Practical implementation of the "LESIDG" functionality in Aladin

(SI scheme with non-uniform linearized map-factor)

P. Bénard, K. Yessad

file : designLESIDG.tex

13 October 2006

ABSTRACT

In this memo we choose, among several possible methods, one practical method for implementing in Aladin the LESIDG functionality, that is, the possibility of using a SI scheme with a non-uniform version of the linearized map-factor. The algorithms for the chosen method are then described. In Appendix A are some remarks by Karim on spectral computational aspects. In Appendix B is presented an algorithm for a robust fitting of the map factor.

1 Introduction

LESIDG is the name of the future logical switch for activating a version of the SI scheme of Aladin in which the squared linearized map-factor is not taken to a constant ($\mathbf{m}^{*2}$), as currently, but rather to a y-varying field $\overline{\mathbf{m}^2}(y)$ optimally fitted to the exact value $\mathbf{m}^2$. This functionality is therefore the LAM counterpart of the LSIDG functionality for the global stretched model. The aim of this functionality in a LAM is to prevent possible instabilities arising from the explicit treatment of quite large residuals linked to the map-factor linearization in the SI scheme, when the map-factor itself has big variations inside the LAM domain. Namely, a possible instability of the model with the current SI scheme is anticipated (through analyses) when using the NH version of Aladin with a large domain (Voitus, 2004).

Similarly to the global approach, the fit $\overline{\mathbf{m}^2}$ of the squared map-factor $\mathbf{m}^2$ is performed in the sub-space of the spectral space which is restricted the two first (largest scale) Fourier components, in order to keep tracatble computations. For the global LSIDG scheme, the fit was exact, due to the special form of the map-factor in the spherical-harmonics representation, but with the projections available in the Aladin model, the fit cannot be exact, and will therefore be only a best approximate.

Note that here we restrict the possibility of using the LESIDG functionality to the case of a Mercator projection (being rotated-tilted or not). The LESIDG functionality could be used for some particular cases of other projections existing in Aladin (unrotated Lambert or Polar Stereographic), but the benefit would be almost vanishing. For instance, for a Lambert projection m is a field which has variations along both the x and y coordinates of the LAM representation. Then a simple fit of m along a unique coordinate direction is not possible in the general case. However, we could imagine to allow the application of the LESIDG functionality for Lambert projection in the case of small domains when the reference longitude is at the center of the domain, because in this case, the variation of m is almost restricted to the y direction (if the domain is small enough, parallels are almost straigth lines along the x direction, and meridians straight lines almost along the y direction). But since the domain must be small for the map-factor to be reasonably fitted in the whole domain, the advantage of the functionality vanishes by nature.

Conversely, in the Mercator projection, the map-factor $\mathbf{m}$ is always dependant on the y direction only. We simply have :

$$\mathbf{m}(y) = \cosh\left(\frac{y}{a}\right) \tag{1}$$

where a is the earth radius, and y is the natural coordinate of the Mercator projection along the ordinate direction :

$$y = a \ln \left[\tan \left(\frac{\pi}{4} + \frac{\theta''}{2} \right) \right] \tag{2}$$

where θ'' is the apparent latitude of the projected point. If there is non tilting/rotation, then θ'' is the geographical latitude (for more details on rotated-tilted Mercator projection, see Bénard 2004).
The coordinate y has the dimension of a physical length and is the coordinate used along the apparent ordinate axis for the LAM domain (the x coordinate is simply $x = a\lambda''$ where λ'' is the apparent longitude). On the apparent Equator ($\theta'' = 0$), y is numerically coincident with distances measured directly on the ground of the (spherical) earth, e.g. :

$$dy|_{\theta''=0} = ad\theta'' \tag{3}$$

2 Choosing a method

The goal of LESIDG is two minimize the problems arising in the SI scheme from the explicit residuals of terms involving the map-factor. In the SI scheme, the map-factor is involved through its square. Moreover, we know that the stability requires that the linearized map-factor is always larger than the actual one. Hence three guidelines are followed :
(i) the total variation and the maximum value of **m** itself must be minimal inside the domain (for a given domain-size).
(ii) $\overline{\mathbf{m}^2}$ must be larger or equal than $\mathbf{m}^2$ in the whole domain.
(iii) $\overline{\mathbf{m}^2}$ must be optimally fitted to $\mathbf{m}^2$ by the two first Fourier components, in the sense that the maximum deviation of $(\overline{\mathbf{m}^2} - \mathbf{m}^2) > 0$ must be minimal.

As a consequence of the first point, the physical domain (C+I zones) must be centred on the apparent equator ($\theta'' = 0$) of the projection. This, in turn, implies that the map factor is 1 at the center of the physical domain (C+I), and has the same maximum value at the (apparent) "northest" and "southest" edges of the physical domain.

One practical problem for implementing LESIDG is that the map factor is defined (and has to be fitted) only over the physical domain (C+I zones), while the Fourier components are defined on the whole domain (C+I+E zones). This causes some difficulties for spectral computations in comparison to the global case. Assuming the (C+I+E) domain is defined by $y \in [-y_{\max}, y_{\max}]$, let us define an adimensional coordinate Y varying from 0 to 1 in the whole (C+I+E) domain :

$$Y = \frac{1}{2}\left(1 + \frac{y}{y_{\max}}\right) \qquad \Rightarrow \qquad Y \in [0, 1] \tag{4}$$

The line $Y = 0$ corresponding to the "apparent southest" edge of the total (C+I+E) domain , and the line $Y = 1$ corresponding to the "apparent northest" edge of the total (C+I+E) domain. In the spectral part of the model and in the spectral transforms, the y-direction Fourier components are defined over the interval $Y \in [0, 1]$, Therefore, $\overline{\mathbf{m}^2}$ writes :

$$\overline{\mathbf{m}^2} = \Re[a_0 + z_1 \exp(2i\pi Y) + z_2 \exp(4i\pi Y)] \tag{5}$$

In the latter expression, a_0 is a real number, while z_1 and z_2 are complex numbers. Similarly, a given field is defined in the spectral space by its representation in the Fourier basis :

$$\psi = \Re\left[\sum_{n=0}^{N} \widehat{\psi}_n \exp(2in\pi Y)\right] \tag{6}$$

where $\widehat{\psi}_n$ are complex numbers (except for zero, where it is real).

Let us now define the physical part of the domain (C+I) by $Y \in [0, 1-2\epsilon]$ (hence, ϵ is the half-width of the E zone in terms of the Y coordinate).

2.1 Multiplication by the squared map-factor in spectral space

For the global stretched model, due to the special form of the decomposition of $\mathbf{m}$ in the basis of spherical harmonics, the multiplication of a field by the squared map-factor in the spectral space is quite simple, as seen in Eq. (8) of Yessad and Bénard (1996) : for a given spectral component $\psi_{(m,n)}$ of a field ψ, the corresponding spectral component of $\left[\mathbf{m}^2\psi\right]_{(m,n)}$ involves only the neighbouring components $\psi_{(m,n-2)}$, $\psi_{(m,n-1)}$, $\psi_{(m,n)}$, $\psi_{(m,n+1)}$ and$\psi_{(m,n+2)}$. It is important to note that in the Legendre Polynomials representation, the spectral component for a given total-wave-number n is purely real. This contrasts with the Fourier component for a given zonal-wave-number m, which is complex by nature (this complex is however treated as a pair of real numbers labelled "m" and "-m" in ARPEGE). As a consequence, Eq. (8) in Yessad and Bénard (1996) is a purely real equation which indifferently applies to positive of negative values of m, resulting in a 5-diagonal form of the corresponding operator. This latter property is lost for Aladin if the squared map-factor has itself a truly complex Fourier representation, because real and imaginary parts of the map-factor and of the field then have to mix themselves in addition to the mixture of adjacent meridional wave-numbers. This in the general case, results in an operator which can be seen as 9-diagonal, and acting in a space which encompasses both real and imaginary parts of the spectral representation (for more details see Karim's note –in french– in Appendix A).

2.2 First method : blind coding ... but costly

If we would like to stick to the simple and direct methodology used in the global model for LSIDG, we would use the decomposition of $\overline{\mathbf{m}^2}$ given by (5) and then define directly the matrix operators required for the SI scheme, in the spectral space. These operators are twofold :
- multiplication of a field ψ by $\overline{\mathbf{m}^2}$: $\quad \psi \longrightarrow \overline{\mathbf{m}^2}.\psi$
- multiplication of a field ψ by the inverse of $\left(1-\alpha\overline{\mathbf{m}^2}\right)$: $\quad \psi \longrightarrow \left(1-\alpha\overline{\mathbf{m}^2}\right)^{-1}.\psi$
where α is a real number. These two operators appear in both H and NH cases.
However, with this method, the map factor is not a symmetric function in the interval [0,1], i.e. the minimum of $\mathbf{m}$ is not located at $Y = 0.5$. As a consequence, for the optimal fit of $\mathbf{m}^2$, (z_1, z_2) in (5) are true complex numbers as illustrated in Fig. 1. In this plot, for the sake of clarity, the fit is performed only with wave numbers 0 and 1 (i.e. $z_2 = 0$). The thick line is the squared map factor, and the two dashed lines are the "cos" and "sin" Fourier components which must be summed to obtain the optimal fit. The thin line is the optimal fit.

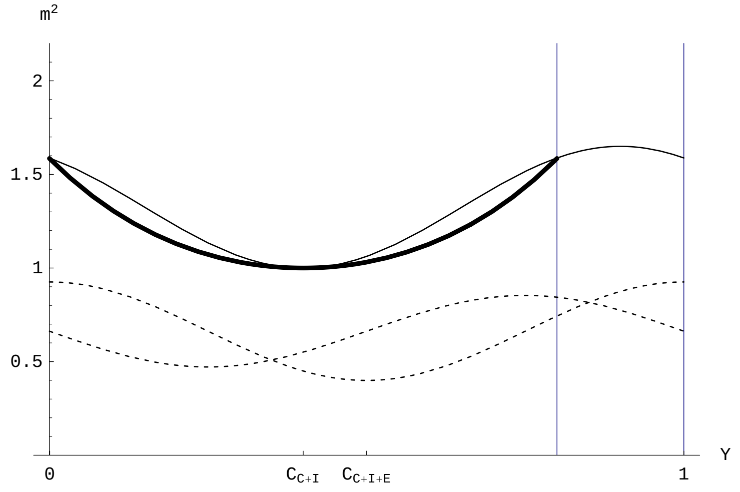

FIG. 1 – Sketch of the fit of $\mathbf{m}^2$ in the spectral space with the first method. The thick line is the squared map-factor $\mathbf{m}^2$, the thin line is the (one-component) optimal fit $\overline{\mathbf{m}^2}$. The two dashed lines below are the "sin" and "cos" Fourier components which must be summed to obtain the optimal fit.

In these conditions, as stated above, the computation of the second operator implies the inversion of 9-diagonal matrices of size 4N+1 (once again, for more details on these computational aspects, see Karim's note in Appendix A, in french). The computations in this case appear to be dramatically different from the global case, where the computations result in the inversion of 5-diagonal matrices of size 2N+1. The code for such a method is not ready, and preparing this code would represent a large work, and a large modification of the spirit of the method compared to the global version. Moreover, although this method would result only in "slightly" heavier computations (mostly in the setup), it woul require much more storage, since twice more matrices are needed, each of them being roughly twice bigger in size (9-diag instead of 5-diag).

2.3 Second method : *ad hoc* and ... a bit unsound

In order to escape the above difficulty, one could imagine to remove the problem at its origin, and accept to define the (C+I) domain as a bit unsymmetric with respect to the apparent equator, in order that the map-factor becomes symmetric when considered in the whole (C+I+E) domain. In such a method, the apparent equator of the projection would be located at $Y = 0.5 + \epsilon$. The (C+I) domain is then more extended toward the southern apparent hemisphere, as seen in Fig. 2. In this method, the minimum of the map-factor is realized at the center of the (C+I+E) domain, therefore the best fit of $\mathbf{m}^2$ on (C+I+E) would be achieved for a Fourier component containing only "cos" functions (by symmetry). As a consequence, the two abovementioned operators become of a similar nature than their global counterpart with LSIDG (i.e. the second operator requires inversion of 5-diag matrices with same dimension as in LSIDG).

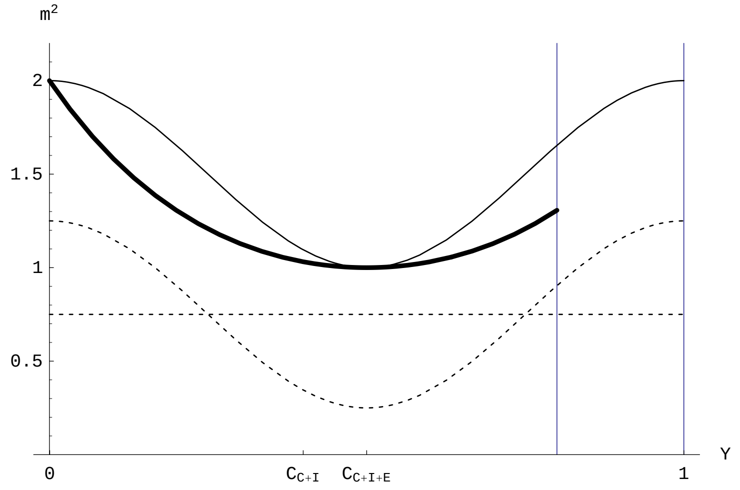

FIG. 2 – Same as Fig. 1, but for the second method. Note the fit is optimal only when considered on the (C+I+E) domain.

However, this method, although efficient in CPU and storage, has several disadvantages. First it implies that the ratio between the width of the C+I domain and C+I+E domain is known very early in the course of an application, that is, when EGGX is invoked for the first time to define the physical domain : we would have to anticipate the width of the E zone as soon as we decide on which physical domain we would like to run the LAM. This is not in agreement with the spirit of the system for designing domains and applications with Aladin, and could lead to practical problems in some cases (for instance, trying to evaluate the impact of broadening the E zone would imply to compulsorily change the location of the physical domain !). Morover, the physical domain would be unsymmetric around the apparent equator, thus violating the first requirement above. In addition, the fit obtained with "cos" components is optimal in the whole (C+I+E) zone, but there is no special reason why to include the E zone in the fitting area, and when considered in the (C+I) area only, this symmetric fit is no longer optimal. Of course, the relative width of the E zone is exaggerated in the figure, but this all these facts represent an inconvenience, especially if we notice that this dissymetry (or its larger domain, according to the adopted point of view) significantly deteriorates the quality of the fit, as can be seen by comparing to Fig. 1.

This method does not bring all the benefits we could expect from the implementation of the LESIDG functionality. It is constraining from a practical point of view, and finally rather inelegant. It should be chosen only in case nothing better is found.

2.4 third method : looks strange, but finally quite simple and efficient

In this method, we try to keep the advantage of a physical domain (C+I) centred on the equator of the projection, the advantage of a best quality fit, and the advantage of simple and minimally-modified SI operators.

For this, we apply a simple "translation" operator $\mathcal{T}$, to all fields $\widetilde{\psi^+}$ entering in the spectral space (which are indeed the RHSs of the SI system). This translation is the one which brings the equator of the projection [i.e. the center of the (C+I) domain] from its initial position $Y = 0.5 - \epsilon$ to the center of the (C+I+E) domain, at the position $Y = 0.5$. The representation of physical domains in the spectral space is illustrated in Fig. 3 and 4, for the untranslated and translated cases.

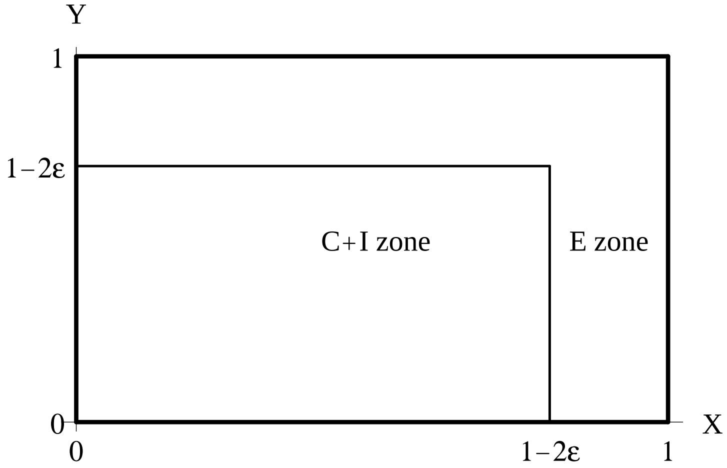

FIG. 3 – Representation of the physical space in the untranslated spectral-space.

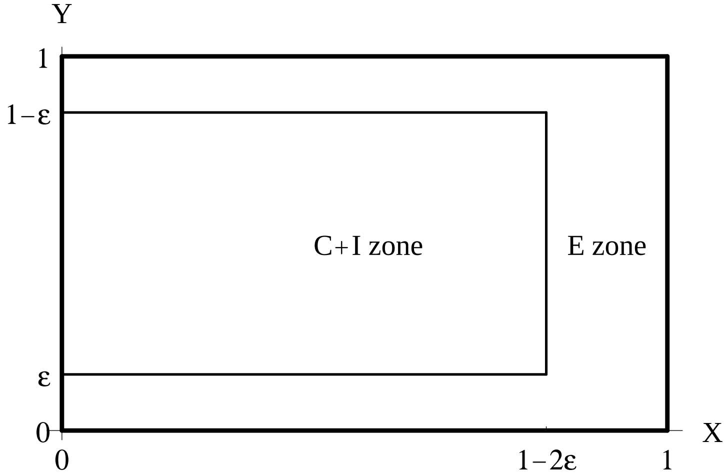

FIG. 4 – Representation of the physical space in the translated spectral-space.

Then in the spectral space, all computations are made with translated fields. Doing this, the fields become consistent with a translated version of $\mathbf{m}^2$ having its minimum value at the center of the Fourier domain $Y = 0.5$, hence being fitted exclusively with "cos"–type components. As a consequence, the operators involved in the SI scheme keep their relatively simple form, similar to the form occuring in the global LSIDG functionality. When the spectral computations are completed, the fields ψ^+ (and their derivatives) are translated back to their normal position by application of $\mathcal{T}^{-1}$.

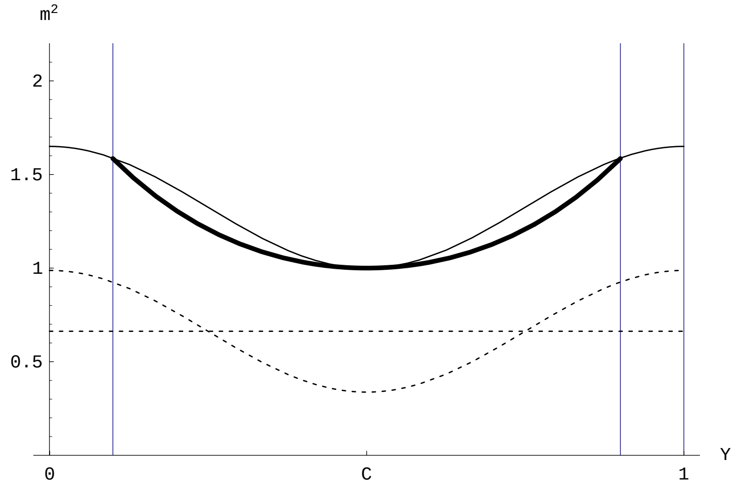

FIG. 5 – Same as Fig. 1, but for the third method.

The situation of the map-factor with respect to "cos" and "sin" Fourier components in the spectral space after the translation is illustrated in Fig. 5. A physical field which would exhibit a minimum at the center of the (C+I) zone in the physical space, would have its minimum at the center of the figure once in the (translated) spectral-space.
The spectral formulation of operators $\mathcal{T}$ and $\mathcal{T}^{-1}$ is quite simple and only involves a "local" mixing of "sin" and "cos" parts of each individual Fourier component $\psi(n,m)$, without any interaction with adjacent Fourier components $\psi(n',m')$. These operator can be anticipated as easy to code, and efficient to apply. Implementing the application of these operators to all fields at the very beginning (and very end) of spectral computations should also not be a big problem, due to their simplicity.

The reason for doing the translation in the spectral space rather than in the physical space, where it would seem to be "more natural" is threefold :
- The translation in physical space would probably be much harder to practically implement due to the parallel computing segmentation : this would require the knowledge of values for grid-points which are not necessarily in the same "packet". Conversely, in the spectral space, the translation process is purely "local" as mentioned above, and therefore, much easier to implement.
- If the translation was made in the physical space, then only translated fields would be available in the spectral space. Since the historical files are written at the level of spectral space and filled with spectral values, these files would necessarily contain translated fields, unless some special operations are performed in order to prevent this.
Doing the translations in the spectral space, although strange at first glance, is finally more natural, when considering globally all the constraints of the model.

A last comment concerns the interest of applying $\mathcal{T}$ "globally", i.e. to all spectral prognostic vectors, instead of "locally", i.e. only when needed. In effect we could imagine that we still continue to work with normal (untranslated) vectors in the spectral-space, and that we just apply locally the sequence $\mathcal{T}$, $\mathcal{T}^{-1}$ to the involved vector when an occurence of $\overline{\mathbf{m}^2}$ is encountered in the code of spectral computations (for one of the two operators mentioned at the beginning). This would minimize the places where the data is modified compared to the case LESIDG=.F. However, in terms of code-management, this would probably be not very interesting. The conceptual simplicity (readability) and the code modularity would become much poorer, and these parts of the code would become H/NH specific although there is no special reason for it. One could argue that it would be more efficient, at least if the occurences of $\overline{\mathbf{m}^2}$ operators are less numerous than the number of spectral prognostic variables. This is maybe true in the H mode, but not in NH mode, where many occurences of the map-factor are indeed encountered in spectral computations. Moreover, even in H mode, the benefit would be small because this represents a reduction of the CPU by a factor 2/3 only for this transformation the cost of which is anticipated as very small. Balancing the pros and contras, the solution of a global transformation to all vectors seems to be more appropriate and more rational.

3 Coding aspects (discretization aspects

3.1 Origin of indices in Fourier Transforms

The appropriateness of the implementation of the LESIDG functionality requires an accurate placement of the center of the physical domain (C+I) exactly at the same place than the center of the segments used in the Fourier tansforms : the translation operator $\mathcal{T}$ must be carefully defined in order to insure an accurate fit of the map-factor. In other words the success of LESIDG requires the accurate knowledge of the origin of the Y coordinate used in Fourier space.

The comments in Fourier transforms code (`FFT992`) indicate that the Fourier vector is : `[X(0), ..., X(N-1)]`, and that the `K`-eth "cos" Fourier functions is `COS(2*J*K*PI/N)` with `(J=0,...,N-1)`, thus suggesting that the Fourier segment actually starts at `J=0`. However, the problem is to know what is actually stored in the so-called quantity "`X(J=0)`", when performing the Fourier Transforms. Is it the first value in the (C+I) domain (the one called `X(JGL=1)` in the grid-point part), or is it the last value of the (C+I) domain (the one called `X(JGL=NDGUX)` in the grid-point part), or even something else ?

Logically, the origin of "sin" and "cos" Fourier components should be the position corresponding either to the indice $J = 0$ or $J = 1$ in grid-point part. Since the code is quite cumbersome on this point, we choose, by safety, to let this origin undetermined (i.e. tunable) and therefore, we define a specific variable `JORIG` which value should be either 0 or 1, and which indicates the index of the origin of Fourier segments in grid-point vectors.

4 Conclusions

The transposition of the LSIDG functionality from the global (stretched) model to the (projected) LAM model can be made straightforward if some special care is taken in order to allow a "symmetric" representation of the map-factor in the spectral space along the y direction. This can be easily achieved through additional translations of all spectral fields at the very beginning and near the end of spectral computations (but before writing historical files for the inverse translation).

In this case, the methodology of the LESIDG functionality becomes very similar to the one of the LSIDG functionality. The main difference is that the linearized map-factor $\overline{\mathbf{m}^2}$ is no longer equal to its exact counterpart, but only approximated through an optimal two-components fit. Due to the symmetry of $\overline{\mathbf{m}^2}$ over the y-Fourier domain, all operators involved remain 5-diagonal matrices with a level of complexity equal or smaller to that of the LSIDG scheme. Therefore, the same coding framework could be kept for the point of view of spectral linear operators.

References

Bénard, 2004 : New "Rotated/Tilted Mercator" geometry in Aladin. *Internal GMAP note, available at :*`http ://www.cnrm.meteo.fr/gmapdoc/IMG/ps/rotmer.ps`

Voitus, F., 2004 : Liens entre la géométrie horizontale et le schéma semi-implicite dans les modèles ARPEGE, ARPEGE/ALADIN, et ARPEGE/ALADIN-NH. *Rapport de Stage de Fin d'Etudes n° 851. Ecole Nationale de la Météorologie, Toulouse.*

Yessad, K, and P. Bénard, 1996 : Introduction of a local mapping factor in the spectral part of the Météo-France global variable mesh numerical forecast model. *Q. J. R. Meteorol. Soc.*,**122**, 1701–1719

Appendix A : Karim's computation sheet

Preliminary note : Here MAPapprox is the *SQUARE* of the approximated map-factor.

A.1 : First method

```
MAPapprox = MAP0 + MAP1C * C1 * cos(2 PI Y)
                 + MAP1C * S1 * sin(2 PI Y)
                 + MAP2C * C2 * cos(4 PI Y)
                 + MAP2C * S2 * sin(4 PI Y)
avec C1**2 + S1**2 = 1; C2**2 + S2**2 = 1
et deux autres relations liant (C1, C2, S1, S2)
puisque (C2,S2) concerne des angles doubles par
rapport à (C1,S1)

Apres moult calculs cela doit nous donner
(je ne garantis pas pleinement le signe des termes
en S1 et S2 mais ce qui est certain c'est qu'on
retranche le terme en n+1 (resp. n+2) au terme en n-1
(resp. n-2).

[MAPapprox F]cc (m,n) = MAP0 Fcc(m,n)
 + 0.5 MAP1C * C1 * [ Fcc(m,n-1) + Fcc(m,n+1) ]
 + 0.5 MAP2C * C2 * [ Fcc(m,n-2) + Fcc(m,n+2) ]
 + 0.5 MAP1C * S1 * [ Fcs(m,n-1) - Fcs(m,n+1) ]
 + 0.5 MAP2C * S2 * [ Fcs(m,n-2) - Fcs(m,n+2) ]

Meme type de formule pour le premier index egal a 's'
au lieu de 'c' (qui fournit [MAPapprox F]sc (m,n)).

[MAPapprox F]cs (m,n) = MAP0 Fcs(m,n)
 + 0.5 MAP1C * C1 * [ Fcs(m,n-1) + Fcs(m,n+1) ]
 + 0.5 MAP2C * C2 * [ Fcs(m,n-2) + Fcs(m,n+2) ]
 - 0.5 MAP1C * S1 * [ Fcc(m,n-1) - Fcc(m,n+1) ]
 - 0.5 MAP2C * S2 * [ Fcc(m,n-2) - Fcc(m,n+2) ]

Meme type de formule pour le premier index egal a 's'
au lieu de 'c' (qui fournit [MAPapprox F]ss (m,n)).

Les commentaires qu'on peut faire sur ces formules
sont les suivants:
- il n'y a plus separabilite des termes en Fcc et Fcs
   (resp. Fsc et Fss) qui sont maintenant couples.
- Les termes en S1 et S2 sont affectes du signe +
  dans la premiere expression et du signe - dans la
  seconde.

Matriciellement cela s'ecrit, pour "m" donne

[MAPapprox F] = MAPapproxSYM (VECFcc,VECFcs,VECFsc,VECFcs)
 + MAPapproxANT (-VECFcc,VECFcs,-VECFsc,VECFss)

ou par exemple VECFcc est le vecteur des coefficients Fcc(m,n)
MAPapproxSYM est la matrice symetrique:
- de diagonale principale comportant des MAP0
- de premiere diagonale laterale comportant des 0.5*MAP1C*C1
```

- de seconde diagonale laterale comportant des 0.5*MAP2C*C2

MAPapproxANT est la matrice antisymetrique:

- de diagonale principale comportant des zeros
- de premiere diagonale laterale comportant des +/- 0.5*MAP1C*S1
- de seconde diagonale laterale comportant des +/- 0.5*MAP2C*S2

Pour les multiplications spectrales par MAPapprox, cela se gere assez bien car au lieu d'avoir une seule multiplication matricielle par MAPapproxSYM (appel a MXPTMA) on en a 2 (2 appels a MXPTMA, l'un avec MapproxSYM, l'autre avec MapproxANT, au prix de duplications de tableaux)

Pour les inversions de facteurs du style I - alpha MAPapprox c'est une toute autre paire de manche car au lieu d'avoir une matrice penta-diagonale de dimension (NISNAX(m)+1)**2 a inverser et a appliquer a 4 vecteurs de longueur NISNAX(m)+1, on a une matrice a 9 diagonales (et non symetrique de surcroit car il y a symetrie de certaines diagonales et antisymetrie des autres), de dimension 4 * (NISNAX(m)+1)**2, qu'on applique a 2 vecteurs de longueur 2*(NISNAX(m)+1). Ceci oblige a reecrire tout le code pour les calculs matriciels (routines SUHER, SUHES, MXTURE, MXTURS) et par voie de consequence toutes les interfaces a ces calculs. Ceci nous vaut egalement la presence de tableaux plus volumineux.

A.2 : Second method

(not documented).

A.3 : Third method

Il existe une transformation spectrale equivalente a une translation meridienne point de grille de longueur E/2 (application d'une transformation matricielle utilisant des coefficients Cn=cos(2 PI n ee) et Sn=sin(2 PI n ee) ou ee = 0.5 * (largeur meridienne de E) / (largeur meridienne de C+I+E), "ee" etant exprimee dans la meme coordonnee que Y, c.a.d. une coordonnee apparente qui suppose l'equidistance des points; par exemple si on a 100 "latitudes" dont 10 pour la zone d'extension alors ee=0.05): au prix d'un changement de variable dans l'espace spectral on se ramenerait au cas S1=S2=0, mais cela voudrait dire qu'il faudrait faire une transformation vectorielle sur tous les champs au debut de ESPCSI/ESPNHSI, et refaire la transformation inverse a la fin de ces memes routines.

Dans l'espace spectral translate, on a:

```
MAPapprox = MAP0 + MAP1 *  cos(2 PI Y)
                                + MAP2 *  cos(4 PI Y)
```

Soit F un vecteur (non translaté) dans l'espace spectral, avec pour chaque nombre d'onde (m,n) les quatres coefficients reels:

```
F(m,n) =  [Fcc(m,n); Fcs(m,n); Fsc(m,n); Fss(m,n)]
```

Les coefficients du vecteur translate G(m,n)
seront alors:

```
Gcc(m,n) = cos(2 PI n ee) Fcc(m,n) - sin(2 PI n ee) Fcs(m,n)
Gcs(m,n) = sin(2 PI n ee) Fcc(m,n) +cos(2 PI n ee) Fcs(m,n)

Gsc(m,n) = cos(2 PI n ee) Fsc(m,n) - sin(2 PI n ee) Fss(m,n)
Gss(m,n) = sin(2 PI n ee) Fsc(m,n) +cos(2 PI n ee) Fss(m,n)
```

La multiplication par M^2 pour le vecteur G s'ecrit alors:

```
[MAPapprox G]xx (m,n) = MAP0 Gxx(m,n)
 + 0.5 MAP1  * [ Gxx(m,n-1) + Gxx(m,n+1) ]
 + 0.5 MAP2  * [ Gxx(m,n-2) + Gxx(m,n+2) ]
```

ou xx represente indifferemment: cc, cs, sc ou ss.

Les commentaires qu'on peut faire sur cette formule
sont les suivants:
- il y a separabilite totale des termes en Fcc, Fcs, Fsc et Fss
  pour les opérateurs faisant intervenir MAPapprox

Matriciellement cela s'ecrit, pour "m" donne:

```
[MAPapprox G]xx = MAPapproxSYM (VEC_Gxx)
```

ou VEC_Gxx est le vecteur des coefficients Gxx(m,n),
MAPapproxSYM est la matrice symetrique:
- de diagonale principale comportant des MAP0
- de premiere diagonale laterale comportant des 0.5*MAP1
- de seconde diagonale laterale comportant des 0.5*MAP2

Pour les multiplications spectrales par MAPapprox, cela
se gere bien.

Pour les inversions de facteurs du style (I - alpha MAPapprox)
cela se gere bien aussi.

Ensuite, vers la fin des calculs spectraux , il faut revenir
aux vecteurs non translatés, ce qui se fait par la translation
inverse.

A.3 : Other computational aspects and conclusions

L'essentiel des calculs concernant la prise en compte de MAPapprox
se fait dans l'espace spectral, qui ne connait pas le decoupage
C+I+E. Il y a quelques multiplications a prevoir dans
l'espace point de grille: par exemple pour le modele hydrostatique

```
la linearisation fournit un terme en MAP_REFSI * D', mais la
partie explicite des calculs point de grille ne connait que la
divergence geographique D, qui vaut D = M**2 * D'
Donc MAP_REFSI * D' = (MAP_REFSI / M**2) * D
MAP_REFSI est la valeur de reference de MAP=M**2 pour le semi-implicite.

Dans ARPEGE:
 LSIDG=F: MAP_REFSI=c**2, donc MAP_REFSI * D' = (c**2/M**2)*D
          C'est donc (c**2/M**2)*D qu'on trouve dans les calculs pdg.
 LSIDG=T: MAP_REFSI=M**2, donc MAP_REFSI * D' = D
          C'est donc D qu'on trouve dans les calculs pdg.

Dans ALADIN:
 RSTRET, c'est la valeur maximale du facteur d'echelle
 sur  C+I
 LESIDG=F: cf. LSIDG=F d'ARPEGE
 LESIDG=T: MAP_REFSI=MAP_approx,
  donc MAP_REFSI * D' = (MAP_approx/M**2)*D
  C'est donc le facteur (MAP_approx/M**2) qui va apparaitre
  dans les calculs point de grille, ce qui suppose de connaitre
  M (calcule dans le tableau GM) et MAP_approx (a calculer
  et a stocker dans un tableau SI_GMAPP egalement
  dimensionne avec NGPTOT).
  Comme les calculs point de grille ne sont faits que sur C+I
  il n'y a pas besoin de calculer la valeur point de grille
  de MAP_approx sur la zone "E" et de toute facon la valeur
  point de grille va etre calculee avec la formule utilisant
  les MAP0, MAP1 et MAP2.
  Pour M, on n'a pas plus besoin de la representation point
  de grille dans la zone "E", pas plus qu'on en a besoin pour
  LESIDG=F ou dans le restant des calculs.
```

5 Appendix B : Algorithm for the fit of $\mathbf{m}^2$ by $\overline{\mathbf{m}^2}$

We assume that the relevant translation has been performed at the beginning of spectral computations, as explained in section 2, thus insuring that the physical field $\mathbf{m}^2$ is centered in the space of Fourier segments defined by $Y \in [0,1]$. Therefore, $Y = 0.5$ is assumed to correspond to the place where $\mathbf{m}^2$ reaches its minimum. However, the proposed algorithm is designed in view of beint quite robust and to garantee a proper fit even if this latter condition is not fulfilled exactly. As a consequence, we will assume, as a starting hypothesis that the map factor is "almost" centered in the (C+I) domain.
As mentionned in section 3 a "universal" solution is adopted here in order to garantee an easy tuning of the scheme with respect to the origin of Fourier segments in grid-point vectors : a tunable value of the origin `JORIG` for the Fourier segment is defined as an extra variable of the problem. The value of `JORIG` may be either 1 or 0.
A last remark before beginning the design of the fitting algorithm is that the proposed algorithm should be appropriate (i.e. should provide a relevant fit) even if the domain is not correctly centered at the apparent equator of the tilted Mercator projection. This last requirement is imposed for the sake of robustness, because the environnement of Aladin allows bugs or "boulettes" to be made by users, and it is preferible to have robust algoritms in NWP.
The data of the problem are :

- `JORIG`, which is a "tuning" parameter allowed to be 0 or 1.
- `NDGL, NDGUX`, the dimensioning parameters for (C+I+E) and (C+I) domains.
- $\mathbf{m}^2_j$, the list of $\mathbf{m}^2$ values on the latitudes of the (C+I) domain $j \in \{1, 2, ..., \mathtt{NDGUX}\}$

From now on, `JGL` always represents the index used in Aladin for describing the (C+I) domain latitudes, with $\mathtt{JGL} \in (1, 2, ..., \mathtt{NDGUX})$. The non-translated coordinate Y introduced in the above sections, may be defined, in terms of model variables, by :

$$Y^{NT}(\mathtt{JGL}) = \frac{\mathtt{JGL} - \mathtt{JORIG}}{\mathtt{NDGL}} \tag{7}$$

This coordinates always varies between 0 and 1 in the whole Fourier segment, and is 0 at the point corresponding to the index `JORIG`.
The real parameter ϵ introduced above writes :

$$\epsilon = \frac{1}{2}\left(\frac{\mathtt{NDGL} - \mathtt{NDGUX} + 1}{\mathtt{NDGL}}\right) \tag{8}$$

and the translated coordinate which we will work with, is defined by :

$$Y^T(\mathtt{JGL}) = \left[Y^{NT}(\mathtt{JGL}) - Y^{NT}(\mathtt{JGL} = 1) + \epsilon\right] = \frac{\mathtt{JGL} - 1}{\mathtt{NDGL}} + \epsilon \tag{9}$$

The (C+I) domain is defined by $Y^T \in [\epsilon, 1 - \epsilon]$.

We want to optimally fit $\mathbf{m}^2$ by a combination of functions $\{1,\ \cos 2\pi Y^T,\ \cos 4\pi Y^T\}$, i.e. we want $(a_0,\ z_1\ , z_2)$ in (5) to be three real numbers. The constraints are that $\overline{\mathbf{m}^2}$ is always bigger than $\mathbf{m}^2$, and that the maximum (positive) deviation $(\overline{\mathbf{m}^2} - \mathbf{m}^2)$ is as small as possible.
We rewrite (5) in the following form :

$$\overline{\mathbf{m}^2}(a, b, r, Y^T) = a + b\left[\cos(2\pi Y^T) + r\cos(4\pi Y^T)\right] \tag{10}$$

The values taken by $\overline{\mathbf{m}^2}$ on the points of the physical grid are :

$$\overline{\mathbf{m}^2}_{\mathtt{JGL}}(a, b, r) = a + b\left[\cos 2\pi Y^T(\mathtt{JGL}) + r\cos 4\pi Y^T(\mathtt{JGL})\right] \tag{11}$$

5.1 Positiveness of $(\overline{\mathbf{m}^2} - \mathbf{m}^2)$ at the edges

First, we notice that in case of an exact centering, the optimal fit is always obtained when $\overline{\mathbf{m}^2}$ is equal to $\mathbf{m}^2$, at the edges of the (C+I) domain, as illustrated in Fig. 6.
This latter remarks allows to write a first constraint which relates a to (b, r) for an exact centering :

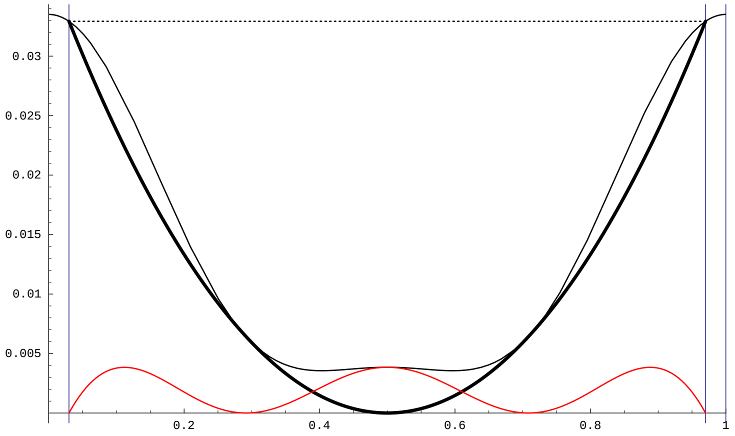

FIG. 6 – Case of an exactly centered field $\mathbf{m}^2$. The domain represented horizontally is the total domain (C+I+E), i.e. $Y \in [0,1]$, the two vertical lines show the edges of the (C+I) domain. Thick line : $(\mathbf{m}^2 - 1)$; thin line $(\overline{\mathbf{m}^2} - 1)$, defined in the total domain ; the dashed lines joins the two maximums of $(\mathbf{m}^2 - 1)$. The red (bottom) curve is the residual $(\overline{\mathbf{m}^2} - \mathbf{m}^2)$. The values of domain parameters are NDGL=200, NDGUX=89,

$$a(b,r) = \mathbf{m}_1^2 - b\,(\cos 2\pi\epsilon + r\cos 4\pi\epsilon) = \mathbf{m}_1^2 - \overline{\mathbf{m}^2}_1(0,b,r) \tag{12}$$

As outlined above, when using the LESIDG functionality, the only allowed projection is the rotated Mercator projection. In addition, at the level of domain design (EGGX), this projection imposes that the physical domain (C+I) is centered around the apparent equator :

$$\min_{(\mathrm{C+I})}(y) = -\max_{(\mathrm{C+I})}(y) \tag{13}$$

or equivalently, the value of the map factor is exactly 1 for $y = 0$. However, an imperfect fulfilment of this property cannot be totally excluded, due to roundoff error, or to any other cause. Hence, for the sake of robustness, it is preferible to assume that the condition is not necessarily fulfilled exactly. In such a case of an imperfect centering of $\mathbf{m}$ around the physical domain, we notice that the optimal fit is always obtained when the deviation $(\overline{\mathbf{m}^2} - \mathbf{m}^2)$ vanishes at the edge of the (C+I) domain where $\mathbf{m}$ is the largest, as illustrated in Fig. 7.

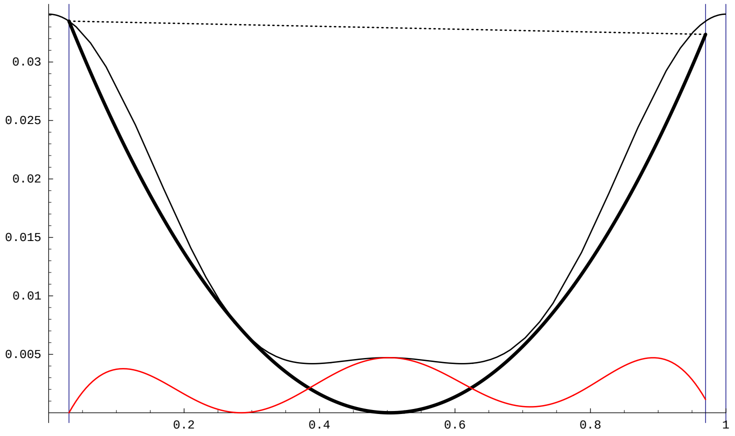

FIG. 7 – Case where the physical domain is not correctly centered. Same conventions as in Fig. 6.

Therefore, for the sake of robustness, we rewrite (12) in a more general form :

$$a(b,r) = \max\left[\mathbf{m}_1^2 - \overline{\mathbf{m}^2}_1(0,b,r)\ ,\ \ \mathbf{m}^2_{\texttt{NDGUX}} - \overline{\mathbf{m}^2}_{\texttt{NDGUX}}(0,b,r)\right] \tag{14}$$

A similar remark occurs in case of an imperfect translation $\mathcal{T}$. In this case, the map factor is symmetric in the (C+I) domain, but its minimum does not coincide with the one of "cos" functions, as illustrated in Fig. 8.

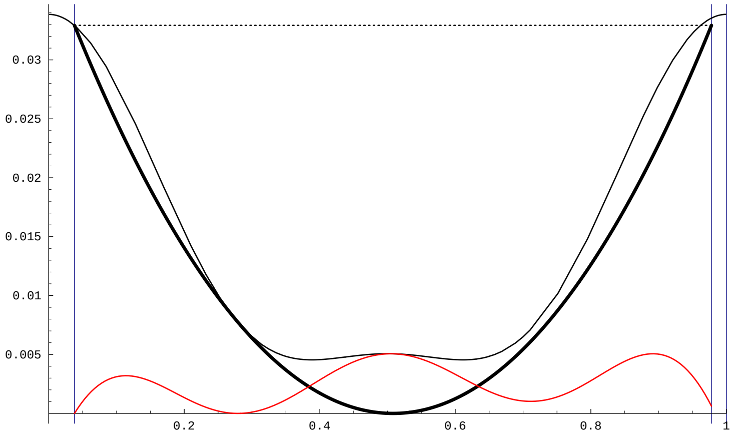

FIG. 8 – Case where the translation $\mathcal{T}$ is not correct. Same conventions as in Fig. 6.

Nevertheless, the above definition of a insures that the deviation $(\overline{\mathbf{m}^2} - \mathbf{m}^2)$ is always positive even at both edges, in any of these two pathological situations.

5.2 Optimal fit $(a,\ b)$ for a given value of r

In this sub-section we assume a given value of r, and we seek the optimal value of b , i.e. the one which insures that the maximum deviation of $\delta = (\overline{\mathbf{m}^2} - \mathbf{m}^2)$ is as small as possible, while positive everywhere. For a given b, we determine a using (14), which insures positiveness of δ at the edges (C+I) domain. The principle is that if b increases, the minimum value of δ becomes negative, and vice-versa. For $b_1 = 0$ the minimum value δ_{min} of δ on the (C+I) domain (excluding the two edges) is positive. For large values of b, as e.g. $b_2 = 2(1 + \mathbf{m}^2_{\text{max}})$, the value of δ_{min} is negative. Therefore, we can find the optimal value of b by a simple dichotomic process, starting with these two values (b_1, b_2) stopping when the residual is smaller than a chosen value. Then take the last value of b which insures a positive value of δ_{min}.
Finally, this process provides tho optimal $b(r)$ for the given value of r.

5.3 Optimal fit $(a,\ b,\ r)$

We define $\delta_{\text{max}}(r)$ as the maximum (positive) value reached by δ for a given value of r , when choosing b as $b(r)$ and a as $a(b,r)$. The function $\delta_{\text{max}}(r)$ is highly convex, and has a unique minimum, as illustrated in Fig. 9.
We start from two values r_1 and r_4 which are certainly smaller and larger than the optimum and we apply an iterative process of descent along the gradient to find this optimum. Since the function $\cosh(x)$ is more flat than the function $\cos(2\pi x)$ at $x = 0$, the optimum value of r is always positive, hence we can take $r_1 = 0$. At the other edge, we take $r_4 = (\texttt{NDGUX}/\texttt{NDGL})$, which is always larger than the optimal value. Then we define $r_2 = (r_1 + r_4)/2$ and $r_3 = r_2 + 0.01(r_1 - r_4)$. We then replace r_1 by r_2 or r_4 by r_3, depending on the sign of $\delta_{\text{max}}(r_3) - \delta_{\text{max}}(r_2)$. This process converges quickly, and is robust for any configuration of the data.

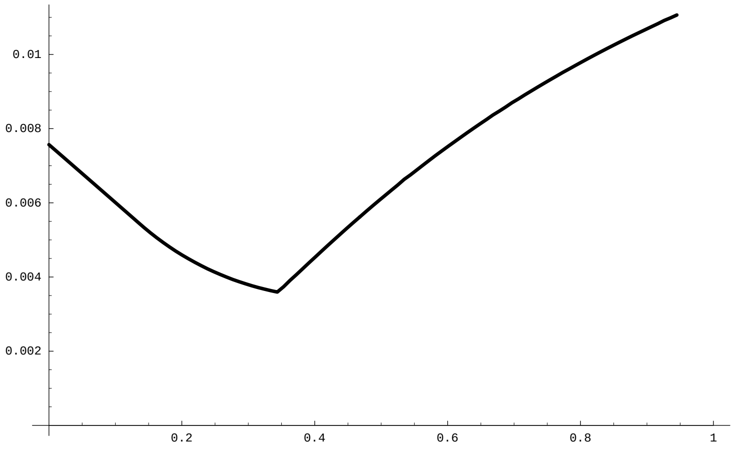

FIG. 9 – Typical shape of the function $\delta_{\max}(r)$. The configuration is as in Fig. 6