x$ are available in array ZXPP, copied into ZHOFX which enters HDEPART. This routine is finally computing the departures in observation

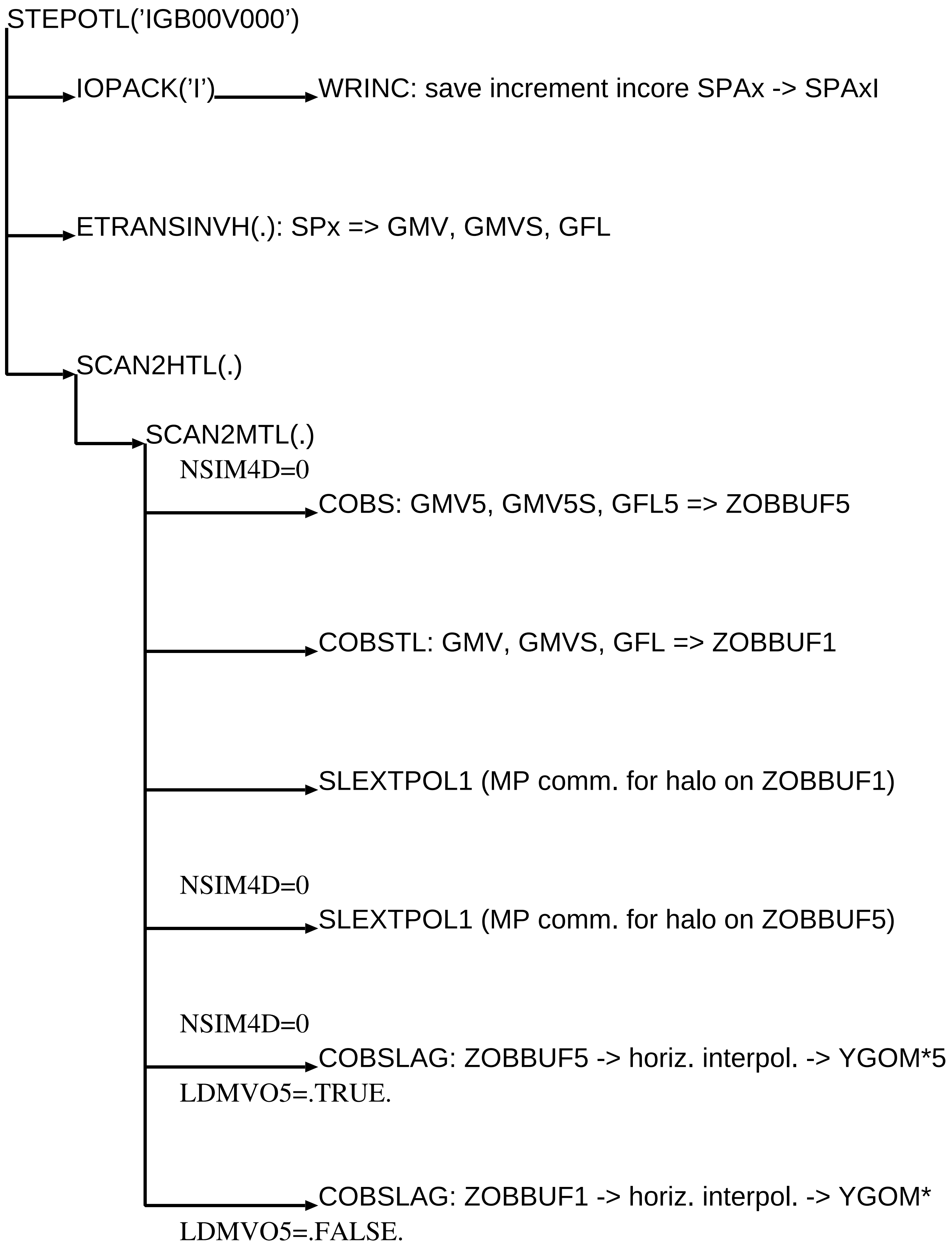



FIG. 4.14 – Calling sequence for STEPOTL/SCAN2MTL.

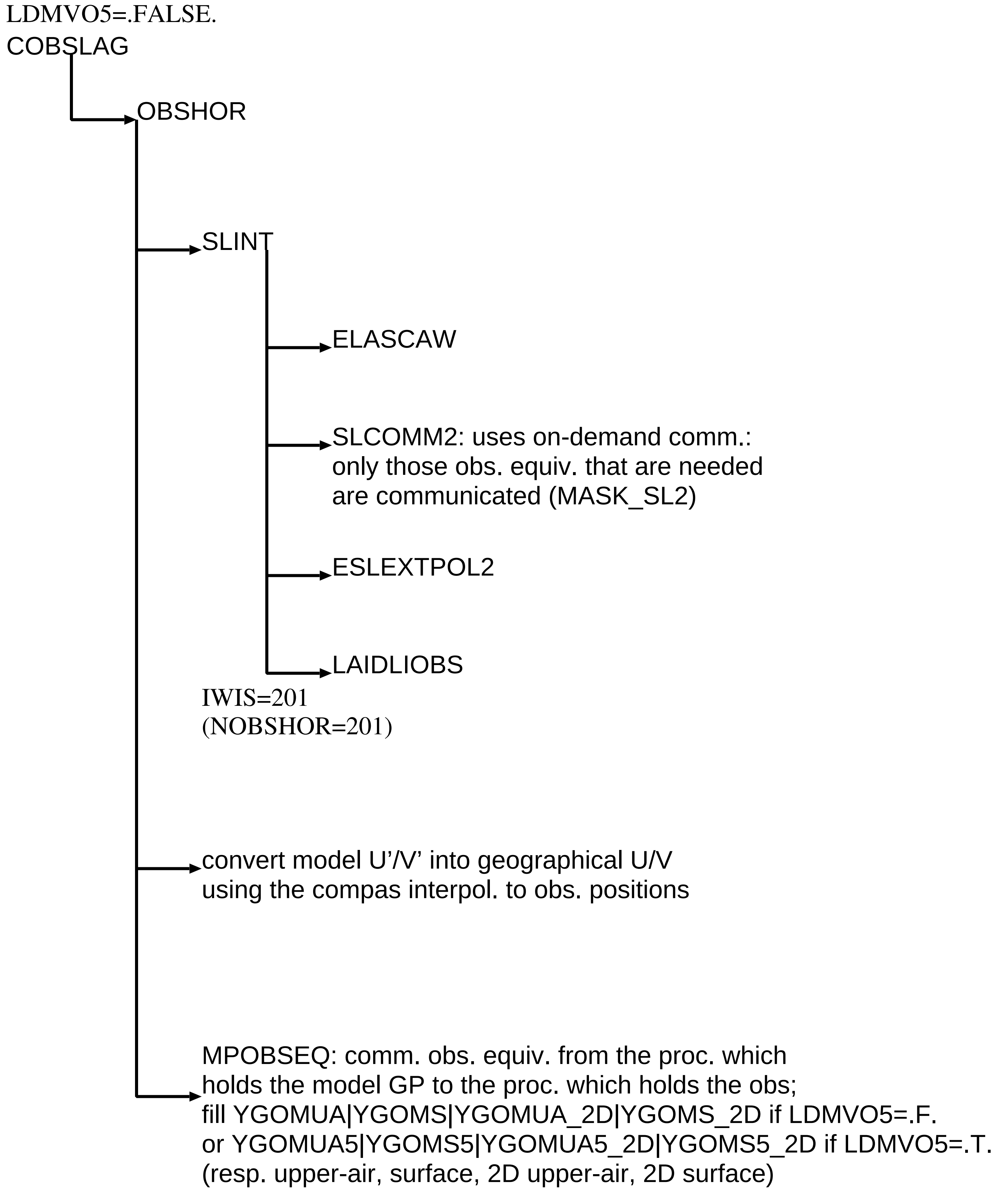



Fig. 4.15 – Calling sequence for COBSLAG/SLINT.

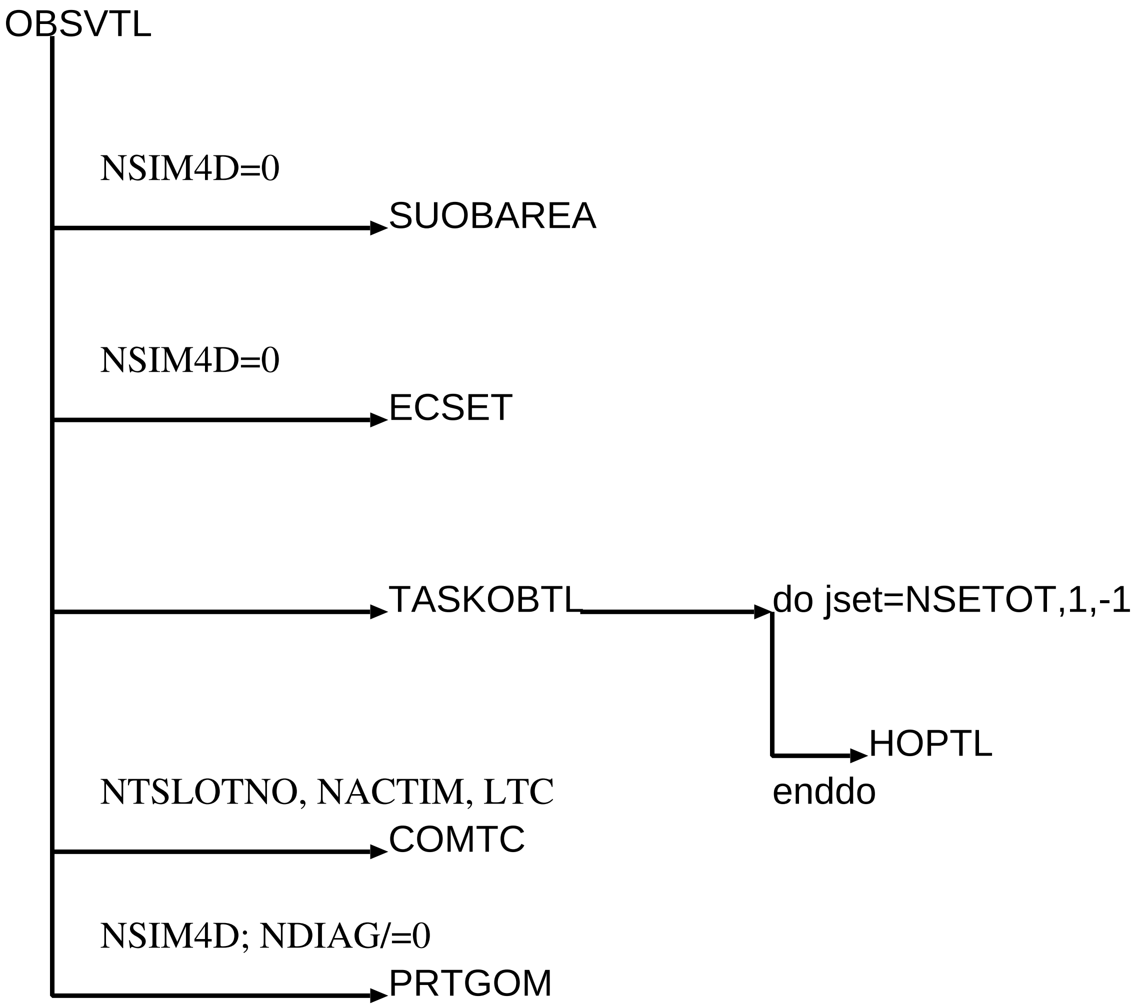



Fig. 4.16 – Calling sequence for OBSVTL/TASKOBTL.

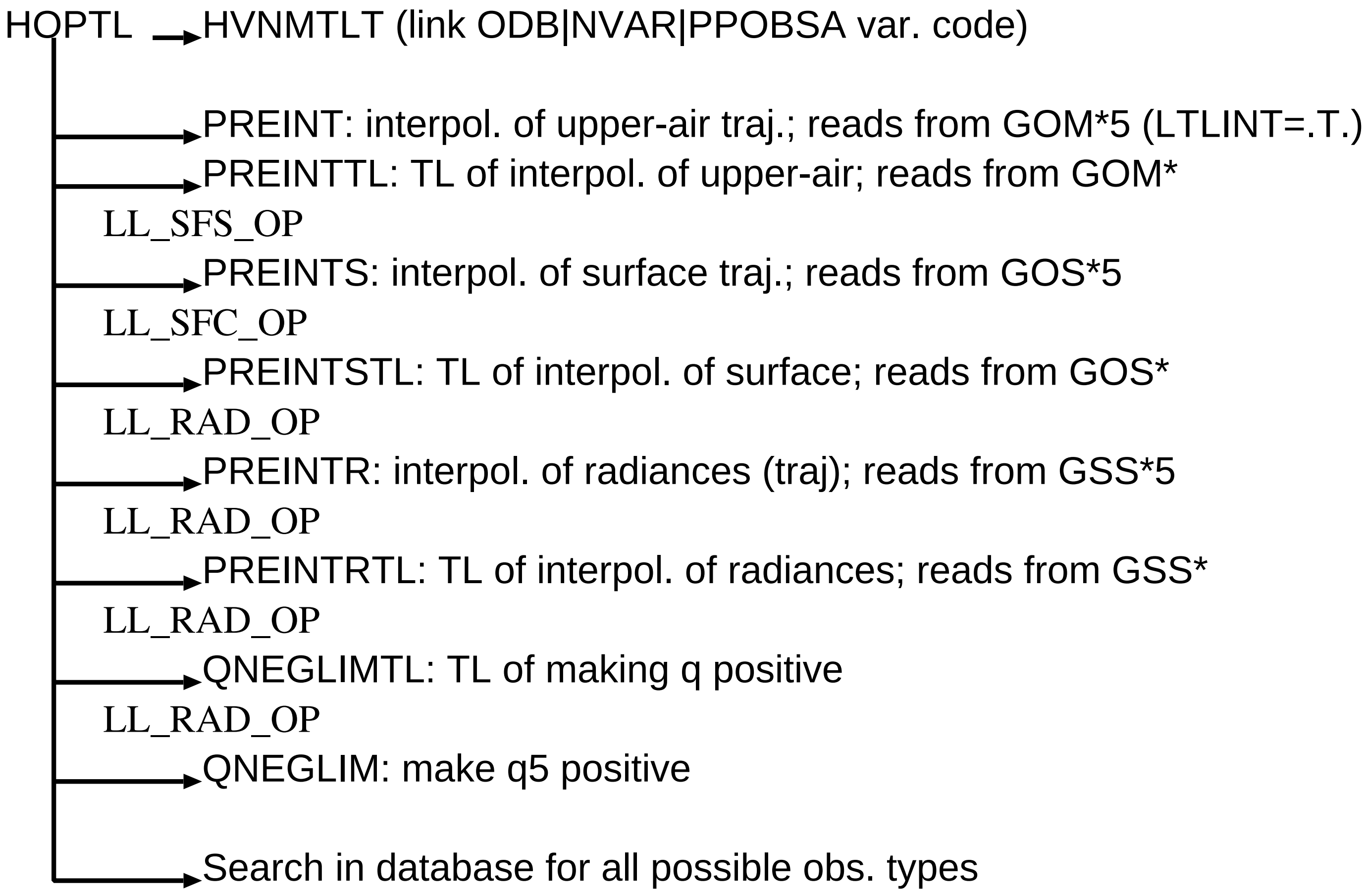



Fig. 4.17 – Calling sequence for HOPTL (first part).

space, weighted by the $\sigma_o$. Note here the quite non-trivial addition and subtraction of the interpolated first-guess (address MBDIFC1), due to the specific operations needed for the multi-incremental 4dvar. In that latter case, the low resolution departure from the previous outer loop is also added to the actual departure. This results in an initial increment which can be non-zero in multi-incremental variational assimilation, for NUPTRA$\geq$1. In Aladin 3dvar, however, this field simply contains the high resolution departures $y - \mathcal{H}(x^b)$. The $J_o$ cost function is set in HJO, where the contributions are stratified by observation type and subtype. Furthermore, HJO writes the "effective" departure $z = d - H\delta x$ into the database (address MBDIOM0(NUPTRA+1)).

Please note that there are two arrays where the $J_o$ cost function is stored : one purely diagnostic (JOT, constructed as a genuine fortran 90 generic variable) is used for the printouts in PRTJO, the other one is the real computational $J_o$ (FJOS). JOT is stratified by type, subtype, variable and area (see module YOMCOSJO). FJOS is indexed by the total number of observations, and summed up in always the same order in EVCOST, to ensure reproducibility in distributed memory.

### 4.2.7 AD observation operators

As usual, the adjoint code is organised in reverse order from the TL code. This section therefore presents the "adjoint" order of the previous section. Please remember that the adjoint is performing a projection from observation space towards model space. Indeed, we first compute the gradient with respect to observation space variations, and then we project this sensitivity field back into the spectral model space. Also, now, the vertical operations are done before the horizontal ones. Let us review the algorithmic sequence :

– The adjoint observation operator is called via the model interface CNT3AD, and computes the gradient of $J_o$ (Figure 4.19) :

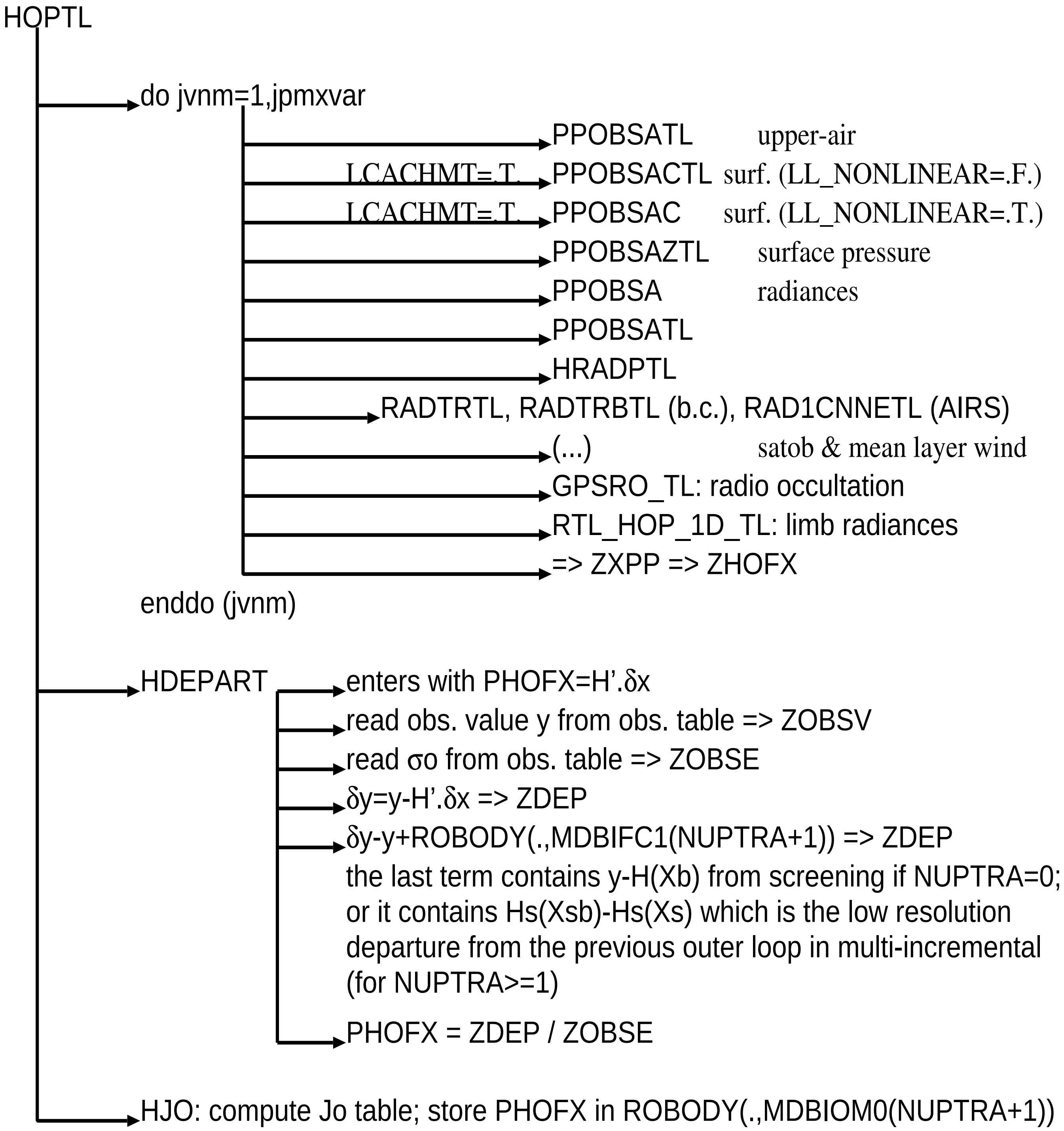



FIG. 4.18 – Calling sequence for HOPTL/HDEPART (second part).

$$\nabla_{\delta x} J_o = H^t R^{-1}(H\delta x - d) = -H^t R^{-1} z$$

At first, the trajectory model-to-observation equivalents are copied from the YGOM*5 to the YGOM* arrays. Thus, the adjoint projection starts with having the interpolated model trajectory in the YGOM* arrays. This is reflected in the output listing, since there is an initial print out of the YGOM* root mean square values per vertical level by PRTGOM, called by OBSVAD. OBSVAD is called before STEPOAD, so that the adjoint of the vertical interpolations are indeed performed before the adjoint of the horizontal interpolations, in accordance with the rule of transposition. Note that in CNT4AD, there is one single call to the adjoint of the vertical operators OBSVAD, followed by one specific call to STEPOAD, outside the time loop. Thus, only this sequence is actually active in Aladin 3dvar. The time loop itself is not active, except if a 3dfgat or a 4dvar are run. Some specific Aladin routines are present in CNT4AD : ELSIN0TA (set perturbations in the coupling zone to zero), EWRLSGRAD (write out the gradient with respect to the lateral boundary conditions). These routines are only useful when the propagator models are run (configurations e131, e501, e401, e601, e801, where NSTOP>0).

- OBSVAD prints out the YGOM* arrays twice : first before calling the adjoint interpolations (thus providing the root mean squares of the trajectory fields), then after the adjoint interpolations, providing the root mean squares of the gradient projected to vertical levels, but not yet to model grid points. The adjoint vertical interpolations are again called in a loop, for each "set of observations" (see Figure 4.20).
- By comparing the figures 4.17/4.18 and 4.21/4.22, one might convince himself that HOPAD indeed acts as the adjoint of HOPTL. Note that the operations on the trajectory fields are done first. Then the database is searched for all possibly present observations. The gradient in observation space is then initialized by a simple reading of the effective departure $z$ from the database (word MDBIOM0(NUPTRA+1)), multiplied by $-2/\sigma_o$. Local arrays are zeroed, then the calls to the vertical physical operators (mostly adjoints, PPOBSA*AD now) are done in a loop over all possible variable-numbers. At the end of HOPAD, the YGOM* arrays are zeroed again, and then the PREINT*AD routines fill the YGOM* with the new contributions. Note also that the gradient with respect to the surface temperature, taken as a rough estimate of a skin temperature, is stored in the TOVSCVX vector, as part of the control variable for the next iteration of the simulator (LTOVSCV=.TRUE.).
- STEPOAD (Figure 4.23) is used to run the adjoint of the horizontal interpolations. In its most general principle, the perturbations stored in the YGOM* arrays (by HOPAD) are read, and then projected back to the model gridpoints (in COBSLAGAD). Message passing is needed for the halo points. In COBSAD, the gradients on model gridpoints are copied into (GMV, GMVS, GFL), which is in turn read by (E)TRANSINVHAD and transformed into spectral space. Finally, one recovers the gradient of $J_o$ in the SPAx arrays.
- COBSLAGAD (Figure 4.24) runs the horizontal interpolations : message passing to send the model equivalents to the processor that holds the model gridpoint (if different), conversion of the geographical wind in non-uniform grids (as is the case in the Aladin plane geometry), adjoint of bilinear interpolations (SLINTAD). One extra difficulty is the need to assure numerical reproducibility of the results. Thus, the message passing is a bit more complicated than in the direct code (in OBSHORAD) so that the contributions of different points to one gradient are summed always in the same order. Note that throughout the adjoint code, the name of the gradient array changes several times : POBB1, PB1, ZOBBUF1.

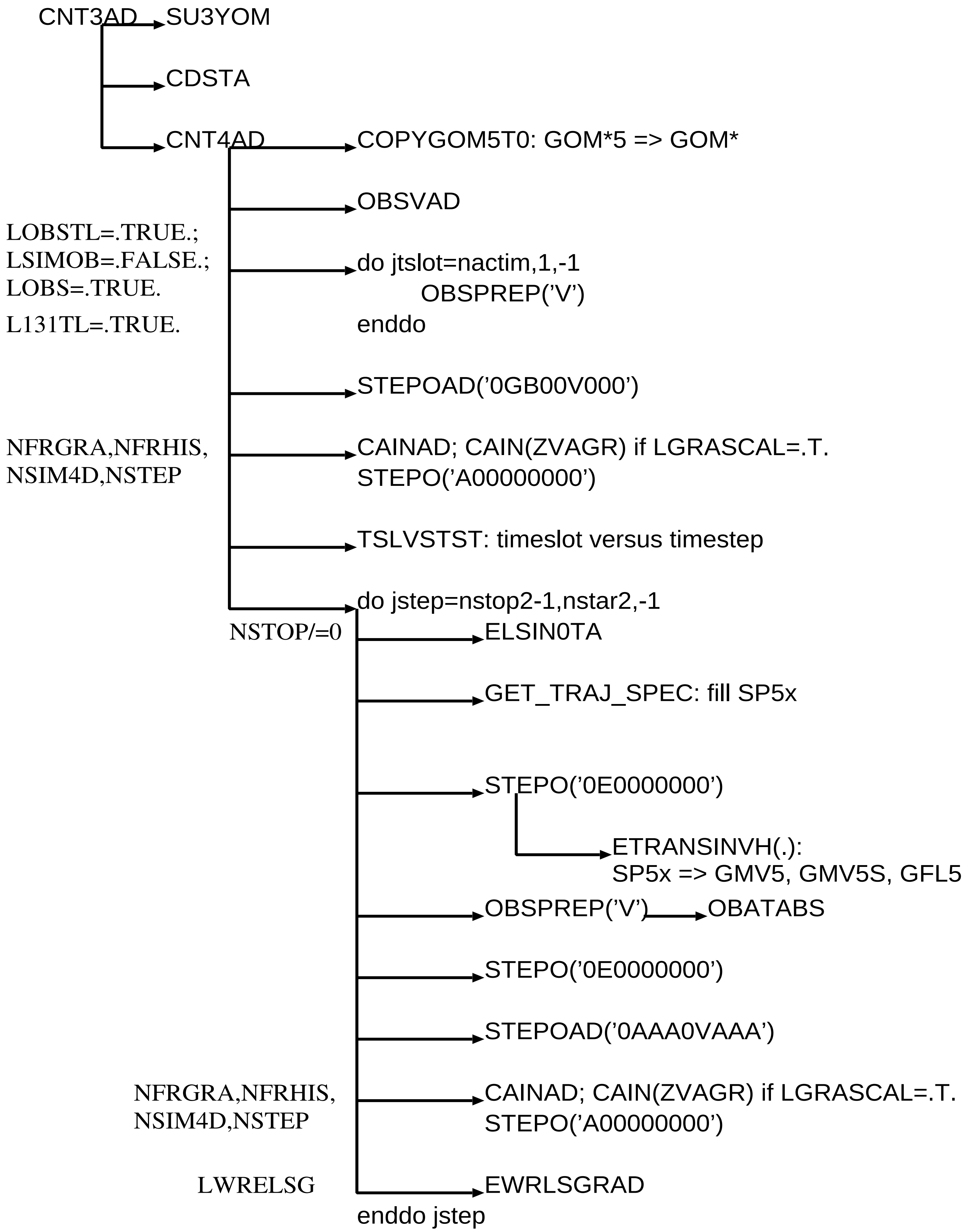



Fig. 4.19 – Calling sequence for CNT3AD/CNT4AD.

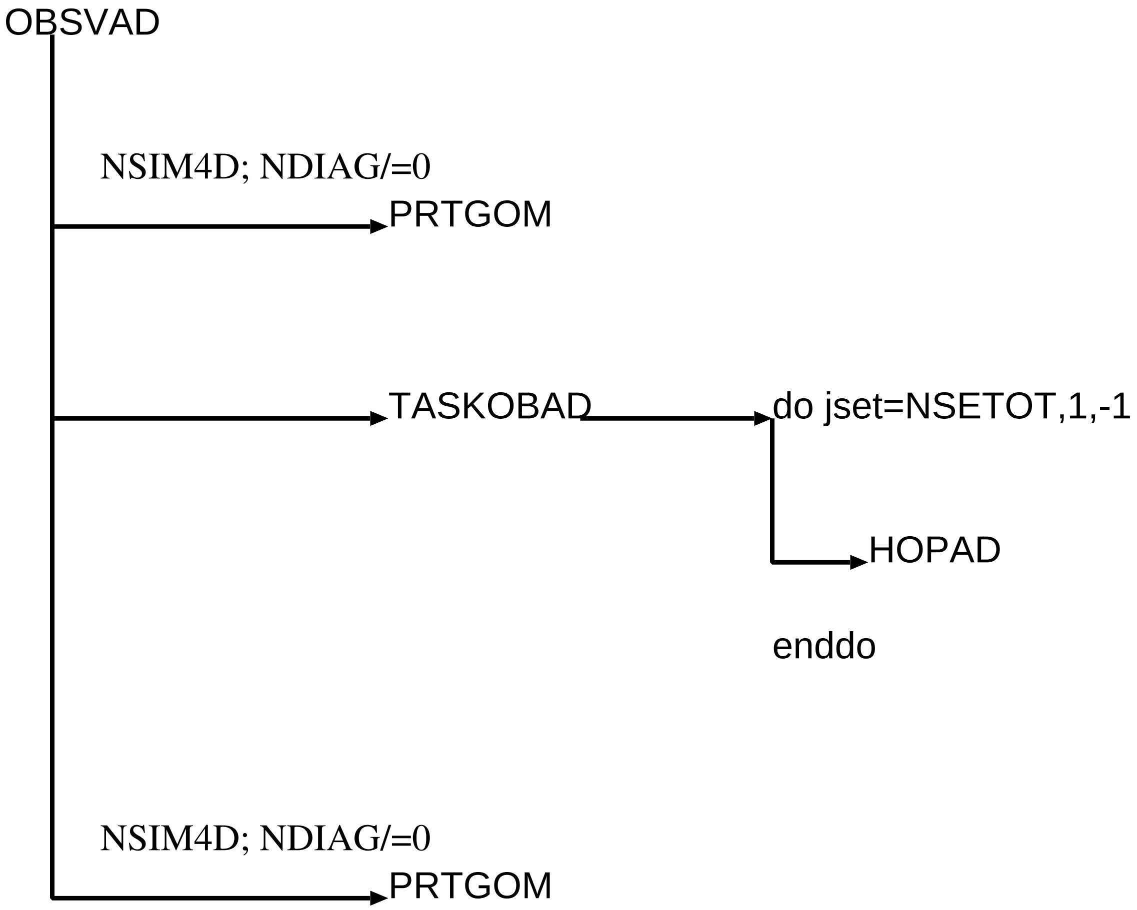



FIG. 4.20 – Calling sequence for OBSVAD/TASKOBAD.

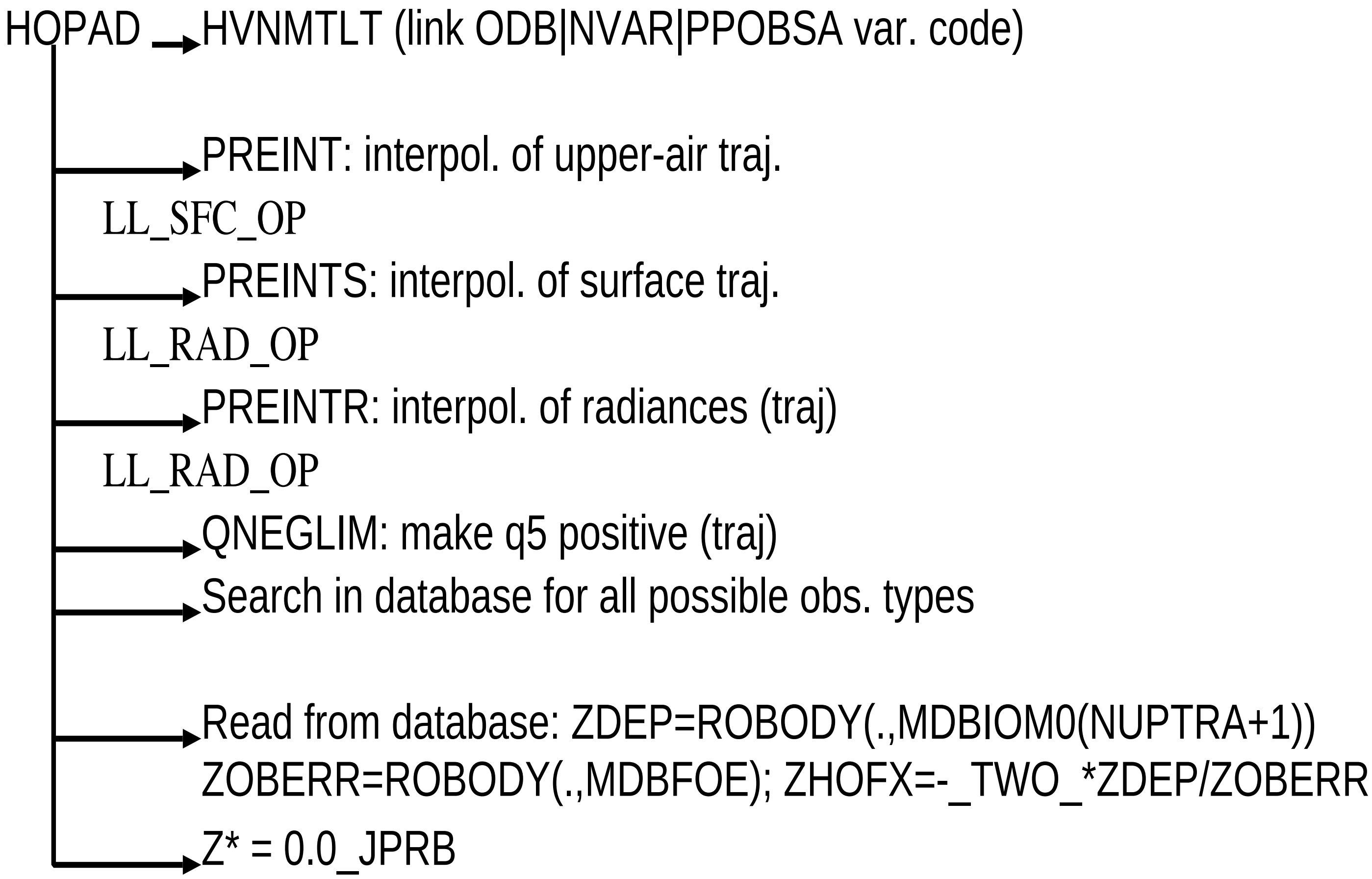



Fig. 4.21 – Calling sequence for HOPAD (first part).

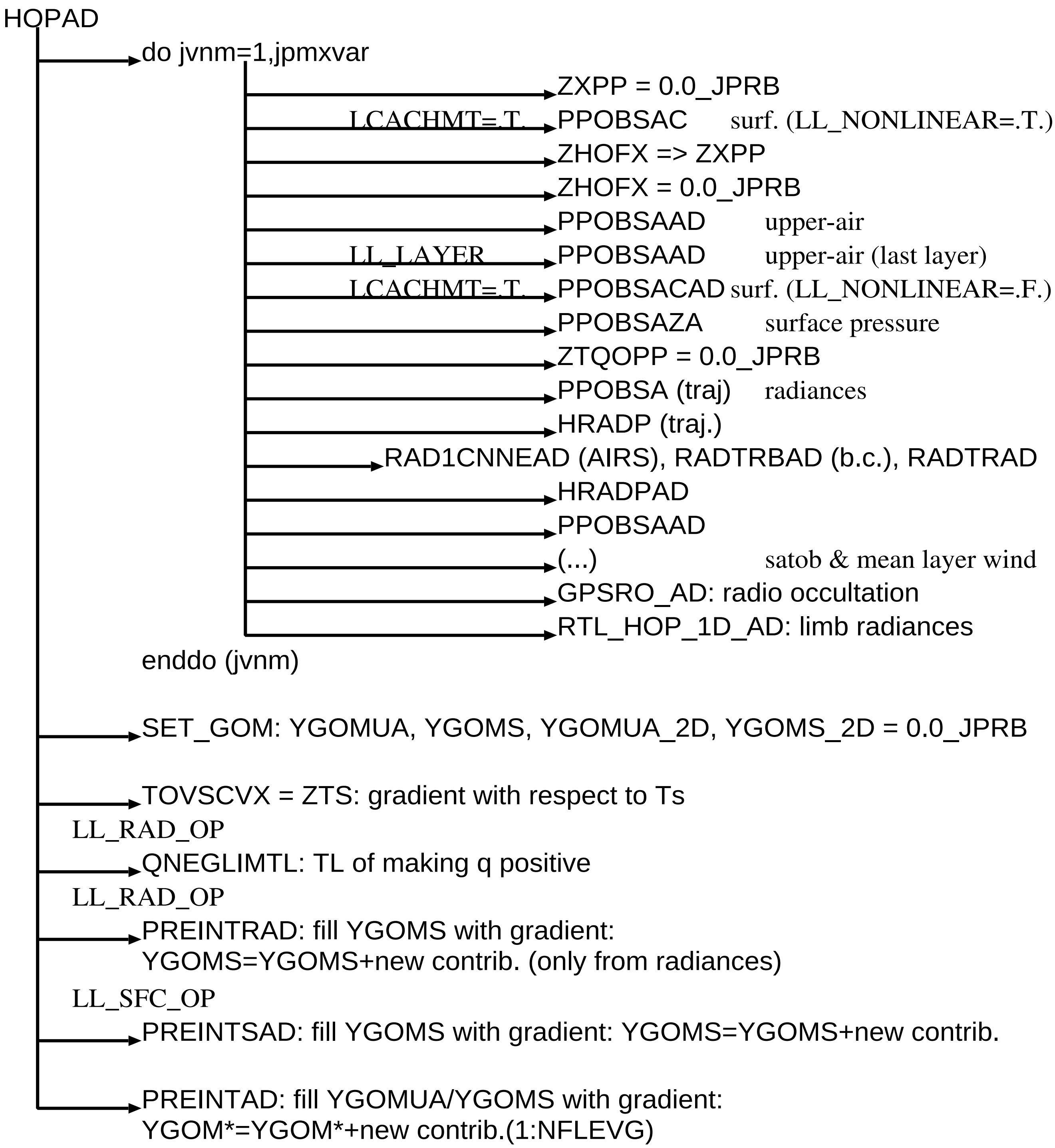



FIG. 4.22 – Calling sequence for HOPAD (second part).

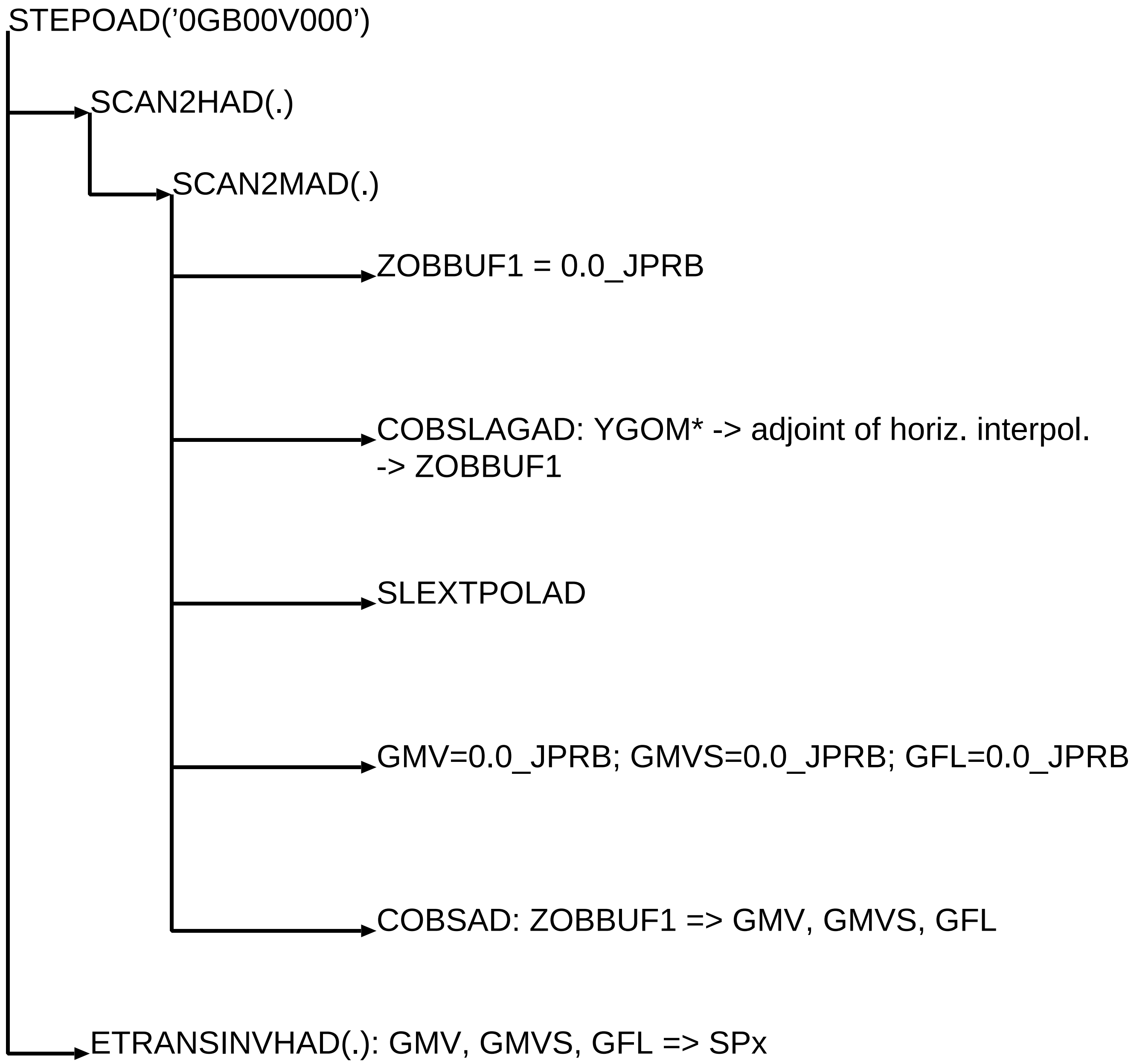



Fig. 4.23 – Calling sequence for STEPOAD/SCAN2MAD.

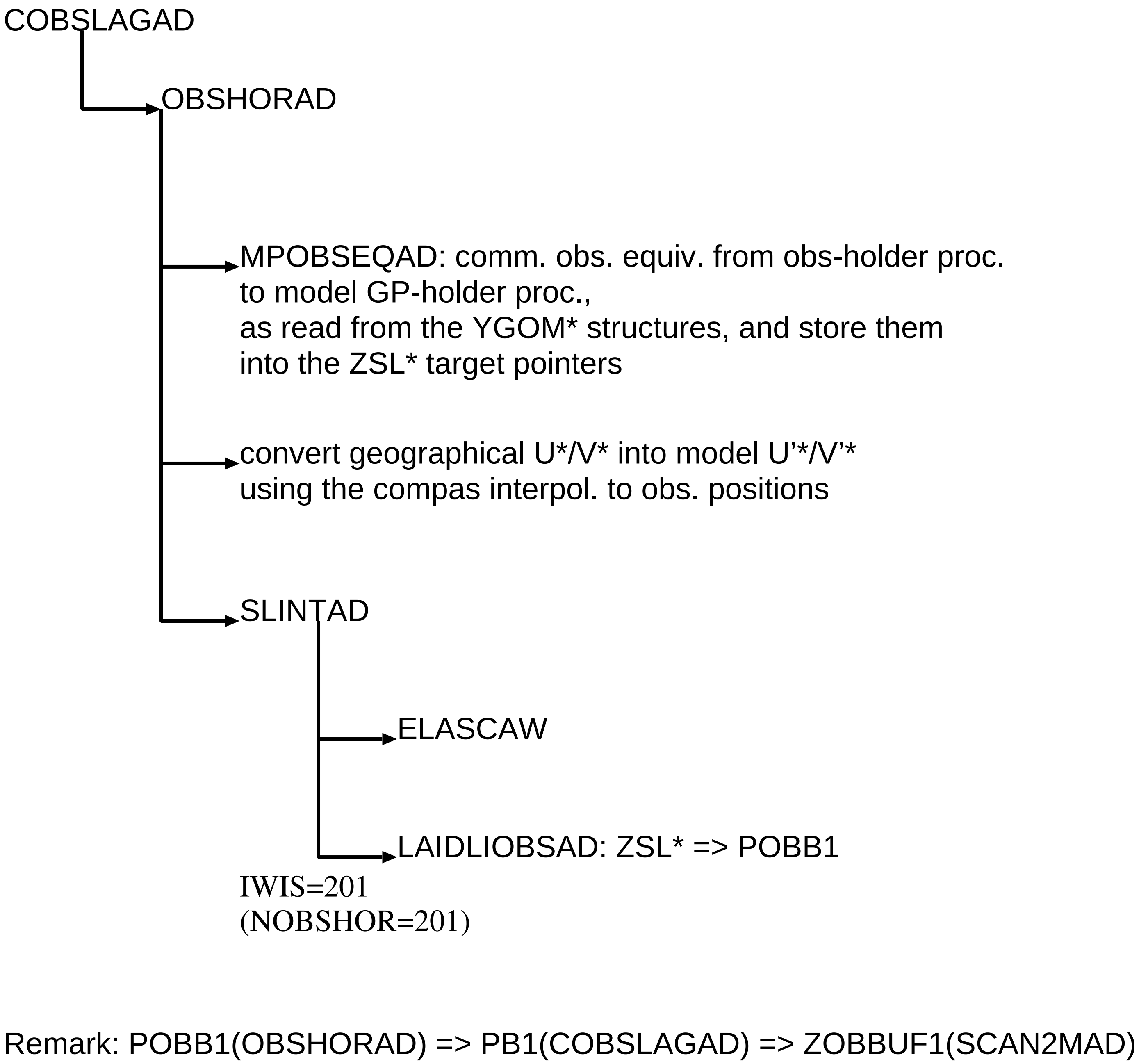



Fig. 4.24 – Calling sequence for COBSLAGAD/SLINTAD.

## 4.3 4dvar minimisation

### 4.3.1 Trajectory handling

### 4.3.2 A very short overview of simplified physics

### 4.3.3 Specific multi-incremental features

## 4.4 conjugate gradient

Useful algorithmic information about the conjugate gradient methods, and their link with the Lanczos method, can be found in Fisher (1998) and in Yessad (2005).

## 4.5 3dvar/fgat

# Chapitre 5

# User's guide : scripts and namelists

## 5.1 Sequences of the execution and Input/Output

There are three main steps to compute one 3dvar analysis : creation of the LAM selected database (not for Arpège obviously !), screening of data, minimization. We will now describe the major sources of information that are used or output by these three steps.

The user's guide heavily focuses on Aladin variational analysis, in 3D-VAR mode. Information about the Arpège 4D-VAR system may be obtained from Desroziers and Riber (2002) . The latter is a technical description of the 4D-VAR scripts, as they used to be run before the implementation onto OLIVE. Similarly, the 3D-VAR presentation below does not mention the OLIVE installation.

### 5.1.1 Creation of a limited area observational database in ODB format (Aladin only)

This database is obtained via the LAM_FLAG program, prepared by Wafaa Sadiki and Sandor Kertesz. The program scans the global database (lat,lon) positions in entry, and runs the Aladin geometry package (Jean-Daniel Gril). For each observation position, it decides whether the data are in or out the computational C+I zone. A flag (located at ROBHDR(JOBS,MDB_ZONE_AT_HDR) in the database) is set to 1 for the C+I observations. Then the ODB shuffle is run, with the environmental parameter TO_ODB_REDUC set to 1 in the script. This shuffle creates the limited area filtered ODB database for Aladin. The main input (I) and output (O) from the script `Obs2LamOdb.ksh` are summarized in Table 5.1.

Please note that the script can run in several modes, depending on the actual input table. We have focused our attention to "mode 1". Other modes include the conversion from an already existing global ODB base to a LAM database, the conversion from one LAM base to another etc... (see the header description of the script for more details).

### 5.1.2 Screening of the data

Since cycle AL15, it is forbidden to treat observations that are outside the C+I zone, though some "safety" code still is kept inside the Arpège/IFS to try to get on with this atypical situation. However, in general, any obs position outside C+I firstly makes some setup loops to cycle without initialization, which is not desirable, and secondly causes hard fortran crashes at the level of SUFGER/GEFGER, where the consequences for non-initialization are felt first. Thus, a "golden rule" for the screening execution is that : "shall thou fail in GEFGER in the screening, then thy C+I database is corrupted". Mostly, the corruption could come from a too tightly or badly defined geometry in the namelist for LAMFLAG, or no C+I filtering at all, or some mismatches at the edges of the domain when different geometry versions are used (for example a mixture of old EGGX and new EGGPACK geometries, in between cycles AL15 and AL25T1).

| | | |
|---|---|---|
| $DIR_obs/obsoul | global ascii obs file | I |
| $DIR_namelist/lamflag_odb.nml | control namelist | I |
| $DIR_ref/file_date | file with date and experiment | I |
| assimDate<br>TYPRES<br>NJOB<br>NMXJOB<br>expName | 2002103106<br>assim<br>0 (fixed)<br>1 (fixed)<br>min4 | $1<br>$2<br>$3<br>$4<br>$5 |
| $EXE_bat | executable bator | I |
| $EXE_lamflag_odb | executable LAMFLAG | I |
| $EXE_shuffle | executable shuffle | I |
| $DIR_saveGlobodb/ecma_${assimDate}.${assimType} | global ODB | O |
| $DIR_saveLamodb/ECMA_LAM.tar | LAM C+I database | O |

Tab. 5.1 – Input/output for the production script of a LAM database.

The screening needs a number of input data files, usually taken from the global model assimilation galaxy, along with a first guess file (6 hour forecast, blended state etc ...), and a namelist specific to the screening. Screening can be viewed in short as a model run, immediatly followed by a setup and a quality control of the observations with respect to the trajectory. One needs to give an observation database in entry, which will have its quality control flags updated by the screening. As output, one retrieves the flagged database, but no new database is created physically. The entry database is generally named "ECMA" (extended) and the output is still "ECMA". The database is reduced (compressed) physically by a run of the ODB shuffle, to obtain the "CCMA" base. The flow of information for the screening is summarized in Table 5.2.

### 5.1.3 3dvar/4dvar/FGAT minimizations

The minimization (configuration 131) reads the observation minus first-guess departures from the CCMA database, and runs the inner loop of the minimization. The algorithm consists actually of two iterative processes : the "inner-inner" loop is the iterative quasi-Newton or conjugate gradient search (inside M1QN3 and CONGRAD, respectively for Aladin and Arpège). This innermost loop is embedded in the iterative recomputation of gradients by the simulator. Thus, two maximum number of iterations have to be defined, and are given in the "file_envir" ascii file : NITER is the maximum number of inner-inner minimization loops, while NSIMU is the maximum number of simulations of the gradient. Additionally, the 3dvar needs to read the binary files which contain the regression coefficients and the covariance tables for the $B$ matrix. These data are read from two so-called "stabal96" files for the homogeneous, isotropic $B$ (Derber and Bouttier, 1999, Berre, 2000, Sadiki etal., 2001, Široka etal., 2003). The flow of information for the minimization is summarized in Table 5.3 :

| | | |
|---|---|---|
| files read in SUADVAR called by SURAD if LSCREEN | | |
| $DIRBCOR/bcor_noaa.dat | bias correction for radiances/$T_b$ | I |
| $dir_const/correl.dat | background correlations | I |
| $dir_const/sigmab.dat | background $\sigma_b$ | I |
| $dir_const/chanspec_noaa.dat | channel specifications on all satellites | I |
| $dir_const/rmtberr_noaa.dat | random $T_b$ errors for all satellites | I |
| $dir_const/rmtberr_airs.dat | random $T_b$ errors for AIRS channels | I |
| $dir_const/cstlim_noaa.dat | cost limits for all sat. and subtypes | I |
| grib file read in INQGRIB called by SCREEN/INIFGER | | |
| $dir_const/errgrib0scr | forecast errors on model grid | I |
| file read in TOVCF called by SURAD/RTSETUP/RTTVI | | |
| $dir_const/rt_coef_atovs_newpred_ieee.dat | RT coefficients | I |
| $dir_const/rszcoef_fmt | | I |
| $dir_const/rtcoef.01.tgz | set of RTTOV8 coef files | I |
| file read in SUNNE called by SURAD/RTSETUP | | |
| $dir_const/nn_weights.dat | coef for neural network (b.c. AIRS) | I |
| $DIRGUESS/ICMSHALAD+0006 | first guess FA file | I |
| $DIR_saveLamodb/ECMA_LAM.tar | LAM C+I database | I |
| $DIR_namelist/scr_3d.nml | control namelist | I |
| $DIR_ref/file_date | file with date and experiment | I |
| assimDate | 2002103106 | $1 |
| TYPRES | assim | $2 |
| NJOB | 0 (fixed) | $3 |
| NMXJOB | 1 (fixed) | $4 |
| expName | min4 | $5 |
| $EXE_screen | executable model screening | I |
| $EXE_toccma | executable shuffle to CCMA | I |
| $XJAG/ECMA.tar | updated database ($y - \mathcal{H}(x^b)$) | O |
| $XJAG/CCMA.tar | compressed ODB base | O |

Tab. 5.2 – Input/output for the screening of observations.

| file read in TOVCF called by SURAD/RTSETUP/RTTVI | | |
|---|---|---|
| $dir_const/rt_coef_atovs_newpred_ieee.dat | RT coefficients | I |
| $dir_const/rszcoef_fmt | | I |
| $dir_const/rtcoef.01.tgz | set of RTTOV8 coef files | I |
| $DIRGUESS/ICMSHALAD+0006 | first guess FA file | I |
| $XJAG/CCMA.tar | compressed ODB base | I |
| $DIR_namelist/assim.nml | control namelist | I |
| $DIR_ref/file_date | file with date and experiment | I |
| assimDate | 2002103106 | $1 |
| TYPRES | assim | $2 |
| NJOB | 0 (fixed) | $3 |
| NMXJOB | 1 (fixed) | $4 |
| expName | min4 | $5 |
| $DIR_ref/file_envir | file with max iteration numbers | I |
| NB_ITER | maximum no. of iterations (60) | $1 |
| NB_SIMUL | maximum no. of simulations (70) | $2 |
| $DIR_work/stabal96.cv | vertical covariances for predictors | I |
| $DIR_work/stabal96.bal | regression coefficients for stat. bal. | I |
| $EXE_minim | executable 3dvar | I |
| $XJAG/CCMA.tar | updated base $(y - \mathcal{H}(x_a))$ | O |
| $XJAG/POS1 | background fields in FA file | O |
| $XJAG/POS2 | analysis fields in FA file | O |

TAB. 5.3 – Input/output for the minimization e131.

## 5.2 Most important namelist parameters

The list of namelist parameters for the screening and the minimization is obviously not exhaustive. The Arpège/IFS software allows for a multitude of configurations and switches, many of them not active in Aladin. We have listed here only those parameters which either are crucial (usually, because only one value is actually possible), or control a major aspect of the execution. When only one value is indicated, then it is understood that either the other (Arpège) possible values do not work for Aladin, or never were tried. Otherwise, possible alternative values are indicated, but the default value is given first.

### 5.2.1 LAMFLAG (Aladin only)

Table 5.4 lists the namelist parameters to run the C+I data selection.

### 5.2.2 Screening

Table 5.5 lists the namelist parameters to run the screening configuration.

### 5.2.3 3dvar/4dvar/FGAT minimizations

Tables 5.6 and 5.7 list the namelist parameters to run the minimization configuration.

| Name of parameter | Namelist | Possible values | Definitions |
|---|---|---|---|
| LOBSONLY | NAMFCNT | F/T | if true, only select per obs type |
| ELAT0 | NAMFGEOM | degrees | latitude of reference point of LAM domain |
| ELON0 | NAMFGEOM | degrees | longitude of reference point of LAM domain |
| ELATC | NAMFGEOM | degrees | latitude of centre point of LAM domain |
| ELONC | NAMFGEOM | degrees | longitude of centre point of LAM domain |
| ELAT1 | NAMFGEOM | degrees | latitude of SW corner of LAM domain |
| ELON1 | NAMFGEOM | degrees | longitude of SW corner of LAM domain |
| EDELX | NAMFGEOM | meters | $\delta x$ of the C+I grid |
| EDELY | NAMFGEOM | meters | $\delta y$ of the C+I grid |
| NDLUN | NAMFGEOM | 1 | abscissa of first C+I point |
| NDGUN | NAMFGEOM | 1 | ordinate of first C+I point |
| NDLUX | NAMFGEOM | integer | abscissa of last C+I point |
| NDGUX | NAMFGEOM | integer | ordinate of last C+I point |
| CANZONE | NAMFGEOM | kilometers | width for a wider selection (for CANARI) |
| REDZONE | NAMFGEOM | kilometers | width for a tighter selection (reduce C+I) |
| REDZONE_S | NAMFGEOM | kilometers | REDZONE only from the Southern border |
| REDZONE_N | NAMFGEOM | kilometers | REDZONE only from the Northern border |
| REDZONE_W | NAMFGEOM | kilometers | REDZONE only from the Western border |
| REDZONE_E | NAMFGEOM | kilometers | REDZONE only from the Eastern border |
| LVAR | NAMFGEOM | T/F | selection will be done for 3dvar applications |
| LNEWGEOM | NAMFGEOM | F/T | origin of coordinate system will be SW corner when false, and center point when true |
| LSYNOP | NAMFOBS | T/F | false if synop obs are to be skipped |
| LAIREP | NAMFOBS | T/F | false if airep obs are to be skipped |
| LSATOB | NAMFOBS | T/F | false if satob obs are to be skipped |
| LDRIBU | NAMFOBS | T/F | false if dribu obs are to be skipped |
| LTEMP | NAMFOBS | T/F | false if temp obs are to be skipped |
| LPILOT | NAMFOBS | T/F | false if pilot obs are to be skipped |
| LSATEM | NAMFOBS | T/F | false if sat radiances are to be skipped |
| LPAOB | NAMFOBS | T/F | false if paob obs are to be skipped |
| LSCATT | NAMFOBS | T/F | false if scatt obs are to be skipped |
| LSLIMB | NAMFOBS | F/T | false if limb sounder obs are to be skipped |
| LOBT11 | NAMFOBS | F/T | false if obs type 11 is to be skipped |
| LOBT12 | NAMFOBS | F/T | false if obs type 12 is to be skipped |
| LRADAR | NAMFOBS | F/T | false if radar obs are to be skipped |
| LOBT14 | NAMFOBS | F/T | false if obs type 14 is to be skipped |
| LOBT15 | NAMFOBS | F/T | false if obs type 15 is to be skipped |

Tab. 5.4 – Namelist parameters for the C+I LAM selection.

| Name of parameter | Namelist | Possible values | Definitions |
|---|---|---|---|
| NPROC | NAMPAR0 | integer | total number of processors |
| NPROMA | NAMDIM | integer | length of vectorization in SCAN2M |
| NUPTRA | NAMVAR | -1 | number of updates of the trajectory |
| LTOVSCV | NAMVAR | T/F | use surface temperature extra control<br>variable (for raw radiance RT) |
| NOBSHOR | NAMOBS | 201/203 | type of horiz. interpol.<br>(201=bilinear in Aladin) |
| LOBS | NAMCT0 | TRUE | real observations (read from database) |
| LOBSC1 | NAMCT0 | TRUE | screening configuration 001<br>(traj is computed) |
| LSIMOB | NAMCT0 | FALSE | simulated observations<br>(TRUE in e401, e801) |
| LSCREEN | NAMCT0 | TRUE | quality control and thinning switched on |
| NCNTVAR | NAMCT0 | 2 | control variable normalized by $B$ |
| LGOM* | NAMDIM | T-F | activate 3D field for obs equivalent (conv data) |
| LGSM* | NAMDIM | T-F | activate 3D field for obs equivalent (sat data) |
| LSCRE4D | NAMSCC | F/T | true if 4d screening |
| LSCDPR | NAMSCC | T/F | level of verbose in the screening return listing |
| RFIND_AIREP | NAMSCC | meters | size of thinning boxes for airep<br>(now : 60000. in Aladin) |
| RFIND_RAD1C | NAMSCC | meters | average distance between radiance data |
| RMIND_RAD1C | NAMSCC | meters | minimum distance between radiance data |
| ISATTV | NAMTOVS | array of integers | satellite identification |

TAB. 5.5 – Namelist parameters for the screening of observations.

| Name of parameter | Namelist | Possible values | Definitions |
|---|---|---|---|
| NPROC | NAMPAR0 | integer | total number of processors |
| NPROMA | NAMDIM | integer | length of vectorization in SCAN2M |
| NUPTRA | NAMVAR | 0 | number of updates of the trajectory |
| LBGOBS | NAMVAR | FALSE | if true, compute background errors for observed quantities |
| LGRTEST | NAMVAR | F/T | run the test of the gradient in e131 : |
| NTESTVAR | NAMVAR | 0/1/2/3 | 0 - never (!)<br>1 - before minim<br>2 - after minim<br>3 - both before and after minim |
| NITER | NAMVAR | integer (60) | max number of iterations in the minimizer |
| NSIMU | NAMVAR | integer (70) | max number of iterations in the simulator |
| NGRATS | NAMVAR | array of integers | write-out events for the gradient (w/r to simulator) |
| NFRGRA | NAMVAR | integer | frequency of write-out of the gradient (w/r to simulator) |
| NANATS | NAMVAR | array of integers | write-out events for the analysis (w/r to simulator) |
| NFRANA | NAMVAR | integer | frequency of write-out of the analysis (w/r to simulator) |
| LJC | NAMVAR | FALSE | switch off $J_c$ in Aladin |
| RCVGE | NAMVAR | 1.E-3 | stop criterion for the gradient norm reduction in the minimizer |
| L_CHECK(...)<br>(...)_CONVERGENCE | NAMVAR | F/T | if true, abort if the minimization fails |
| LTOVSCV | NAMVAR | T/F | use surface temperature extra control variable (for raw radiance RT) |
| LOBSTL | NAMVRTL | T/F | activate the TL obs op. |
| L131TL | NAMVRTL | T/F | run incremental with TL model |
| NOBSHOR | NAMOBS | 201/203 | type of horiz. interpol. (201=bilinear in Aladin) |
| LOBCLN | NAMOBS | FALSE | if true, clean the unwanted wagons from the obs array |
| LCAPACH | NAMOBS | FALSE | if true, use APACHE |
| LCACHMT | NAMOBS | TRUE | if true, use ACHMT in obs operators |
| LOBS | NAMCT0 | TRUE | real observations (read from database) |
| LOBSREF | NAMCT0 | TRUE | comparison of observations with the current incore trajectory |
| LSIMOB | NAMCT0 | FALSE | simulated observations (TRUE in e401, e801) |
| NCNTVAR | NAMCT0 | 2 | control variable normalized by $B$ |
| ISATTV | NAMTOVS | array of integers | satellite identification |
| NEINI | NAMINI | 0 | choice of digital filter for $J_{cdfi}$ (only Arpège) |

TAB. 5.6 – Namelist parameters for the 3dvar minimization.

| | | | |
|---|---|---|---|
| NMUPD | NAMIOMI | integer (5) | number of expected updates of trajectory |
| CFVATRA | NAMIOMI | 'VATRH' | filename where to write out the Hessian for a warm restart |
| L3DBGERR | NAMJG | TRUE | use 3d (nonseperable) variances of background error |
| LCORCOSU | NAMJG | FALSE | enforce compactly supported correlations |
| REDNMC | NAMJG | 1.0 | multiplying factor for all $\sigma_b$ |
| LSPFCE | NAMJG | TRUE | use horizontally constant background errors |
| LJBBALBETA | NAMJG | F/T | $\beta$-plane $J_b$ in Aladin |
| NPSMAX | NAMJG | 15 | number of waves to describe $\beta$-effect |
| NOTVAR | NAMJO | array of 0/-1 | switch on/off the use of the obs type and variable in $J_o$ (see YOMCOSJO for the ordered list of variables) |
| LEVARGP | NEMVAR | F/T | force $\sigma_b$ to a vanishing value in a rim zone around LAM borders |
| NBZONVAR_EW | NEMVAR | 0 | gridpoint width of $\sigma_b$ rim zone |
| EXPVAR | NEMVAR | 0. | exponent of the rim zone sine function |
| EVAREPSILON | NEMVAR | 1. | pseudo-zero of the forced $\sigma_b$ (usually set to $10^{-2}$ if LEVARGP) |
| LCORCOSU_AUTO | NEMVAR | FALSE | compact support on auto-correlations |
| LCORCOSU_STAT | NEMVAR | FALSE | compact support on multivariate regressions |
| LCORCOSU_HBAL | NEMVAR | FALSE | compact support on horizontal balance |
| ENIL1 | NEMVAR | 0. | start length for compact support (auto and stat) |
| ENIL2 | NEMVAR | 0. | radius for compact support (auto and stat) |
| ENILHBAL1 | NEMVAR | 0. | start length for compact support (hbal) |
| ENILHBAL2 | NEMVAR | 0. | radius for compact support (hbal) |
| LSTREAM | NEMVAR | FALSE | apply compact support on $\Psi$ instead of $\zeta$ (case auto) |

TAB. 5.7 – Namelist parameters for the 3dvar minimization (continued).

| Obs type | Analyzed fields | Position in NOTVAR |
|---|---|---|
| Synop | relative humidity | 5 |
| | relative humidity at 2 m | 6 |
| | temperature | 7 |
| | geopotential | 8 |
| | temperature at 2 m | 11 |
| Airep | wind U | 1 |
| | 10 m wind U10 | 2 |
| | temperature | 7 |
| Satob | wind U | 1 |
| | 10 m wind U10 | 2 |
| Dribu | wind U | 1 |
| | 10 m wind U10 | 2 |
| | geopotential | 8 |
| Temp | wind U | 1 |
| | 10 m wind U10 | 2 |
| | temperature | 7 |
| | specific humidity | 19 |
| Pilot | wind U | 1 |
| | 10 m wind U10 | 2 |
| Rad | radiance R-1C | 13 |
| PAOB | none | |
| Scatt | none | |
| Limb sounder | none | |
| none | none | |
| none | none | |
| Radar | none | |
| none | none | |
| none | none | |

Tab. 5.8 – Analyzed variables per obs type and rank in the NOTVAR table. These choices are similar to the Arpège 4dvar oper assimilation, plus the inclusion of 2m observations (T and RH). SEVIRI radiances fall into the general radiance type.

## 5.3 How to check for the execution ?

There are a number of useful recipies for checking the result of your analysis. Mostly, we will concentrate on the interesting features that are readily obtained from the return listings of screening (configuration 002/001) and minimization (conf. 131) :

– completion of the execution successful : grep for 'END CNT0'
– completion of the model setup successful : grep for 'END OF SETUP'. Before this line, you can find all the printouts from your model setup. Below this line, the printouts of the execution of any configuration do start. For screening and minimization, the NOTVAR tables are repeated just below, which can be useful.
– list of the identification numbers of satellites in your database : grep for 'ISATIDLIST
– check for the proper execution of the configuration, with an emphasis on each step of the algorithm : grep for 'STEPO'. You will then see all the calls to STEPO, which is the "toolbox" of the model, and its associated 9-character control string. The 9-character control string tells you which parts of the model were activated in STEPO. Their meaning is briefely recalled in Table 5.9 :

| position | function | usual values encoutered in e002 or e131 |
|---|---|---|
| 1 | input/output and incore read/write | A (write FA file), 0 (trajectory module!!) |
| 2 | spectral North/South inverse transforms | A, G (SPA*/GMV, GMVS, GFL),<br>B, K (SP5A*/GMV5, GMV5S, GFL5) |
| 3 | spectral East/West inverse transforms | A (SPA*/GMV, GMVS, GFL),<br>0 (N/S continued) |
| 4 | model gridpoint computations | A (full model), S, T (predictor/corrector) |
| 5 | gridpoint projections | L (localize norm) |
| 6 | observation operators | V (basic call to obs op.) |
| 7 | spectral East/West direct transforms | A (SPA*/GMV, GMVS, GFL) |
| 8 | spectral North/South direct transforms | A (SPA*/GMV, GMVS, GFL) |
| 9 | model spectral computations | A (full model), T (pred./corr.), I (DFI) |

Tab. 5.9 – 9-character string CDCONF.

– spectral $L^2$ norms of the data stored in the SPA* arrays : grep for 'SPECTRAL NORMS', and check for 'START CNT4', 'START CNT4TL', 'START CNT4AD', to locate the corresponding model
– check for the correctness (at least the physical dimensions) of the observation equivalents : grep for 'RMS OF GOM-ARRAYS'. You will see, below that line, the vertical profiles of root mean squared model values, horizontally interpolated to the observation points. However, the vertical profiles are still in model levels (vertical interpolations have not yet taken place). Thus, the printout in the direct model validates the horizontal interpolations. The profiles are repeated for conventional upper-air data, surface data and radiance data.
  The same printouts are also performed in the adjoint model, twice. The first time, you will again see the rms of the direct interpolated model values (trajectory). The second time, you will see the rms of the gradients. You need then to remember that the gradients are obtained after the adjoint of the vertical obs operators, but before calling the horizontal adjoint code. Thus, the second rms printout validates the adjoint of the vertical operators.
– check the results from screening : grep for 'SCREENING STATISTICS'. Below this line, you will find a few useful tables which summarize the behaviour of the screening of your observations. The tables are ordered per observation type, per variable field, per screening event (reasons for rejection) and so on.
– in the listing of screening, search for the GREPGEOM string. If present, the text will indicate observations which were found outside of C+I (your job must have crashed later on ...).
– the final values of the $J_o$ can be seen, stratified by obs type and subtype, in the final

printout of the "Jo-table". For this, simply grep for 'Diagnostic JO-table'. In screening, you will find only one such printout. In the minimization, you will find two diagnostic printouts of $J_o$ : one before and one after the minimization. You might check that the ratios $J_o/P$, where $P$ is the number of observations under consideration, are always close, or usually below, 1. Ratios of $J_o/P$ of about 10, 50 or 100, or reversely of about 0.1, 0.01, should all be considered as suspicious.

- the evolution of the cost functions, and of the gradient norm reduction can be seen by the following greps : grep for 'GREPCOST' to list the evolution of $J_b$ and $J_o$ with iterations, and grep for 'GREPGRAD' to visualize the evolution of the norm of the total gradient.
- if the test of the gradient is asked, then the results can be checked by grepping for 'GRTEST gradient T1'. Around this line, one will also find more information about the local topological properties of the total cost function. However, the most important parameters to check for are the continuity and the verification of the first-order Taylor expansion :

$$T1 = -\frac{J(x) - J(x + \delta x)}{< \nabla_{\delta x} J, \delta x >}$$

Using LOBSTL=.TRUE., that is the "exact" TL obs operators, this test should be very accurately fulfilled in the proper range of magnitude for $\delta x$, by up to 5-6 digits.

- the final information tells you about the overall effieciency of the minimization. Just before 'END CNT0', you get the information about the ratio of final to initial cost function, as well as the ratio of the final to initial gradient norm. Just grep for 'FINAL GRADIENT NORM SQUARED' and for 'RATIO FINAL/INITIAL'.

## 5.4 Frequently encountered ... crashes

Since we all started as beginners one time, here are the most trivial crashes one could encounter, and that should be fixed "easily" :

- crash with "BUS ERROR" in READOBA, called by OBADAT : this is actually the very first place where the IFS tries to read the ODB database in screening and minimization. If you fail here, either the ODB library was not loaded in your executable (in this case, you should also have "GETDB should not be called" and "ABORDB should not be called" messages in your standard output), or the ODB base is corrupted (try to read it separately with MANDAODB for instance)
- call abort in GEFGER : typical place where the screening stops if you have data outside the C+I zone in your ECMA_LAM database (grep for GREPGEOM to verify that this is indeed the case)
- call abort in ECHIEN : incompatible geometries between some of your input fields (climatology versus first guess, or namelist versus first guess)
- call abort in EDOG : incompatible geometry between the "stabal96" files and the assumed geometry in e131, as read from the first guess. This message is always right. It means that you either need to adapt your first guess domain (by fullpos or a new e927/forecast procedure), or you should change the "stabal96" if available on the new domain.
- zero iteration in the minimization, $J = 0$ from scratch and most likely a floating point exception (division by zero) in PRTJO. Here, you probably also have NOBTOT=0 (that is, zero observations in your database). This uggly situation could be encountered if you have mixed up ECMA and CCMA databases, or the shuffle to produce the CCMA base failed. Another good reason for a null cost function (with NOBTOT>0) is a mismatch between the types and fields present in the database on one hand (as a result from LAMFLAG and screening, for instance), and the types and fields required in the minimisation (NOTVAR array) on the other hand.
  Reversely, it could happen that all observations, kept or rejected by the screening step, could be used in the minimization. The latter would be the case if somehow your ODB pointers in the script indicate ECMA adresses instead of CCMA.
- zero data in ODB pool : having no observation in a pool is not supported by the ODB. This produces a crash in routine MAKE_TIMESLOT_INDEX

# Annexe A

# Configurations of the Arpège/Aladin executables

| Conf number | Action | arp or ald only (default : both) |
| --- | --- | --- |
| 001 | forecast | |
| 001/002 | screening and trajectory | |
| 131 | variational minimization | |
| 2xy | barotropic model | arp |
| 401 | test of the adjoint | |
| 501 | test of the tangent linear | |
| 601 | singular vectors (Lanczos algo.) | |
| 701 | optimal interpol. (CANARI) | |
| 801 | gradient comp. (with resp. to IC) | |
| 901 | grib to FA conv. | |
| 911 | | |
| 923 | Arpège/Aladin clim file prod. | |
| 927 | Full-Pos | |

TAB. A.1 – List of existing configurations in Arpège/Aladin

# Annexe B

# Domain definitions in Aladin

Following the ideas of Machenhauer and Haugen (1987), the gridpoint domain of Aladin is separated into three functional areas :

– the Central area where the pure gridpoint dynamics apply (”C“ zone)
– the Intermediate area where the Davies relaxation applies (”I“ zone)
– the Extension area where the biperiodisation of the fields applies, to ensure the proper bi-Fourier transformations (”E“ zone)

This is further illustrated in Figure B.1. In short, the Aladin topology is equivalent to a torus.

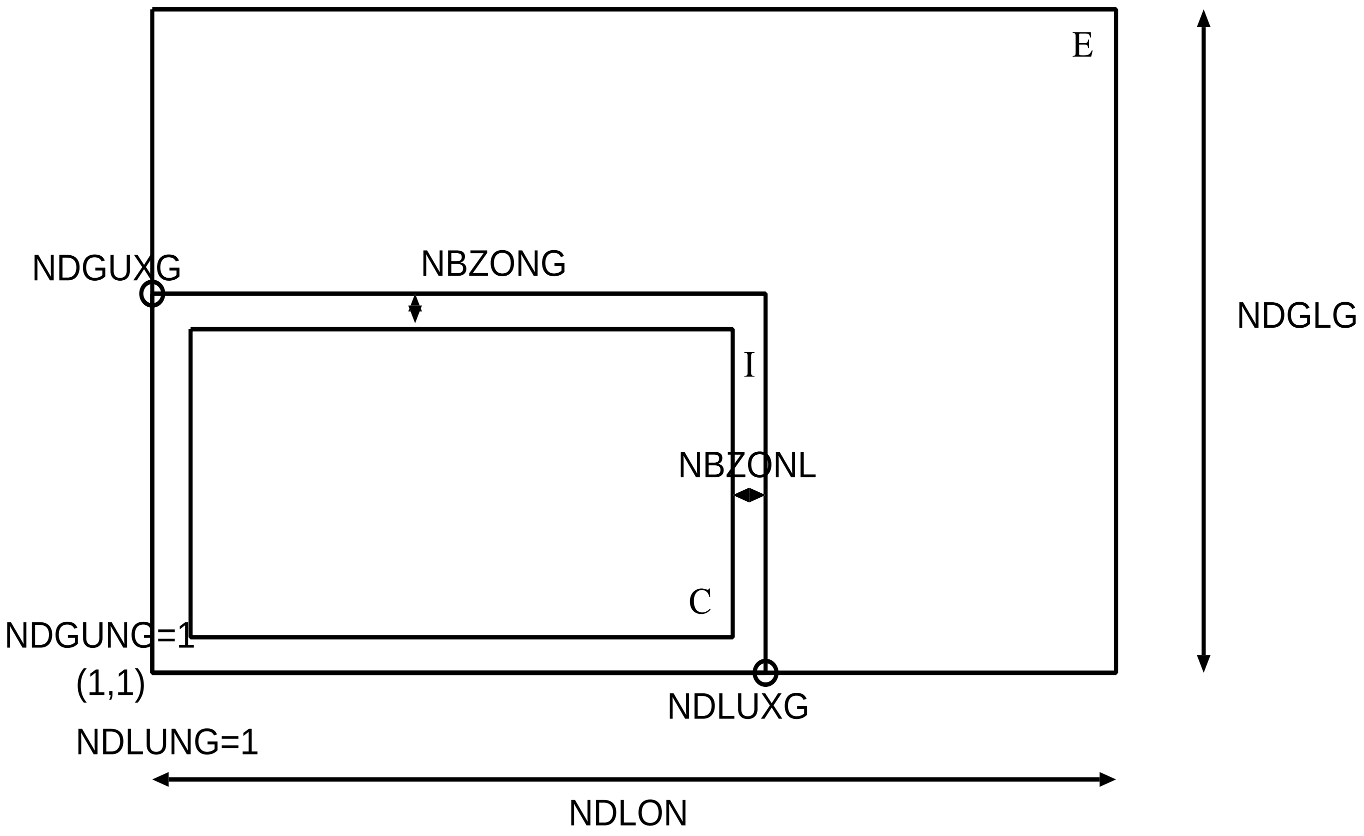



Fig. B.1 – Gridpoint domain definitions.

# Annexe C

# Communications between observation and model spaces in DM mode

A key routine to set the communications between data distributed in observation space and data distributed in model gridpoint space is the setup routine MKGLOBSTAB. This is where you define the mapping that tells which obsveration positions belong to different processors, and therefore need to be communicated from one proc to the other in order to compute the $y - H(x^b)$ departures (as well as the associated sensitivity in the adjoint).

For more details, let us cite Mats Hamrud : *before CY30, each processor had a global view of the observations and found out which of all the observations overlapped with your own processor's gridpoint area. Now instead, each processor only has a local view of the observations and finds out which processors need to do the interpolation of the model equivalent for its observations. Then it proceeds to "tell" that processor what to do (the lat,long to interpolate to, the observation type etc...). The main concern is to make sure each observation is accounted for once and only once through the use of the correct ¡ or ¡= and possibly epsilons.*

*The adjoint part is more complicated : we need a halo of observations (that's where SLCSET comes in) and we can no longer protect against double counting by simple checks ( an observation can be in the extended area of many processors e.g 4 in a "corner"). What will protect us in the end from "double counting" is that, after the adjoint of the interpolation of observation equivalents, we throw away what is in the halo and retain only the contribution at core points.*

*There is actually a piece of unsafe code in MKGLOBSTAB that I intend to correct in cycle 30. As we don't know in advance how many instances of double counting there will be, the array IPROCPROFAD is arbitrarily dimensioned to NOBPROFG. This has proved not to be enough for some artificial cases (running with one obs on one processor etc...) and we need to extend it and put in a check that we are not going out of bounds.*

# Annexe D

# Aspects of the data distribution on MPP machines

This appendix deals with the description of the data distributed aspects, and the message passing elements. It is indeed helpful to have some basic ideas in mind, since the ARPEGE/ALADIN software is, from the coding point of view, rather in an intermediate stage : though some elementary message passing routines do exist, and could be used by any developper in a somewhat transparent way, the practical coding requires a good overview of the organisation of data. Therefore, the reader will find in the appendix a set of graphics which illustrate the data distribution in the code, in gridpoint space, in spectral space and for the control vector of the variational analysis.

For short, some general message passing aspects :

- the vertical covariance blocks of $B$ are diagonalized in distributed way (each processor diagonalizes some blocks). The corresponding setup is in SUEMPVAR for Aladin. The eigenvalues and eigenvectors are then message passed under SU(E)JBCOR, by the routine (E)GATHEREIGMD.
- the control vector is distributed in a way which mimics the data distribution of the spectral model buffer SPA3. In the module procedures SPECTRAL_FIELDS and CONTROL_VECTORS, generic f90 pointer structures are defined, which will receive the control vector data, along with the information about the data distribution. These modules duplicate some IFS code. Mostly, conversions between spectral state vectors and control vectors are done by a mere reorganisation of the coefficients, locally to one processor. Only DM global operations such as the computation of a scalar product between two CONTROL_VECTORS or the computation of the $J_b$ cost function involve message passing (using MPL_ALLGATHERV). The Aladin specific code is deeply inserted in the heart of Y. Trémolet's module procedures for Arpège/IFS. A correct knowledge of the distribution in spectral space is required to understand the details.
- In Aladin, there is a small contradiction between the distribution of the spectral space state vector (stratified along zonal waves $m$ from 0 to NMSMAX), and the distribution of the cost function elements (originally defined along total spherical waves $n$ from 0 to NSMAX). Thus, cost functions have been defined from 0 to NVARMAX, where NVARMAX is NMSMAX in Aladin and NSMAX in Arpège. This allows for a proper distribution and collection of cost function elements from the spectral state vector SPAx. This problem concerned both $J_b$ and $J_c$ until AL25T2, and now only concerns $J_c$ since AL26T1 ($J_b$ DM aspects are implicitly treated by the CONTROL_VECTOR approach).

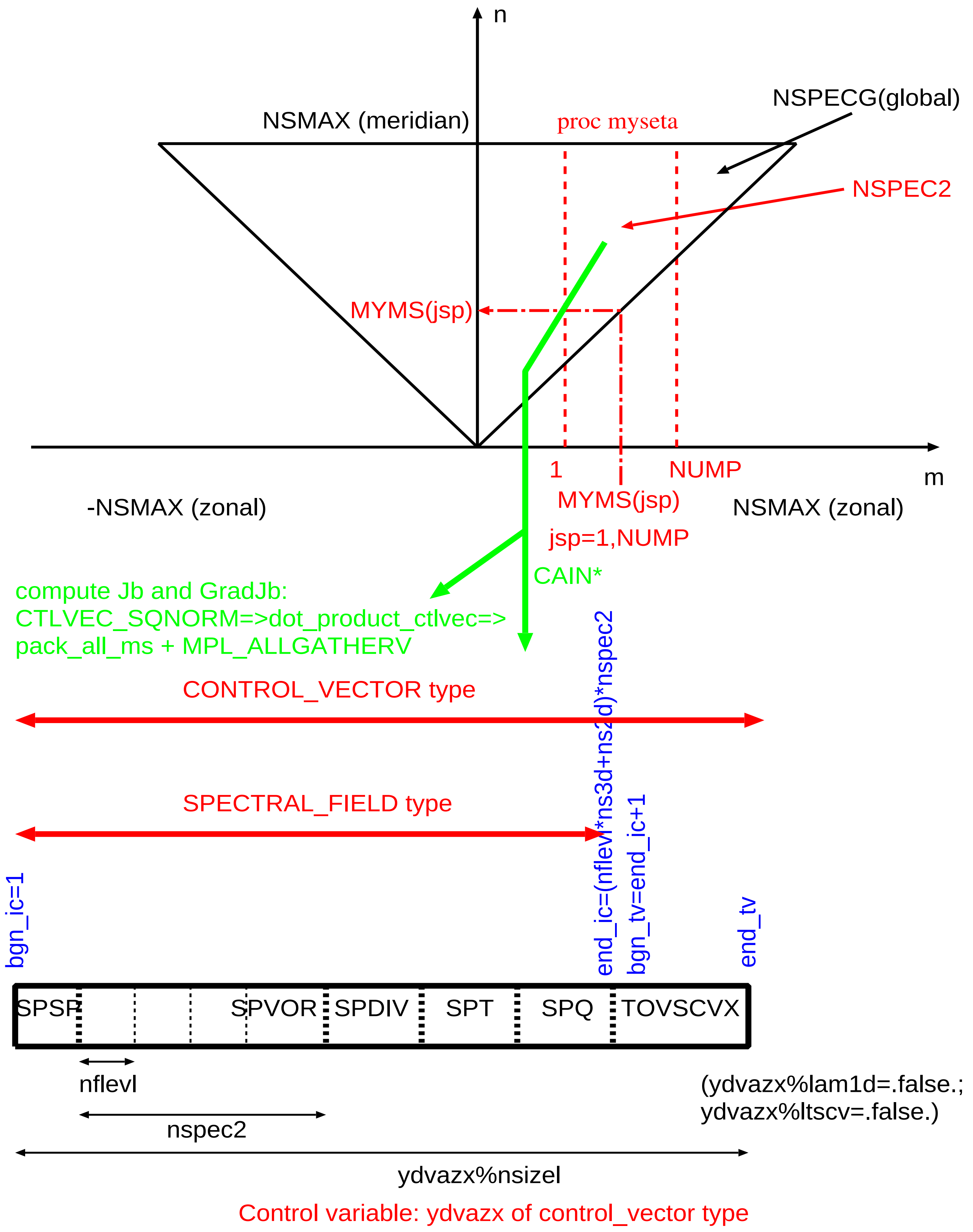



FIG. D.1 – Data distribution for canonical injections in Arpège.

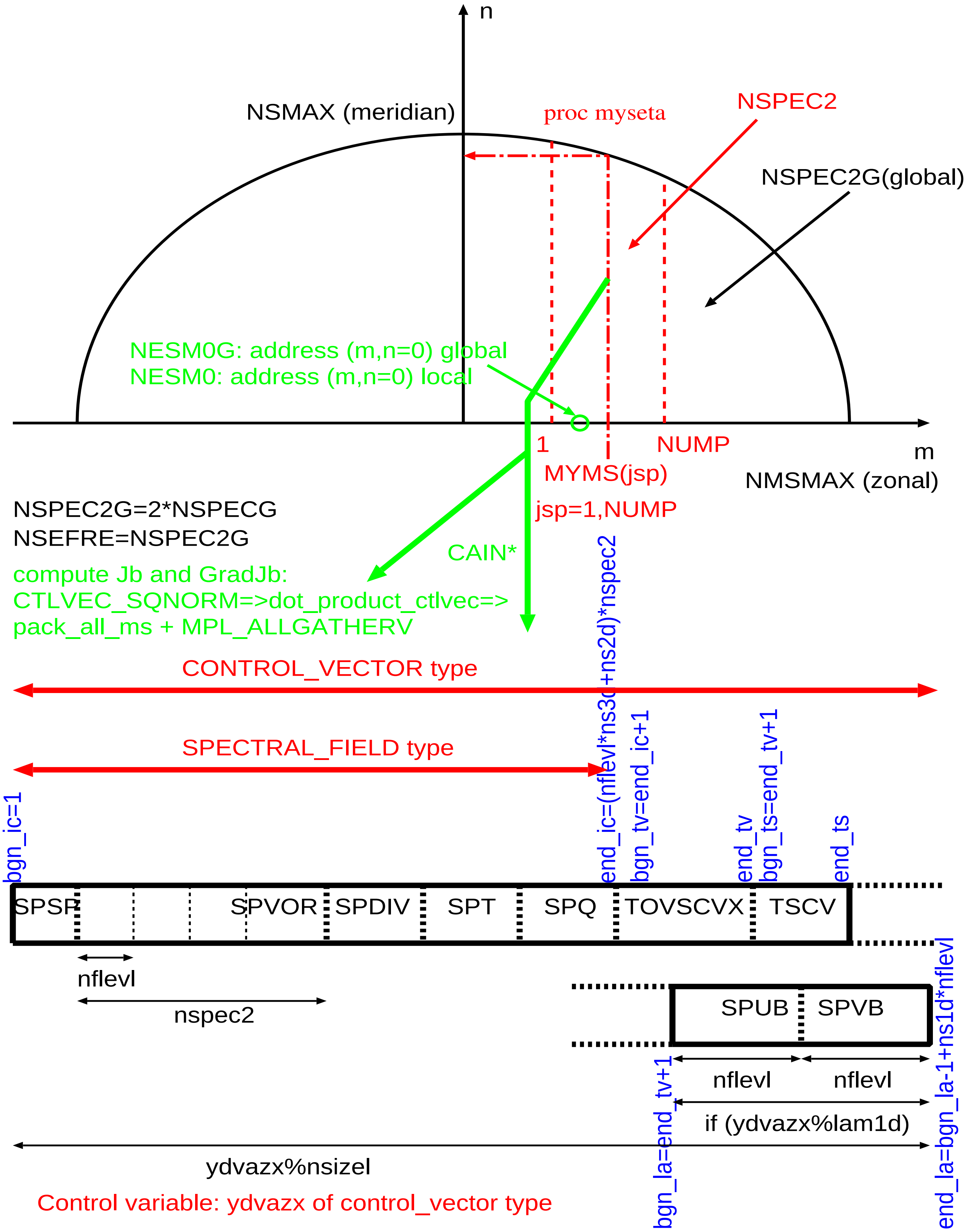



Fig. D.2 – Data distribution for canonical injections in Aladin.

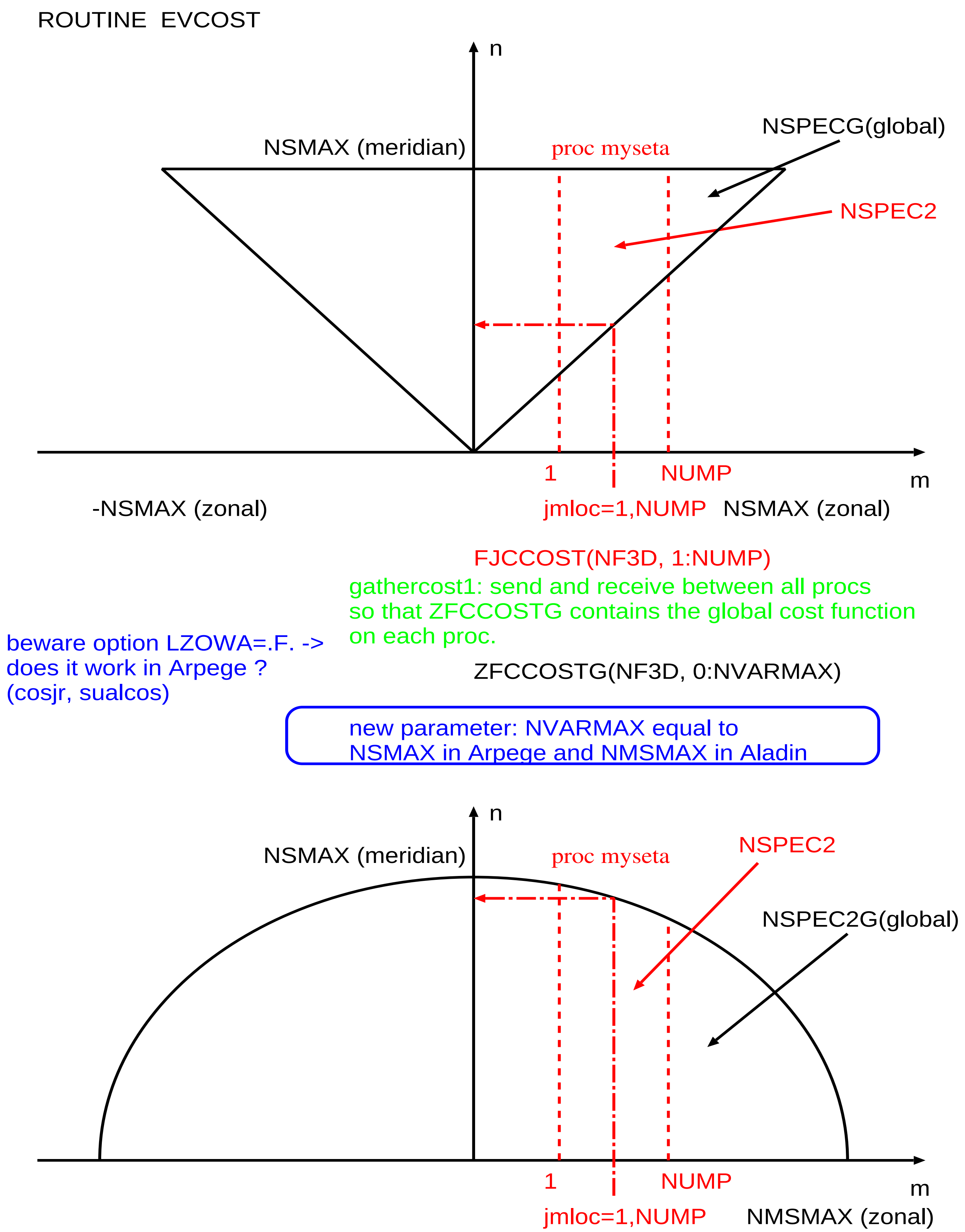
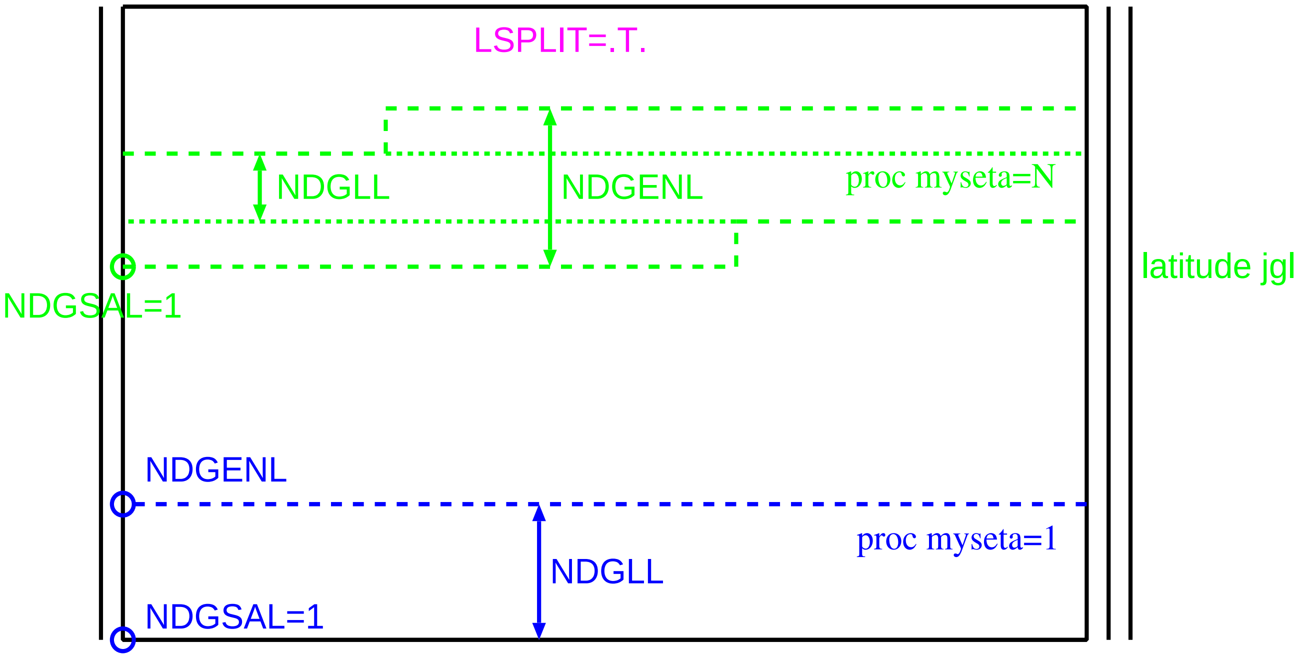



FIG. D.3 – Data distribution for $J_b$ cost function.




DM: GP area is cut into bands of latitudes in first level parallelization: for all jgl, NONL is equal to NDLON if LSPLIT=.F., otherwise latitudes may be uncomplete on the local proc

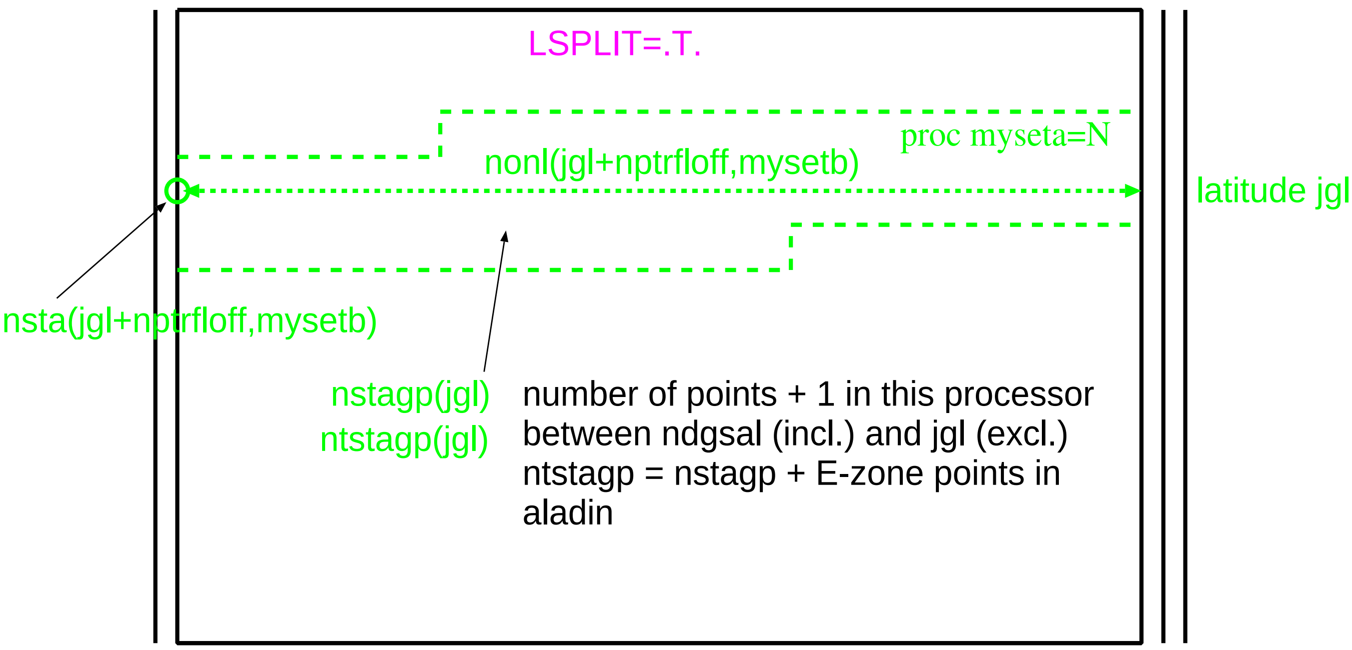



FIG. D.4 – Data distribution for latitudes in gridpoints.

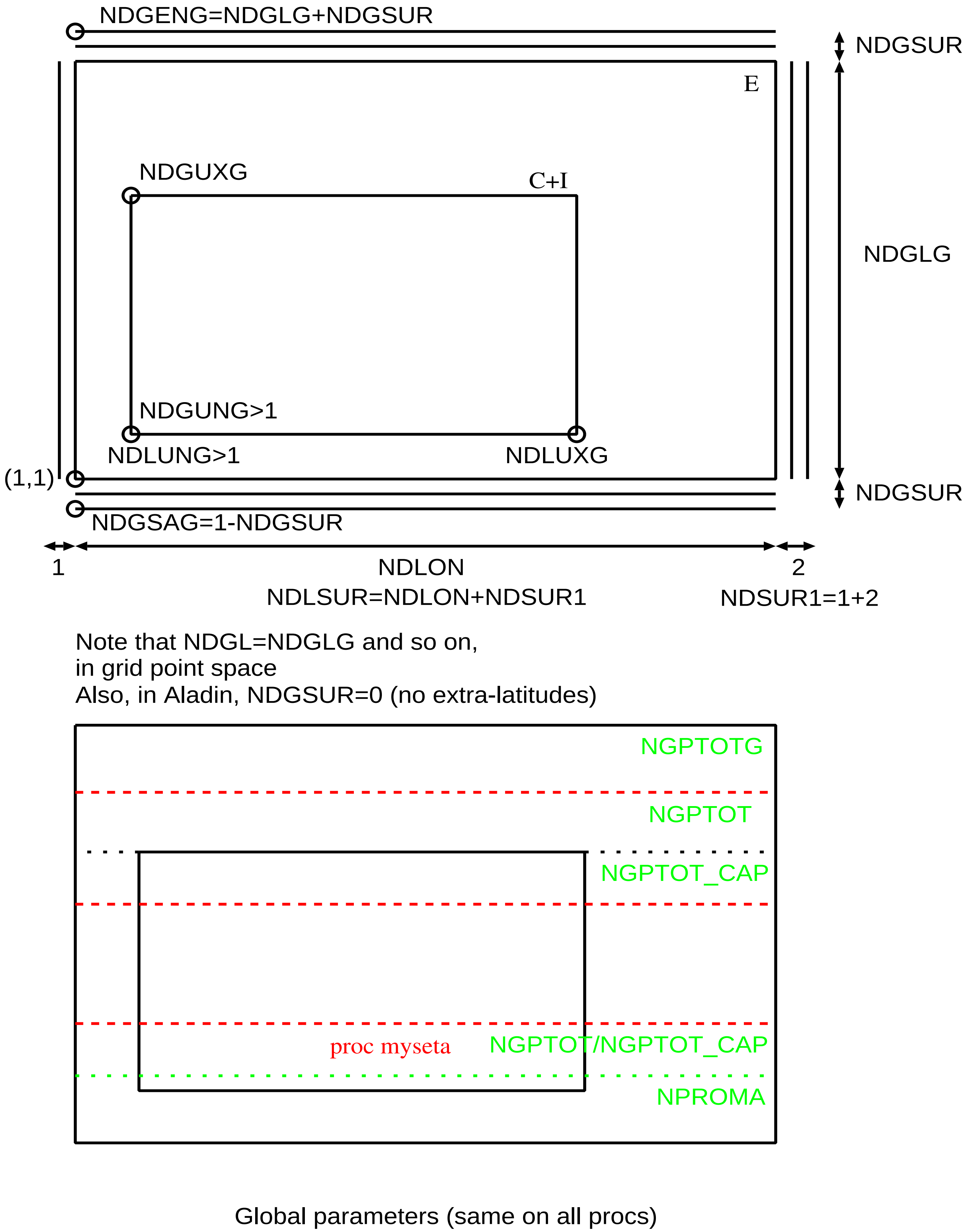
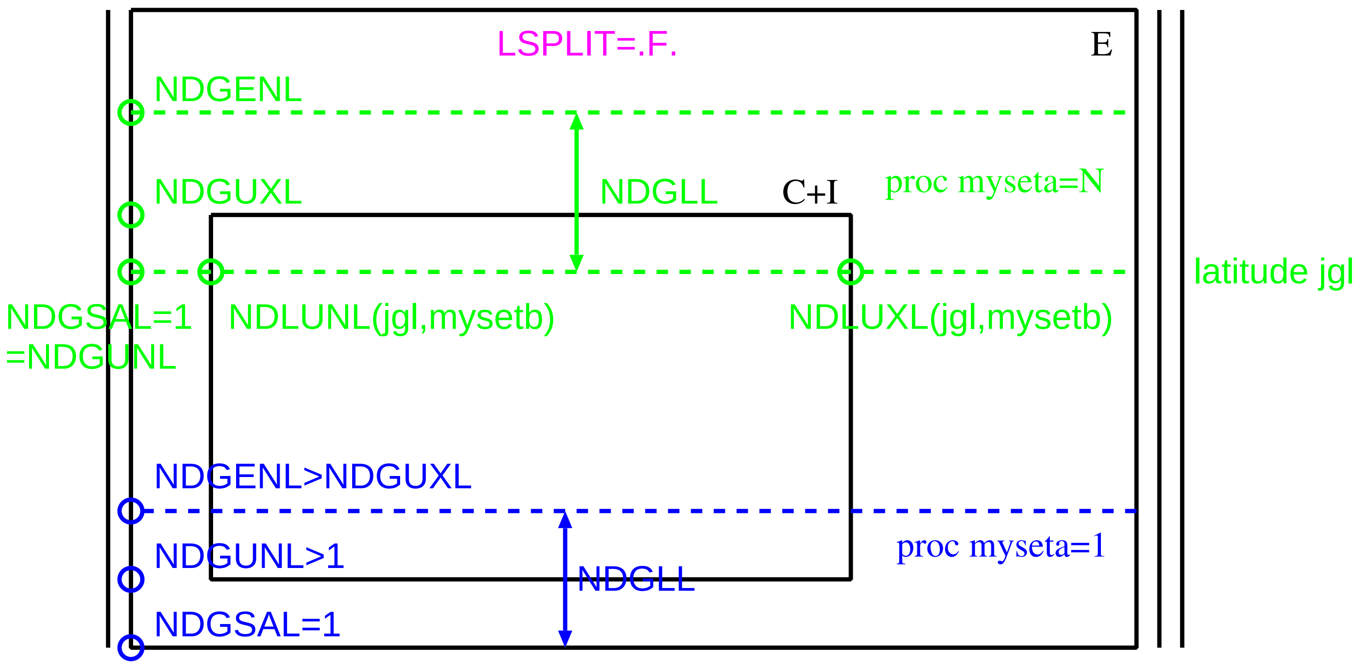



FIG. D.5 – Data distribution for C+I and E zones in SCAN2MDM.




DM: C+I+E is cut into bands of latitudes in first level parallelization (for all jgl, NONL is equal to NDLON):
- above are pathological cases (including E-zone)
- below is the usual case (including Full-Pos parameters)

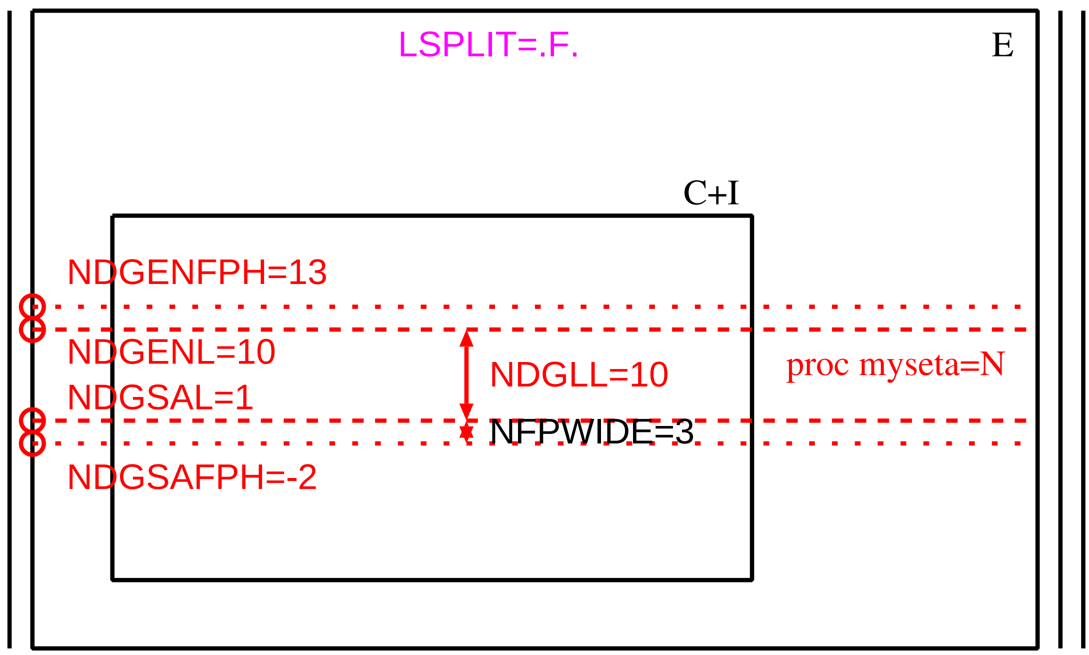



Fig. D.6 – Data distribution for latitudes in gridpoints (bis).

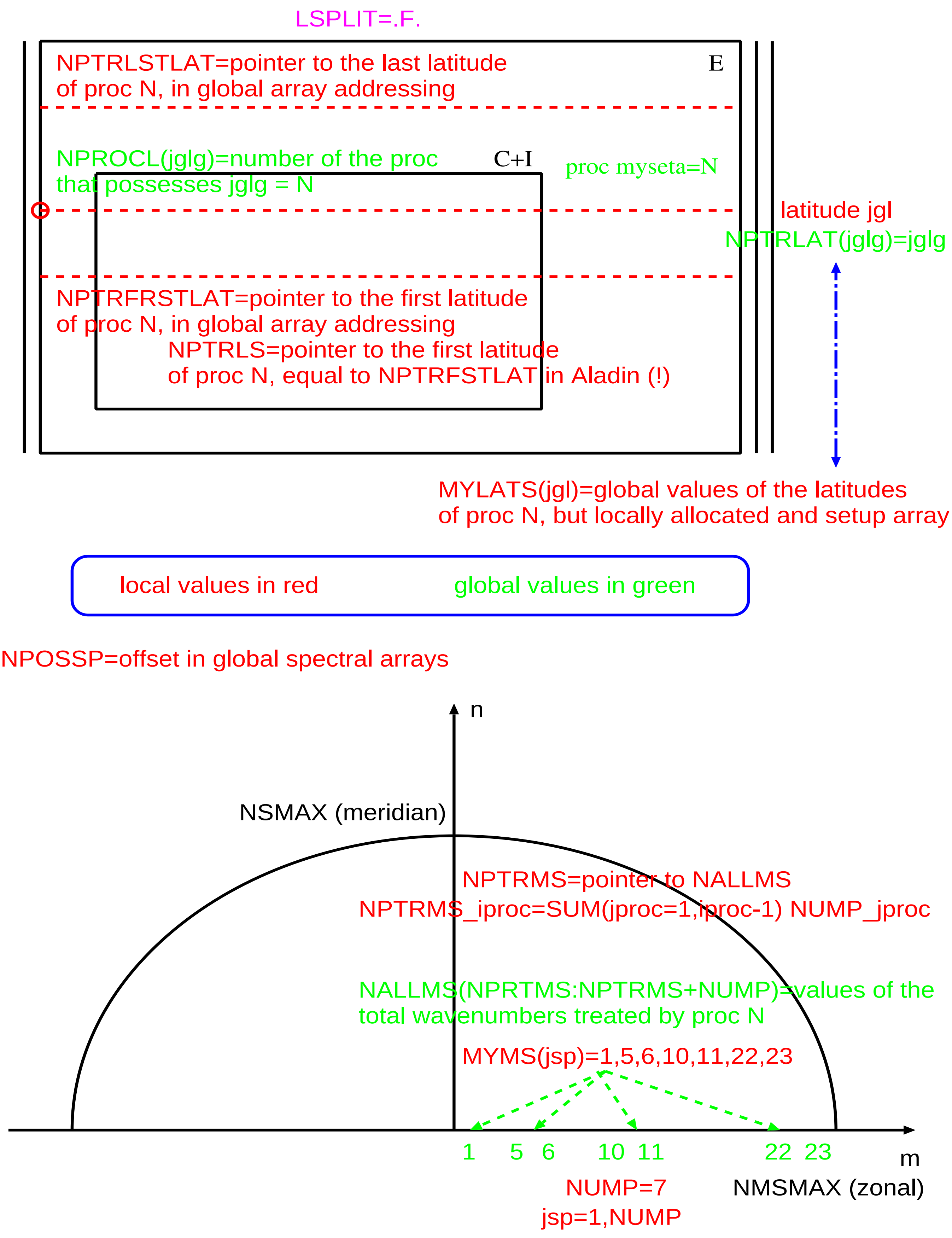



FIG. D.7 – Data distribution for full latitudes and zonal wavenumbers.

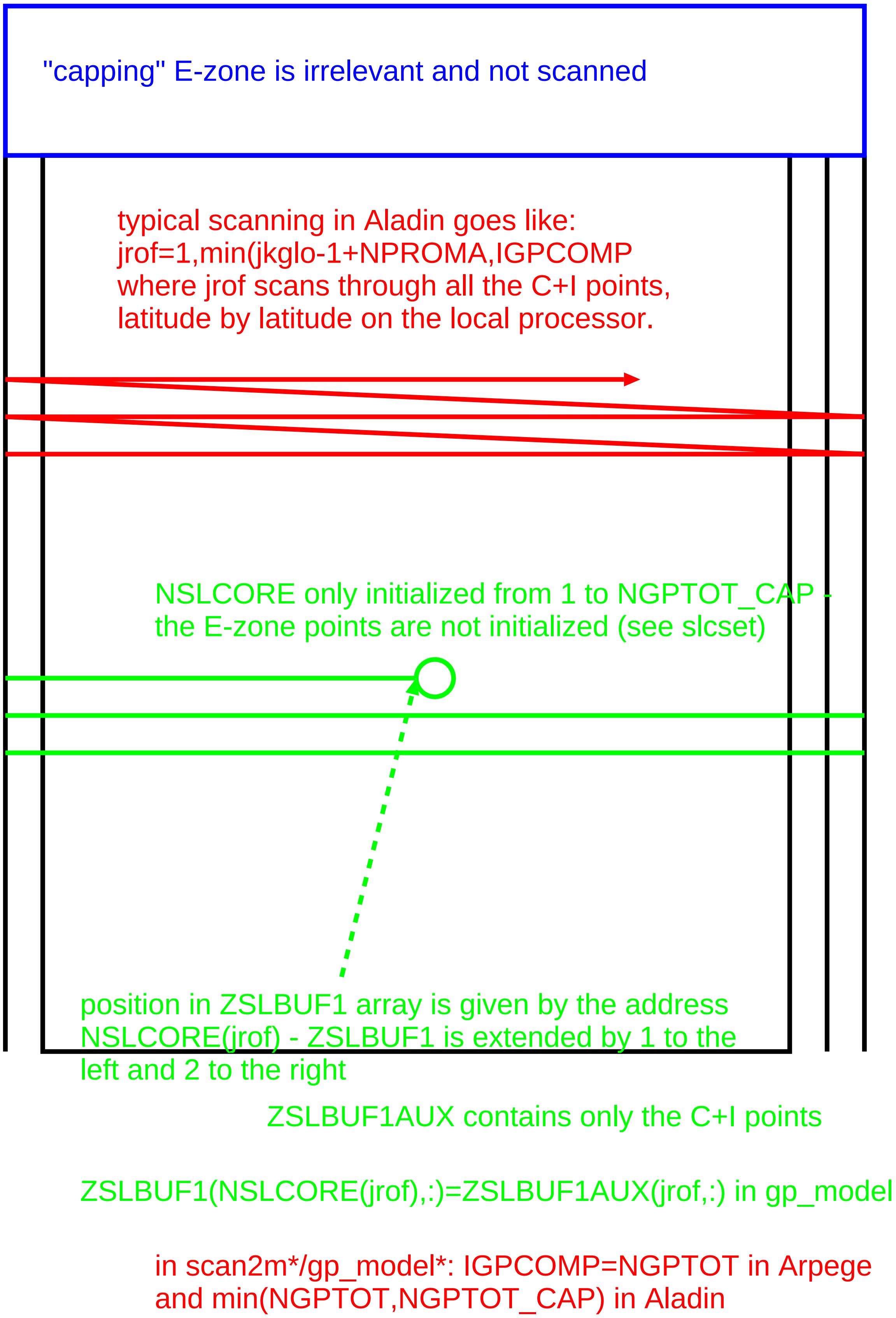



Fig. D.8 – Data distribution for SCAN2MDM parameters.

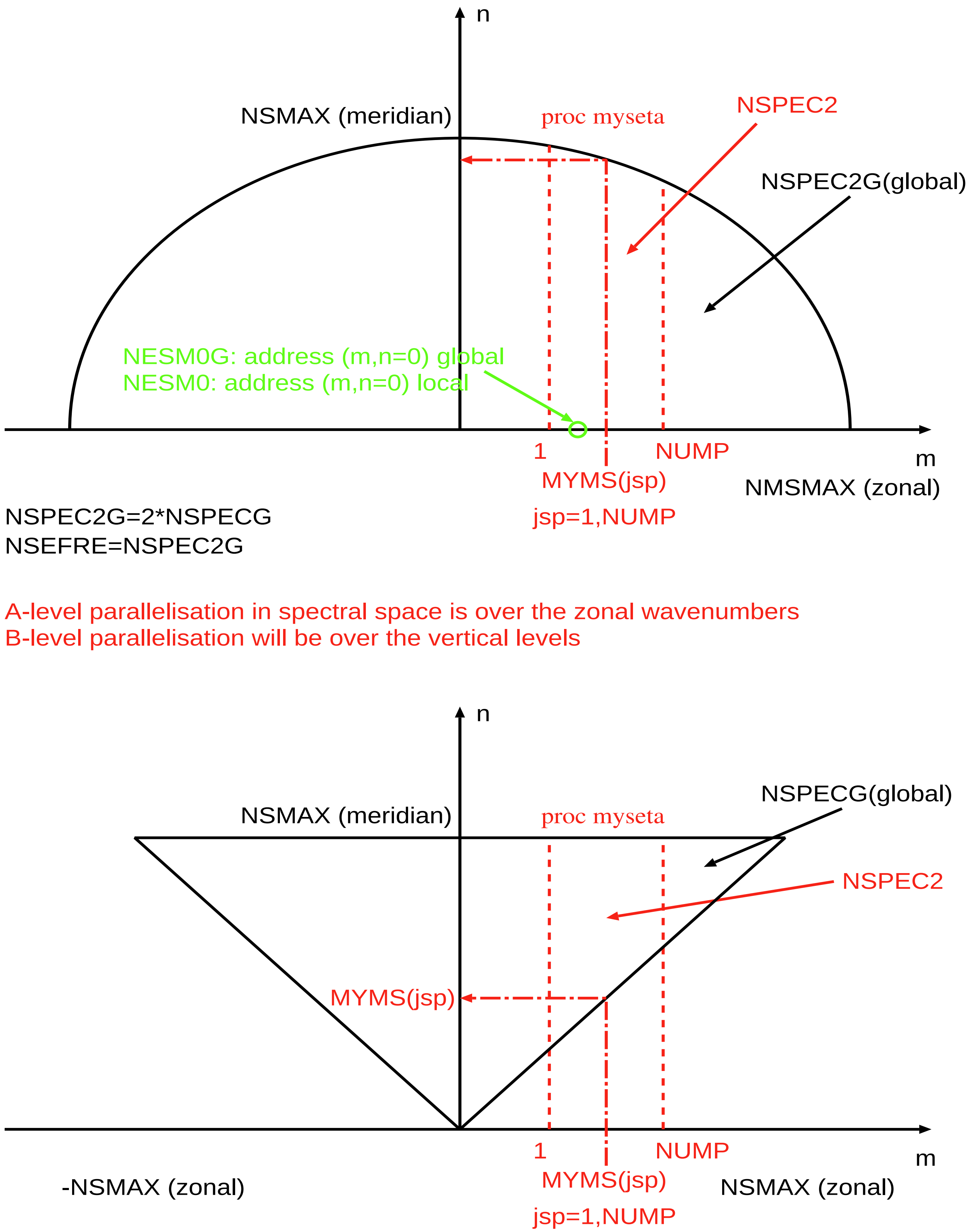



Fig. D.9 – Data distribution for spectral coefficients.

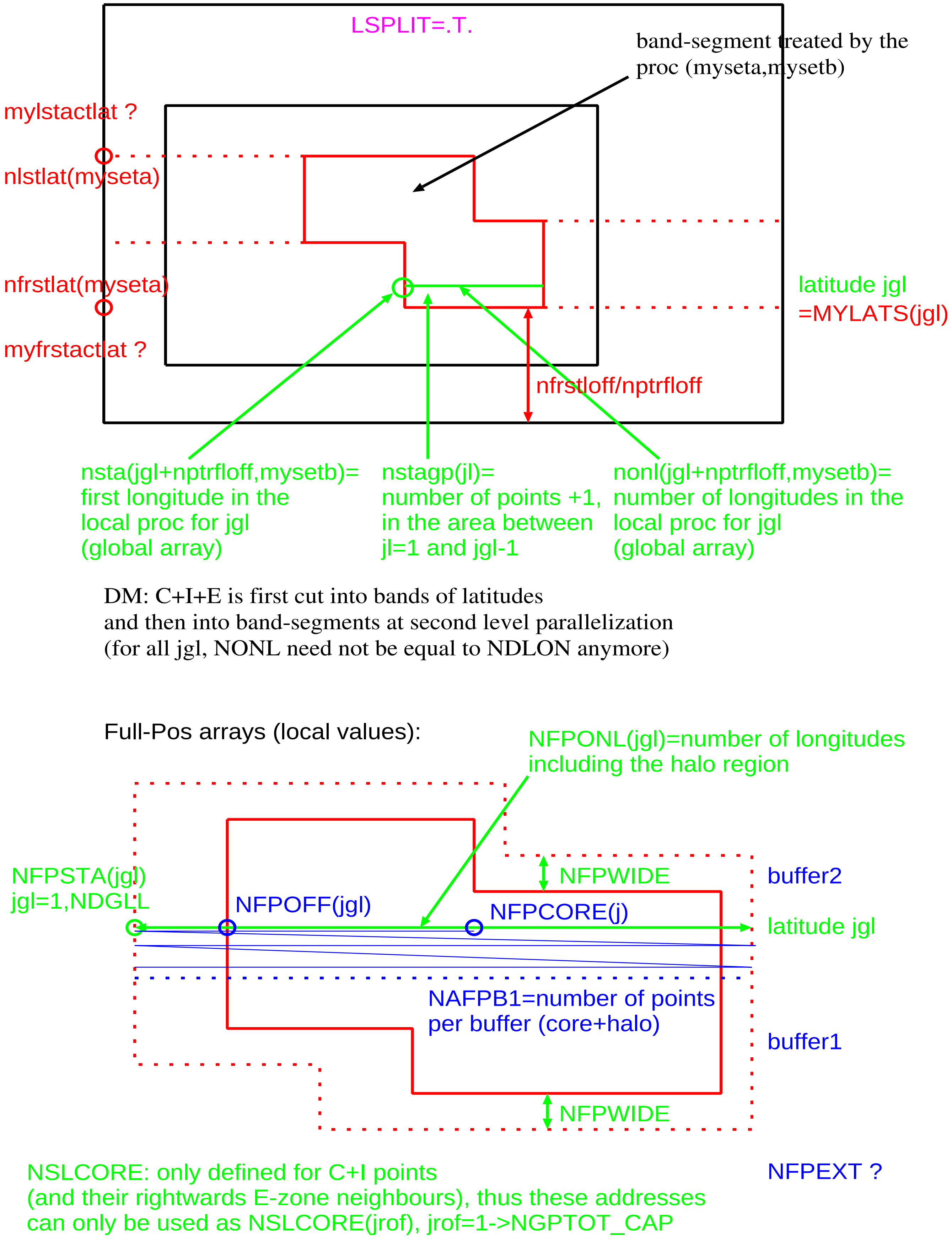



Fig. D.10 – Data distribution for halo in gridpoints.

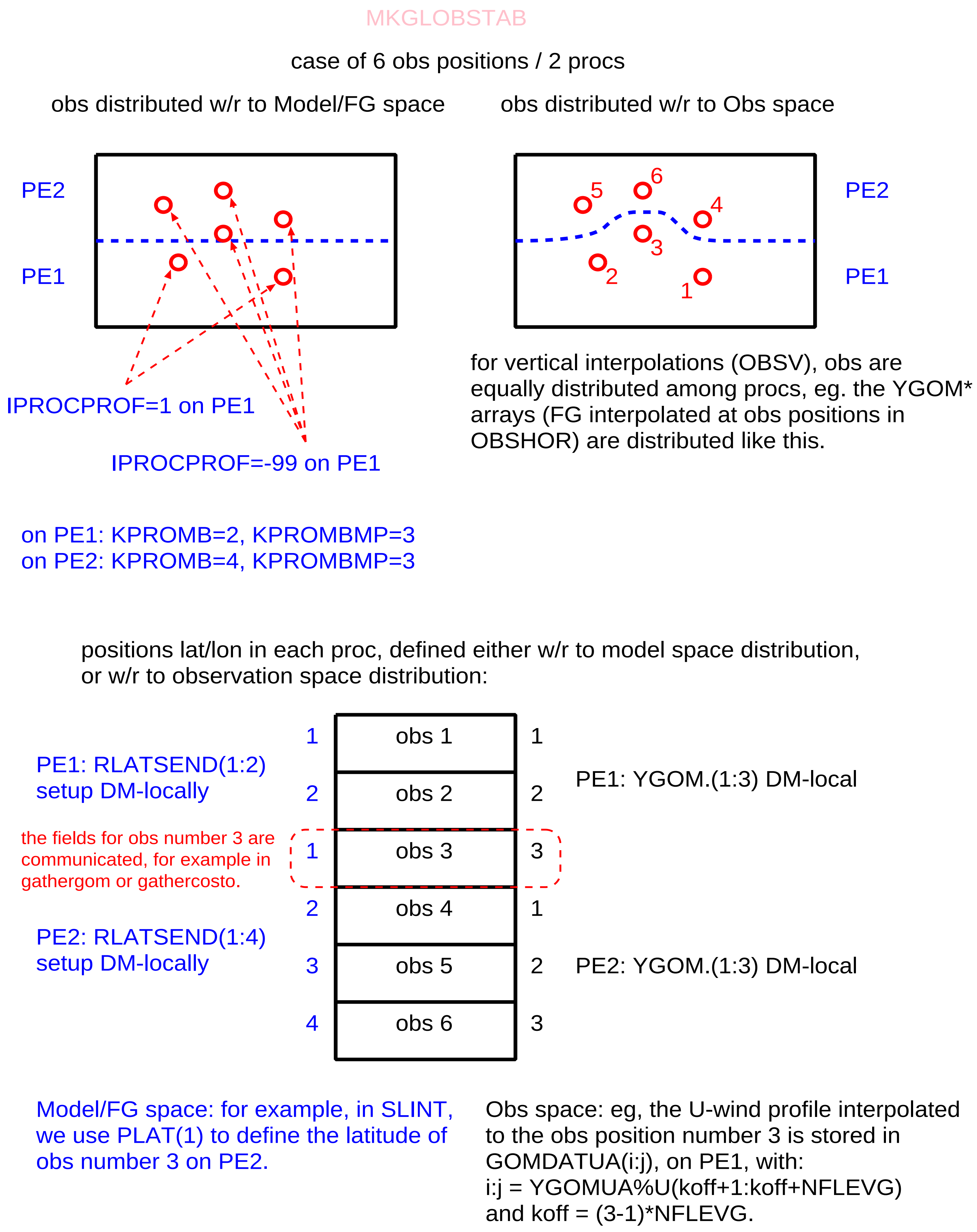



Fig. D.11 – Data distribution for observation space.

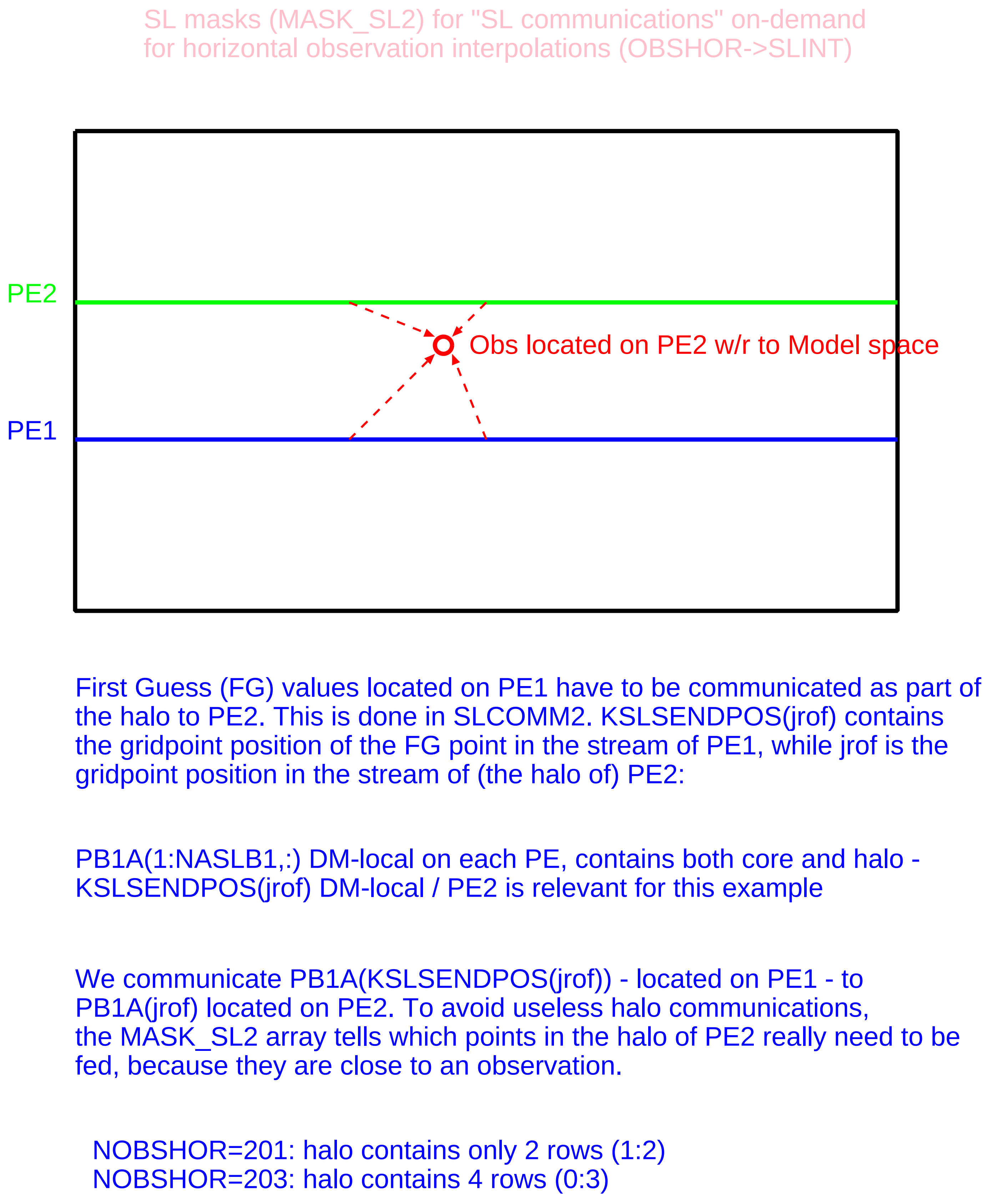



Fig. D.12 – Data distribution for observation communications.