TranSpHO

Transformation along the Spherical Harmonics for Ocean applications

User's guide

Jean-Michel Brankart

`http://pp.ige-grenoble.fr/pageperso/brankarj/`

Institut des Géosciences de l'Environnement
Université Grenoble Alpes, CNRS, France

The main purpose of the TranSpHO modules is to provide all tools needed to transform the ocean data assimilation problem by projection on the spherical harmonics. The objective is to separate scales and to make the assimilation system behave differently for different scales (e.g. different localization algorithm, different error parameterization,...).

The tools are provided as a library of modules, which can be easily plugged in any assimilation software. Examples are provided to illustrate how the main routines can be used an to check that the library has been installed correctly.

This library includes:

- the forward and backward transformation of gridded data,
- the regression of observation data on the spherical harmonics. and

1 Description of the modules

In this section, the modules are described one by one, giving for each of them: the method that has been implemented, the list of public routines that can be called by the user (with a description of input and output data), and an estimation of the computational cost as a function of the size of the problem.

1.1 Module: `sphylm`

The purpose of this module is to perform (i) the forward and backward transformation of a two-dimansional field on the sphere in the basis of the spherical harmonics (section 1.1.1), and (ii) the regression of observation data on the spherical harmonics (section 1.1.2).

1.1.1 Forward and backward transformation

Method

The approach is to project the two-dimensional field in spherical coordinates $f(\theta, \varphi)$ on the spherical harmonics $Y_{lm}(\theta, \varphi)$:

$$f_{lm} = \int_{\Omega} f(\theta, \varphi) Y_{lm}(\theta, \varphi)\, d\Omega \tag{1}$$

where l and m are the degree and order of each spherical harmonics. The integral is over the whole sphere but $f(\theta, \varphi)$ can be extended with zeroes outside of the available domain, at least for our assimilation purpose (since only differences between two model state play a role in the assimilation algorithm). From the spectrum f_{lm} can then be reconstructed using the inverse transformation:

$$f(\theta, \varphi) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} f_{lm} Y_{lm}(\theta, \varphi) \tag{2}$$

so that any spectral band can be extracted by limiting the sum over a specific range of l (remembering that the wavelength $\lambda = 2\pi R/l$, where R is the earth radius).

Routines

The implementation of the above equations is made of 3 public routines that can be called by an outside program, and 2 private routines that are only called inside the module. The public routines are:

init_ylm: to precompute the Legendre functions [used in the computation of the spherical harmonics: $Y_{lm}(\theta, \varphi)$] up to the required degree (`kjpl`), on the required latitude range (`latmin` to `latmax`) and with the required resolution (`dlatmax`);

proj_ylm: to project a gridded two-dimensional field on all spherical harmonics up to degree `kjpl`: `ktab` is the array with the field data (already weighted with the spherical surface of each grid cell, so that the integral simplifies to a simple sum), `klon` is the array with the corresponding longitudes, `klat` is the array with the corresponding latitudes, and `kproj` is the resulting projection array (or spectrum in the basis of spherical harmonics).

back_ylm: to perform the inverse transformation from the spectrum (`kproj`) to the spherical coordinates (specified in `klon` and `klat` as above), `ktab` is here the output data (containing only the scales between degrees `jlmin` and `kjpl`).

The private routines are:

plm: to evaluate the Legendre functions, recursively up to the required degree (`jpl`);

ylm: to evaluate the spherical harmonics (degree: `kl`, order: `km`) at the specified spherical coordinates (longitude: `klon`, latitude: `klat`).

Computational cost

The computational complexity (leading asymptotic behaviour for large systems) is given by:

$$C \sim knl_{\max}^2 = kn\left(\frac{2\pi R}{\lambda_c}\right)^2 \tag{3}$$

where n is the number of grid points (size of `ktab`), $l_{\max}$ is the maximum degree of spherical harmonics (`kjpl`), and k is the number of operation required for one single evaluation of spherical harmonics (i.e˙Â one call to `ylm`). In this code, $k \simeq 5$, plus the cost of one evaluation of the sine or cosine function. In the second formula, R is the earth radius and λ_c is the cutting wave length.

1.1.2 Regression of observations

Method

The approach is to look for the spectral amplitudes f_{lm} so that the corresponding field $f(\theta, \varphi)$ (up to degree $l_{\max}$):

$$f(\theta, \varphi) = \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} f_{lm} Y_{lm}(\theta, \varphi) \tag{4}$$

minimizes the following distance to observations (f_k^o at coordinates θ_k, φ_k, $k = 1, \ldots, p$):

$$J^o = \sum_{k=1}^{p} \frac{1}{\sigma_k^{o2}} \left[f(\theta_k, \varphi_k) - f_k^o\right]^2 \tag{5}$$

where σ_k^o is typically the observation error standard deviation (including the representativity error corresponding to the signal above degree $l_{\max}$).

If the observation system is insufficient to control all spectral components with sufficient accuracy, the penalty function J can include a regularization term J^b: $J = J^o + \rho J^b$, where the parameter ρ can be tuned (between 0 and 1) to modify the importance of J^b with respect to J^o. The regularization term J^b is the following norm of the spectral amplitudes f_{lm}:

$$J^b = \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} \frac{f_{lm}^2}{\sigma_{lm}^2} \tag{6}$$

where σ_{lm} is typically the standard deviation of the signal along each spherical harmonics.

Scale localization of the regression. In order to improve the efficiency of the above algorithm (if $l_{\max}$ and p are large), options to apply the above algorithm locally have been implemented. This corresponds to solving the regression problem for a limited block of degrees l in Eq. (4), and loop over all blocks from 0 to $l_{\max}$. This is controlled by two parameters specifying the number of degrees l in the last block (`kmaxbloc`) and the overlap between the blocks (`koverlap`). (The number of degrees in each block is automatically modified as a function of l to keep about the same number of spherical harmonics in each block of degrees, and thus minimize the overall cost.)

In addition to that, this localization approximation requires iterating on the local regressions until covergence. Two possibilities have been implemented: (i) perform the loop from 0 to $l_{\max}$ several times until convergence (this is the `'local'` option), and (ii) perform the loop from 0

to current block several times until convergence, before going to the next block of degrees (this is the `'sublocal'` option). These iterations are controlled by two parameters specifying the maximum number of iterations (`kmaxiter`), hopefully never reached, and the maximum relative variation for convergence (`kepsilon`).

This makes quite a large number of parameters to specify, but it can be expected that the default values (see below) should be sufficient for most applications (especially if the weights `kwei` and `kobswei` are correctly tuned).

Routines

The implementation of the above equations is made of 2 public routines that can be called by an outside program, and 3 private routines that are only called inside the module. The public routines are:

init_regr_ylm: to modify the default parameters in the regression of observations: the type of regression (`ktype` = `'global'`, `'local'` or `'sublocal'`), the maximum number of iterations (`kmaxiter`), the number of degrees l in the last block (`kmaxbloc`), the overlap between the blocks (`koverlap`), the maximum relative variation for convergence (`kepsilon`), the weight of the background term in the cost function (`krho`). The default values are: `ktype='local'`, `kmaxiter`=50, `kmaxbloc`=1, `koverlap`=1, `kepsilon`=0.01, `krho`=1;

regr_ylm: to perform the regression of observations: `kregr` is the resulting array with the amplitudes (or spectrum in the basis of spherical harmonics), `kwei` is the weight to give to each spherical harmonics (σ_{lm}), `kobs` is the array with observations, `klon` is the array with the corresponding longitudes, `klat` is the array with the corresponding latitudes, and `kproj` is the resulting projection array (or spectrum in the basis of spherical harmonics). `kobswei` is the weight to give to each observation ($1/\sigma_k^o$),

The private routines are:

regr_ylm_loc: to perform the regression over a local range of degrees of the spherical harmonics;

regr_calc_ab: to compute the matrix **A** and vector **b** of the linear system **Ax**=**b**.

regr_calc_x: to solve of the linear system **Ax**=**b**.

Computational cost

The computational complexity (leading asymptotic behaviour for large systems) of the global regression algorithm is given by:

$$C \sim k\,p\,l_{\max}^4 + \frac{1}{6}\,l_{\max}^6 \tag{7}$$

where p is the number of observations (size of `kobs`), $l_{\max}$ is the maximum degree of spherical harmonics (`kjpl`), and k is the number of operation required for one single evaluation of spherical harmonics (i.eˆ Â one call to `ylm`). In this code, $k \simeq 5$, plus the cost of one evaluation of the sine or cosine function. The first term corresponds to the computation of the **A** matrix in routine **regr_calc_ab**, and the second term to the resolution of the linear system in routine **regr_calc_x**.

For the local regression, the computational complexity of each iteration is given by:

$$C \sim k\,p\,l_{\max}^3 \frac{(b+o)^2}{b} + \frac{1}{6}\,l_{\max}^4 \frac{(b+o)^3}{b^2} \tag{8}$$

where b is the size of the local blocks (parameter `kmaxbloc`) and o is overlap between the blocks (parameter `koverlap`). The defaults values are $b = o = 1$.

1.2 Module: `spharea`

The purpose of this module is to compute the spherical area $\Delta\Omega_i$ associated to each data point $f(\theta_i, \varphi_i)$ in the numerical evaluation of the integral in equation (1).

Method

For gridded data the method is to use spherical trigonometry formula to compute the surface of each grid cell. The standard formula applied in the code are:

- Great-circle distance between two points on a sphere (with unit radius):

$$\Delta\sigma = \arccos\left(\sin\phi_1 \cdot \sin\phi_2 + \cos\phi_1 \cdot \cos\phi_2 \cdot \cos(\Delta\lambda)\right) \tag{9}$$

  where ϕ_1, λ_1 and ϕ_2, λ_2 are the gerographical latitudes and longitudes in radians of the two points, and $\Delta\lambda = \lambda_1 - \lambda_2$.

- Area of a spherical triangle (on a sphere with unit radius):

$$\Delta\Omega = 4\arctan\sqrt{\tan\frac{s}{2}\tan\frac{s-a}{2}\tan\frac{s-b}{2}\tan\frac{s-c}{2}} \tag{10}$$

  where a, b, c are the great-circle lengths of the edges of the triangle, and $s = (a+b+c)/2$.

Routines

The module implementing the above equations contains one public routine:

mesh_area: to compute the spherical area (`area`) associated to each quadrangle of the grid from the latitude and longitude of the mesh vertices (`lon` and `lat`).

Computational cost

Negligible.

2 Examples

In this section, the examples provided with the library are described, giving for each of them: the purpose of the examples, the list of modules/routines that are illustrated, the input parameters and data, the calling sequence of the library routines, with a description of inputs and outputs for each of them, and a description of the final result that is expected.

2.1 Low-pass filter on the sphere

The purpose of this example is to illustrate how to use the module `sphylm` to extract the large scale component of an anomaly field. This is done by calling successivley the following routines from the module `sphylm`:

init_ylm: to initialize the computation of the spherical harmonics;

proj_ylm: to compute the spectrum of the input field;

back_ylm: to reconstruct the original field up to a given scale.

Input data are: the random field generated by the example given in StochTools, and the maximum degree of the spherical harmonics to keep in the filtered field. The output is the filtered field written in NetCDF.

2.2 Observation regression on the sphere

The purpose of this example is to illustrate how to use the module `sphylm` to perform the regression of observations on the spherical harmonics up to a specified degree. This is done by calling successivley the following routines from the module `sphylm`:

init_ylm: to initialize the computation of the spherical harmonics;

init_regr_ylm: to define the parameters of observation regression;

regr_ylm: to perform the regression of observations;

back_ylm: to reconstruct the original field up to a given scale.

Input data are: the large-scale field generated by the previous example, and the observation sampling ratio (e.g. one observation every 50 grid points). The output is the regressed field written in NetCDF.