Projections used in ALADIN & AROME limited area models (LAM)

Generality & description of the code.[#]

by Jean-Daniel GRIL - May 2009

[#] Translation by MF/CNRM/GMAP/JAM

CONTENT

1.Introduction

This document describes the Fortran package for the management of the "geometry" of the ALADIN & AROME LAM. This "geometry" is a projection of the earth on a plane with linked functions ensuring a smooth shift between representations.

A brief annal will recall few notions and formulae linked to used projections, followed by the general structure of the code, its constraints and its goals. The projections described here and available within the models are as follow:

- ✓ Polar Stereographic projection (SP)
- ✓ Conform Conical Lambert projection (LCC)
- ✓ Mercator projection (M)
- ✓ Mercator Rotated Tilted projection (MRT)

The conclusion will deal with the potentialities and limitations of this package and also the possible adaptation of the code to other languages.

2.Background

First in the ALADIN, then in the AROME LAM, the "geometry" was rewritten in 2002 and plugged in in cycle *al25t1bf.02*.This rewriting was led by several findings:

- ✓ As the ALADIN model had been operational for many years, every "open" options (rotation of the Pole, secant projections) associated to projections from the beginning of the model were reduced to most frequent cases.
- ✓ the reorganization of variables associated to the projection and to the features of the domain, in order to simplify and to minimize the number of errors due to redundant information (ERPK, LAT0, KGIVO, KSTROP,...)
- ✓ a more structured code (in the Fortran sense) as well as an object approach (attributes & method) have made it possible to erase the "COMMON" notion and to entirely externalize the package from the model, so as to be used separately.
- ✓ To eliminate some impossible cases, more numerous safety checks were added (overlapping domains along the LCC cone split, choice of the polar projection...)
- ✓ as the code is free from the one of the model, it can be used within a toolbox prior or after the model run without using the model libraries, and also on multiple platforms (saving space, lightweight, portability)

In 2004, with cycle *cy29t2_main.01*, a new kind of projection was added (a Mercator projection generalisation) called Mercator Rotated Tilted projection (MRT).

3.Formulae and Reminder about the used projections:

3.1.Reminder:

The earthly projections are made either on a plane (a) or on simple shapes (cones, cylinders) (b,c) which are, should there be a need, cut up (vertical dashed lines) and spread over as sections of the plane (plane less an angular sector (b), plane strip (c)).

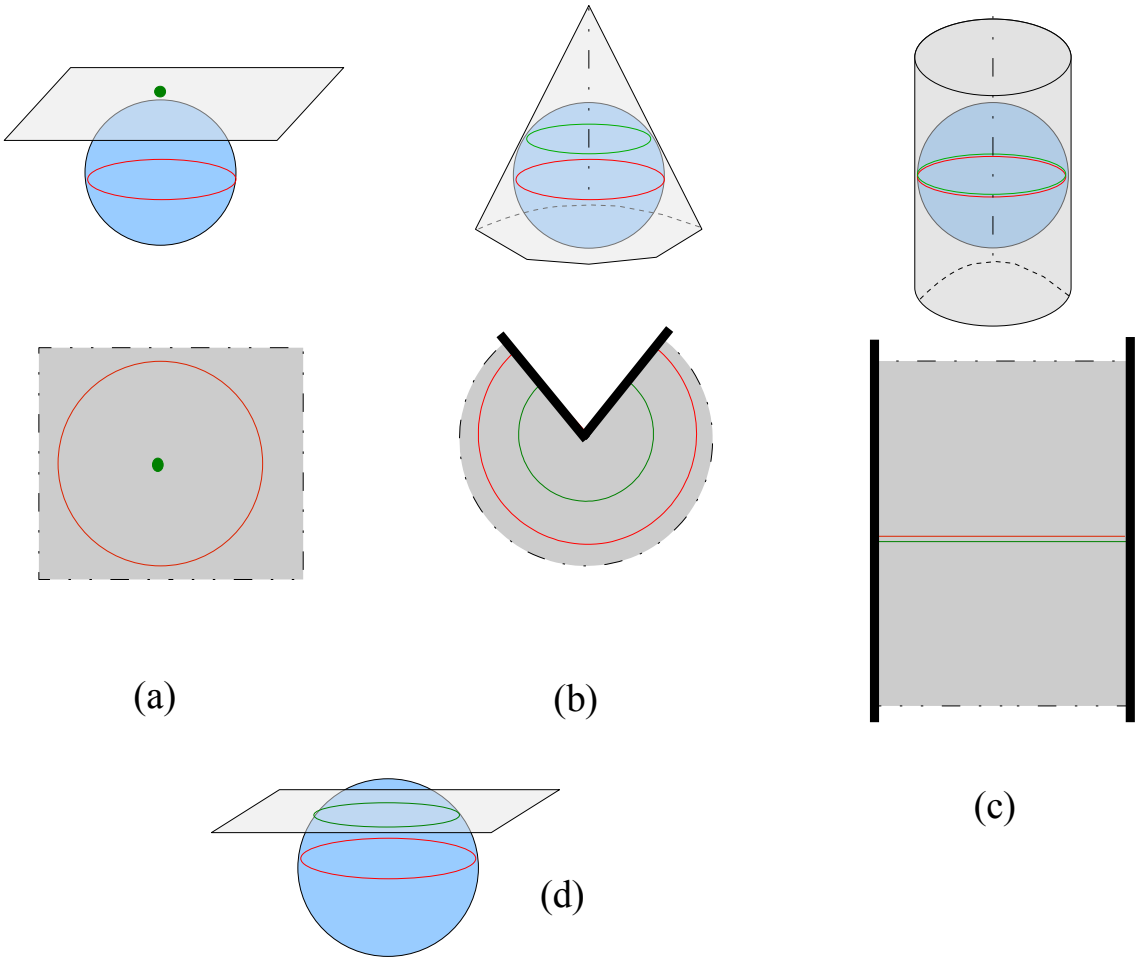


Illustration 1: Diagram of projections

The above projections are boundless where the outline is dashed and limited where there is a solid bold black line. The red line is either the equator or its projection, the green line is where the sphere and the shape used as support for the projection merged. The map factor = 1 at these meeting points.

A map factor is defined as the ratio between the distances on the projection plane to the same distances on the sphere. From the above examples, for tangent cases (a,b,c), the map factor is superior or equal to 1; for secant cases (d) the map factor may be either inferior or superior to 1 depending on one's position in relation to the cross lines.

Tangent cases occur when the support of the projection plane is tangent to the sphere (a,b,c). *Secant cases* occur when the shape crosses the sphere (d).

No mention will be made of the MRT projection since it is only a Mercator projection (M) where by the earth has rotated from a coordinate point (λ,φ) to (0,0), this is the Rotated part of MRT. The MRT Tilted part is a second rotation along a perpendicular axis to the Poles ones and going by coordinates (0,0). The Mercator projection is therefore made after this shift (Rotated) and rotation (Tilted) of the sphere.

The cross section figures from Table 1 (below) shows that, for secant and tangent cases, several types of projection are available.

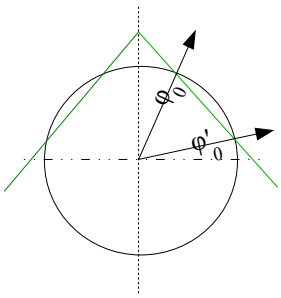
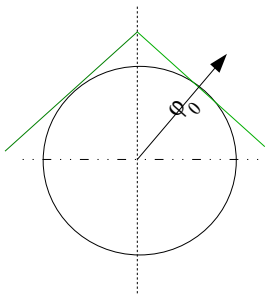
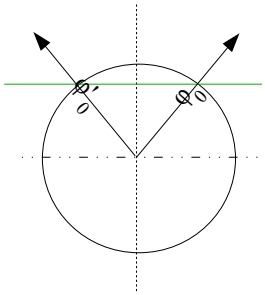
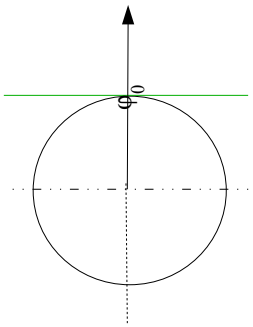
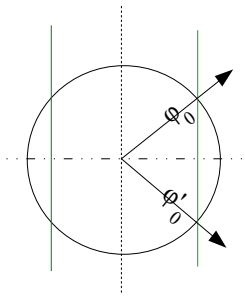
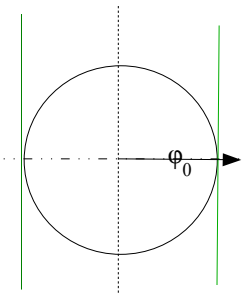
Projections	Secant case	Tangent case
Conform Conical Lambert	 $Kl = \frac{\ln\left(\frac{\cos(\varphi_0)}{\cos(\varphi'_0)}\right)}{\ln\left(\frac{\tan(\pi/4 - Pole.\varphi_0/2)}{\tan(\pi/4 - Pole.\varphi'_0/2)}\right)}$	 $Kl = Pole \cdot \sin(\varphi_0)$
Polar Stereographic	 *Kl = 1*	 *Kl = 1*
Mercator	 *Kl = 0*	 *Kl = 0*

Table 1: The projections

These can be deduced from the latitudes where the map factor = 1 (cross section point: 2 latitudes φ_0 et φ_0'; tangent point: 2 merged latitudes φ_0). If:

$$\left|\frac{\varphi_0+\varphi_0'}{2}\right|=90°\rightarrow projection\ SP$$

$$\frac{\varphi_0+\varphi_0'}{2}=0°\rightarrow projection\ M$$

$$\left|\frac{\varphi_0+\varphi_0'}{2}\right|\in]0°,90°[\rightarrow projection\ LCC$$

Kl (see the above table for formulae) is another parameter that can be deduced from the 2 preceding latitudes and that can be found in the projection formulae. To be noted that:

- ✓ KL = 1 for a SP projection
- ✓ KL = 0 for a M projection
- ✓ Kl $\in]0,1[$ for a LCC projection

Only 2 out of 3 of these parameters (φ_0, φ_0', Kl) are enough to define the type of projection. For the LCC case, it is better to know φ_0 et φ_0' (Kl can be easily deduced) than Kl and one of the two latitudes (φ_0 or φ_0') because the computation of the other latitude (either φ_0 or φ_0') is neither easy nor always possible.

Unfortunately, in the first version of the geometry package of the ALADIN model, the kind of projection was defined only by Kl et φ_0, which could lead to dead lock[1] inside the model external software using φ_0 et φ_0' . Kl is more often than not a kind of projection private attribute.

A new geometry package has been written to avoid this kind of occurrence. From now on, only projections in tangent mode will be used. φ_0 by itself fully quantifies the projection:

- ✓ $|\varphi_0| = 90°$ => SP
- ✓ $|\varphi_0| = 0°$ => M
- ✓ $|\varphi_0| \in]0°,90°[$ => LCC

The geometry package is mainly used to shift from «longitude,latitude» (λ,φ) coordinates on the sphere to Cartesian coordinates (x,y) on the plane and conversely. Other parameters on each grid points, besides latitude and longitude values, are of interest for meteorological models. These are the scale ratio and the "compass"[2]. The geometry package contains functions that enable these computations.

3.2.Formulae:

Table 2 shows all the projection formulae, for general cases (secant) as well as for specific ones (tangent) where Kl=sin(φ_0).

For SP and LCC projections, the equations are identical but for the value of Kl and/or φ_0. The map factor formula is the same for the 3 projections and will be used in a simplified version depending on the projection.

---

1 For a tangent LCC projection, compute Kl=sin(φ_0) on a pocket calculator (precision 10^{-6}) in order to initialize the parameters of the model. As the latter is running on a super computer (precision 10^{-15}), a secant LCC projection will be defined with φ_0 and φ_0' very close because of the precision differences between (Kl ≠ sin(φ_0)). More often than not, third part programs use φ_0 and φ_0'; they uselessly try to recompute φ_0' from the formula (table below), which results in a blocked situation: the model did run but data cannot be used.

2 Compass: sinus and cosine of the rotation matrix between the axes $(\vec{u},\vec{v})$ of the grid of the model and the geographical North. They can be used to compute the wind in regard to the North, knowing its coordinates on the grid $(\vec{u},\vec{v})$.

	Lambert	Polar stereographic.	Mercator
Sécant	$Kl = \dfrac{\ln\left[\dfrac{\cos(\varphi_0)}{\cos(\varphi_0')}\right]}{\ln\left[\dfrac{\tan\left(\frac{\pi}{4}-\frac{Pole\cdot\varphi_0}{2}\right)}{\tan\left(\frac{\pi}{4}-\frac{Pole\cdot\varphi_0'}{2}\right)}\right]}$	$Kl = 1 \quad , \quad \varphi_0 \neq \pm\dfrac{\pi}{2}$	$Kl = O \quad , \quad \varphi_0 \neq \pm\dfrac{\pi}{2} \quad , \quad \varphi_0 \neq 0$
	$R_{Eq} = \dfrac{R_T\cdot\cos(\varphi_0)^{(1-Kl)}}{Kl}\cdot\left[1+Pole\cdot\sin(\varphi_0)\right]^{Kl}$		
	$R_\varphi = R_{Eq}\cdot\tan\left(\dfrac{\pi}{4}-\dfrac{Pole\cdot\varphi}{2}\right)^{Kl} = R_{Eq}\cdot\left[\dfrac{\cos(\varphi)}{1+Pole\cdot\sin(\varphi)}\right]^{Kl}$ $\Theta_\lambda = Kl\cdot(\lambda-\lambda_0)$		
	$x = R_\varphi\cdot\sin(\Theta_\lambda)$ $y = -Pole\cdot R_\varphi\cdot\cos(\Theta_\lambda)$		$x = R_T\cdot\cos(\varphi_0)\cdot(\lambda-\lambda_0)$ $y = -R_T\cdot\cos(\varphi_0)\cdot\ln\left[\tan\left(\frac{\pi}{4}-\frac{\varphi}{2}\right)\right]$
	$m_\varphi = \left[\dfrac{\cos(\varphi_0)}{\cos(\varphi)}\right]^{(1-Kl)}\cdot\left[\dfrac{1+Pole\cdot\sin(\varphi_0)}{1+Pole\cdot\sin(\varphi)}\right]^{Kl}$		
	$m_\varphi = \dfrac{Kl\cdot R_\varphi}{R_T\cdot\cos(\varphi)}$	$m_\varphi = \dfrac{R_\varphi}{R_T\cdot\cos(\varphi)}$	$m_\varphi = \left[\dfrac{\cos(\varphi_0)}{\cos(\varphi)}\right]$
Tangent	$Kl = Pole\cdot\sin(\varphi_0)$	$Kl = Pole\cdot\sin(\varphi_0) = 1$ $\varphi_0 = \pm\dfrac{\pi}{2}$	$Kl = \sin(\varphi_0) = 0$ $\varphi_0 = 0$
	$R_{Eq} = \dfrac{R_T\cdot\cos(\varphi_0)^{(1-Kl)}}{Kl}\cdot(1+Kl)^{Kl}$	$R_{Eq} = 2\cdot R_T$	
	$R_\varphi = R_{Eq}\cdot\left[\dfrac{\cos(\varphi)}{1+Pole\cdot\sin(\varphi)}\right]^{Kl}$ $\Theta_\lambda = Kl\cdot(\lambda-\lambda_0)$	$R_\varphi = R_{Eq}\cdot\left[\dfrac{\cos(\varphi)}{1+Pole\cdot\sin(\varphi)}\right]$ $\Theta_\lambda = (\lambda-\lambda_0)$	
	$x = R_\varphi\cdot\sin(\Theta_\lambda)$ $y = -Pole\cdot R_\varphi\cdot\cos(\Theta_\lambda)$		$x = R_T\cdot(\lambda-\lambda_0)$ $y = -R_T\cdot\ln\left[\tan\left(\frac{\pi}{4}-\frac{\varphi}{2}\right)\right]$
	$m_\varphi = \dfrac{Kl\cdot R_\varphi}{R_T\cdot\cos(\varphi)}$	$m_\varphi = \dfrac{2}{1+Pole\cdot\sin(\varphi)}$	$m_\varphi = \dfrac{1}{\cos(\varphi)}$

Table 2: Formulae[3]

3 The different shades of the table describe (from top to bottom) : Kl calculus, the intermediary calculus of the polar coordinates $(R_\varphi,\Theta_\lambda)$ in LCC and SP, the Cartesian coordinates (x,y) and the map factor calculus.

R_{eq} *(Illustration 2)* is a specific item for any given projection (φ_0, KL) in the equations for LCC and SP projections. R_{eq} is the distance of the Equator projection in polar coordinates. The main projection methods consist in a shift from geographical coordinates (λ,φ) into Cartesian coordinates (x,y). For LCC and SP projections, this shift uses a go between step in polar coordinate (R_φ,$\Theta\lambda$).

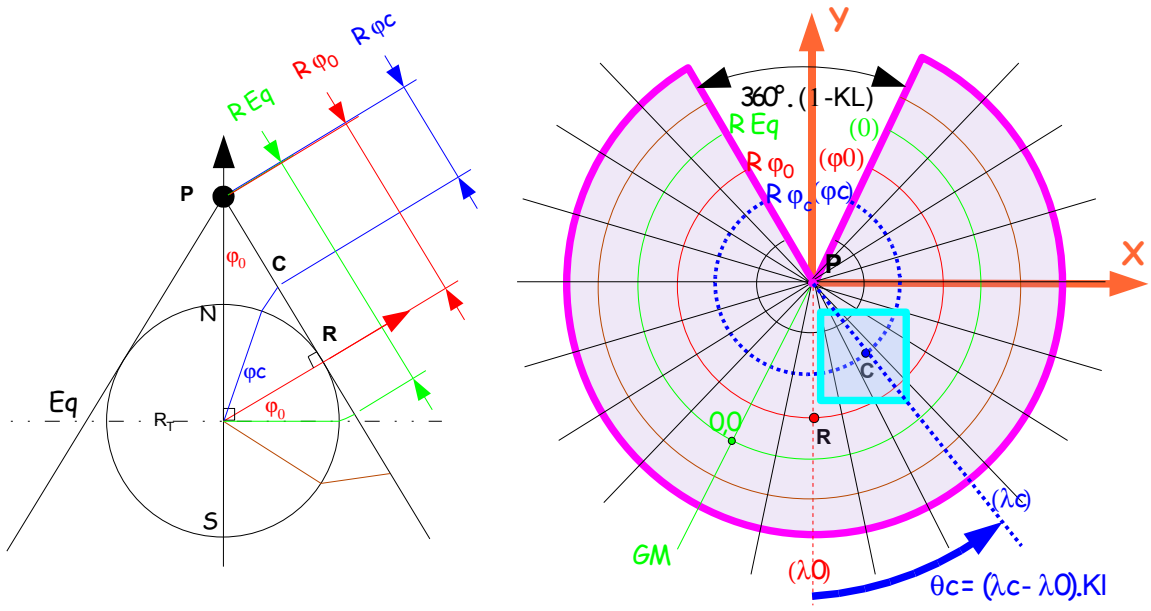


Illustration 2: Correspondences for the Lambert case

Two other parameters are used in these formulae:

✓ *Pole* that defines in LCC and SP projections, the pole belonging to the projection plane[4]. *Pole* will be chosen, according to the reference latitude φ_0,, its absolute value =1 and will get the sign of the reference latitude whose definition domain is [-90°,90°]. This parameter is valueless in the Mercator projection.

✓ λ_0, in the LCC and SP projections, is the meridian parallel to the lefthand and righthand sides of the domain. In the 3 projections, it defines the Y axis (abscissa x=0). On the opposite meridian (the one forming with it a great circle) will take place the slicing of the solid (cylinder or conic) to make the projection plane. This parameter is not essential for the projection but for a usage in a LAM and for LCC and SP projections, it can give an idea of the rotation of the domain on its centre and prevents the slicing to be inside the domain.

If SP is a true central projection whose centre is the opposite pole of the projection plane[5], it is not the case for the LCC and M projections which impose some constraints to be conformal[6]. For example, should a LCC projection be represented on a cone in the usual way, then it will not have a projection centre; should one be needed then the cone will look like a candle flame (Illustration 3).

4 Only the pole of the sphere tangent to the plane (SP) or on the same side of the tip of the cone (LCC) can be projected.

5 When the latitude of the projection plane is situated in one hemisphere, then the centre of projection is the pole of the opposite hemisphere.

6 Conform: conservation of the $\delta x/\delta y$ ratio between the sphere and the projection plane, also called angle conservation.

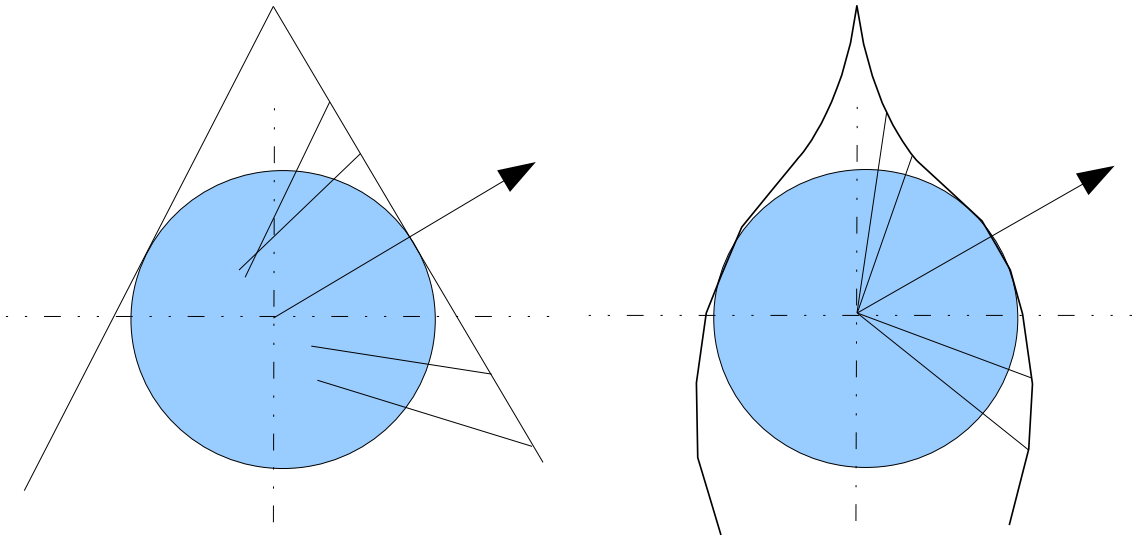
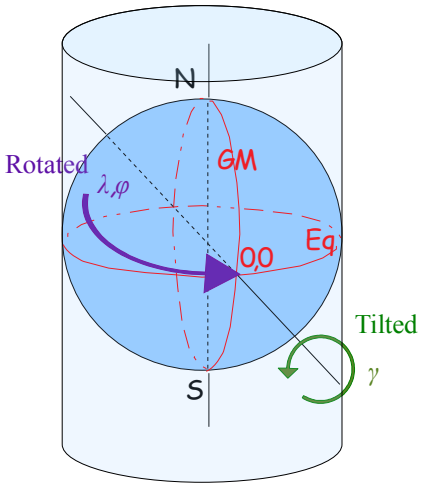


Illustration 3: Centre of the projection LCC or cone

3.3.The Mercator Rotated Tilted «projection»

As already described, this projection is solely another usual Mercator projection whose centre of interest has been set at (0,0), which is the intersection of the Equator with the Greenwich meridian (Illustration 4). The parameters describing this projection are identical as the ones for the M projection.



However the domain of interest must also be described so as to specify the "Rotated" part, this will be done using the data of coordinates (λ,φ) of a point[7] specific of the domain which will be moved to (0,0). The rotated part of the MRT projection is given by any point coordinates.

The "Tilted" part which is the the rotation of the sphere on the [(0,0),(180,0)] axis (after being "Rotated"),will be represented by its angular value γ oriented trigonometric wise (anti clockwise). This enables the projected domain to be reoriented in regard of the cylinder axis.

Illustration 4: The MRT projection

The "Rotated" part of the projection corresponds to what is sometimes called the relocation of the polar of the projection[8]. But it is easier to define the point (thus the zone) to be shifted to coordinates (0,0) than to define the position the pole must reach to characterize this rotation.

7 For the geometry of the LAM ALADIN-AROME models, it is the centre C(λ_c,φ_c) of the domain.
8 i.e. for the ARPEGE model.

The formulae that described the “Rotated” and Tilted” notions can be found in P. Bénard's documentation: New «Rotated/Tilted Mercator» geometry in Aladin (27 Octobre 2004).

4.The «geometry» package

4.1.The geometry in the ALADIN-AROME model

In a meteorological model, the geometry is a way by which a working computer array, indexed by *x et y,* finds a correlation with an array of geographical data: position on the real sphere (longitude, latitude) and data from the type of projection chosen (map factor, compass).

The functions by which one goes from the projection to the other are define by:

- ✔ the kind of projection (L, SP, M, MRT[9]) spotted by the reference point R of coordinates (λ_0, φ_0)

The domain is defined by:

- ✔ position of the domain on the sphere identified by the coordinates (λ_c, φ_c) of its centre[10] C
- ✔ the number of points of the domain[11] along the axis x and y noted *nbptx* and *nbpty*
- ✔ the resolutions[12] on the x and y axis are respectively noted Δx and Δy.

To define the geometry of an ALADIN-AROME model means to define a domain and its associated projection so that the relation between an computing array of points and their geographical characteristics (see i*llustration 6*).

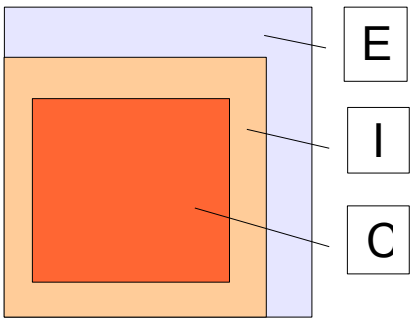


Illustration 5: The C,I,E zones

The working array of the ALADIN-AROME models is divided into 3 zones: C+I+E (see illustration 5). the E zone is a truly mathematical extension zone, with no meteorological validity. It enables the “bi-periodicity” of the data fields according to the x and y axis and thus to use them in a spectral representation (FFT) for the computations.

The size of the C+I+E array is subject to constrains (see the ALADIN documentation) which limit the choice of the size of the domain C+I.

When the geographical domain is mentioned, it will be associated with the C+I zones only.

9 For a MRT projection, the centre C of the domain also specify the «Rotated» part.

10 The central point of the array associated to the domain is called centre of the domain. The number of points on either side of the centre are identical along the two axes of the array.

11 LAM will be represented by 2 dimensional arrays of axes $(\vec{u}, \vec{v})$, of respective indexes *x* et *y*, and of respective sizes *nbptx* et *nbpty*.

12 The resolution is the distance between the points along the two axes in the projection plane. Potentially two values are to be specified, one for each axis; but as more often than not these values will be identical, the term isotropic grid will be used to designate “**the**” resolution of the model.

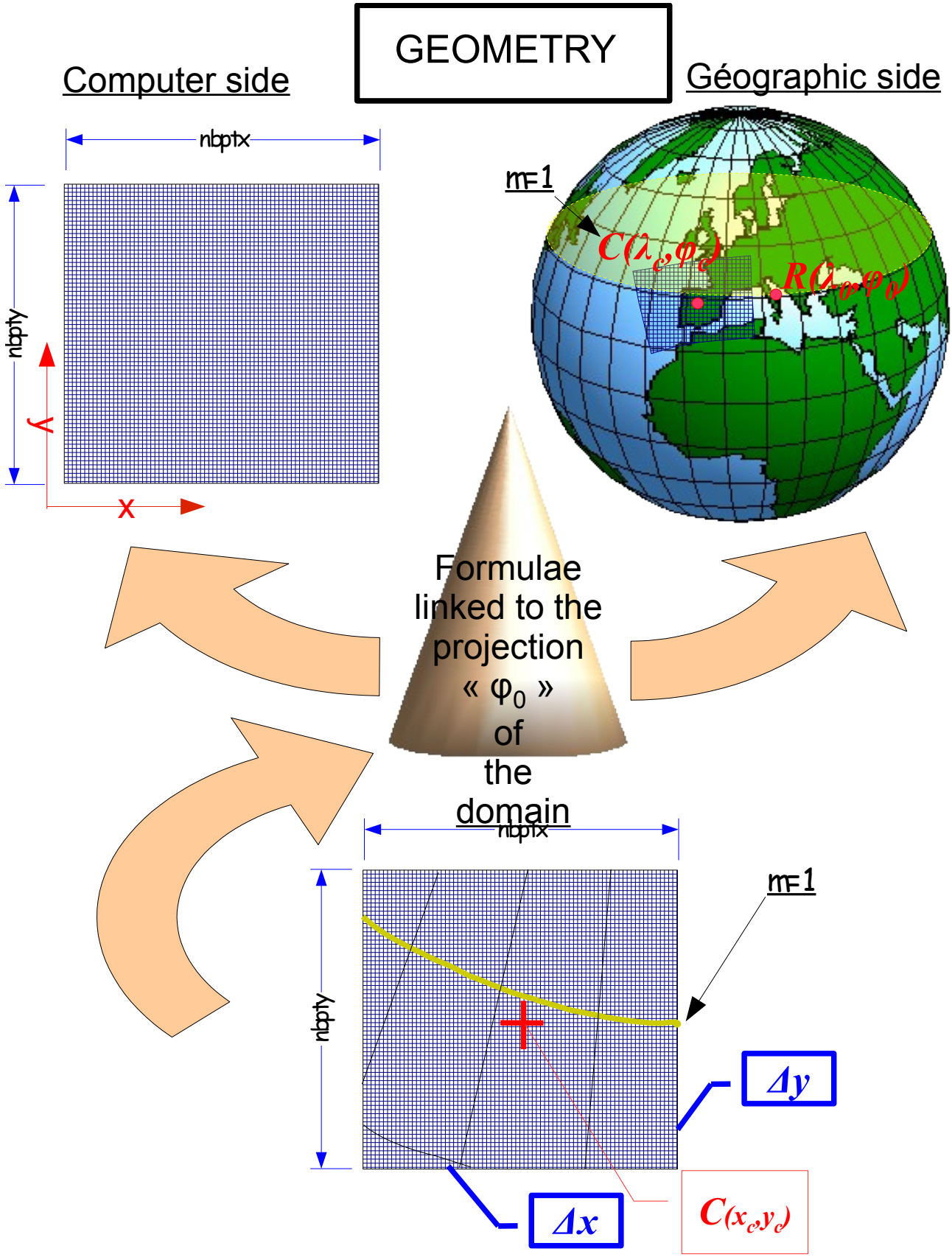


Illustration 6: Relations between computer arrays, domain, projection

4.2.The package design

Fortran 90 is the language used in the "geometry" package of the ALADIN-AROME model. It has been thus possible to somehow make an oriented object design. First of all, a structure that define the projection is initialized (as an Oriented Object Language constructor). The transformations concerning the projections are set as functions using this structure (see the methods associated to this object). Thus, the code framework can very easily be rewriten in a true Oriented Object Language[13]

Several other types of Fortran (likened to object classes) have also been defined with their own functions (methods), for example, LOLA which represents a "geopoint" (**lo**ngitude, **la**titude) with its associated functions from the EGGANGLES module.

Initially the code was split into two modules:

- ✓ EGGANGLES: additional module composed of either several functions about angles or more specific functions used in the package (likened to one of the "private" methods).
- ✓ EGGPACK: module about the definition of the structure of the projection, about depictive functions, about specific functions for SP & LCC projections (corresponding to intermediate computations in (R, Θ), see *table 2*), about main functions enabling to switch from coordinates (x,y) to coordinates (λ, φ), main functions on map factor computation and of compass. This module also possess a MAKDO routine (**mak**e **do**main) which is specific to the ALADIN-AROME model[14] that draws latitude, longitude, map factor, compass arrays linked with a specific domain (zone C+I, see *illustrations 5 et 6*). All other functions can be used on any points, vector of points («array» Fortran) of the sphere where the projection is delimited.

Because of the design of the code it has been possible to insert a new additional module (EGGMRT) which ismade up of the «Rotated» and «Tilted» parts when the MRT projection was added. Calls to the EGGPACK functions have not been modified, only internal tests have been added as well as an extension of the structure depicting the projection so as to have new attributes[15] ("inheritance" OOL notion).

4.3.Description of the elements of the three modules of the package

4.3.1.The EGGPACK module

A.pre-compiler key:

Name	Default value	Explanation
DEBUG	FALSE	Validate the automatic impression of information in the MAKDO routine (LD_LIP default value)

B.External modules :

Specific modules for ALADIN-AROME which can be easily redefined for external use:

13 A PYTHON version of this package is planned.
14 For robustness sake could be moved into another file.
15 Specially, the coordinates of the starting point of the "Rotated" rotation.

Module	Elements	Type[16]	Description
PARKIND	JPIM	INT	Kind of integer
	JPRB	INT	Kind of real
YOMHOOK	LHOOK	LOGI	Implementation or not
	DR_HOOK	PROC	Trace routine

C.Package modules:

Module	Elements	Type[16]	Description
EGGANGLES	LOLA	TYPE	Geopoint longitude, latitude
	VAL_COORD	FUNC	Validity of coordinates
	LOLAR	FUNC	Geopoint from degree to radian
	LOLAD	FUNC	Geopoint from radian to degree
	ANGLE_DOMAIN	FUNC	Domain of validity of geopoint
	DIST_2REF	FUNC	Distance along longitude to the reference point
EGGMRT	MEROTIL	FUNC	Transformation => «ROTATED-TILTED»
	METILROT	FUNC	Inverse transformation

D.Parameters:

Name	Value	Type[16]	Description
R_EARTH	6371229._JPRB	REAL	Earth radius assumed spherical

E.Structures:

Name	Elements	Type[16]	Description
ERROR	NUM	INT	Error index
	TXT	CHAR	Error description
PGN	ONX	REAL	Compass on X
	ONY	REAL	Compass on Y
NBPTS	ONX	INT	Number of points along X
	ONY	INT	Number of points along Y
DELTA	ONX	REAL	Resolution along X (meter)
	ONY	REAL	Resolution along Y (meter)
RTETA	R	REAL	Radial polar coordinate (meter)
	TETA	REAL	Angular polar coordinate (radian)

16 See table of types at the end of chapitre, § 4.3.4.

XY	X	REAL	Cartesian coordinates X (meter)
	Y	REAL	Cartesian coordinates Y (meter)
PARAM_PROJ (structure outlining a specific projection)	REF_PT	LOLA	Reference geopoint of the projection
	TZO_PT	LOLA	Geopoint to be set to (0,0), MRT only.
	KL	REAL	Attributes of the projection.
	R_EQUATEUR	REAL	Radial coordinates of the projection of the Equator
	POLE	REAL	1=N;-1=S only for LCC and SP
	TYPE_PJ	CHAR	Belongs to S, L, M, W (for MRT)
DOMI (Information about the projected domain according to the characteristics of the projection described in INFO_PROJ)	G_SIZE	NBPTS	Size of the domain
	CT_COORD	LOLA	Coordinates of the centre point
	RF_COORD	LOLA	Reference point coordinates
	SW_COORD	LOLA	Bottom left point coordinates
	SE_COORD	LOLA	Bottom right point coordinates
	NE_COORD	LOLA	Top right point coordinates
	NW_COORD	LOLA	Top left point coordinates
	MF_CT	REAL	map factor of the centre point
	MF_RF	REAL	map factor of the reference point
	MF_SW	REAL	map factor of the bottom left point
	MF_SE	REAL	map factor of the bottom right point
	MF_NE	REAL	map factor of the top right point
	MF_NW	REAL	map factor of the top left point
	INFO_PROJ	PARAM_PROJ	Characteristics of the projection

F. Functions:

"ABOR1(«Display text before switch off»)" is an external function which is used within the module. Not used with the model program, it can be replaced by "PRINT" followed by "STOP".

- **Functions for inner usage:**

These functions are used internally by either others functions or procedures of the module. Usually speaking, they are of no use outside the module.

Name	Type[16] returned	Description
RETURN_PRINT	LOGICAL	Display function for errors, informations or alarms with possible execution abort in case of error; an ERROR type given in input. A TRUE value is sent in case of error, FALSE in every other cases.
Input parameters	Type[16]	Description
YD_CODE_ERR	ERROR	An integer and its associated message delivering status information: <0 : ERROR 0 : OK 1 : INFORMATION >1 : ALARM
K_NUM_TEST	INT	Associated test number
AUTO_STOP	LOGICAL	By default TRUE. If the value is TRUE then STOP if ERROR%NUM<0. Display in any case.
KOUT	INT	Output unit, by default it is the standard output unit (6).

Name	Type[16] returned	Description
TYPE_PROJ	CHAR	Function attributing the kind of projection (in the shape of a letter L,S,M) in regard of the latitude of the reference point.
Input parameters	Type[16]	Description
REF_COORD	LOLA	Reference point geographical coordinates in degree.

Name	Type[16] returned	Description
POLE_IS	REAL	Function for: -1: South hemisphere latitude reference point 1 : North hemisphere latitude reference point 0 : latitude = 0 (but unused with Mercator)
Input parameters	Type[16]	Description
REF_COORD	LOLA	Reference point geographical coordinates in degree.

•Specific functions for LCC & SP projections:

For a given projection, the shift from geographical coordinates (λ, φ) to polar ones (R, Θ) and from polar coordinates (R, Θ) to Cartesian ones (x,y) and conversely are made possible by these functions specific to LCC & SP projections.

They are applied either to a point (suffix **_S** for **S**calar) or to a one dimension array of points (suffix **V** for **V**ector...-). The x axis is oriented W/E, the y one, S/N. The axes origin is the projection of the pole associated to the hemispheric that contains the Reference Latitude. This origin is called ***standard origin (STD)***. Θ is the angle made with λ_0 trigonometrically wise (see *illustration 2*).

Selecting points (x,y) inside the missing sector of the LCC projection plane induces an abort error of the functions STLP_XY_TO_RTETA

Name	Type[16] returned	Description
STPL_LATLON_TO_RTETA_S	RTETA	Shift into a projection defined by P_PJ, of coordinates (λ,φ) to (R, Θ).

Input parameters	Type[16]	Description
PT_COORD	LOLA	Radian geographical coordinates of a geopoint
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_LATLON_TO_RTETA_V	RTETA (:)	Shift into a projection defined by P_PJ, of an array of coordinates (λ,φ) to (R, Θ) - Array.

Input parameters	Type[16]	Description
PT_COORD(:)	LOLA	Radian geographical coordinates of an array of geopoints
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_RTETA_TO_LATLON_S	LOLA	Shift into a projection defined by P_PJ, of coordinates (R, Θ) to (λ,φ).

Input parameters	Type[16]	Description
PT_RTETA	RTETA	Points in polar coordinates
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_RTETA_TO_LATLON_V	LOLA (:)	Shift into a projection defined by P_PJ, of an array of coordinates (R, Θ) to (λ,φ).

Input parameters	Type[16]	Description
PT_RTETA(:)	RTETA	Array of points in polar coordinates
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_XY_TO_RTETA_S	RTETA	Shift into a projection defined by P_PJ, of coordinates (x,y) to (R, Θ).
Input parameters	Type[16]	Description
PT_XY	XY	Point in Cartesian coordinates
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_XY_TO_RTETA_V	RTETA (:)	Shift into a projection defined by P_PJ, of an array of coordinates (x,y) vers (R, Θ).
Input parameters	Type[16]	Description
PT_XY(:)	XY	Array of points in Cartesian coordinates
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_RTETA_TO_XY_S	XY	Shift into a projection defined by P_PJ, of coordinates (R, Θ) to (x,y) .
Input parameters	Type[16]	Description
PT_RTETA	RTETA	Points in polar coordinates
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
STPL_RTETA_TO_XY_V	XY (:)	Shift into a projection defined by P_PJ, of an array of coordinates (R, Θ) to (x,y).
Input parameters	Type[16]	Description
PT_RTETA(:)	RTETA	Arrays of points in polar coordinates
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

•**Generic Functions:**

The first generic function, REF_DATAS is the "constructor" (OOL sense) of the PARAM_PROJ type structure. It must be called first and foremost before using the specific functions and especially before the generic ones as it will be these ones which will be mainly used. Aforementioned specific functions are essentially used, once combined, to define the generic functions enabling to go from the (x,y) coordinates to the (λ, φ) ones and conversely.

Functions expecting (λ, φ) geographical coordinates, do so in radian in the [-π, π[interval for longitudes (negative west of the Greenwich Meridian) and between [-π/2, π/2] from the south to the north for latitudes.

These functions work with x & y referred to the standard origin **(STD)** in LCC & SP projections. This origin point is defined in a Mercator projection by the reference point (λ_0, 0) and in MRT by (0,0). In all cases, axes are oriented as explained above (W/E & N/S) (See *illustration 7*).

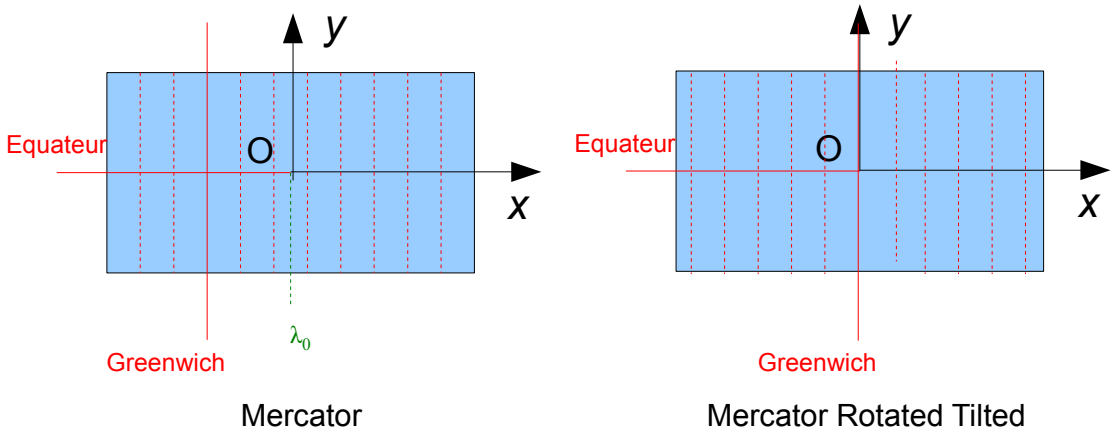


Illustration 7: STD standard origin in Mercator and in Mercator Rotated Tilted

Conversion functions (and the reverse functions) about Cartesian coordinate (x,y) to the standard origin **(STD)** towards a new origin (**NEW**) set by its geographical coordinates (λ, φ) are available.

Other functions are the map factor and compass.

All these functions can be found either as a geopoint (suffix **_S**) or as an array of points (suffix **_V**). They can be called using, for example, the «MODULE PROCEDURE» generic Fortran interface.

In these functions, UNIT & DOM attributes (see *table 3*) will be set for parameters of the LOLA type. For example: degre in [-180°, 180°[or radian in [-π, π[; degre in [0, 360°[or radian in [0, 2π[. DOM attribute specifies only the range of the longitude. According to the UNIT value (degre or radian), the latitude is always in [-90°, 90°] or [-π/2, π/2].

Name	Type[16] returned	Description
REF_DATAS	PARAM_PROJ	Sets the structure defining the projection and used by the following functions.
Input parameters	Type[16]	Description
REF_COORD	LOLA	Reference point in degree [-180°,180°[
RA	REAL	Earth radius in metres. (optional)
TOZERO_COORD	LOLA	With MRT projection, point to be shifted in (0,0) in degree [-180°,180°[. (optional)
LRT	LOGICAL	TRUE if projection MRT, need the above parameter (optional)

Note: *For a MRT projection,* ***the value of λ_0 represent the angle γ*** *of the «TILTED» (see illustration 4, §3.3). The origin of the Cartesian axes x & y is, in this case, always (0°,0°).*

Name	Type[16] returned	Description
XY_NEW_TO_STD_ORIGIN_S	XY	$(x,y)_{NEW}$ to $(x,y)_{STD}$.
Input parameters	Type[16]	Description
NEW_ORIGIN_COORD	LOLA	Geographical coordinates (λ, φ) of the origin NEW in degree inside [-180°,180°[
PT_XY_IN_NEW_ORIGIN	XY	Point in Cartesian coordinates (x,y) in regard to the origin NEW
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
XY_NEW_TO_STD_ORIGIN_V	XY (:)	$(x,y)_{NEW}$ to $(x,y)_{STD}$. -Array-
Input parameters	Type[16]	Description
NEW_ORIGIN_COORD	LOLA	Geographical coordinates (λ, φ) of the origin NEW in degree within [-180°,180°[
PT_XY_IN_NEW_ORIGIN(:)	XY	Array of points in Cartesian coordinates (x,y) in regard to the origin NEW
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
XY_STD_TO_NEW_ORIGIN_S	XY	$(x,y)_{STD}$ to $(x,y)_{NEW}$.
Input parameters	Type[16]	Description
NEW_ORIGIN_COORD	LOLA	Geographical coordinates (λ, φ) of the origin NEW in degree within [-180°,180°[
PT_XY_IN_STD_ORIGIN	XY	Points in Cartesian coordinates (x,y) in regard to the origin STD
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
XY_STD_TO_NEW_ORIGIN_V	XY (:)	$(x,y)_{STD}$ to $(x,y)_{NEW}$. -Array-
Input parameters	Type[16]	Description
NEW_ORIGIN_COORD	LOLA	Geographical coordinates (λ, φ) of the origin NEW in degree within [-180°,180°[
PT_XY_IN_STD_ORIGIN(:)	XY	Array of points in Cartesian coordinates (x,y) in regard to the origin STD
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Note : A shift from an A origin to a B origin is possible when combining two inverse functions STD_TO_NEW et NEW_TO_STD:

```
PT_XY_IN_B_ORIGIN = XY_STD_TO_NEW_ORIGIN(B_ORIGIN_COORD,
                    XY_NEW_TO_STD_ORIGIN(A_ORIGIN_COORD, PT_XY_IN_A_ORIGIN, P_PJ, PI)
                    , P_PJ, PI)
```

Name	Type[16] returned	Description
LATLON_TO_XY_S	XY	(λ, φ) to $(x,y)_{STD}$
Input parameters	Type[16]	Description
PT_COORD	LOLA	Geopoint in geographical coordinates (λ, φ) in radian on [-π, π[
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
LATLON_TO_XY_V	XY (:)	(λ, φ) to $(x,y)_{STD}$ -Array-
Input parameters	Type[16]	Description
PT_COORD(:)	LOLA	Array of geopoints in geographical coordinates (λ, φ) in radian on $[-\pi, \pi[$
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
XY_TO_LATLON_S	PT_COORD	$(x,y)_{STD}$ to (λ, φ) in radian
Input parameters	Type[16]	Description
PT_XY	XY	Geopoint in Cartesian coordinates (x,y)
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
XY_TO_LATLON_V	PT_COORD (:)	$(x,y)_{STD}$ to (λ, φ) in radian -Array-
Input parameters	Type[16]	Description
PT_XY(:)	XY	Array of de points in Cartesian coordinates (x,y)
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)

Note: *Calls to the EGGMRT module are made inside the above mentioned functions, see § 4.3.2.*

Name	Type[16] returned	Description
MAP_FACTOR_S	REAL	map factor calculus for (λ, φ) in radian
Input parameters	Type[16]	Description
PT_COORD	LOLA	Geopoint in geographical coordinates (λ, φ) in radian on $[-\pi, \pi[$
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)
RA	REAL	Earth radius in metres. (optional)

Name	Type[16] returned	Description
MAP_FACTOR_V	REAL (:)	map factor calculus for (λ, φ) in radian -Array-
Input parameters	Type[16]	Description
PT_COORD(:)	LOLA	Array of geopoints in geographical coordinates (λ, φ) in radian on [-π, π[
P_PJ	PARAM_PROJ	Attributes of the projection
PI	REAL	Value of π (optional)
RA	REAL	Earth radius in metres. (optional)

Name	Type[16] returned	Description
GN_S	PGN	Compass calculus (**G**eographic **N**orth)
Input parameters	Type[16]	Description
PT_COORD	LOLA	Geopoint in geographical coordinates (λ, φ) in radian on [-π, π[
P_PJ	PARAM_PROJ	Attributes of the projection

Name	Type[16] returned	Description
GN_V	PGN (:)	Compass calculus (**G**eographic **N**orth) -Array-
Input parameters	Type[16]	Description
PT_COORD(:)	LOLA	Array of geopoints in geographical coordinates (λ, φ) in radian on [-π, π[
P_PJ	PARAM_PROJ	Attributes of the projection

G.Procedures:

- **Display Procedures:**

The characteristics of a domain or of a projection can be displayed using these two procedures.

Name	Description		
INFO_DOMI_PRINT	Display of the attributes of the domain and its associated projection		
Input parameters	Type[16]	I/O	Description
YD_G_INFO	DOMI	I	Structure of a MAKDO made domain
KOUT	INT	I	Output unit, by default it is the standard output unit (6).
PI	REAL	I	Value of π (optional)

Name	Description		
INFO_PP_PRINT	Display of projection's features		
Input parameters	Type[16]	I/O	Description
P_P	PARAM_PROJ	I	Structure of a REF_DATAS projection
KOUT	INT	I	Output unit, by default it is the standard output unit (6).
PI	REAL	I	Value of π (optional)

•Creation of an ALADIN-AROME domain using the MAKDO routine:

As already seen, to create this kind of domain, three 2D, i & j, arrays must be filled in. The first one with the geographical coordinates (λ,φ) linked to each point of index coordinates (i,j) of the grid of the domain, the second one with the map factors, the third one with the compass for each of these points.

These index coordinates (i,j) vary respectively from *1* to *nbptx* and from *1* to *nbpty*. If x_{sw}[17] & x_{se} are the Cartesian coordinates, along the x axis, of the points on the bottom left and the bottom right of the domain, then[18] $x_{se} - x_{sw} = (nbptx-1).\Delta x$ where Δx is the resolution along the x axis. Ditto for the y axis: $y_{nw} - y_{sw} = (nbpty-1).\Delta y$ where y_{nw} and y_{sw} are the Cartesian coordinates along the y axis, respectively for points on the top left and top right of the domain. The Cartesian coordinates (x_{ij}, y_{ij}) of a point of index (i, j) can be written as:

$x_{ij} = x_{sw} + (i-1).\Delta x$

$y_{ij} = y_{sw} + (j-1).\Delta y$

Output arrays are two dimensional ones (size of the C+I zone, see *illustration 5*) and, in the MAKDO routine, are used with generic functions within a loop on the second dimension to optimize the "vectorisation" of the Fortran code.

Name	Description		
MAKDO	Creation of an ALADIN-AROME domain		
Input parameters (dimension i, dimension j)	Type[16]	I/O	Description
YD_REF_COORD	LOLA	IO	Geographical coordinates in degree of the reference point in [-180°,180°[
YD_CENTER_COORD	LOLA	IO	Geographical coordinates in degree of the centre of the domain in [-180°,180°[
YD_PDEL	DELTA	I	Resolution in metre
YD_NB_PTS	NBPTS	I	Number of points in x et y
YD_GRID_COORD (YD_NB_PTS%ONX,YD_NB_PTS%ONY)	LOLA	O	Geographical coordinates in radian of index points (i, j), value in [0°,360°[

17 The four corners of the domain are named SW,SE,NE, NW. Some mistakes can occur in case of domains with a wide range along the longitude. Therefore, at all times the SW point will refer to the Botton Left corner, the SE to the Bottom Right one, and so on.

18 Whatever the origin (STD or NEW), the x and y axes are respectively oriented from West to East and from South to North so as to have: $x_{se} > x_{sw}$ et $y_{nw} > y_{sw}$

P_GRID_MF (YD_NB_PTS%ONX,YD_NB_PTS%ONY)	REAL	O	map factor of index points (i, j)
YD_GRID_PGN (YD_NB_PTS%ONX,YD_NB_PTS%ONY)	PGN	O	Compass of index points (i, j)
YD_GRID_INFO	DOMI	O	Information about the domain[19]
YD_ERR_CODE	ERROR	O	Returned error code
LD_LIP	LOGICAL	I	Call INFO_PRINT or not (optional)
LD_AUTO_STOP	LOGICAL	I	Stop on error or not (optional)
PI	REAL	I	Value of π (optional)
P_RA	REAL	I	Earth radius in metres. (optional)
KOUT	INT	I	Output unit, by default it is the standard output (6).
LD_LMRT	LOGICAL	I	Flag specifies the projection kind is Mercator « Rotated/Tilted » (optional either FALSE)

The code of this routine creates the domain from the centre, from the number of points and from the resolution along the two directions (from index coordinates (i, j) to Cartesian coordinates (x, y)). This is followed by the definition of a projection and the expected arrays are created by the generic functions.
Validity tests are carried out to check the relevance and the values of the choice of the domain and its associated projection. The following ERROR parameters are defined, depending on cases (LD_AUTO_STOP value), either they stop the execution or print an alert message (see above the RETURN_PRINT function for internal use)

Table of ERROR type parameters	
NUM	TXT
-6	With Mercator, distortions are too big outside [-85.0,85.0] (case MRT) !
-5	With Lambert, the pole must be outside the domain!
-4	With Mercator, distortions are too big outside [-85.0,85.0]!
-3	Coordinates in degree of the Centre outside the boundaries!
-2	Coordinates in degree of the Reference point outside the boundaries!
-1	Abort Routine on Error!
0	Routine ends successfully!
1	Test OK, go ahead!
2	With Mercator, better use Lambert or St.Pol if ABS(CENTER_LAT) > 20.0 !
3	With St.Pol., better use Lambert or Mercator if ABS(CENTER_LAT) < 70.0 !
4	With Lambert, better use St.Pol. or Mercator if ABS(REF_LAT) outside [20.0,70.0] !

19 Input geographical coordinates are in degree on [-180°,180°[. Output geographical coordinates are in radian on the interval [0, 2π[. DOMI geographical coordinates are in degree on [-180°, 180°[for Reference and Centre points, on [0°,360°[for others points.

4.3.2. Module EGGMRT:

Developed in a second phase, this module houses the transition formulation "Rotated/Tilted" (see the relevant above chapter). As seen earlier, the switch and the rotation take place prior to[20] or after[21] making a classical Mercator projection. This explains why these are called from the generic functions LATLON_TO_XY and XY_TO_LATLON for respectively before and after the request for the direct or reverse Mercator projection functions.

The functions which are called by functions such as LATLON_TO_XY are named METILROT and those by XY_TO_LATLON are named MEROTIL.

> The METILROT functions[22], that are linked to a shift from (λ,φ) to (x,y), use a mix[23] of the ANTI_TILT et ANTI_ROTATE functions; the MEROTIL ones, that are linked to a shift from (x,y) to (λ,φ), use ROTATE and TILT. There is an unfortunate inversion of names between the descriptions and *Illustration 4*. "Rotated" is applied to the ANTI_ROTATE function and "Tilted" to the ANTI_TILT one, the same goes for the reverse functions.

A. External Modules:

ALADIN-AROME own modules easily redefined for outer use:

Module	Elements	Type[16]	Description
PARKIND	JPRB	INT	Kind of real
YOMHOOK	LHOOK	LOGI	Used or not
	DR_HOOK	PROC	Display routine

B. Package Modules:

Module	Elements	Type[16]	Description
EGGANGLES	LOLA	TYPE	Type Geopoint longitude, latitude
	COSIN_TO_ANGLE	FUNC	Give the value of an angle knowing its cosine and its sinus
	P_ASIN	FUNC	Refer to an sinus arc securing that input values be [-1,1] (the P of P_ASIN means Protected)

C. Functions:

All the following functions work on a single geo-point[24] (suffix **_S**) or an array of them (suffix **_V**). They are collated under a Fortran generic interface such as "MODULE PROCEDURE".

- **Reverse functions:**

20 In the sense of : from spheric surface to the plane of projection (direct way)
21 In the sense of,: from the plane of projection to spheric surface (retrograde way)
22 They exist as one geopoint (suffixe **_S**) or array of geopoints (suffixe **_V**)
23 The « mix » is a drawing up of functions as: METILROT(...) = ANTI_TILT(..., ANTI_ROTATE(...))
24 Geographical coordinate in radian.

Name	Type[16] returned	Description
TILT_S	LOLA	Return geopoint after **reverse**«TILT» .
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
PT_COORD2	LOLA	Geopoint subjected to the **reverse** «TILT».

Name	Type[16] returned	Description
TILT_V	LOLA (:)	Return geopoint after **reverse** «TILT» . -Array-
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
PT_COORD2(:)	LOLA	Array of geopoints subjected to the **reverse** «TILT».

Name	Type[16] returned	Description
ROTATE_S	LOLA	Return geopoint after **reverse** «ROTATE» from (0°,0°) to (λ_c, φ_c) .
Input parameters	Type[16]	Description
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **reverse** shift.
PT_COORD1	LOLA	Geopoint subjected to the **reverse** «ROTATE».

Name	Type[16] returned	Description
ROTATE_V	LOLA (:)	Return an array of geopoints after **reverse** «ROTATE» from (0°,0°) to (λ_c, φ_c) .
Input parameters	Type[16]	Description
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **reverse** shift.
PT_COORD1(:)	LOLA	Array of geopoints subjected to **reverse** «ROTATE».

Name	Type[16] returned	Description
MEROTIL_S	LOLA	Return geopoint after **reverse** «ROTATED/TILTED».
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **reverse** shift.
PT_COORD2	LOLA	Geopoint subjected to **reverse** «ROTATED/TILTED».

Name	Type[16] returned	Description
MEROTIL_V	LOLA (:)	Return an array of geopoints after **reverse** «ROTATED/ TILTED».
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **reverse** shift.
PT_COORD2(:)	LOLA	Array of geopoints subjected to the **reverse** «ROTATED/TILTED».

Note:
PT_COORD=MEROTIL(REF_COORD,CENTER_COORD,PT_COORD2)
equivalent to :
PT_COORD=ROTATE(CENTER_COORD,TILT(REF_COORD,PT_COORD2))

- **Direct functions:**

Name	Type[16] returned	Description
ANTI_TILT_S	LOLA	Return geopoint after **direct** «TILT» .
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
PT_COORD1	LOLA	Geopoint subjected to the **direct** «TILT».

Name	Type[16] returned	Description
ANTI_TILT_V	LOLA (:)	Return geopoint after **direct** «TILT» . -Array-
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
PT_COORD1(:)	LOLA	Array geopoints subjected to the **direct** «TILT».

Name	Type[16] returned	Description
ANTI_ROTATE_S	LOLA	Return geopoint after **direct** «ROTATE» from (λ_c, φ_c) vers (0°,0°).
Input parameters	Type[16]	Description
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **direct** shift..
PT_COORD	LOLA	Geopoint subjected to **direct** «ROTATE».

Name	Type[16] returned	Description
ANTI_ROTATE_V	LOLA (:)	Return an array of geopoints after **direct** «ROTATE» from (λ_c, φ_c) vers (0°,0°).
Input parameters	Type[16]	Description
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **direct** shift.
PT_COORD(:)	LOLA	Array of geopoint subjected to **direct** «ROTATE».

Name	Type[16] returned	Description
METILROT_S	LOLA	Return geopoint after **direct** «ROTATED/TILTED».
Input parameters	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **direct** shift.
PT_COORD	LOLA	Geopoint subjected to **direct** «ROTATED/TILTED».

Nom	Type[16] returned	Description
METILROT_V	LOLA (:)	Return an array of geopoints after a **direct** « ROTATED/ TILTED ».
Paramètres d'entrée	Type[16]	Description
REF_COORD	LOLA	Geographical coordinates (λ_0, 0) in radian of geopoint where λ_0 represents the «TILT» angle.
CENTER_COORD	LOLA	Geographical coordinates (λ_c, φ_c) in radian of geopoint, depicting the ending point of the **direct** shift.
PT_COORD(:)	LOLA	Array of geopoint subjected to «ROTATE D/TILTED» **direct**

Note:
PT_COORD2=METILROT(REF_COORD,CENTER_COORD,PT_COORD)
equivalent to :
PT_COORD2=ANTI_TILT(CENTER_COORD,ANTI_ROTATE(REF_COORD,PT_COORD))

4.3.3. EGGANGLES module:

This module houses the functions for:

✓ the handling of angles (domain of validity, switch radians ↔ degrees)

✓ the protection (definition of the domain) during the numerical computations of the trigonometric functions

✓ the management of angular distances (longitudes) on the sphere

The terms DOM (domain) and UNIT (unit) appear in several functions. They specify the domain of validity of the geo-point in geographical coordinates. (see *table 3* below)

DOM	UNIT	Longitudes	Latitudes
'-+'	'D'	[-180.0°, 180.0°[	[-90.0°, 90.0°]
'0+'	'D'	[0.0°, 360.0°[	[-90.0°, 90.0°]
'-+'	'R'	[-π, π[	[-π/2, π/2]
'0+'	'R'	[0, 2π[	[-π/2, π/2]
'-+'	'D'	← Values by default	

Table 3: Domains of validity

A. External modules:

Specific ALADIN-AROME modules easily redefined for outer use:

Module	Elements	Type[16]	Description
PARKIND	JPIM	INT	Kind of integer
	JPRB	INT	Kind of real
YOMHOOK	LHOOK	LOGI	Set up or no
	DR_HOOK	PROC	Display routine

B. Structures:

The main structure of the package represents a geo-point with its geographical coordinates; the unit, either radian or degree is left to users.

Name	Eléments	Type[16]	Description
LOLA	LON	REAL	Longitude of the geopoint
	LAT	REAL	Latitude of the geopoint

C. Functions:

All the following functions work about a single geo-point (suffix_**S**) or an array of geo-points (suffix_**V**). They are collated under a generic interface such as “MODULE PROCEDURE”.

There is only one version for the SIZE_W2E function with the geographical coordinates of the geo-points W and E (suffix **_S**), and only one with the longitudes of these geopoints (suffix **_L**).

•**Functions for the management of angles:**

Name	Type[16] returned	Description
ANGLE_DOMAIN_RS	REAL	Return angle according to chosen input values of unit UNIT[25] and domain definition DOM
Input parameters	Type[16]	Description
ALPHA	REAL	Angle in unit UNIT with value according to domain definition DOM
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
ANGLE_DOMAIN_RV	REAL (:)	Return an array of angle according to chosen input values of unit UNIT[24] and domain definition DOM
Input parameters	Type[16]	Description
ALPHA(:)	REAL	Array of angle in unit UNIT with value according to domain definition DOM
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
ANGLE_DOMAIN_LOLAS	LOLA	Return a geopoint in geographical coordinates in chosen unit UNIT[24] and domain definition DOM
Input parameters	Type[16]	Description
ALPHA	LOLA	Geopoint in geographical coordinates in chosen unit UNIT[24] and domain definition DOM
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

25 **CAUTION**: this function does not switch units. To do it, you must use the LOLAR and/or LOLAD functions.

Name	Type[16] returned	Description
ANGLE_DOMAIN_LOLAV	LOLA (:)	Return an array of geopoints in geographical coordinates in chosen unit UNIT[24] and domain definition DOM
Input parameters	Type[16]	Description
YL_ALPHA(:)	LOLA	Array of geopoints in geographical coordinates in choosen unit UNIT[24] and domain definition DOM
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
VAL_LAT_S	INTEGER	Return 1 if the latitude is a valid one ([-90°, 90°] or [-π/2, π/2] according to UNIT) else NUM_ERR
Input parameters	Type[16]	Description
LAT	REAL	Latitude to be tested
NUM_ERR	INT	Returned value if an error occurs, -1 by default, (optional).
PI	REAL	Value of π (optional)
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
VAL_LAT_V	INTEGER	Return 1 if all the values of the latitude array are valid ([-90°, 90°] or [-π/2, π/2] according to UNIT) else NUM_ERR
Input parameters	Type[16]	Description
P_LAT(:)	REAL	Array of latitude to be tested
NUM_ERR	INT	Returned value if an error occurs, -1 by default, (optional).
PI	REAL	Value of π (optional)
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
VAL_LON_S	INTEGER	Return 1 if the longitude is a valid one (according to chosen values of DOM and UNIT, see *table 3*) else NUM_ERR
Input parameters	Type[16]	Description
LON	REAL	Longitude to be tested
NUM_ERR	INT	Returned value if an error occurs, -1 by default, (optional).
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
VAL_LON_V	INTEGER	Return 1 if all the values of a longitude array are valid (according to chosen values of DOM and UNIT, see *table 3*) else NUM_ERR
Input parameters	Type[16]	Description
LON(:)	REAL	Array of longitudes to be tested
NUM_ERR	INT	Returned value if an error occurs, -1 by default, (optional).
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
VAL_COORD_S	INTEGER	Return 1 if geopoint (latitude and longitude) is valid (according to chosen values of DOM and UNIT, see *table 3*) else NUM_ERR
Input parameters	Type[16]	Description
PT_COORD	LOLA	Geopoint in geographical coordinates to be tested
NUM_ERR	INT	Returned value if an error occurs, -1 by default, (optional).
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
VAL_COORD_V	INTEGER	Return 1 if all values of a geopoints (latitude and longitude) array are valid (according to chosen values of DOM and UNIT, see *table 3*) else NUM_ERR
Input parameters	Type[16]	Description
PT_COORD(:)	LOLA	Array of geopoints in geographical coordinates to be tested
NUM_ERR	INT	Returned value if an error occurs, -1 by default, (optional).
PI	REAL	Value of π (optional)
DOM	CHAR	'-+' ou '0+'. '-+' by default, (optional).
UNIT	CHAR	'D' ou 'R'. 'D' by default, (optional).

Name	Type[16] returned	Description
LOLAR_S	LOLA	Return geopoint value in radian
Input parameters	Type[16]	Description
COORD_DEG	LOLA	Geopoint in degree

Name	Type[16] returned	Description
LOLAR_V	LOLA (:)	Return an array of geopoints in radian
Input parameters	Type[16]	Description
COORD_DEG(:)	LOLA	Array of geopoints in degree

Name	Type[16] returned	Description
LOLAD_S	LOLA	Return a geopoint in degree
Input parameters	Type[16]	Description
COORD_RAD	LOLA	Geopoint in radian

Name	Type[16] returned	Description
LOLAD_V	LOLA (:)	Return an array of geopoints in degree
Input parameters	Type[16]	Description
COORD_RAD(:)	LOLA	Array of geopoints in radian

•**"Protective" functions:**

Name	Type[16] returned	Description
COSIN_TO_ANGLE_S	REAL	Angle in radian defined by its Cosine and Sinus
Input parameters	Type[16]	Description
COSINUS	REAL	Value of the cosine
SINUS	REAL	Value of the sinus

Note: In the above and below functions,the following formulation is used:
*ANGLE=ArcCOS(COSINUS)*SIGN(1,SINUS)*

Name	Type[16] returned	Description
COSIN_TO_ANGLE_V	REAL (:)	Array of angles in radians defined by arrays of their Cosine et Sinus values
Input parameters	Type[16]	Description
COSINUS(:)	REAL	Array of values of the cosine
SINUS(:)	REAL	Array of values of the sinus

Name	Type[16] returned	Description
P_ACOS_S	REAL	Return the Arc Cosine with a bounded value of Cosine in [-1, 1] : (**P**rotected_**Arc**COS**)**
Input parameters	Type[16]	Description
COSINUS	REAL	Value of the cosine

Name	Type[16] returned	Description
P_ACOS_V	REAL (:)	Return an array of Arc Cosine with a bounded array of Cosine in [-1, 1] : (**P**rotected_**Arc**COS**)**
Input parameters	Type[16]	Description
COSINUS(:)	REAL	Array of values of the cosine

Name	Type[16] returned	Description
P_ASIN_S	REAL	Return the Arc Sinus with a bounded value of Sinus in [-1, 1] (**P**rotected_**Arc**SIN**)**
Input parameters	Type[16]	Description
SINUS	REAL	Value of the sinus

Name	Type[16] returned	Description
P_ASIN_V	REAL (:)	Return an array of Arc Sinus with a bounded array of Sinus in [-1, 1] (**P**rotected_**Arc**SIN**)**
Input parameters	Type[16]	Description
SINUS(:)	REAL	Array of values of the sinus

Name	Type[16] returned	Description
MINIMAX_S	REAL	Return « minimax[26] » value in [-LIM, LIM] of a value VAL
Input parameters	Type[16]	Description
VAL	REAL	Input value
LIM	REAL	Limit, by default 1 (optional)

Name	Type[16] returned	Description
MINIMAX_V	REAL (:)	Return an array of « minimax[25] » values in [-LIM, LIM] of an array of values VAL
Input parameters	Type[16]	Description
VAL(:)	REAL	Array of input values
LIM	REAL	Limit, by default 1 (optional)

•Calculation of distances functions

The DIST_2REF functions can be used to compute an oriented distance between two longitudes on the earth: the geo-points longitudes PT_COORD and REF_COORD. This distance can be assimilated to a coordinate within an interval [- π,π[in radian and whose origin is the longitude of the REF_COORD parameter. Negative values are to the West of this origin.

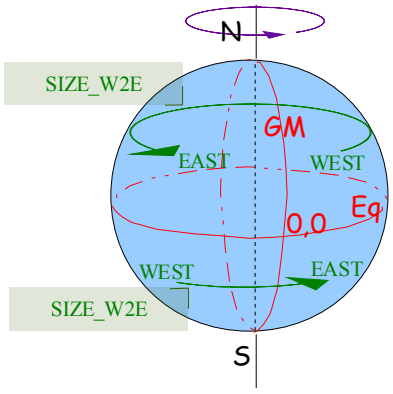


Illustration 8: Way to direct route

Caution: the longitudes of the PT_COORD and REF_COORD geo-points coordinates are normal **geographical coordinates**, in the [- π,π[interval, in radian and whose origin is the Greenwich meridian. Negative values are to the West of this origin.

SIZE_W2E is another function that calculate a distance or a size (notion of norm within interval [0,2π[radian) between two longitudes on the Earth: WEST_COORD%LON and EAST_COORD%LON. When moving from WEST_COORD%LON to EAST_COORD %LON, the choice between the two possible ways on the sphere will be from West to East linked to a direct trihedron (see *illustration 7*).

These functions are required because of the "side effects"[27] when the difference is calculated:

PT_COORD%LON – REF_COORD%LON

26 We name « minimax » the value of VAL according the bound interval [-LIM, LIM] (LIM positive):
- VAL if VAL ∈ [-LIM, LIM]
- -LIM if VAL < -LIM
- LIM if VAL > LIM

27 Because two distances exist (one on each side of the Earth), but also, because of the position of one point against the other, and these positions against the origin (Greenwich meridian) and the limits (- π ,π) of the values (modulo).

Name	Type[16] returned	Description
DIST_2REF_L	REAL	Return oriented distance between 2 longitudes COORD_LON and REF_LON
Input parameters	Type[16]	Description
COORD_LON	REAL	Longitude of a geopoint in geographical coordinates (in radian within [- π,π[).
REF_LON	REAL	Longitude (in radian within [-π,π[) of a reference geopoint in geographical coordinates, used as origin for the computed oriented distance.
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
DIST_2REF_S	REAL	Return oriented distance between longitude of 2 geopoints PT_COORD and REF_COORD
Input parameters	Type[16]	Description
PT_COORD	LOLA	Geopoint in geographical coordinates (longitude in radian within [- π,π[).
REF_COORD	LOLA	Reference geopoint in geographical coordinates, used as origin for the computed oriented distance.
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
DIST_2REF_V	REAL (:)	Return an array of oriented distances between longitude of an array of geopoints PT_COORD and one geopoint REF_COORD
Input parameters	Type[16]	Description
PT_COORD(:)	LOLA	Array of geopoints in geographical coordinates (longitudes in radian within [- π,π[).
REF_COORD	LOLA	Reference geopoint in geographical coordinates, used as origin for the computed oriented distance.
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
SIZE_W2E_L	REAL	Return size between 2 longitudes WEST_LON and EAST_LON in the defined right way *illustration 7*
Input parameters	Type[16]	Description
WEST_LON	REAL	Longitude (in radian within [-π,π[) called «WEST».
EAST_LON	REAL	Longitude (in radian within [-π,π[) called «EAST».
PI	REAL	Value of π (optional)

Name	Type[16] returned	Description
SIZE_W2E_S	REAL	Return size between 2 geopoints WEST_COORD and EAST_COORD in the defined right way *illustration 7*
Input parameters	Type[16]	Description
WEST_COORD	LOLA	Geopoint called «WEST» in geographical coordinates (longitude in radian within [- π,π[).
EAST_COORD	LOLA	Geopoint called «EAST» in geographical coordinates (longitude in radian within [-π,π[).
PI	REAL	Value of π (optional)

4.3.4. Table of «Types»:

Types	**Description**		**Modules**
LOGI	Booleans	Generic types	FORTRAN
INT	Integers		
REAL	Reals		
CHAR	Characters		
TYPE	Structures	ERROR	EGGPACK
		PGN	
		NBPTS	
		DELTA	
		RTETA	
		XY	
		PARAM_PROJ	
		DOMI	
		LOLA	EGGANGLES

PROC	Procedures	INFO_DOMI_PRINT	EGGPACK
		INFO_PP_PRINT	
		MAKDO	
FUNC	Functions	ANGLE_DOMAIN (_RS, _RV, _LOLAS, _LOLAV)	EGGANGLES
		VAL_LAT (_S, _V)	
		VAL_LON (_S, _V)	
		VAL_COORD (_S, _V)	
		LOLAR (_S, _V)	
		LOLAD (_S, _V)	
		COSIN_TO_ANGLE (_S, _V)	
		P_ACOS (_S, _V)	
		P_ASIN (_S, _V)	
		MINIMAX (_S, _V)	
		DIST_2REF (_L, _S, _V)	
		SIZE_W2E (_L, _S)	
		RETURN_PRINT	EGGPACK
		TYPE_PROJ	
		POLE_IS	
		STPL_LATLON_TO_RTETA (_S, _V)	
		STPL_RTETA_TO_LATLON (_S, _V)	
		STPL_XY_TO_RTETA (_S, _V)	
		STPL_RTETA_TO_XY (_S, _V)	
		REF_DATAS	
		XY_NEW_TO_STD_ORIGIN (_S, _V)	
		XY_STD_TO_NEW_ORIGIN (_S, _V)	
		LATLON_TO_XY (_S, _V)	
		XY_TO_LATLON (_S, _V)	
		MAP_FACTOR (_S, _V)	
		GN (_S, _V)	
		TILT (_S, _V)	EGGMRT
		ROTATE (_S, _V)	
		MEROTIL (_S, _V)	
		ANTI_TILT (_S, _V)	
		ANTI_ROTATE (_S, _V)	
		METILROT (_S, _V)	

5. Conclusion

The implementation of the various geometry packages have always reached its goals:

- to simplify the parameters that define domains
- a better consistency between parameters and the formulae that are used (i.e. Kl et φ_0)
- to reduce the number of errors, thanks to the above points
- specific bugs for the most atypical domains (South hemisphere) have been removed

"COMMON" Fortran has been suppressed

- an "ORIENTED OBJECT" pseudo approach applicable to more oriented OO languages
- the "externalisation" of functions in the shape of a package in the AROME-ALADIN model
- consequently, its use outside the model, by others applications (PINUTS, products from NCAR tools such as Chagal_MeteoFrance or Goeview …)
- its potential for the future (branching of the MRT projection)

Using the structures of Fortran 90 together with a functional approach have had a cost in memory space and computing time. Using MAKDO presents the biggest hindrance, but it is used only once (or not very often anyway) to create the domain.

The computing optimisation with 2D arrays (loop on the second dimension) along with the "vectorization" of almost all the functions, are at an affordable cost.

The creation of a domain has been automatically tested by a script for several cases with different longitudes and latitudes from the centre, for not only "normal" longitudinal extensions but also for complementary domains (see *Illustration 8*).

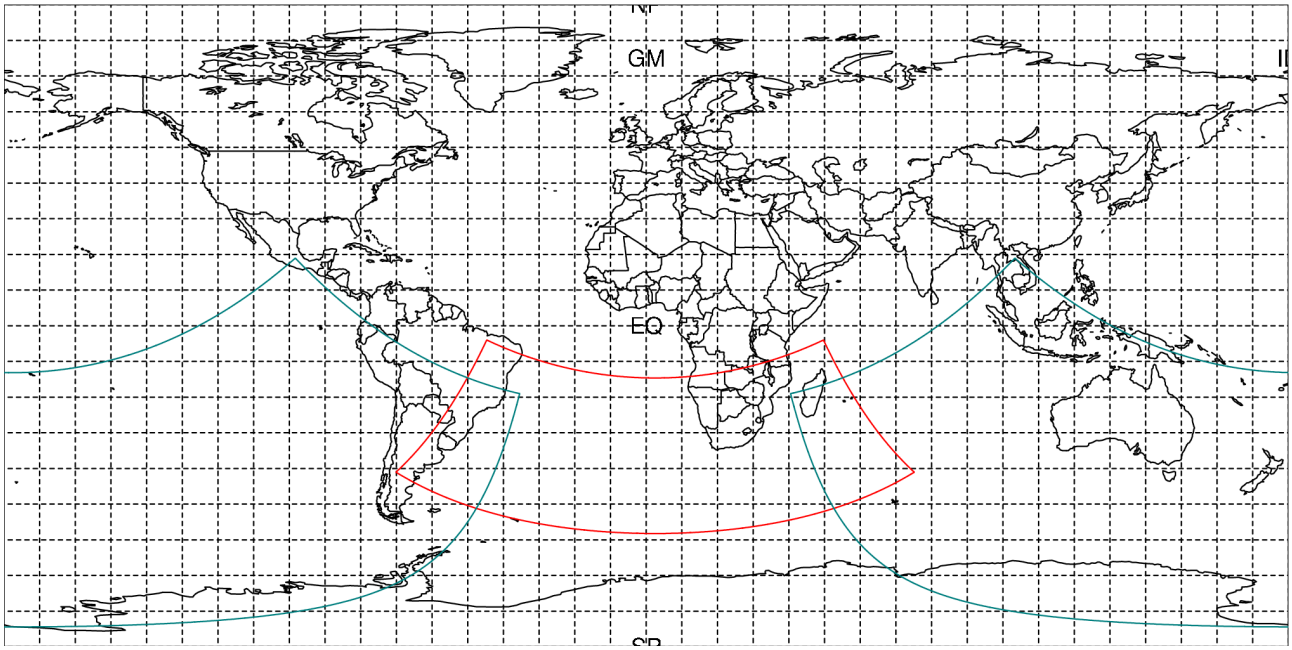


Illustration 8: Example about one test, one domain and its "counterpart"

The domains have been created using the DOMOLALO program from the PINUTS toolbox. This tool creates an ALADIN-AROME domain covering a rectangular area[28] defined by extreme North and South latitudes and extreme West and East longitudes whose values are listed below.

Domain	Red	Green
North	15°	15°
South	-50°	-50°
West	-50°	55°
East	55°	-50°

GEOVIEW which uses the NCAR libraries as been used for the drawings.

Potential evolution:

✓ MAKDO cold be taken out of the EGGPACK module as it is completely independent from any meteorological model whereas the routine is specific to the ALADIN-AROME models.

✓ the switch to a more friendly OO such as PYTHON and/or C++

✓ the possible use in a secant mode of these functions is not very useful right now even if it is not very difficult to implement them as the formulae are the same. Only the relation Kl=sin(φ_0) is no longer valid.

28 in «LATLON» format in Illustration 8

Index of illustrations

Index of tables