SPECTRAL TRANSFORMS IN THE CYCLE 46T1R1 OF ARPEGE/IFS.

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

February 12, 2019

Abstract:

ARPEGE/IFS, ALADIN and AROME are spectral models, so this documentation has for purpose to describe the spectral transforms done. The present note has for aim to give a brief summary of the spectral method (how to compute Legendre polynomials, spectral transforms) and to describe parts of the code performing spectral transforms (organigramme).

Résumé:

ARPEGE/IFS, ALADIN et AROME sont des modèles spectraux, en conséquence cette documentation a pour but de décrire les transformées spectrales nécessaires pour passer de l'espace spectral vers l'espace point de grille. On y fait un bref rappel de la méthode spectrale (comment calculer les polynômes de Legendre et les transformées spectrales). On y décrit les parties de code faisant des transformées spectrales, avec fourniture de quelques organigrammes.

Contents

1 Introduction.

ARPEGE/IFS, ALADIN and AROME are spectral models. One part of computations is done in spectral space (semi-implicit scheme, horizontal diffusion scheme for model integration), the other part in grid-point space on a grid defined by a Gaussian quadrature if global model, by the plane projection if LAM model. So it is necessary to perform spectral transforms from spectral space to grid-point space or vice-versa. The present note has for aim to give a brief summary of the spectral method and to describe parts of the code performing spectral transforms. Spectral method is not described with too many details: for more details on can report to (Rochas and Courtier, 1992, note ARPEGE nr 30, in French) or to (Temperton, 1991).

For a global spectral model, spectral transforms are a combination of a Legendre transform and a Fourier transform. A spectral limited-area model like ALADIN uses a double Fourier representation for spectral fields.

∗ **Fast Legendre transforms (FLT):** They are now implemented and use specific routines. They are not extensively documented there; one can refer to section 2 of (Wedi et al., 2013).

2 Theoretical aspects.

2.1 Notations.

- a is the Earth mean radius. Equations are written for models with thin layer approximation and deep-layer effects are ignored.
- Θ is the latitude on the Gaussian collocation grid.
- $\mu = \sin\Theta$.
- Λ is the longitude on the Gaussian collocation grid.
- n is the total wave number, m is the zonal wave number.
- M is the mapping factor.
- $\mathbf{V}$ is the horizontal geographical wind. Its zonal component is U. Its meridian component is V. Reduced components $(U^{'}, V^{'})$ are linked to unreduced components (U, V) by relationships $U = M * U^{'}$, $V = M * V^{'}$.
- D is the unreduced divergence of horizontal wind, $D^{'}$ is the reduced divergence: $D = M^2 * D^{'}$.
- ζ is the unreduced vorticity of horizontal wind, $\zeta^{'}$ is the reduced vorticity: $\zeta = M^2 * \zeta^{'}$.
- ∇^2 is the horizontal unreduced Laplacian operator and $\nabla^{'2}$ its reduced counterpart: $\nabla^2 = M^2 * \nabla^{'2}$.
- x and y are coordinates on the plane projection (LAM models).

2.2 Spectral representation of a scalar field.

$*$ Spherical geometry.

A field f can be decomposed into its spectral components according to the following expression:

$$f(\Lambda,\Theta) = \sum_{m=-\infty}^{m=\infty} \sum_{n=|m|}^{n=\infty} f_{(n,m)} P_{(n,m)}(\mu) \exp(im\Lambda) \tag{1}$$

$f_{(n,m)}$ are spectral coefficients. $P_{(n,m)}(\mu)$ are the first kind normalised polynomials of Legendre and are normalised by the following relationship:

$$\frac{1}{2}\int_{\mu=-1}^{\mu=1} (P_{(n,m)}(\mu))^2 d\mu = 1 \tag{2}$$

That yields the following expression for $P_{(n,m)}(\mu)$, if m is a positive integer:

$$P_{(n,m)}(\mu) = \frac{1}{2^n n!}\sqrt{(2n+1)\frac{(n-m)!}{(n+m)!}}(1-\mu^2)^{m/2}\frac{d^{n+m}}{d\mu^{n+m}}(\mu^2-1)^n \tag{3}$$

For a real scalar field, $P_{(n,-m)}(\mu)$ is set equal to $P_{(n,m)}(\mu)$, $f_{(n,-m)} = \overline{f}_{(n,m)}$, so only the real and imaginary parts of $f_{(n,m)}$ for $m \geq 0$ are computed and stored. The algorithm of computation of $P_{(n,m)}(\mu)$ is described in appendix 3.

$*$ LAM models with plane projection.

Coordinates x and y are assumed to vary between 0 and 1 in the limited-area domain. A bi-periodic field f (assumed to match $f(1,0) = f(0,0)$ and $f(0,1) = f(0,0)$) can be decomposed into its spectral components according to the following expression:

$$f(x,y) = \sum_{m=-\infty}^{m=\infty} \sum_{n=-\infty}^{n=\infty} f_{(n,m)} \exp(i2\pi mx) \exp(i2\pi my) \tag{4}$$

$f_{(n,m)}$ are spectral coefficients.

2.3 Truncation.

$*$ Spherical geometry.

In practical the expression of f is limited to a finite set of harmonics corresponding to $0 \leq n \leq N_{\rm s}$ and $-n \leq m \leq n$. That defines a triangular truncation $N_{\rm s}$. Fourier representation of the transmission coefficients for simplified physics is a very low truncation $N_{\rm tc}$ always lower than the model truncation $N_{\rm s}$. From now on formulae and algorithms will be written with truncation $N_{\rm s}$. So equation (1) becomes:

$$f(\Lambda,\Theta) = \sum_{m=-N_{\rm s}}^{m=N_{\rm s}} \sum_{n=|m|}^{n=N_{\rm s}} f_{(n,m)} P_{(n,m)}(\mu) \exp(im\Lambda) \tag{5}$$

Due to the aforementioned properties of $P_{(n,m)}$, expression of f becomes for a real scalar field:

$$f(\Lambda,\Theta)=\sum_{m=0}^{m=N_{\rm s}}\sum_{n=|m|}^{n=N_{\rm s}} f_{(n,m)}P_{(n,m)}(\mu)\exp\left(im\Lambda\right) \tag{6}$$

Some other types of truncations exist (for example rhomboidal truncation). Such truncations will not be detailed there.

∗ LAM models with plane projection.
Elliptic truncation: in practical the expression of f is limited to a finite set of harmonics corresponding to: $(n/N_{\rm s})^2+(m/N_{\rm ms})^2\leq 1$.
It is also possible (not coded) to use a rectangular truncation: $n\leq N_{\rm s}$ and $m\leq N_{\rm ms}$.

2.4 Horizontal derivatives.

∗ Meridian derivative relative to Gaussian latitude Θ in spherical geometry: for a variable f, meridian derivative is discretised in spectral space by the following formula:

$$\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)_{(n,m)}=-(n-1)e_{(n,m)}f_{(n-1,m)}+(n+2)e_{(n+1,m)}f_{(n+1,m)} \tag{7}$$

where $e_{(0,0)}=0$ and:

$$e_{(n,m)}=\sqrt{\frac{n^2-m^2}{4n^2-1}} \tag{8}$$

∗ Zonal derivative relative to Gaussian longitude Λ in spherical geometry: for a variable f, zonal derivative is discretised in spectral space by the following formula:

$$\left(\frac{\partial f}{\partial\Lambda}\right)_{(n,m)}=imf_{(n,m)} \tag{9}$$

Such a derivation can be done on Fourier coefficients by a multiplication by im.

∗ Derivatives for LAM models: example is given for zonal derivative; for a variable f, zonal derivative is discretised in spectral space by the following formula:

$$\left(\frac{\partial f}{\partial x}\right)_{(n,m)}=i2\pi mf_{(n,m)} \tag{10}$$

∗ Horizontal laplacian in the computational sphere for spherical geometry: for a variable f, expression of the laplacian is:

$$\left(\nabla'^2 f\right)_{(n,m)}=\left(\frac{1}{(a\cos\Theta)^2}\frac{\partial^2 f}{\partial\Lambda^2}\right)_{(n,m)}+\left(\frac{1}{a^2\cos\Theta}\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} \tag{11}$$

One can show (for more details see appendix 1) that its discretisation writes:

$$\left(\nabla'^2 f\right)_{(n,m)}=-\frac{n(n+1)}{a^2}f_{(n,m)} \tag{12}$$

∗ Horizontal laplacian in the computational sphere for LAM models: for a variable f, expression of the laplacian is:

$$\left(\nabla'^2 f\right)_{(n,m)}=\left(\frac{\partial^2 f}{\partial x^2}\right)_{(n,m)}+\left(\frac{\partial^2 f}{\partial y^2}\right)_{(n,m)} \tag{13}$$

2.5 Spectral relationship between (divergence, vorticity) and (velocity potential, stream function).

- Relationships between: reduced divergence and velocity potential χ; reduced vorticity and stream function ψ:

$$D'=\nabla'^2\chi\ ;\quad \zeta'=\nabla'^2\psi$$

- These relationships can be rewritten, for each complex spectral component (spherical geometry):

$$D'_{(n,m)}=-\frac{n(n+1)}{a^2}\chi_{(n,m)}\ ;\quad \zeta'_{(n,m)}=-\frac{n(n+1)}{a^2}\psi_{(n,m)}$$

2.6 Spectral relationship giving wind components once known (velocity potential, stream function).

- Global models: relationship between $U^{'}$, $V^{'}$, ψ and χ:

$$(U^{'} a\cos\Theta) = \frac{\partial\chi}{\partial\Lambda} - \cos\Theta\frac{\partial\psi}{\partial\Theta}\ ; \quad (V^{'} a\cos\Theta) = \frac{\partial\psi}{\partial\Lambda} + \cos\Theta\frac{\partial\chi}{\partial\Theta}$$

the spectral discretisation is:

$$(U^{'} a\cos\Theta)_{(n,m)} = im\chi_{(n,m)} + (n-1)e_{(n,m)}\psi_{(n-1,m)} - (n+2)e_{(n+1,m)}\psi_{(n+1,m)} \tag{14}$$

$$(V^{'} a\cos\Theta)_{(n,m)} = im\psi_{(n,m)} - (n-1)e_{(n,m)}\chi_{(n-1,m)} + (n+2)e_{(n+1,m)}\chi_{(n+1,m)} \tag{15}$$

- LAM models: relationship between $U^{'}$, $V^{'}$, ψ and χ:

$$U^{'} = U^{'}_{\rm moy} + \frac{\partial\chi}{\partial x} - \frac{\partial\psi}{\partial y}\ ; \quad V^{'} = V^{'}_{\rm moy} + \frac{\partial\psi}{\partial x} + \frac{\partial\chi}{\partial y}$$

We need a separate storage of $U^{'}_{\rm moy}$ and $V^{'}_{\rm moy}$ in spectral space.

2.7 Spectral relationship giving (divergence, vorticity) once known wind components.

∗ **Global models:**

- Relationship between $D^{'}$, $U^{'}$ and $V^{'}$:

$$D^{'} = \frac{1}{a\cos\Theta}\left(\frac{\partial U^{'}}{\partial\Lambda} + \frac{\partial(V^{'}\cos\Theta)}{\partial\Theta}\right) \tag{16}$$

which can be rewritten:

$$D^{'} = \frac{1}{a^2\cos^2\Theta}\left(\frac{\partial(U^{'} a\cos\Theta)}{\partial\Lambda} + \cos\Theta\frac{\partial(V^{'} a\cos\Theta)}{\partial\Theta}\right) \tag{17}$$

- Relationship between $\zeta^{'}$, $U^{'}$ and $V^{'}$:

$$\zeta^{'} = \frac{1}{a\cos\Theta}\left(\frac{\partial V^{'}}{\partial\Lambda} - \frac{\partial(U^{'}\cos\Theta)}{\partial\Theta}\right) \tag{18}$$

which can be rewritten:

$$\zeta^{'} = \frac{1}{a^2\cos^2\Theta}\left(\frac{\partial(V^{'} a\cos\Theta)}{\partial\Lambda} - \cos\Theta\frac{\partial(U^{'} a\cos\Theta)}{\partial\Theta}\right) \tag{19}$$

- Spectral discretisations of (17) and (19) allow to retrieve easily spectral components of $D^{'}a^2\cos^2\Theta$ and $\zeta^{'}a^2\cos^2\Theta$, but not directly spectral components of $D^{'}$ and $\zeta^{'}$ (requiring inversion of a pentadiagonal matrix). In practical algorithm involved to retrieve spectral coefficients of $D^{'}$ and $\zeta^{'}$ once knowing values of wind components is slightly different (requiring a division by $a\cos\Theta$ in Fourier space), and is described in detail in (Temperton, 1991) (see equations (2.16), (2.17), (2.21) to (2.27) of Temperton's paper).

∗ **LAM models:** relationship between $D^{'}$, $\zeta^{'}$, $U^{'}$ and $V^{'}$ writes:

$$D^{'} = \frac{\partial U^{'}}{\partial x} + \frac{\partial V^{'}}{\partial y}\ ; \quad \zeta^{'} = \frac{\partial V^{'}}{\partial x} - \frac{\partial U^{'}}{\partial y}$$

2.8 Steps involved in a direct and in an inverse spectral transformation for scalar fields.

∗ **Transformation from spectral to grid point space for a scalar field or a set of scalar fields in a global model.** Transformation from spectral to grid point space needs:

- An inverse Legendre transform, for each zonal wave number m. The following quantity $F_m(\mu)$ is computed for each latitude Θ:

$$F_m(\mu) = \sum_{n=|m|}^{n=N_s} f_{(n,m)} P_{(n,m)}(\mu) \tag{20}$$

One transforms real fields, so $F_{-m}(\mu) = \overline{F_m(\mu)}$ and only $F_m(\mu)$ for $m \geq 0$ is computed.

- An inverse Fourier transform, for each latitude Θ:

$$f(\Lambda, \Theta) = \sum_{m=-N(\Theta)}^{m=N(\Theta)} F_m(\mu) \exp(im\Lambda) \tag{21}$$

One transforms real fields, so formula (21) becomes:

$$f(\Lambda, \Theta) = F_0(\mu) + 2 \sum_{m=1}^{m=N(\Theta)} (\mathcal{R}(F_m(\mu)) \cos(m\Lambda) - \mathcal{I}(F_m(\mu)) \sin(m\Lambda)) \tag{22}$$

$F_0(\mu)$ is the average of f on latitude Θ and is real; $N(\Theta)$ is not always equal to N_s (due to the use of a reduced Gaussian grid). For latitude Θ one obtains $nlon(\Theta)$ grid-point values.

$*$ Transformation from spectral to grid point space for a scalar field or a set of scalar fields in a LAM model. Transformation from spectral to grid point space needs:

- An inverse meridian Fourier transform.
- An inverse zonal Fourier transform.

$*$ Transformation from grid point to spectral space for global models. Transformation from grid point to spectral space needs:

- A direct Fourier transform, for each latitude Θ:

$$F_m(\mu) = \frac{1}{nlon(\Theta)} \sum_{jlon=1}^{jlon=nlon(\Theta)} f(\Lambda(jlon), \Theta) \exp(-im\Lambda(jlon)) \tag{23}$$

One transforms real fields, so formula (23) becomes for non-zero m:

$$\mathcal{R}(F_m(\mu)) = \frac{1}{nlon(\Theta)} \sum_{jlon=1}^{jlon=nlon(\Theta)} f(\Lambda(jlon), \Theta) \cos(m\Lambda(jlon)) \tag{24}$$

$$\mathcal{I}(F_m(\mu)) = -\frac{1}{nlon(\Theta)} \sum_{jlon=1}^{jlon=nlon(\Theta)} f(\Lambda(jlon), \Theta) \sin(m\Lambda(jlon)) \tag{25}$$

and for m=0:

$$\mathcal{R}(F_0(\mu)) = \frac{1}{nlon(\Theta)} \sum_{jlon=1}^{jlon=nlon(\Theta)} f(\Lambda(jlon), \Theta) \tag{26}$$

$$\mathcal{I}(F_0(\mu)) = 0 \tag{27}$$

$nlon(\Theta)$ is the number of grid-points for latitude Θ. Only $N(\Theta)$ Fourier coefficients are kept. $N(\Theta)$ is not always equal to N_s.

- A direct Legendre transform, for each zonal wave number m.

$$f_{(n,m)} = \sum_{jgl=lat1(m)}^{jgl=lat2(m)} F_m(\mu(jgl)) P_{(n,m)}(\mu(jgl)) \tag{28}$$

$lat1$ and $lat2$ satisfy to $\Theta(lat2) = -\Theta(lat1)$. $lat1(m)$ is the first northern latitude where the wave number m is represented and is not always equal to 1.

$*$ Transformation from grid point to spectral space for LAM models. Transformation from grid point to spectral space needs:

- A direct zonal Fourier transform.
- A direct meridian Fourier transform.

∗ **Relationship between N and $lat1$ (global models):**

$$N(lat1(m)-1) < m \leq N(lat(m)) \tag{29}$$

∗ **Use of symmetry properties of the polynomials of Legendre (global models):** Due to relation (30) scalar fields to be transformed are split into symmetric and antisymmetric parts $f_{\rm sym}$ and $f_{\rm ant}$ matching $f_{\rm sym}(\Lambda,-\Theta) = f_{\rm sym}(\Lambda,\Theta)$ and $f_{\rm ant}(\Lambda,-\Theta) = -f_{\rm ant}(\Lambda,\Theta)$. Spectral expression of $f_{\rm sym}$ (resp. $f_{\rm ant}$) include all coefficients with even (resp. odd) value of $m+n$. Legendre transforms are done on $f_{\rm sym}$ and $f_{\rm ant}$ and scalar products defined by equations (20) and (28) can reduce to latitudes $lat1$ to $ndgnh$.

2.9 Relationship between dimension of spectral space and grid point space for Gaussian grid.

∗ **Quadratic grid, linear grid and cubic grids in global models:**

- Spectral space is defined by a triangular truncation $N_{\rm s}$. Grid point space has $ndglg$ latitudes and a maximum number of longitudes equal to $ndlon$. $ndlon$ and $ndglg$ are always even integers: if $ndlon$ is a multiple of 4, $ndglg = ndlon/2$; if $ndlon$ is not a multiple of 4, $ndglg = ndlon/2+1$. The latitudes of the Gaussian grid are computed by an algorithm detailed in appendix 2.
- For a quadratic Gaussian grid, there is a relationship between these parameters to avoid aliasing on quadratic terms. If the stretching coefficient c is equal to 1 (resp. >1) $N_{\rm s}$ is the biggest integer which matches the relationship $3*N_{\rm s} \leq (ndlon-1)$ (resp. $3*N_{\rm s} \leq \min(2*ndglg-3, ndlon-1)$).
- In a semi-Lagrangian scheme the advective quadratic terms disappear, so it is possible to use a smaller grid-point space. The most frequently grid used is the "linear grid": If the stretching coefficient c is equal to 1 (resp. >1) $N_{\rm s}$ is the biggest integer which matches the relationship $2*N_{\rm s} \leq (ndlon-1)$ (resp. $2*N_{\rm s} \leq \min(2*ndglg-3, ndlon-1)$).
- Recently, the concept of cubic grid has been introduced. If the stretching coefficient c is equal to 1 (resp. >1) $N_{\rm s}$ is the biggest integer which matches the relationship $4*N_{\rm s} \leq (ndlon-1)$ (resp. $4*N_{\rm s} \leq \min(2*ndglg-3, ndlon-1)$).
- We generally use a reduced Gaussian grid; the number of grid-points is progressively reduced as we go towards computational poles.

∗ **Quadratic grid, linear grid and cubic grids in LAM models:** similar notions exist in LAM models. For example:
For a quadratic grid:

- $N_{\rm s}$ is the biggest integer which matches the relationship $3*N_{\rm s} \leq (ndglg-1)$.
- $N_{\rm ms}$ is the biggest integer which matches the relationship $3*N_{\rm ms} \leq (ndlon-1)$.

For a linear grid:

- $N_{\rm s}$ is the biggest integer which matches the relationship $2*N_{\rm s} \leq (ndglg-1)$.
- $N_{\rm ms}$ is the biggest integer which matches the relationship $2*N_{\rm ms} \leq (ndlon-1)$.

For a cubic grid:

- $N_{\rm s}$ is the biggest integer which matches the relationship $4*N_{\rm s} \leq (ndglg-1)$.
- $N_{\rm ms}$ is the biggest integer which matches the relationship $4*N_{\rm ms} \leq (ndlon-1)$.

∗ **Admissible dimensions for spherical geometry:** If FFT992 is used, the codes of FFT (fast Fourier transforms) used allows to use integers $ndlon$ which can write as $2^{1+p_2}*3^{p_3}*5^{p_5}$. That limits the possibility of choosing the dimensions in a discontinuous subset of truncations and dimensions for Gaussian grid. In the following table one can find the admissible values for $(ndlon, ndglg)$ for $ndglg$ between 32 and 8000, and the corresponding truncations for a quadratic grid and a linear grid. This constraint does not exist with FFTW.

∗ **Admissible dimensions for LAM models:** If FFT992 is used, the codes of FFT (fast Fourier transforms) used allows to use integers $ndlon$ and $ndglg$ which can write as $2^{1+p_2}*3^{p_3}*5^{p_5}$. This constraint does not exist with FFTW.

$(ndglg, ndlon)$	Cub	Quad $c>1$	Quad $c=1$	Lin $c>1$	Lin $c=1$		$(ndglg, ndlon)$	Cub	Quad $c>1$	Quad $c=1$	Lin $c>1$	Lin $c=1$
(32, 64)	15	20	21	30	31		(1250,2500)	624	832	833	1248	1249
(36, 72)	17	23	23	34	35		(1280,2560)	639	852	853	1278	1279
(40, 80)	19	25	26	38	39		(1296,2592)	647	863	863	1294	1295
(46, 90)	22	29	29	44	44		(1350,2700)	674	899	899	1348	1349
(48, 96)	23	31	31	46	47		(1440,2880)	719	959	959	1438	1439
(50, 100)	24	32	33	48	49		(1458,2916)	728	971	971	1456	1457
(54, 108)	26	35	35	52	53		(1500,3000)	749	999	999	1498	1499
(60, 120)	29	39	39	58	59		(1536,3072)	767	1023	1023	1534	1535
(64, 128)	31	41	42	62	63		(1600,3200)	799	1065	1066	1598	1599
(72, 144)	35	47	47	70	71		(1620,3240)	809	1079	1079	1618	1619
(76, 150)	37	49	49	74	74		(1728,3456)	863	1151	1151	1726	1727
(80, 160)	39	52	53	78	79		(1800,3600)	899	1199	1199	1798	1799
(82, 162)	40	53	53	80	80		(1876,3750)	937	1249	1249	1874	1874
(90, 180)	44	59	59	88	89		(1920,3840)	959	1279	1279	1918	1919
(96, 192)	47	63	63	94	95		(1944,3888)	971	1295	1295	1942	1943
(100, 200)	49	65	66	98	99		(2000,4000)	999	1332	1333	1998	1999
(108, 216)	53	71	71	106	107		(2026,4050)	1012	1349	1349	2024	2024
(120, 240)	59	79	79	118	119		(2048,4096)	1023	1364	1365	2046	2047
(126, 250)	62	83	83	124	124		(2160,4320)	1079	1439	1439	2158	2159
(128, 256)	63	84	85	126	127		(2188,4374)	1093	1457	1457	2186	2186
(136, 270)	67	89	89	134	134		(2250,4500)	1124	1499	1499	2248	2249
(144, 288)	71	95	95	142	143		(2304,4608)	1151	1535	1535	2302	2303
(150, 300)	74	99	99	148	149		(2400,4800)	1199	1599	1599	2398	2399
(160, 320)	79	105	106	158	159		(2430,4860)	1214	1619	1619	2428	2429
(162, 324)	80	107	107	160	161		(2500,5000)	1249	1665	1666	2498	2499
(180, 360)	89	119	119	178	179		(2560,5120)	1279	1705	1706	2558	2559
(192, 384)	95	127	127	190	191		(2592,5184)	1295	1727	1727	2590	2591
(200, 400)	99	132	133	198	199		(2700,5400)	1349	1799	1799	2698	2699
(216, 432)	107	143	143	214	215		(2880,5760)	1439	1919	1919	2878	2879
(226, 450)	112	149	149	224	224		(2916,5832)	1457	1943	1943	2914	2915
(240, 480)	119	159	159	238	239		(3000,6000)	1499	1999	1999	2998	2999
(244, 486)	121	161	161	242	242		(3072,6144)	1535	2047	2047	3070	3071
(250, 500)	124	165	166	248	249		(3126,6250)	1562	2083	2083	3124	3124
(256, 512)	127	169	170	254	255		(3200,6400)	1599	2132	2133	3198	3199
(270, 540)	134	179	179	268	269		(3240,6480)	1619	2159	2159	3238	3239
(288, 576)	143	191	191	286	287		(3376,6750)	1687	2249	2249	3374	3374
(300, 600)	149	199	199	298	299		(3456,6912)	1727	2303	2303	3454	3455
(320, 640)	159	212	213	318	319		(3600,7200)	1799	2399	2399	3598	3599
(324, 648)	161	215	215	322	323		(3646,7290)	1822	2429	2429	3644	3644
(360, 720)	179	239	239	358	359		(3750,7500)	1874	2499	2499	3748	3749
(376, 750)	187	249	249	374	374		(3840,7680)	1919	2559	2559	3838	3839
(384, 768)	191	255	255	382	383		(3888,7776)	1943	2591	2591	3886	3887
(400, 800)	199	265	266	398	399		(4000,8000)	1999	2665	2666	3998	3999
(406, 810)	202	269	269	404	404		(4050,8100)	2024	2699	2699	4048	4049
(432, 864)	215	287	287	430	431		(4096,8192)	2047	2729	2730	4094	4095
(450, 900)	224	299	299	448	449		(4320,8640)	2159	2879	2879	4318	4319
(480, 960)	239	319	319	478	479		(4374,8748)	2186	2915	2915	4372	4373
(486, 972)	242	323	323	484	485		(4500,9000)	2249	2999	2999	4498	4499
(500,1000)	249	332	333	498	499		(4608,9216)	2303	3071	3071	4606	4607
(512,1024)	255	340	341	510	511		(4800,9600)	2399	3199	3199	4798	4799
(540,1080)	269	359	359	538	539		(4860,9720)	2429	3239	3239	4858	4859
(576,1152)	287	383	383	574	575		(5000,10000)	2499	3332	3333	4998	4999
(600,1200)	299	399	399	598	599		(5120,10240)	2559	3412	3413	5118	5119
(626,1250)	312	416	416	624	624		(5184,10368)	2591	3455	3455	5182	5183
(640,1280)	319	425	426	638	639		(5400,10800)	2699	3599	3599	5398	5399
(648,1296)	323	431	431	646	647		(5626,11250)	2812	3749	3749	5624	5624
(676,1350)	337	449	449	674	674		(5760,11520)	2879	3839	3839	5758	5759
(720,1440)	359	479	479	718	719		(5832,11664)	2915	3887	3887	5830	5831
(730,1458)	364	485	485	728	728		(6000,12000)	2999	3999	3999	5998	5999
(750,1500)	374	499	499	748	749		(6076,12150)	3037	4049	4049	6074	6074
(768,1536)	383	511	511	766	767		(6144,12288)	3071	4095	4095	6142	6143
(800,1600)	399	532	533	798	799		(6250,12500)	3124	4165	4166	6248	6249
(810,1620)	404	539	539	808	809		(6400,12800)	3199	4265	4266	6398	6399
(864,1728)	431	575	575	862	863		(6480,12960)	3239	4319	4319	6478	6479
(900,1800)	449	599	599	898	899		(6562,13122)	3280	4373	4373	6560	6560
(960,1920)	479	639	639	958	959		(6750,13500)	3374	4499	4499	6748	6749
(972,1944)	485	647	647	970	971		(6912,13824)	3455	4607	4607	6910	6911
(1000,2000)	499	665	666	998	999		(7200,14400)	3599	4799	4799	7198	7199
(1024,2048)	511	681	682	1022	1023		(7290,14580)	3644	4859	4859	7298	7299
(1080,2160)	539	719	719	1078	1079		(7500,15000)	3749	4999	4999	7498	7499
(1126,2250)	562	749	749	1124	1124		(7680,15360)	3839	5119	5119	7678	7679
(1152,2304)	575	767	767	1150	1151		(7776,15552)	3887	5183	5183	7774	7775
(1200,2400)	599	799	799	1198	1199		(8000,16000)	3999	5332	5333	7998	7999
(1216,2430)	607	809	809	1214	1214							

3 Set-up routines.

3.1 Simplified call-tree.

The call-tree given below is not an extensive one but a simplified one to show the main routines. Mention "lctnd" means "lower call-tree not described". Abbreviation FLT means "Fast Legendre transforms". Left part is for global model, right part is for LAM model.

```
SUOYOMA ->                                          | SUOYOMA ->
- SUTRANS0 ->                                       | - SUETRANS0 ->
  * SETUP_TRANS0 -> SUMP_TRANS0 -> (lctnd)          |   * SETUP_TRANS0 -> SUMP_TRANS0 -> (lctnd)
- SUGEOMETRY ->                                     | - SUGEOMETRY ->
  * SUTRANS ->                                      |   * SUETRANS ->
    - SETUP_TRANS ->                                |     - ESETUP_TRANS ->
      * SET_RESOL                                   |       * ESET_RESOL
      * SETUP_DIMS                                  |       * ESETUP_DIMS
      * SUMP_TRANS_PRELEG -> (lctnd)                |       * SUEMP_TRANS_PRELEG
      * SULEG ->                                    |       * ESETUP_GEOM
        - PRE_SULEG                                 |
        - INI_POL                                   |
        - SUGAW -> (lctnd)                          |
        - SUPOL                                     |
        - SUTRLE -> (lctnd)                         |
        - SETUP_GEOM                                |
        - SUPOLF, CONSTRUCT_BUTTERFLY (FLT)         |
      * SUMP_TRANS ->                               |       * SUEMP_TRANS ->
        - SUMPLATF -> (lctnd)                       |         - SUMPLATF -> (lctnd)
        - SUMPLAT -> (lctnd)                        |         - SUEMPLAT -> (lctnd)
        - SUSTAONL -> (lctnd)                       |         - SUESTAONL -> (lctnd)
      * SUFFT ->                                    |       * SUEFFT ->
        - SET99 (use of FFT992)                     |         - SET99 (use of FFT992)
        - INIT_PLANS_FFTW (use of FFTW)             |         - INIT_PLANS_FFTW (use of FFTW)
  * SULEGA                                          |   * SUELEGA
```

3.2 Computations done by these routines and remarks.

Upper levels of call-tree:

- **SU(E)TRANS0**: computes not geometry-dependent variables useful for the transform package. There are for example some dimension variables, some distributed memory variables.
- **SUGEOMETRY**: geometry set-up; it includes in particuliar some variables computed by the transform package (for example Gaussian latitudes).
- **SU(E)TRANS**: computes geometry-dependent variables useful for the transform package.
- **SULEGA**: gets some variables from transform package (these variables depend on Gaussian latitudes), copy them in "arpifs" structures.
- **SUELEGA**: is just for security but computed variables inside should not be used in LAM models.

Lower levels of call-tree for Legendre transforms (global models only): the most important features are the Gaussian grid geometry (for example Gaussian latitudes, Gaussian weights) and the Legendre polynomials.

- The main routine is **SULEG**.
- **SUGAW** computes the Gaussian latitudes (Θ, in attribute **RLATIG**) and the Gaussian weights (in attribute **RW**), and some additional variables like $\cos\Theta$ (in attribute **RSQM2**) and $\sin\Theta$ (in attribute **RMU**).
- **SUPOL** computes the polynomials of Legendre (attribute **RPNM**). For each Gaussian latitude, **RPNM** is an array with a structure identical to spectral arrays. **RPNM** is stored only for the Northern Hemisphere (of computational sphere) due to the following property:

$$P_{(n,m)}(-\mu) = (-1)^{(m+n)} P_{(n,m)}(\mu) \tag{30}$$

- **SUPOLF** is an alternate version of **SUPOL** (for fast Legendre transforms).

Lower levels of call-tree for Fourier transforms:

- The main routine is **SUFFT**.
- For "FFT992" Fourier transforms, one needs to compute quantities **TRIGS** and **NFAX** necessary to use routine **FFT992**. These two arrays are computed in routines **SUFFT** and **SUEFFT** (note that a call to **FFT992** reinitialises them).
- For "FFTW" Fourier transforms, the set-up is done in **INIT_PLANS_FFTW**.

Lower levels of call-tree for transpositions and distributed memory features:

- **SUTRLE**: does transpositions for Legendre polynomials, because the distribution required for Legendre polynomials calculation is not the same as the distribution required for Legendre polynomials use. Calculation of Legendre polynomials requires a distribution on latitudes (for a given latitude all the spectral coefficients are treated by the same processor). Use of Legendre polynomials requires a distribution on zonal wavenumbers (for a given zonal wavenumber all the latitudes are treated by the same processor).
- **SU(E)MP_TRANS**: sets-up geometry-dependent distributed environment for the transform package.

Duplicate version of variables between "arpifs" and "trans"/"etrans":

- Most of the variables computed under **SUTRANS** have a "duplicate version" in the "arpifs" library, because they are used in other parts of the ARPEGE/IFS code. The "trans" version (in modules **TPM_...**) is used in the spectral transforms, the "arpifs" version is used elsewhere.
- Most of the variables computed under **SUETRANS** have a "duplicate version" in the "arpifs" library, because they are used in other parts of the ARPEGE/ALADIN code. The "etrans" version (in modules **TPMALD_...**) is used in the spectral transforms, the "arpifs" version is used elsewhere.

4 Spectral transforms routines: the general routines (E)DIR_TRANS and (E)INV_TRANS.

4.1 Range of use.

They can be used for a set of fields (model fields), but they use optional arguments, so they can also be used for only one separate field. **DIR_TRANS** (**EDIR_TRANS** for LAM models) does direct transforms (from grid-point space to spectral space), **INV_TRANS** (**EINV_TRANS** for LAM models) does inverse transforms (from spectral space to grid-point space). These routines have adjoint versions **(E)DIR_TRANSAD** and **(E)INV_TRANSAD**. The tangent linear code is identical to the direct code. These routines and the routines called under them are in the "trans" library for spherical geometry, "etrans" library for LAM models.

4.2 Call tree for direct code.

∗ Call tree of INV_TRANS and EINV_TRANS:

```
INV_TRANS ->                              | EINV_TRANS ->
  * SET_RESOL                             |   * ESET_RESOL
  * INV_TRANS_CTL ->                      |   * EINV_TRANS_CTL ->
    - SHUFFLE (case NPRTRV>1)             |     - SHUFFLE (case NPRTRV>1)
    - FIELD_SPLIT                         |     - FIELD_SPLIT
    - LTINV_CTL ->                        |     - ELTINV_CTL ->
      * LTINV ->                          |       * ELTINV ->
        - PREPSNM                         |
        - PRFI1B                          |         - EPRFI1B
        - VDTUV                           |         - EVDTUV
        - SPNSDE                          |         - ESPNSDE
        - LEINV -> DGEMM                  |         - ELEINV -> FFT992 or EXEC_EFFTW
                   MULT_BUTM (FLT)        |
        - ASRE1B                          |         - EASRE1B
        - FSPGL_INT                       |         - FSPGL_INT
      * TRMTOL -> (MPL routines)          |       * TRMTOL -> (MPL routines)
    - FTINV_CTL ->                        |     - EFTINV_CTL ->
      * FOURIER_IN                        |       * FOURIER_IN
      * FSC                               |       * EFSC
      * FTINV ->                          |       * FTINV ->
        - FFT992 or EXEC_FFTW             |         - FFT992 or EXEC_EFFTW
      * TRLTOG -> (MPL routines)          |       * TRLTOG -> (MPL routines)
```

∗ Call tree of DIR_TRANS and EDIR_TRANS:

```
DIR_TRANS ->                              | EDIR_TRANS ->
  * SET_RESOL                             |   * ESET_RESOL
  * DIR_TRANS_CTL ->                      |   * EDIR_TRANS_CTL ->
    - SHUFFLE (case NPRTRV>1)             |     - SHUFFLE (case NPRTRV>1)
    - FIELD_SPLIT                         |     - FIELD_SPLIT
    - FTDIR_CTL ->                        |     - EFTDIR_CTL ->
      * TRGTOL -> (MPL routines)          |       * TRGTOL -> (MPL routines)
                                          |       * EXTPER
                                          |       * AUX_PROC
      * FTDIR ->                          |       * FTDIR ->
        - FFT992 or EXEC_FFTW             |         - FFT992 or EXEC_EFFTW
      * FOURIER_OUT                       |       * FOURIER_OUT
    - LTDIR_CTL ->                        |     - ELTDIR_CTL ->
      * TRLTOM -> (MPL routines)          |       * TRLTOM -> (MPL routines)
                                          |       * AUX_PROC
      * LTDIR ->                          |       * ELTDIR ->
        - PRFI2 -> PRFI2B                 |         - EPRFI2 -> EPRFI2B
                                          |         - EXTPER
        - LDFOU2                          |
        - LEDIR -> DGEMM                  |         - ELEDIR -> FFT992 or EXEC_EFFTW
                   MULT_BUTM (FLT)        |
        - PREPSNM                         |
        - UVTVD                           |         - EUVTVD
                                          |         - EUVTVD_COMM
        - UPDSP -> UPDSPB                 |         - EUPDSP -> EUPDSPB
```

∗ Work done by routines called under INV_TRANS_CTL and DIR_TRANS_CTL:

- Legendre transforms:
  - **LTINV_CTL**: upper control routine for inverse Legendre transforms.

- **LTDIR_CTL**: upper control routine for direct Legendre transforms.
- **LTINV**: lower control routine for inverse Legendre transforms.
- **LTDIR**: lower control routine for direct Legendre transforms.
- **PREPSNM**: stores useful values of $e_{(n,m)}$ (in array **REPSNM**) for a given zonal wave number m.
- **PRFI1B**: stores useful spectral coefficients of the fields to be transformed for a given zonal wave number m and latitudes $lat1(m)$ to $ndgnh$ in a work array.
- **VDTUV**: computation of $(a\cos\Theta U^{'}, a\cos\Theta V^{'})$ in spectral space knowing $(D^{'}, \zeta^{'})$ when required.
- **SPNSDE**: computation of meridian derivatives when required.
- **LEINV** performs two inverse Legendre transforms, the first one for $f_{\rm sym}$, the second one for $f_{\rm ant}$, that gives the Fourier coefficients for symmetrical and antisymmetrical parts.
- **DGEMM**: product matrix by matrix.
- **MULT_BUTM**: product matrix by matrix with the butterfly method.
- **ASRE1B** recombines symmetrical and antisymmetrical parts to compute Fourier coefficients $F_m(\mu)$.

$$F_m(\mu) = F_{\rm sym\ m}(\mu) + F_{\rm ant\ m}(\mu)\ ; \quad F_m(-\mu) = F_{\rm sym\ m}(\mu) - F_{\rm ant\ m}(\mu)$$

- **FSPGL_INT**: interface for calculations done in Fourier space.
- **PRFI2B**: splits Fourier coefficients into symmetrical and antisymmetrical parts.
- **PRFI2**: interface for **PRFI2B** (one call to **PRFI2B** for scalars, two for vectors).
- **LDFOU2**: division by $a\cos\Theta$ in Fourier space for wind.
- **LEDIR** performs two direct Legendre transforms, the first one for $F_{\rm sym\ m}(\mu)$, the second one for $F_{\rm ant\ m}(\mu)$.
- **UVTVD**: computation of $(D^{'}, \zeta^{'})$ in spectral space knowing $\left(\frac{U^{'}}{a\cos\Theta}, \frac{V^{'}}{a\cos\Theta}\right)$.
- **UPDSPB**: final memory transfer in spectral array containing all coefficients.
- **UPDSP**: interface routine for **UPDSPB**.

- Transpositions between Legendre transforms space and Fourier space:
  - **TRLTOM**: transposition routine for distributed memory. Transpose Fourier buffer data from partitioning over latitudes (necessary for direct Fourier transforms) to partitioning over wave numbers (necessary for direct Legendre transforms).
  - **TRMTOL**: does the inverse transposition of **TRLTOM**.
- Fourier transforms:
  - **FTINV_CTL**: upper control routine for inverse Fourier transforms.
  - **FTDIR_CTL**: upper control routine for direct Fourier transforms.
  - **FTINV**: lower control routine for inverse Fourier transforms.
  - **FTDIR**: lower control routine for direct Fourier transforms.
  - **FOURIER_IN**: separates the real and imaginary parts of the Fourier coefficients.
  - **FOURIER_OUT**: recombines the real and imaginary parts of the Fourier coefficients.
  - **FSC**: division by $a\cos\Theta$ in Fourier space for wind, and first order derivatives.
  - **FFT992** performs Fourier transforms for "FFT992" package.
  - **EXEC_FFTW** is called instead of **FFT992** if "FFTW" package is used.
- Transpositions between Fourier transforms space and grid-point space:
  - **TRGTOL**: transposition routine for distributed memory. Transpose grid-point data from partitioning over sets of **NGPTOT** grid-points containing complete columns (all the **NFLEVG** model layers) and incomplete latitudes to partitioning over complete latitudes (a processor contains a subset of complete latitudes) and layers if **NPRTRV**>1 (a processor contains **NFLEVL** layers).
  - **TRLTOG**: does the inverse transposition of **TRGTOL**.
- Other routines:
  - **SHUFFLE**: re-shuffle fields for load balancing if **NPRTRV**>1. Note that the relative order of the local spectral fields has to maintained.
  - **FIELD_SPLIT**: split fields for case **NPRTRV**>1; simple memory transfer if **NPRTRV**=1.

$*$ Work done by routines called under EINV_TRANS_CTL and EDIR_TRANS_CTL:

We can notice that we generally have the same routines as for "trans" routines, with an additional "E" at the beginning of their names when a specific "etrans" routine is required: for example **ELEDIR** instead of **LEDIR**. Note the following specificies:

- **ELEDIR** and **ELEINV** do meridian Fourier transforms and call **FFT992**.
- **AUX_PROC** is called to do periodicisation of auxiliary fields.
- **EXTPER** is called to do periodicisation when there is no extension zone.

4.3 Call tree for adjoint code.

Call tree is given for **INV_TRANSAD** and **DIR_TRANSAD**; their LAM counterparts **EINV_TRANSAD** and **EDIR_TRANSAD** also exist (organigramme not detailed).

∗ Call tree of INV_TRANSAD:

```
INV_TRANSAD ->
  * SET_RESOL
  * INV_TRANS_CTLAD ->
    - SHUFFLE (case NPRTRV>1)
    - FIELD_SPLIT
    - FTINV_CTLAD ->
      * TRGTOL -> (MPL routines)
      * FTINVAD ->
        - FFT992 or EXEC_FFTW
      * FSCAD
      * FOURIER_INAD
    - LTINV_CTLAD ->
      * TRLTOM -> (MPL routines)
      * LTINVAD ->
        - PREPSNM
        - ASRE1BAD
        - LEINVAD -> DGEMM
                     MULT_BUTM (FLT)
        - VDTUVAD
        - SPNSDEAD
        - PRFI1BAD
```

∗ Call tree of DIR_TRANSAD:

```
DIR_TRANSAD ->
  * SET_RESOL
  * DIR_TRANS_CTLAD ->
    - SHUFFLE (case NPRTRV>1)
    - FIELD_SPLIT
    - LTDIR_CTLAD ->
      * LTDIRAD ->
        - UPDSPAD -> UPDSPBAD
        - PREPSNM
        - UVTVDAD
        - LEDIRAD -> DGEMM
                     MULT_BUTM (FLT)
        - LDFOU2AD
        - PRFI2AD -> PRFI2BAD
      * TRMTOL -> (MPL routines)
    - FTDIR_CTLAD ->
      * FOURIER_OUTAD
      * FTDIRAD ->
        - FFT992 or EXEC_FFTW
      * TRLTOG -> (MPL routines)
```

5 Spectral transforms routines for general application.

5.1 Spectral transforms routines for scalar variables: SPEREE and REESPE.

REESPE does direct transforms (from grid-point space to spectral space), **SPEREE** does inverse transforms (from spectral space to grid-point space). **SPEREE** calls **INV_TRANS**; **REESPE** calls **DIR_TRANS**; the adjoint **SPERAD** of **SPEREE** calls **REESPE**. These routines perform transforms for scalar fields according to the algorithm described in subsection (2.8).

These routines have LAM counterparts: **EREESPE** and **ESPEREE** (direct code and TL code), **ESPERAD** (adjoint code).

5.2 Wind components: routines SPEUV and UVSPE.

∗ **Transformation from spectral to grid point space.** $D^{'}$ is the reduced divergence, $\zeta^{'}$ is the reduced vorticity. χ is the velocity potential, ψ is the stream function. $U^{'}$ and $V^{'}$ are the reduced wind components. $D^{'}$ and $\zeta^{'}$ are available in spectral space. Transformation from spectral to grid point space needs:

- A transformation $(\chi, \psi) \rightarrow (D^{'}, \zeta^{'})$ if required.
- Computation of $(a\cos\Theta U^{'}, a\cos\Theta V^{'})$ in spectral space knowing $(D^{'}, \zeta^{'})$, following formulae (14) and (15).
- Inverse Legendre transform on $a\cos\Theta U^{'}$ and $a\cos\Theta V^{'}$.
- In Fourier space, division by $a\cos\Theta$ for each latitude: yields $U^{'}$ and $V^{'}$ in this space.
- Inverse Fourier transform on $U^{'}$ and $V^{'}$ for each latitude.

∗ **Transformation from grid point to spectral space.** Transformation from grid point to spectral space needs:

- Direct Fourier transform on $U^{'}$ and $V^{'}$ for each latitude.
- Computation of $\left(\frac{U^{'}}{a\cos\Theta}, \frac{V^{'}}{a\cos\Theta}\right)$ in Fourier space for each latitude.
- Direct Legendre transform on $\frac{U^{'}}{a\cos\Theta}$ and $\frac{V^{'}}{a\cos\Theta}$. Spectral coefficients obtained after this transform correspond to the $\tilde{U}_{(n,m)}$ and $\tilde{V}_{(n,m)}$ defined in equations (2.24) and (2.25) of (Temperton, 1991).
- Computation of $(D^{'}, \zeta^{'})$ in spectral space knowing $\left(\frac{U^{'}}{a\cos\Theta}, \frac{V^{'}}{a\cos\Theta}\right)$, following formulae (2.26) and (2.27) of (Temperton, 1991).
- A transformation $(D^{'}, \zeta^{'}) \rightarrow (\chi, \psi)$ if required.

∗ **Routines performing transformations:** **SPEUV** performs a transformation from spectral space (input variables are $(D^{'}, \zeta^{'})$ or (χ, ψ)) to grid-point space (output variables are $U^{'}$ and $V^{'}$). **UVSPE** performs a transformation from grid-point space (input variables are $U^{'}$ and $V^{'}$) to spectral space (output variables are $(D^{'}, \zeta^{'})$ if **KOUTP**=1 or (χ, ψ) if **KOUTP**=2). **SPEUV** calls **INV_TRANS**; **UVSPE** calls **DIR_TRANS**.

∗ **LAM model versions:**

- **ESPEUV** is the LAM counterpart of **SPEUV**.
- There are also **EUVGEOVD** (retrieve divergence and vorticity from wind components in spectral space) and **EVDUVGEO** (retrieve wind components from divergence and vorticity in spectral space).

6 Spectral transforms used in the model under routine STEPO and for specific applications.

6.1 Call tree under STEPO.

In model integration, routines **TRANSINVH** and **TRANSDIRH** play the same part as **SPEREE/SPEUV**, **REESPE/UVSPE**, but they deal with a pre-determined set of variables. Horizontal derivatives of these variables have generally to be computed during the inverse transform. Meridian derivatives are computed after converting $(D^{'}, \zeta^{'})$ into $(a\cos\Theta U^{'}, a\cos\Theta V^{'})$ and before inverse Legendre transforms in spectral space (call to routine **SPNSDE**). These spectral transforms routines are also used for the spectral fit of FULL-POS.

∗ **General organigramme under STEPO:** one starts from routine **STEPO** which controls one time-step.

- **control/STEPO** →
  - Data reading/writing
  - **(E)TRANSINVH** (see below the detailed organigramme)
  - **SCAN2M** (grid point computations)
  - **OBSV** (for assimilation part computing J_o)
  - Diagnostics
  - **(E)TRANSDIRH** (see below the detailed organigramme)
  - Spectral computations

FULLPOS has its own version of **STEPO** (**STEPO_FPOS**) calling **TRANSINV_FPOS** and **TRANSDIR_FPOS**.

∗ Call tree under (E)TRANSINVH:

```
(E)TRANSINVH -> (E)TRANSINV_MDL -> (E)INV_TRANS (model variables)
```

∗ Call tree under (E)TRANSDIRH:

```
(E)TRANSDIRH -> (E)TRANSDIR_MDL -> (E)DIR_TRANS (model variables)
```

∗ **Tangent linear code:** The tangent linear code of **(E)TRANSINVH** (called under **STEPOTL**) is **(E)TRANSINVH** itself. The tangent linear code of **(E)TRANSDIRH** (called under **STEPOTL**) is **(E)TRANSDIRH** itself.

∗ **Adjoint code:** The adjoint code of **(E)TRANSINVH** (called under **STEPOAD**) is **(E)TRANSINVHAD**. The adjoint code of **(E)TRANSDIRH** (called under **STEPOAD**) is **(E)TRANSDIRHAD**. Call trees are:

```
(E)TRANSINVHAD -> (E)TRANSINV_MDLAD -> (E)INV_TRANSAD
```

and:

```
(E)TRANSDIRHAD -> (E)TRANSDIR_MDLAD -> (E)DIR_TRANSAD
```

6.2 Spectral transforms necessary for model integration.

Only model integration is here presented. For other applications where **TRANSDIRH** or **TRANSINVH** are used (assimilation, initialisation by DFI), see corresponding documentations. Variables are divided in three classes:

- GMV (3D) variables: the adiabatic part of the RHS of their equation is non-zero and these variables have always a spectral representation (example: wind components, temperature).
- GMVS (2D) variables: the adiabatic part of the RHS of their equation is non-zero and these variables have always a spectral representation (example: logarithm of hydrostatic surface pressure).
- GFL variables: the adiabatic part of the RHS of their equation is zero and these variables can have a spectral representation or not (example: specific humidity, ozone).

For more details about this classification, see documentation (IDEUL).

6.2.1 3D hydrostatic and non-hydrostatic models.

∗ **Input variables in spectral space to be transformed into grid-point space:** For GMV variables: reduced divergence $D^{'}$, reduced vorticity $\zeta^{'}$, temperature T if **LSPRT**=.F. or RT if **LSPRT**=.T., additional NH variables for non-hydrostatic models. For GMVS variables: pressure surface variable $\log \Pi_s$ (Π_s is the surface pressure). All the GFL variables which have a spectral representation.

∗ **Output variables to be obtained in grid-point space:** Variable X, meridian derivative $\frac{1}{a \cos \Theta} \frac{\partial X}{\partial \Theta}$ and zonal derivative $\frac{1}{a} \frac{\partial X}{\partial \Lambda}$ for the following variables: reduced divergence $D^{'}$, reduced vorticity $\zeta^{'}$, reduced wind components $U^{'}$ and $V^{'}$, temperature T if **LSPRT**=.F. or RT if **LSPRT**=.T., pressure surface variable $\log \Pi_s$, additional NH variables for non-hydrostatic models, all the GFL variables which have a spectral representation. All horizontal derivatives are required in an Eulerian model but some derivatives are useless in a semi-Lagrangian model.

∗ **Input variables in grid-point space to be transformed into spectral space:** Provisional $t + \Delta t$ quantities before semi-implicit scheme and horizontal diffusion scheme for GMV variables equations (momentum equation, temperature equation (virtual temperature equation if **LSPRT**=.T.), additional NH variables for non-hydrostatic models), GMVS variables equations (continuity equation), GFL equations for GFL variables which have a spectral representation. For momentum equation input variables are reduced wind components, output variables are reduced divergence and vorticity.

∗ **Additional spectral transforms necessary for iterative centred-implicit schemes:** The iterative part of these schemes do spectral transforms as for the predictive part (transforms are done on the provisional $t+\Delta t$ quantities obtained after each iteration); transforms involves all variables for the centred-implicit scheme coded.

6.2.2 Shallow-water 2D model.

∗ **Input variables in spectral space to be transformed into grid-point space:** Reduced divergence $D^{'}$, reduced vorticity $\zeta^{'}$, equivalent height Φ.

∗ **Output variables to be obtained in grid-point space:** Variable X, meridian derivative $\frac{1}{a \cos \Theta} \frac{\partial X}{\partial \Theta}$ and zonal derivative $\frac{1}{a} \frac{\partial X}{\partial \Lambda}$ for the following variables: reduced divergence $D^{'}$, reduced vorticity $\zeta^{'}$, reduced wind components $U^{'}$ and $V^{'}$, equivalent height Φ. All horizontal derivatives are required in an Eulerian model but some derivatives are useless in a semi-Lagrangian model.

∗ **Input variables in grid-point space to be transformed into spectral space:** Provisional $t + \Delta t$ quantities before semi-implicit scheme and horizontal diffusion scheme for momentum equation and continuity equation. For momentum equation input variables are reduced wind components, output variables are reduced divergence and vorticity.

6.3 Spectral transforms necessary for particular applications.

One finds some other **TRANSDIR_...** (which call **DIR_TRANS**) and **TRANSINV_...** (which call **INV_TRANS**) routines (resp. **ETRANSDIR_...** and **ETRANSINV_...** routines for LAM models) which are designed for specific applications. For example, for ARPEGE, one finds **TRANSDIR_NHCONV** and **TRANSINV_NHCONV**, **TRANSDIR_WAVELET** and **TRANSINV_WAVELET**.

7 Some distributed memory features.

∗ **Message passing division into processors:** The total number of processors is **NPROC**. There are two levels of parallelisation. Distribution on these two levels is different in the different spaces of calculations.

∗ **Grid-point computations:** The total number of processors involved in the A-level parallelisation is **NPRGPNS**. The total number of processors involved in the B-level parallelisation is **NPRGPEW**. The total number of processors is **NPROC**=**NPRGPNS**∗**NPRGPEW**.

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

A processor works on complete columns, but latitudes are not always complete (split among several processors).

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

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

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

Data reorganisation and transpositions are necessary between the grid-point space (complete columns, incomplete latitudes) and the Fourier space (complete latitudes, possibility to have incomplete columns), using routines **TRGTOL** and **TRLTOG**.

∗ **Fast Fourier transforms:** **NPROC**=**NPRTRNS**∗**NPRTRV**. The total number of processors involved in the A-level parallelisation is **NPRTRNS**. The total number of processors involved in the B-level parallelisation is **NPRTRV**. Fourier transforms are done latitude by latitude for each **NPROMA**-packet. A processor treats a subset of latitudes, one latitude at the time. A processor works on complete latitudes. A processor can treat a subset of the vertical levels (if **NPRTRV**>1).

Data reorganisation and transpositions are necessary in the Fourier space between the zonal wave structure necessary for Legendre transforms, and the latitudinal structure necessary for Fourier transforms (routines **TRMTOL** and **TRLTOM**).

∗ **Legendre transforms (North-South Fourier for LAM models):** **NPROC**=**NPRTRW**∗**NPRTRV**. The total number of processors involved in the A-level parallelisation is **NPRTRW**. The total number of processors involved in the B-level parallelisation is **NPRTRV**. Legendre transforms are done zonal wave number by zonal wave number. A processor treats a subset of zonal wave numbers, one zonal wave number at the time. A processor can treat only a subset of the vertical levels (if **NPRTRV**>1).

∗ **Spectral computations:** The total number of processors involved in the A-level parallelisation is **NPRTRW**. The total number of processors involved in the B-level parallelisation is **NPRTRV** for horizontal diffusion, **NPRTRN** for semi-implicit scheme. Spectral computations are done zonal wave number by zonal wave number. A processor treats a subset of zonal wave numbers. For more details see documentations (IDSI) about semi-implicit scheme and (IDDH) about horizontal diffusion scheme.

∗ **Computation of Legendre polynomials:** This computation is done latitude by latitude for all the spectral coefficients. Data reorganisation and transpositions (done by **SUTRLE**) are necessary for Legendre transforms to gather all latitudes and split into zonal wave numbers because for Legendre transforms one processor treats a subset of zonal wave numbers and all latitudes.

∗ **Additional remarks about LEQ_REGIONS.** This key controls the way of optimising B-level parallelisation in grid-point space for reduced Gaussian grid (global models). More details are given in documentation (IDEUL).

8 Specific variables in modules and namelists.

8.1 Specific variables of the "trans" and "etrans" libraries.

Variables (and comments) can be found in the following modules of the "trans" (modules **TPM_...**) and "etrans" (modules **TPMALD_...**) libraries, see comments in the code for more details. They may have mirror counterparts in "arpifs/module".

∗ **"trans":**

- **TPM_CONSTANTS**: constants.
- **TPM_CTL**.
- **TPM_DIM**: dimensions.
- **TPM_DISTR**: variables describing distributed memory parallelization.
- **TPM_FFT**: setup for fast Fourier transforms. **TRIGS** contains the sines and cosines which are used in the matricial product which defines the Fourier transform.
- **TPM_FIELDS**: Legendre polynomials and some variables linked to the Gaussian grid. For a given total wave number jn, **NLTN**(jn) is the number of available Legendre coefficients $P(n = jn, m)$.
- **TPM_FLT**: variables for fast Legendre transforms.
- **TPM_GEN**: general control variables.
- **TPM_GEOMETRY**: data describing Gaussian grid.
- **TPM_POL**: encapsulated routines for Legendre polynomials.
- **TPM_TRANS**: variables "local" to a specific call to a transform.

∗ **"etrans":**

- **TPMALD_DIM**: dimensions.
- **TPMALD_DISTR**: variables describing distributed memory parallelization.
- **TPMALD_FFT**: setup for fast Fourier transforms. **TRIGSE** contains the sines and cosines which are used in the matricial product which defines the Fourier transform.
- **TPMALD_FIELDS**: contains for example the eigen-values of the inverse Laplace operator.
- **TPMALD_GEO**: data describing plane projection grid.
- **TPMALD_TCDIS** (useless).

8.2 Specific variables of the "arpifs" library.

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

- IFS_INIT object:
  - **YOMARG** (0-level control, former command line) and **YOMCT0** (0-level control): for example NCONF, LSPRT. Some of these variables are in namelist **NAMCT0**.
  - **YOMMP0** ("ifs_init" distributed memory environment, see documentation (IDDM) for more details).
- Geometry:
  - **YOMDIM**, **YOMDIMV** (geometry dimensioning): most of variables. Some of these variables are in namelist **NAMDIM**.
  - **YOMGEM** (computational space grid-point horizontal geometry): all variables. Some of these variables are in namelist **NAMGEM**.
  - **YOMLAP** (constants related to the Laplace space).
  - **YOMLEG** (description of Legendre polynomials).
  - **YOMMP** (geometry-dependent distributed memory environment, see documentation (IDDM) for more details).

  - **YOMTRANS** (spectral transforms control variables). Some of these variables are in namelist **NAMTRANS** and **NAMTRANS0**.
  - **TYPE_GEOMETRY**: geometry (head structure of some geometry modules listed above).
- Other modules:
  - **YOMDIMF** (field dimensioning): most of variables.

8.3 Additional remarks.

- In a spectral array X dimensioned with **NSPEC2**, order of storing spectral coefficients is the following: $\Re(X)_{(n=0,m=0)}$, $\Im(X)_{(n=0,m=0)}$, $\Re(X)_{(n=1,m=0)}$, $\Im(X)_{(n=1,m=0)}$, ..., $\Re(X)_{(n=N_{\rm s},m=0)}$, $\Im(X)_{(n=N_{\rm s},m=0)}$, $\Re(X)_{(n=1,m=1)}$, $\Im(X)_{(n=1,m=1)}$, $\Re(X)_{(n=2,m=1)}$, $\Im(X)_{(n=2,m=1)}$, ..., $\Re(X)_{(n=N_{\rm s},m=1)}$, $\Im(X)_{(n=N_{\rm s},m=1)}$, ..., $\Re(X)_{(n=N_{\rm s}-1,m=N_{\rm s}-1)}$, $\Im(X)_{(n=N_{\rm s}-1,m=N_{\rm s}-1)}$, $\Re(X)_{(n=N_{\rm s},m=N_{\rm s}-1)}$, $\Im(X)_{(n=N_{\rm s},m=N_{\rm s}-1)}$, $\Re(X)_{(n=N_{\rm s},m=N_{\rm s})}$, $\Im(X)_{(n=N_{\rm s},m=N_{\rm s})}$.
- In the array **RPNM** containing the Legendre polynomials, order of storing Legendre coefficients for a given Gaussian latitude is the following for (n,m): $(N_{\rm s}+1,0)$, $(N_{\rm s},0)$, ..., $(1,0)$, $(0,0)$, $(N_{\rm s}+1,1)$, $(N_{\rm s},1)$, ..., $(1,1)$, ..., $(N_{\rm s}+1,N_{\rm s})$, $(N_{\rm s},N_{\rm s})$, ..., $(N_{\rm s},N_{\rm s})$.
- The reader can also look at modules linked to the GMV and GFL structures: modules have names containing root **gfl** or **gmv**. See comments inside these modules and documentation (IDEUL) for more details.

9 Bibliography.

A lot of references are available in note ARPEGE nr 30.

9.1 Publications.

- Courtier, Ph., C. Freydier, J.F. Geleyn, F. Rabier and M. Rochas, 1991: The ARPEGE project at METEO-FRANCE. ECMWF Seminar Proceedings 9-13 September 1991, Volume II, 193-231.
- Courtier, Ph., and J.F. Geleyn, 1988: A global numerical weather prediction model with variable resolution: Application to the shallow-water equations. *Quart. J. Roy. Meteor. Soc.*, **114**, 1321-1346.
- Laprise, R., 1992: The resolution of global spectral models. *Bulletin American Meteor. Society*, **73**, 1453-1454.
- Ritchie, H., C. Temperton, A. Simmons, M. Hortal, T. Davies, D. Dent, and M. Hamrud, 1995: Implementation of the semi-Lagrangian method in a high resolution version of the ECMWF forecast model. *Mon. Wea. Rev.*, **123**, 489-514.
- Schmidt, F., 1977: Variable fine-mesh in spectral global model. *Beitr. Phys. Atmos.*, **50**, 211-227.
- Swarztrauber, P.N., 2002: On computing the points and weights for Gauss-Legendre quadrature. *SIAM J. Sci. Comput.*, **24**, 945-954.
- Temperton, C., 1991: On scalar and vector transform methods for global spectral models. *Mon. Wea. Rev.*, **119**, 1303-1307.
- Wedi, N. P., M. Hamrud and G. Mozdzynski, 2013: A fast spherical harmonics transform for global NWP and climate models. *Mon. Wea. Rev.*, **141**, 3450-3461.
- Williamson, D. L., and J. M. Rosinski, 2000: Accuracy of reduced grid calculations. *Q. J. R. Meteorol. Soc.*, **126**, 1619-1640.
- Yessad, K. and P. Bénard, 1996: Introduction of a local mapping factor in the spectral part of the METEO-FRANCE global variable mesh numerical forecast model. *Quart. J. Roy. Meteor. Soc.*, **122**, 1701-1719.

For non-hydrostatic models aspects:

- Bubnová, R., G. Hello, P. Bénard, and J.F. Geleyn, 1995: Integration of the fully elastic equations cast in the hydrostatic pressure terrain-following coordinate in the framework of the ARPEGE/Aladin NWP system. *Mon. Wea. Rev.*, **123**, 515-535.
- Geleyn, J.F., and R. Bubnová, 1995: The fully elastic equations cast in hydrostatic pressure coordinate: accuracy and stability aspects of the scheme as implemented in ARPEGE/Aladin. *Atm. Ocean, special issue memorial André Robert.*
- Laprise, R., 1992: The Euler equations of motion with hydrostatic pressure as an independent variable. *Mon. Wea. Rev.*, **120**, 197-207.

9.2 Some internal notes and other ARPEGE notes.

- (TDECDYN) 2018: IFS technical documentation (CY45R1). Part III: dynamics and numerical procedures. Available at "https://software.ecmwf.int/wiki/display/IFS/Official+IFS+Documentation".
- (TDECTEC) 2018: IFS technical documentation (CY45R1). Part VI: technical and computational procedures. Available at "https://software.ecmwf.int/wiki/display/IFS/Official+IFS+Documentation".
- (IDTSN) Hamrud, M., 2002: New transform library. Internal note (9pp) available on the intranet server "http://www.umr-cnrm.fr/gmapdoc/".
- (IDEQR) Mozdzynski, G., 2006: A new partitioning approach for IFS. Internal note, 6pp.
- (IDTSAL) Radnoti, G., 2001: The transformation package for ALADIN. Internal note, 6pp.
- (NTA30) Rochas, M., et Ph. Courtier, 1992: La méthode spectrale en météorologie. Note de travail ARPEGE numéro **30**, 58pp.
- (IDBAS) Yessad, K., 2019: Basics about ARPEGE/IFS, ALADIN and AROME in the cycle 46t1r1 of ARPEGE/IFS (internal note).
- (IDEUL) Yessad, K., 2019: Integration of the model equations, and Eulerian dynamics, in the cycle 46t1r1 of ARPEGE/IFS (internal note).
- (IDSL) Yessad, K., 2019: Semi-Lagrangian computations in the cycle 46t1r1 of ARPEGE/IFS (internal note).
- (IDSI) Yessad, K., 2019: Semi-implicit spectral computations in the cycle 46t1r1 of ARPEGE/IFS (internal note).
- (IDDH) Yessad, K., 2019: Horizontal diffusion in the cycle 46t1r1 of ARPEGE/IFS (internal note).
- (IDDM) Yessad, K., 2019: Distributed memory features in the cycle 46t1r1 of ARPEGE/IFS (internal note).
- (IDLAM) Zagar, M., and C. Fischer, 2007: The ARPEGE/ALADIN Tech'Book: Implications of LAM aspects on the global model code for CY33/AL33. Internal note, 31pp, available on "http://www.umr-cnrm.fr/gmapdoc/".

Appendix 1: Expression of the laplacian in spectral space (spherical geometry).

Equation:

$$\left(\nabla'^2 f\right)_{(n,m)} = \left(\frac{1}{(a\cos\Theta)^2}\frac{\partial^2 f}{\partial\Lambda^2}\right)_{(n,m)} + \left(\frac{1}{a^2\cos\Theta}\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} \tag{31}$$

can be rewritten:

$$\left((a\cos\Theta)^2\nabla'^2 f\right)_{(n,m)} = \left(\frac{\partial^2 f}{\partial\Lambda^2}\right)_{(n,m)} + \left(\cos\Theta\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} \tag{32}$$

One assumes that this equation can be rewritten as:

$$\left(\nabla'^2 f\right)_{(n,m)} = Z_{(n,m)} f_{(n,m)} \tag{33}$$

Combining equations (32) and (33) yields:

$$\left((a\cos\Theta)^2 Zf\right)_{(n,m)} = \left(\frac{\partial^2 f}{\partial\Lambda^2}\right)_{(n,m)} + \left(\cos\Theta\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} \tag{34}$$

∗ **LHS of equation (34):** One uses the following relationships:

$$(a\cos\Theta)^2 = a^2(1-\mu^2) \tag{35}$$

and:

$$[\mu X]_{(n,m)} = e_{(n,m)}X_{(n-1,m)} + e_{(n+1,m)}X_{(n+1,m)} \tag{36}$$

Applying μ to equation (36) yields:

$$[\mu^2 X]_{(n,m)} = e_{(n,m)}\left(e_{(n-1,m)}X_{(n-2,m)} + e_{(n,m)}X_{(n,m)}\right) + e_{(n+1,m)}\left(e_{(n+1,m)}X_{(n,m)} + e_{(n+2,m)}X_{(n+2,m)}\right) \tag{37}$$

i.e.

$$[\mu^2 X]_{(n,m)} = e_{(n,m)}e_{(n-1,m)}X_{(n-2,m)} + \left(e^2_{(n,m)} + e^2_{(n+1,m)}\right)X_{(n,m)} + e_{(n+1,m)}e_{(n+2,m)}X_{(n+2,m)} \tag{38}$$

Introducing the results of equations (35) and (38) in the LHS of equation (34) yields:

$$\left((a\cos\Theta)^2 Zf\right)_{(n,m)} = \left(a^2(1-\mu^2)Zf\right)_{(n,m)} =$$
$$-a^2 e_{(n,m)}e_{(n-1,m)}[Zf]_{(n-2,m)} - a^2\left(-1 + e^2_{(n,m)} + e^2_{(n+1,m)}\right)[Zf]_{(n,m)} - a^2 e_{(n+1,m)}e_{(n+2,m)}[Zf]_{(n+2,m)} \tag{39}$$

∗ **RHS of equation (34):** Equation (9) yields:

$$\left(\frac{\partial^2 f}{\partial\Lambda^2}\right)_{(n,m)} = -m^2 f_{(n,m)} \tag{40}$$

Equation (7) yields:

$$\left(\cos\Theta\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} =$$
$$-(n-1)e_{(n,m)}\left(-(n-2)e_{(n-1,m)}f_{(n-2,m)} + (n+1)e_{(n,m)}f_{(n,m)}\right)$$
$$+(n+2)e_{(n+1,m)}\left(-ne_{(n+1,m)}f_{(n,m)} + (n+3)e_{(n+2,m)}f_{(n+2,m)}\right) \tag{41}$$

i.e.

$$\left(\cos\Theta\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} = e_{(n,m)}e_{(n-1,m)}(n-1)(n-2)f_{(n-2,m)} +$$
$$\left(-(n-1)(n+1)e^2_{(n,m)} - n(n+2)e^2_{(n+1,m)}\right)f_{(n,m)} + e_{(n+1,m)}e_{(n+2,m)}(n+2)(n+3)f_{(n+2,m)} \tag{42}$$

The sum of equations (40) and (42) yield:

$$\left(\frac{\partial^2 f}{\partial\Lambda^2}\right)_{(n,m)} + \left(\cos\Theta\frac{\partial\left(\cos\Theta\frac{\partial f}{\partial\Theta}\right)}{\partial\Theta}\right)_{(n,m)} = e_{(n,m)}e_{(n-1,m)}(n-1)(n-2)f_{(n-2,m)}$$
$$+\left(-\frac{m^2}{n(n+1)} - \frac{(n-1)}{n}e^2_{(n,m)} - \frac{(n+2)}{n+1}e^2_{(n+1,m)}\right)n(n+1)f_{(n,m)} + e_{(n+1,m)}e_{(n+2,m)}(n+2)(n+3)f_{(n+2,m)} \tag{43}$$

$*$ **Identity of the LHS and the RHS:** After some strenuous calculations which are not detailed in this documentation and using the expression of $e_{(n,m)}$ (see formula (8)), one can show that the two expressions:

$$-1 + e^2_{(n,m)} + e^2_{(n+1,m)}$$

and:

$$-\frac{m^2}{n(n+1)} - \frac{n-1}{n} e^2_{(n,m)} - \frac{n+2}{n+1} e^2_{(n+1,m)}$$

are equal. So equation (43) can be rewritten:

$$\left(\frac{\partial^2 f}{\partial \Lambda^2}\right)_{(n,m)} + \left(\cos\Theta \frac{\partial\left(\cos\Theta \frac{\partial f}{\partial \Theta}\right)}{\partial \Theta}\right)_{(n,m)} = e_{(n,m)} e_{(n-1,m)} (n-1)(n-2) f_{(n-2,m)}$$
$$+ \left(-1 + e^2_{(n,m)} + e^2_{(n+1,m)}\right) n(n+1) f_{(n,m)} + e_{(n+1,m)} e_{(n+2,m)} (n+2)(n+3) f_{(n+2,m)} \quad (44)$$

The identity of the LHS and RHS of equation (34) can be satisfied taking:

$$Z_{(n,m)} = -\frac{n(n+1)}{a^2} \quad (45)$$

So equation (33) can be rewritten:

$$\left(\nabla'^2 f\right)_{(n,m)} = -\frac{n(n+1)}{a^2} f_{(n,m)} \quad (46)$$

Appendix 2: Computation of Gaussian latitudes and Gaussian weights (spherical geometry).

Appendix 2.1: First method.

Method used until CY35T2.

∗ **Gaussian latitudes:** $ndglg$ is the number of Gaussian latitudes (for convenience it is assumed to be an even number), jgl is the index numbering the latitudes. Θ is the latitude on the computational sphere, $\mu = \sin\Theta$. $\mathcal{P}_{(n,m)}$ are the unnormalized Legendre polynomials. The relationship between $P_{(n,m)}$ and $\mathcal{P}_{(n,m)}$ is:

$$\mathcal{P}_{(n,m)}(\mu) = \frac{1}{\sqrt{(2n+1)\frac{(n-m)!}{(n+m)!}}} P_{(n,m)}(\mu) \tag{47}$$

The Gaussian latitudes Θ_{jgl} match the following relationship:

$$\mathcal{P}_{(n=ndglg,m=0)}\left(\mu_{jgl}\right) = 0 \tag{48}$$

A first guess approximation for Θ_{jgl} (which are "nearly" equidistant) yields:

$$\Theta_{jgl}^{(i=0)} = \frac{\pi}{2} - \frac{4jgl-1}{4ndglg+2}\pi - \frac{1}{8ndglg^2}\frac{\cos\left(\frac{4jgl-1}{4ndglg+2}\pi\right)}{\sin\left(\frac{4jgl-1}{4ndglg+2}\pi\right)} \tag{49}$$

and thus $\mu_{jgl}^{(i=0)} = \sin\Theta_{jgl}^{(i=0)}$. An iterative Newton method is then applied to find the "final" value of μ_{jgl}.

$$\mu_{jgl}^{(i+1)} = \mu_{jgl}^{(i)} - \frac{\mathcal{P}_{(n=ndglg,m=0)}\left(\mu_{jgl}^{(i)}\right)}{\frac{d\mathcal{P}_{(n=ndglg,m=0)}}{d\mu}\left(\mu_{jgl}^{(i)}\right)} \tag{50}$$

This algorithm is used to find the Northern hemisphere latitudes (jgl from 1 to $ndglg/2$). The southern hemisphere latitudes are given by:

$$\mu_{ndglg+1-jgl} = -\mu_{jgl}$$

One can see that this algorithm needs the calculation of $\mathcal{P}_{(n=ndglg,m=0)}$ and $\frac{d\mathcal{P}_{(n=ndglg,m=0)}}{d\mu}$. This calculation is done with some recurrence formulae which will be detailed in appendix 3 for the normalized Legendre polynomials. The derivatives of the unnormalized Legendre polynomials are given by the following formula, once knowing the unnormalized Legendre polynomials:

$$\frac{d\mathcal{P}_{(n,m)}}{d\mu} = -\frac{n\mu}{1-\mu^2}\mathcal{P}_{(n,m)} + \frac{n+m}{1-\mu^2}\mathcal{P}_{(n-1,m)} \tag{51}$$

∗ **Gaussian weights:** They are given by the following formula:

$$\omega_{jgl} = \frac{1-\mu_{jgl}^2}{\left[ndglg * \mathcal{P}_{(n=ndglg,m=0)}\left(\mu_{jgl}\right)\right]^2} \tag{52}$$

Appendix 2.2: Second method.

Method used by the current version of **SUGAW**. Uses a method described in (Swarztrauber, 2002).

Appendix 3: Computation of the normalized Legendre polynomials (spherical geometry).

Appendix 3.1: First method.

Method used until CY35T2.

∗ **Introduction:** The algorithm uses recurrence formulae. There are several possible solutions, the following one is used in the code. The order is the following one:

- $P_{(n=0,m=0)}$; $P_{(n=1,m=0)}$; $P_{(n=1,m=1)}$.
- $P_{(n,m=0)}$ where $n > 1$.
- $P_{(n,m=1)}$ where $n > 1$.
- $P_{(n,m=n)}$ where $n > 2$.
- The remaining Legendre polynomials in the following order: $P_{(n=3,m=2)}$; $P_{(n=4,m=2)}$; $P_{(n=4,m=3)}$; $P_{(n=5,m=2)}$; $P_{(n=5,m=3)}$; $P_{(n=5,m=4)}$; ...; $P_{(n,m=2)}$; $P_{(n,m=3)}$; ...; $P_{(n,m=n-1)}$; $P_{(n+1,m=2)}$; $P_{(n+1,m=3)}$; ...; $P_{(n+1,m=n-1)}$; $P_{(n+1,m=n)}$; etc.

∗ **Wave numbers 0 and 1:**

$$P_{(n=0,m=0)} = 1$$
$$P_{(n=1,m=0)} = \sqrt{3}\mu$$
$$P_{(n=1,m=1)} = \sqrt{1.5\,(1-\mu^2)}$$

∗ **Other polynomials for $m = 0$:** $P_{(n,m=0)}$ is computed knowing $P_{(n-1,m=0)}$ and $P_{(n-2,m=0)}$. The following recurrence formula is used (formula (1.20) of (Rochas and Courtier, 1992)):

$$\mu P_{(n,m)} = \sqrt{(n^2-m^2)/(4n^2-1)}P_{(n-1,m)} + \sqrt{((n+1)^2-m^2)/(4(n+1)^2-1)}P_{(n+1,m)} \tag{53}$$

Changing n into $n-1$ in equation (53), then putting $P_{(n,m)}$ in the LHS, yields:

$$P_{(n,m)} = \mu\sqrt{(4n^2-1)/(n^2-m^2)}P_{(n-1,m)} - \sqrt{((n-1)^2-m^2)/(4(n-1)^2-1)}\sqrt{(4n^2-1)/(n^2-m^2)}P_{(n-2,m)} \tag{54}$$

For $m = 0$ equation (54) becomes:

$$P_{(n,m=0)} = \mu\sqrt{(4n^2-1)/n^2}P_{(n-1,m=0)} - \sqrt{(n-1)^2/(4(n-1)^2-1)}\sqrt{(4n^2-1)/n^2}P_{(n-2,m=0)} \tag{55}$$

∗ **Other polynomials for $m = 1$:** $P_{(n,m=1)}$ is computed knowing $P_{(n,m=0)}$ and $P_{(n-1,m=0)}$. The following recurrence formula is used (formula (1.22) of (Rochas and Courtier, 1992)):

$$\sqrt{1-\mu^2}P_{(n,m)} = \sqrt{[(2n+1)(n+m-1)]/[(2n-1)(n+m)]}P_{(n-1,m-1)} + \sqrt{(n-m+1)/(n+m)}P_{(n,m-1)} \tag{56}$$

For $m = 1$ equation (56) becomes:

$$\sqrt{1-\mu^2}P_{(n,m=1)} = \sqrt{[(2n+1)n]/[(2n-1)(n+1)]}P_{(n-1,m=0)} + \sqrt{n/(n+1)}P_{(n,m=0)} \tag{57}$$

∗ **Other polynomials for $m = n$:** The following recurrence formula is used:

$$P_{(n,n)} = \sqrt{1-\mu^2}\sqrt{(2n+1)/(2n)}P_{(n-1,n-1)} \tag{58}$$

∗ **Other polynomials:** The following recurrence formula is used (formula (1.31) of (Rochas and Courtier, 1992)):

$$P_{(n,m)} = \sqrt{\frac{(2n+1)(n+m-1)(n+m-3)}{(2n-3)(n+m)(n+m-2)}}P_{(n-2,m-2)} - \sqrt{\frac{(2n+1)(n+m-1)(n-m+1)}{(2n-1)(n+m)(n+m-2)}}P_{(n-1,m-2)} + \sqrt{\frac{(2n+1)(n-m)}{(2n-1)(n+m)}}P_{(n-1,m)} \tag{59}$$

Appendix 3.2: Second method.

Method used by the current version of **SUPOL**. Uses a method described in (Swarztrauber, 2002) and briefly described in (Wedi et al., 2013).

Appendix 4: Fair numbers between 1 and 50000 writing $2^{(1+p)}3^q5^r$.

2	4	6	8	10	12	16	18	20	24	30	32	36	40	48
50	54	60	64	72	80	90	96							
100	108	120	128	144	150	160	162	180	192	200	216	240		
250	256	270	288	300	320	324	360	384	400	432	450	480	486	
500	512	540	576	600	640	648	720							
750	768	800	810	864	900	960	972							
1000	1024	1080	1152	1200	1250	1280	1296	1350	1440	1458				
1500	1536	1600	1620	1728	1800	1920	1944							
2000	2048	2160	2250	2304	2400	2430	2500	2560	2592	2700	2880	2916		
3000	3072	3200	3240	3456	3600	3750	3840	3888						
4000	4050	4096	4320	4374	4500	4608	4800	4860						
5000	5120	5184	5400	5760	5832	6000	6144	6250	6400	6480	6750	6912	7200	7290
7500	7680	7776	8000	8100	8192	8640	8748	9000	9216	9600	9720			
10000	10240	10368	10800	11250	11520	11664	12000	12150	12288					
12500	12800	12960	13122	13500	13824	14400	14580							
15000	15360	15552	16000	16200	16384	17280	17496	18000	18432	18750	19200	19440		
20000	20250	20480	20736	21600	21870	22500	23040	23328	24000	24300	24576			
25000	25600	25920	26244	27000	27648	28800	29160							
30000	30720	31104	31250	32000	32400	32768	33750	34560	34992					
36000	36450	36864	37500	38400	38880	39366								
40000	40500	40960	41472	43200	43740	45000	46080	46656	48000	48600	49152	50000		