Chapter 1: Introduction to Spherical Harmonics

The Slepian Working Group

October 25, 2021

1 Initialization

Start Matlab or Octave and switch to the folder `Slepian`

In Matlab to initialize the Slepian software in Matlab, run

```
initialize
```

For Octave, I recommend running

```
more off
```

and

```
warning off
```

before initialization, as otherwise it will print a long list of unimportant messages. You can then initialize the Slepian software in Octave using

```
initialize_octave
```

The first time you run `initialize_octave`, it will download and install missing packages. This will take a few minutes. Make sure you have internet access otherwise Octave will not be able to install the necessary packages

2 Testing the software

To check if the software installed correctly you can run a few demos using the function `demos_slepian_golf`

```
    help demos_slepian_golf
```

This will show you a description of what the program does. For example, try

```
    demos_slepian_golf(6)
```

This will calculate and plot a single gradient vector Slepian function for Eurasia and maximum spherical-harmonic degree $L = 10$. You will learn later in this tutorial what the spherical-harmonic degree is. We leave the tutorial for gradient vector Slepian functions for a later time. If you are interested you can check out for example the article "Potential-field estimation using scalar and vector Slepian functions at satellite altitude" (doi: `10.1007/978-3-642-27793-1_64-2`) or the information including links on the website `www.alainplattner.net`.

3 Introduction to spherical harmonics

Spherical harmonics are a counterpart to monomials (building blocks of polynomials, the "wiggly" lines) but on the sphere. Remember that polynomials were always sums of monomials multiplied by some numbers. For example $3x^2-20.95x+5.11113$ is a *polynomial* consisting of the *monomials* x^2, x, and 1 multiplied by the *coefficients* $3, -20.95, 5.1113$ and then summed up. Spherical harmonics do exactly the same. Each of them has a degree, we multiply them with coefficients and then sum them up. One thing that is different is that besides having a degree, spherical harmonics also have an order. This is because they live on a sphere which is a surface and therefore has two directions in which it can "wiggle".

Here we denote spherical harmonics with the capital letter Y followed by its degree l and its order m as indices. Both degree l and order m are integers. The degree l is 0 or greater whereas the order can be positive, zero, or negative. So we generally write $Y_{l\,m}$. For example for degree 4 and order 2 it will be $Y_{4\,2}$ (note the small gap between 4 and 2). The *degree* l of a spherical harmonic says, how many zero-crossing lines it has in total. The *order* m of a spherical harmonic says how many of the zero-crossings are perpendicular to the equator. The sign of the order says if one of the zero-crossings is along the prime-meridian (negative order m) or if the maximum value is on the prime meridian (positive order m). From this explanation it is clear that the order m must always be equal or smaller to the degree l (there can't be more zero-crossings perpendicular to the equator than there are zero-crossings altogether).

Let's look at a few examples. To make it easier to visualize the spherical harmonics, I am plotting them on a sphere and in Mollweide projection over Earth's map. $Y_{0\,0}$ is just a constant number over the entire planet. That's a bit boring so I won't plot it here.

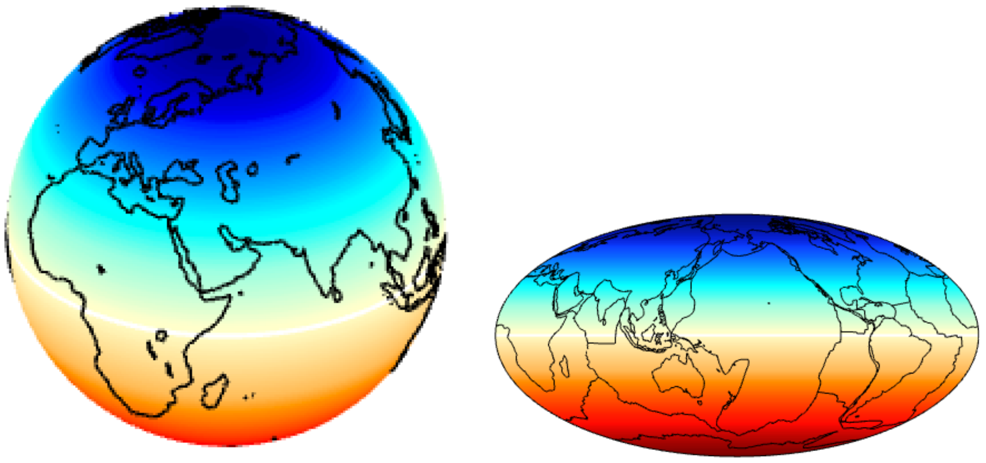

Figure 1: This is spherical harmonic $Y_{1\,0}$. It's a dipole with its north pole over the geographical North Pole and its south pole over the geographical South Pole. In the left panel I am plotting it over a sphere, in the right panel I am using Mollweide projection

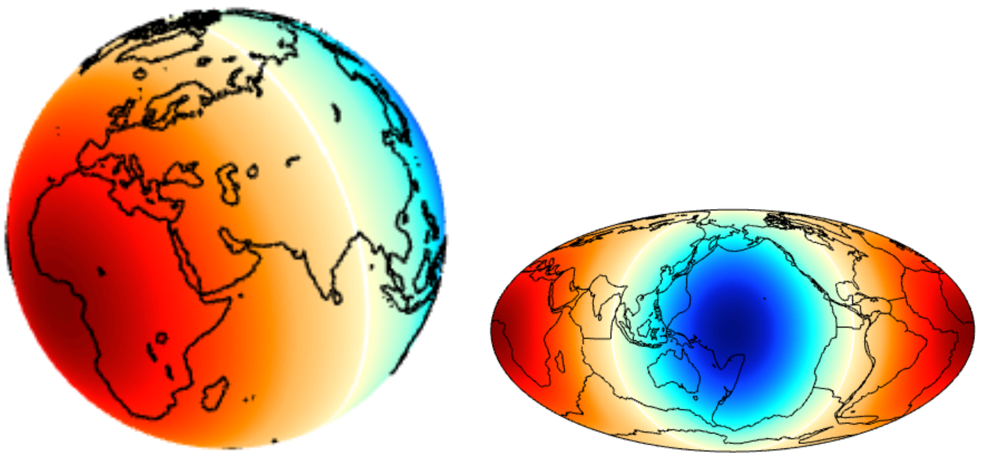

Figure 2: This is spherical harmonic $Y_{1\,-1}$. It's a dipole with its north pole over the Pacific and its south pole over Africa. In the left panel I am plotting it over a sphere, in the right panel I am using Mollweide projection.

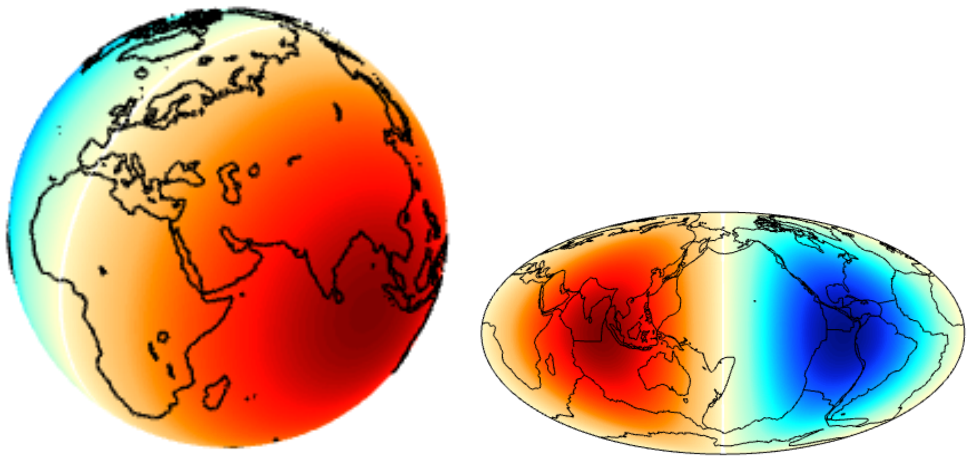

Figure 3: This is spherical harmonic Y_{11}. It's a dipole with its north pole over the Indian Ocean and its south pole over Central America. In the left panel I am plotting it over a sphere, in the right panel I am using Mollweide projection.

Question: How can we make a dipole with its south pole over the geographical North Pole and its North pole over the geographical South Pole?

I made these plots using our codes that you just installed. Once you chose a degree l and an order m, you must select a plotting grid. Let's make a pixel every half degree. From your Octave tutorial you remember that

```
lat=-90:0.5:90;
```

Sets up such a vector for the latitudes and

```
lon=0:0.5:360;
```

for the longitudes. We will use the function `ylm.m` to calculate spherical-harmonic values on the sphere on the grid we just defined. Notice that the symbol `l` is the letter l. Run

```
l=3
```

and

```
m=-2
```

and then

```
Y3m2=ylm(l,m,pi/180*(90-lat),pi/180*lon);
```

This will calculate the values of Y_{lm} for your chosen l and m on the grid you defined with latitudes `lat` and longitudes `lon`. We need the factors $\pi/180$ because the function `ylm.m` requires the grid in radians and we need the $90-$ `lat` because the function `ylm.m` works with colatitudes instead of latitudes.

To plot the resulting evaluated spherical harmonic `Y3m2` on a Mollweide projection, run

```
    plotplm(Y3m2,pi/180*lon,pi/180*lat,1)
```

Let's change the color to a bit a nicer color scheme, run

```
  kelicol(1)
```

To plot the resulting evaluated spherical harmonic on a rotatable sphere, you need to first open a new figure or close the old one and then run

```
    plotplm(Y3m2,pi/180*lon,pi/180*lat,2)
```

Exercise: Reproduce figures 1, 2, and 3. Plot different spherical harmonics for different degrees and orders. Plot sums of spherical harmonics. Plot linear combinations of spherical harmonics. How does for example
$4Y_{3\,1} - 0.2Y_{1\,-1} + 2Y_{5\,-2}$ look like?

If it looks like in figure 4, then your programing was correct.

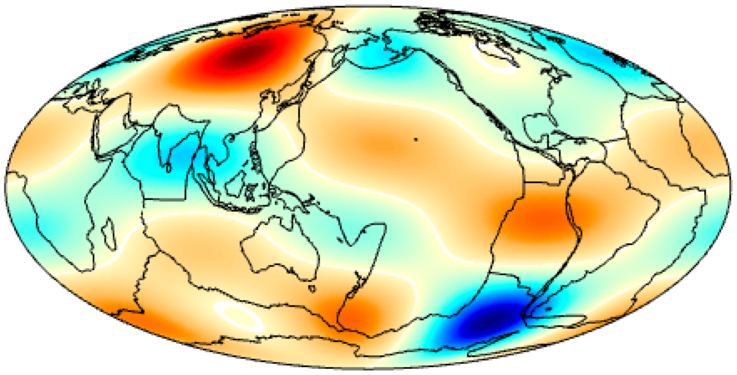

Figure 4: Linear combination of spherical harmonics $4Y_{3\,1} - 0.2Y_{1\,-1} + 2Y_{5\,-2}$ in Mollweide projection

What we learn from this is that even though the individual spherical harmonics look very regular with their zero-crossings either parallel or perpendicular to the equator, linear combinations of spherical harmonics can look very arbitrary. In fact, every reasonably behaved function on the sphere (no infinite values and no sudden jumps, etc.) can be expressed as a linear combination of spherical harmonics. For some we might need spherical harmonics with very high degrees.

Exercise: Make your own linear combination of spherical harmonics. Play around with the degrees and coefficients. Can you make up some shape in your mind that you want to draw with spherical harmonics and then reproduce it?

Calculating spherical harmonics individually, multiplying them with coefficients, and then summing them up is cumbersome. Luckily, our software allows us to do this more efficiently. Instead of just calculating a single spherical harmonic we can calculate all of them up to a any degree L we choose. Run for example

```
  L=10
```

and then

```
  Y=ylm([0 L],[],pi/180*(90-lat),pi/180*lon);
```

This will calculate a matrix `Y` in which each row represents an evaluated spherical harmonic. the rows are ordered in what we call here the ADDMOUT format (more about this in Section 4). This is an informal term used within the software suite and is not generally used in the community. ADDMOUT means the following ordering for the (degree/order) pairs: (0/0), (1/-1), (1/0), (1/1), (2/-2), (2/-1), etc. This means that the spherical harmonic with degree 2 and order 1 is in the 8th row of the matrix `Y`.

Question: In which row is degree 3, order -2? Which degree/order pair do we find in row 20?

If we now have a vector of coefficients that we call `c`, then we can calculate the linear combination of the spherical harmonics evaluated in matrix `Y` with the coefficients in vector `c` using matrix-vector multiplication. For a maximum spherical harmonic degree L there are $(L+1)^2$ spherical harmonics including all degrees between 0 and L and all corresponding orders. We can calculate a random row vector with the correct length by running

```
  c=randn(1,(L+1)^2);
```

and evaluate the linear combination of the spherical harmonics as

```
  F=c*Y;
```

The linear combination `F` has the form of a vector. We need to put it into the right shape by running

```
  Fp=reshape(F,length(lat),length(lon));
```

Plot this function with the command we have seen before

```
    plotplm(Fp,pi/180*lon,pi/180*lat,1)
```

Question: Why are there $(L+1)^2$ spherical harmonics for degrees 0 to L and all coprresponding orders?

Question: Why does the matrix vector multiplication `F=c*Y;` calculate a linear combination of the spherical harmonics with the corresponding coefficients in `c`?

Exercise: If I give you a random list of data point values together with their locations, can you find a way to calculate best-fitting spherical-harmonic coefficients? Remember least-squares fitting.

To solve this exercise, you need to evaluate the spherical harmonics Y_{lm} on arbitary points, not just the regular grids we have seen. To do that, you first need to have the longitudes and latitudes of your arbitary ("irregularly spaced") points organized as two vectors list. For example, you may want to calculate the spherical harmonics at the (lon, lat) locations (286, 40), (18, -34), and (0, 52). In this case, your longitude vector would be `lon=[286,18,0]` and your latitude vector would be `lat=[40,-34,52]`. To evaluate all spherical harmonic functions between degree 0 and L at these locations, run the following:

```
Y=ylm([0,L], [], pi/180*(90-lat), pi/180*lon, [], [], [], 1);
```

The `1` (the number "one") at the end of this function call is to indicate to `ylm` that the points we provide are not for a grid but irregular points. The open and close brackets before that are to just use the standard values for all the other, here irrelevant, values that `ylm` needs as input parameters.

You will also need to make sure that the matrix dimensions agree. If you want to solve the classical $Ac = b$ system of equation where A is the matrix, x is the column vector of unknowns, b is the column right hand side vector, then you need to define the matrix A and the right hand side vector b accordingly. The matrix A needs to be the transpose of `Y`.

```
A = Y';
```

Then you can obtain the spherical-harmonic coefficients `c` through

```
c = A\b;
```

4 Organization and ordering of spherical-harmonic coefficients

Some representations of spherical harmonics split the spherical-harmonic coefficients up into coefficients for the cos longitudinal variation and coefficients for the sin longitudinal variation, each with a positive order. In our representation, the order m can be positive or negative. The positive orders correspond to the sin coefficients and the negative coefficients correspond to the cos coefficients.

Our software has three ways to store the spherical-harmonic coefficients in a variable, LMCOSI, ADDMOUT, and ADDMON.

The first, most intuitive one is LMCOSI. As the name says, this variable, a matrix, has four columns. The first one indicating the degree l, the second column contains the (non-negative) order m, the third column contains the coefficients for the negative or zero order (corresponding to the cos longitudinal variation) and the 4-th column contains the coefficients for the positive order (corresponding to the sin longitudinal variation).

We often need to work with the spherical-harmonic coefficients as a single vector to be able to carry out linear algebraic operations. To do that, we have two different **ordering schemes**: ADDMOUT and ADDMON.

In the ADDMOUT ordering scheme, we order the coefficients first by their degree and then by their order, from negative to positive:

l	0	1	1	1	2	2	2	2	2	etc.
m	0	-1	0	1	-2	-1	0	1	2	etc.

For example, if the coefficients organized in LMCOSI are:

l	m	cos	sin
0	0	-1.2	0
1	0	3.1	0
1	1	0.01	1.35

then their corresponding ADDMOUT representation is

```
[-1.2, 0.01, 3.1, 1.35]
```

A second ordering scheme is ADDMON. As for ADDMOUT, the primary ordering scheme is using the degrees. But here, the orders are arranged by their absolute values:

l	0	1	1	1	2	2	2	2	2	etc.
m	0	0	-1	1	0	-1	1	-2	2	etc.

Our coefficients from before would thus be ordered as

```
[-1.2 , 3.1, 0.01, 1.35]
```

Luckily, this transformation does not need to be done by hand. Our functions `coef2lmcosi` and `lmcosi2coef` can transform the vector form to and back from the LMCOSI form. You can choose whether you want ADDMON or ADDMOFF by supplying an extra flag:

`lmcosi2coef(lmcosi,0)` translates into ADDMON, and `lmcosi2coef(lmcosi,1)` translates to ADDMOFF. The same flag applies when you transform your vector of coefficients into LMCOSI. You need to let the program know if the vector was in ADDMON or ADDMOFF ordering scheme: `coef2lmcosi(coef,0)` or `coef2lmcosi(coef,1)`, respectively.

5 Normalization of the spherical harmonics

In the scientific literature contains a myriad of different normalizations for the spherical harmonics. Here we use "unit sphere normalization". But even this normalization is not unique. It could mean "normalization with respect to the sphere of unit radius", or "normalization with respect to the sphere of unit surface area".

So specifically: Our function `ylm` normalizes with respect to the unit-area sphere. Mathematically, this means that if Y_{lm} and $Y_{l'm'}$ are outputs of the function `ylm`, then their inner product over the sphere is either 1 (if $Y_{lm} = Y_{l'm'}$) or zero (if they are not the same):

$$\int_\Omega Y_{lm} Y_{l'm'} \, d\Omega = \delta_{ll'} \delta_{mm'}$$

Our collection of Matlab/Octave codes also contains the highly-efficient `plm2xyz`, which can plot spherical-harmonic coefficients on the sphere in a computationally more efficient way than `ylm`. But `plm2xyz` is normalized for the "unit radius sphere", thus

$$\int_\Omega Y_{lm} Y_{l'm'} \, d\Omega = 4\pi \delta_{ll'} \delta_{mm'}$$

The plotting routine `plotplm` uses `plm2xyz` and thus follows the same normalization.

So if you obtain spherical-harmonic coefficients through inversion using `ylm` (see Section 3) and you want to plot them using `plm2xyz`, then you must divide the coefficients by `sqrt(4*pi)` before plotting to obtain the correct spatial representation:

```
  coef = Y'\b;

  lmcosi = coef2lmcosi(coef/sqrt(4*pi) ,0); % Y is in ADDMON

  plotplm(lmcosi,[],[],[],1)
```