

# Estimation of Laplace Series Coefficients via 2-D Numerical Quadrature

Evan Voyles

November 20, 2022

## 1 Abstract

This paper provides an alternative approach to the computation of Laplace Series coefficients  $C_l^m$  and  $S_l^m$  with algorithmic complexity  $O(Nl^2)$  in time and  $O(Nl^2)$  in space. We make use of the fact that the spherical harmonics form a complete and orthogonal set to directly compute  $C_l^m$  as a 2-dimensional integral. We estimate this integral using numerical quadrature techniques and the resulting model is blazing fast compared to a least squares model running in  $O(Nl^4)$ . Harnessing the structure of the data sets, we perform many

## 2 Preliminaries

Any sufficiently regular function defined on a sphere can be written

$$f(\theta, \phi) = \sum_{l=0}^{+\infty} \sum_{m=0}^l \bar{P}_l^m(\cos \theta) [C_l^m \cos m\phi + S_l^m \sin m\phi]. \quad (1)$$

Where  $C_l^m, S_l^m$  are *spherical harmonic* coefficients (also referred to as Laplace series coefficients) of degree  $l$  and order  $m$ ;  $\bar{P}_l^m$  are the fully normalized Associated Legendre Functions of degree  $l$  and order  $m$ ;  $\theta$  is the polar angle and  $\phi$  is the azimuthal angle, as per ISO convention<sup>1</sup>.

The project proposal suggests computing the coefficients  $C_l^m, S_l^m$  by setting up an overdetermined linear system of equations and finding the least squares solution. Unfortunately, this problem runs in  $O(Nl_{max}^4)$  time and requires about  $8N(l_{max} + 1)^2$  bytes of memory. That is absurdly prohibitive. Furthermore, there are no *self-evident* parallelization techniques to speed up computation time. In this report we propose an alternative approach to computing the Laplace series coefficients  $C_l^m$  and  $S_l^m$  that runs in  $O(Nl_{max}^2)$  time complexity with trivial parallelization models.

---

<sup>1</sup>We have chosen the ISO standard coordinate system to ensure the integrity of future computations. For a discussion on transforming the  $(\phi, \lambda)$  coordinates provided by NASA, see Appendix A

### 3 Data Sets

NASA data sets contain the altitude information for a spherical coordinate pair. The spherical coordinates can be represented as the discretization of a 2-d rectangle with width  $2\pi$  and height  $\pi$ . NASA's data sets contain observations over a discretization of  $[-\frac{\pi}{2}, \frac{\pi}{2}] \times [-\pi, \pi]$ , represented as a  $n_\theta \times n_\phi$  grid with equidistant points.

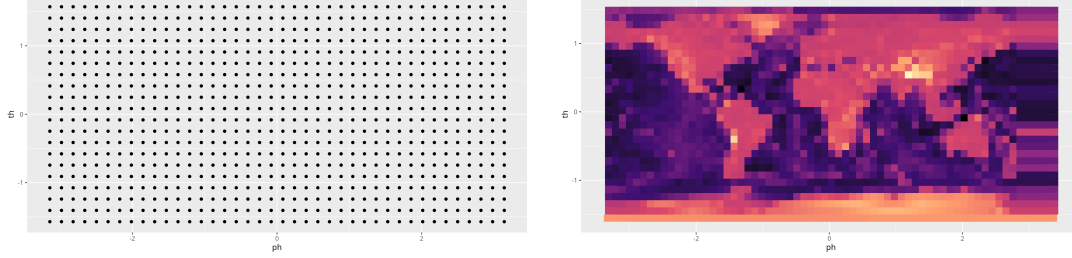


Figure 1: Discretization points (left) of a reduced data set with  $n_\theta = 20$ ,  $n_\phi = 40$ . On the vertical axis we have  $\theta \in [-\pi/2, \pi/2]$ , or the latitude; On the horizontal axis we have  $\phi \in [-\pi, \pi]$ , the longitude.

Due to the linear spacing, we only need to know the number of points  $n_\theta$  and  $n_\phi$  to recreate the spherical coordinate pairs provided by the data sets, allowing us to decrease the running memory footprint of our program. In order study informative analytics about our model, we first enumerate the characteristics of each data set.

	$n_\theta$	$n_\phi$	$N$
small	180	360	64800
med	540	1080	583200
hi			
ultra			

$$\theta \in [0, \pi], \quad \phi \in [0, \pi/2]$$

We'll use these values later when we study the time it takes to compute our model.

#### 3.1 Visualization

TODO: [ ] add 3-d maps

### 4 Mathematical Model

#### 4.1 Coefficients

The coefficients  $C_l^m$  and  $S_l^m$  are defined as the orthogonal projection of  $f(\theta, \phi)$  onto the basis functions  $Y_l^m$ , the spherical harmonics of degree  $l$  and order  $m$ . This definition is perfectly analogous to the computation of fourier coefficients for  $2\pi$  periodic functions. Written in terms of the associated legendre polynomials, these integrals are defined as

$$C_l^m = \int_0^{2\pi} \int_0^\pi f(\theta, \phi) P_l^m(\cos \theta) \cos(m\phi) \sin \theta d\theta d\phi$$

$$S_l^m = \int_0^{2\pi} \int_0^\pi f(\theta, \phi) P_l^m(\cos \theta) \sin(m\phi) \sin \theta d\theta d\phi$$

Taking advantage of the structure of the data sets, we approximate these coefficients via numerical quadrature.

### 4.1.1 Computation via Numerical Integration

We approximate the Laplace Series coefficients via a 2 dimensional analogue of the rectangle rule.

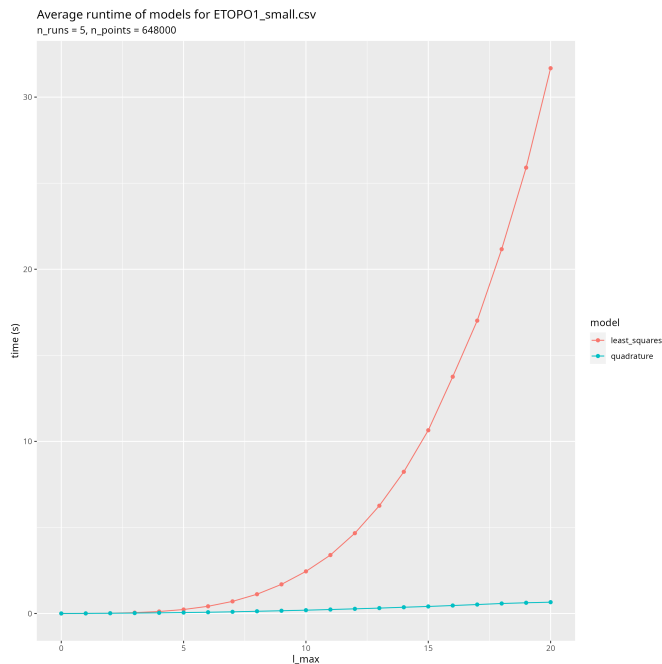
$$\begin{aligned} C_l^m &\approx \sum_{i=1}^{n_\phi} \sum_{j=1}^{n_\theta} f(\theta_j, \phi_i) P_l^m(\cos \theta_j) \cos(m\phi_i) \sin \theta_j \Delta\theta \Delta\phi \\ &\approx \Delta\theta \Delta\phi \sum_{i=1}^{n_\phi} \cos(m\phi_i) \sum_{j=1}^{n_\theta} f(\theta_j, \phi_i) P_l^m(\cos \theta_j) \sin \theta_j \end{aligned}$$

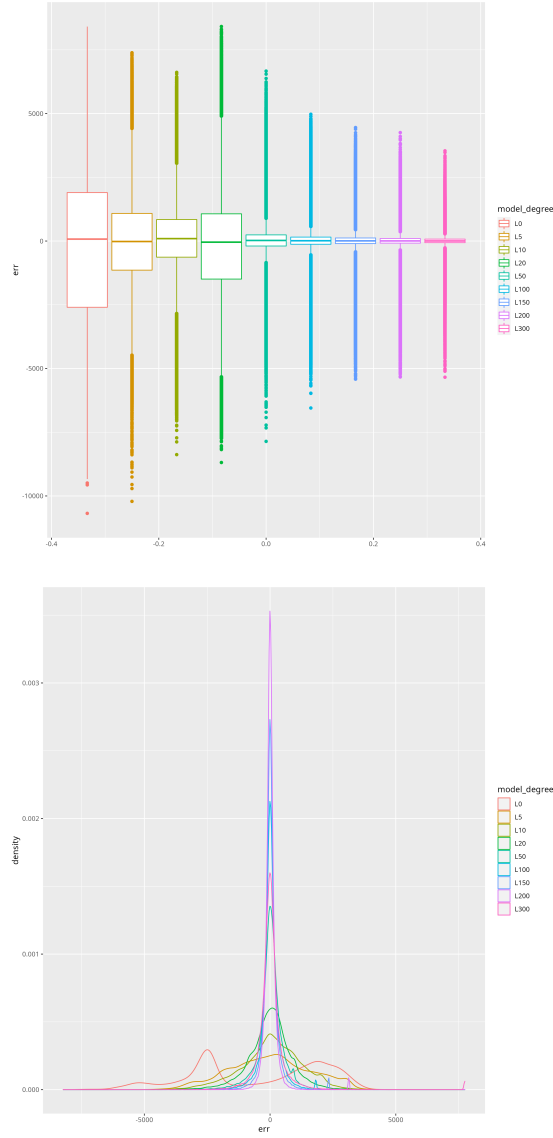
$$\begin{aligned} S_l^m &\approx \sum_{i=1}^{n_\phi} \sum_{j=1}^{n_\theta} f(\theta_j, \phi_i) P_l^m(\cos \theta_j) \sin(m\phi_i) \sin \theta_j \Delta\theta \Delta\phi \\ &\approx \Delta\theta \Delta\phi \sum_{i=1}^{n_\phi} \sin(m\phi_i) \sum_{j=1}^{n_\theta} f(\theta_j, \phi_i) P_l^m(\cos \theta_j) \sin \theta_j \end{aligned}$$

This method computes *individual* Laplace series coefficients in  $O(N)$  time! Thus, computing the coefficients for a Discrete Laplace Series (DLS) of order  $l_{max}$  runs in  $O(Nl_{max}^2)$ . More interestingly, each  $C_l^m$  or  $S_l^m$  can be computed **independently** of any other coefficient. Whereas solving the linear least squares problem requires computing every  $C_l^m$  and  $S_l^m$  in a single operation, this method allows us the ability to compute any arbitrary  $C_l^m$  directly. Why do we care? If I already have the first 500  $C_l^m$  coefficients computed and stored in a text file, I can *directly* compute the 501st in  $O(N)$  time. Furthermore, this method gives rise to obvious and easy-to-implement parallelization schemes.

## 4.2 Comparison with Linear Squares

Let's start by exploring the small data set and gathering data for how long it takes to compute a model.





Tested on the ETOP01\_small.csv data set with  $l_{max} = 20$ , the model computed via numerical integration had an average error of about 2 times that of the least squares approach, but ran over 100 times faster.

**TODO**

- Provide more detailed numeric comparison
- Provide timing analysis

section

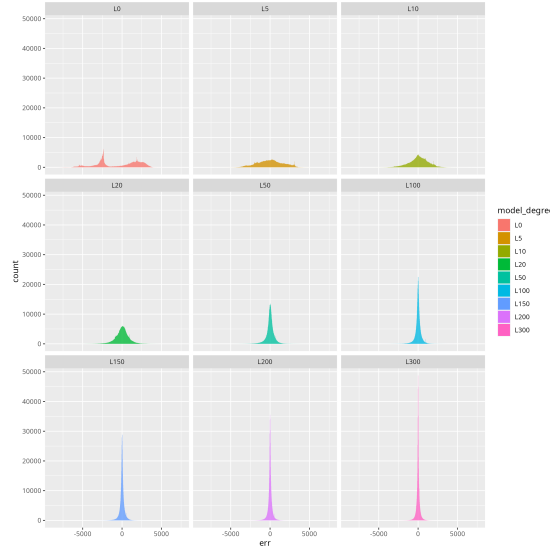
## 5 Improvements to the model

The least squares method provides a more accurate model, but runs in  $O(Nl_{max}^4)$  time. In this section we propose improvements to our model computed via quadrature.

### 5.1 Gradient Descent

We can apply methods of optimization to minimize the mean squared error (MSE) of our model. To begin we explicitize the MSE then compute its gradient

**TODO** I have already derived the gradient of the MSE with respect to all  $C_l^m$  and  $S_l^m$  on paper



- Write up derivation in latex

Computing the gradient vector runs in  $O(Nl_{max}^2)$  time

### 5.1.1 Stochastic Gradient Descent

Computing the gradient with respect to a single  $C_l^m$  requires summing up certain values for all  $N$  points of the data set. For larger data sets (e.g.  $N_{high} = 1800 * 3600 = 6480000$ ), this computation can become intractable. We can speed up our gradient computation by computing as estimate of the gradient,  $\hat{V}MSE$ , by randomly sampling  $n$  points from our entire data set. This reduces the computation of a gradient vector from  $O(Nl_{max}^2)$  to  $O(nl_{max}^2)$ . This is an **insane** improvement to the run time of our algorithm. Instead of summing all 6480000 points of the large data set, we could fix  $n$  at say  $n = 1000$  instead, sacrificing accuracy for a speedup of 6,480.

## 5.2 MCMC Algorithms

In initial tests my gradient descent learning is *sometimes* reducing the MSE but other times it's increasing it... If I don't figure out exactly what's going wrong then I plan on implementing MCMC algorithms like Gibb's sampling to explore the sample space and only accept proposal vectors that decrease the MSE.

## 6 Parallelization

TODO (trivial)

## 7 Conclusion

Instead of calculating the model with a direct method that runs in  $O(Nl_{max}^4)$  time, We propose a hybrid model that first directly computes the coefficients in  $O(Nl_{max}^2)$  time then iteratively improves the MSE via stochastic methods running in  $O(n_{training}n_{sample}l_{max}^2)$  time.

Results are to follow :)

## A Spherical Coordinates Transformation

The data sets ETOP01\_\*.csv provided by NASA represent spherical coordinates in an unconventional format:

$$(\phi, \lambda) \in \left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \times [-\pi, \pi]$$

where  $\phi$  refers to the latitude and  $\lambda$  refers to the longitude. In this section we outline the transformation from NASA's coordinate system to the ISO standard representation and prove equivalence between (1) and the Project presentation's  $f(\phi, \lambda)$ .

To avoid confusion, we denote the spherical coordinate pair in ISO coordinates as  $(\theta_{iso}, \phi_{iso})$  and NASA's coordinate scheme as  $(\phi_{nasa}, \lambda_{nasa})$ . We remark that the polar angle  $\theta_{iso}$  is simply equal to  $\frac{\pi}{2} - \phi_{nasa}$ . We corroborate this fact by remarking that a latitude of  $\phi_{nasa} = 0$  corresponds to a polar angle of  $\phi_{iso} = \frac{\pi}{2}$ . Conveniently, the longitude  $\lambda_{nasa}$  is equivalent to the azimuthal angle  $\phi_{iso}$ .

**Proposition 1.** Let the spherical coordinate pairs  $(\theta_{iso}, \phi_{iso})$ ,  $(\phi_{nasa}, \lambda_{nasa})$  be defined as above.

$$f(\theta_{iso}, \phi_{iso}) \equiv f(\phi_{nasa}, \lambda_{nasa})$$

*Proof.*  $f(\phi_{nasa}, \lambda_{nasa})$  is defined in the Project proposal as

$$f(\phi_{nasa}, \lambda_{nasa}) = \sum_{l=0}^{+\infty} \sum_{m=0}^l \bar{P}_l^m(\sin \phi_{nasa}) [C_l^m \cos m \lambda_{nasa} + S_l^m \sin m \lambda_{nasa}] \quad (2)$$

$$= \sum_{l=0}^{+\infty} \sum_{m=0}^l \bar{P}_l^m(\sin(\pi/2 - \theta_{iso})) [C_l^m \cos m \phi_{iso} + S_l^m \sin m \phi_{iso}] \quad (3)$$

$$= \sum_{l=0}^{+\infty} \sum_{m=0}^l \bar{P}_l^m(\cos \theta_{iso}) [C_l^m \cos m \phi_{iso} + S_l^m \sin m \phi_{iso}] \quad (4)$$

$$\equiv f(\theta_{iso}, \phi_{iso}) \quad (5)$$

□

## References

- [1] V. Rokhlin and M. Tygert, "Fast Algorithms for Spherical Harmonic Expansions," *SIAM Journal on Scientific Computing*, vol. 27, pp. 1903–1928, Jan. 2006. Publisher: Society for Industrial and Applied Mathematics.
- [2] R. Blais, "Discrete Spherical Harmonic Transforms: Numerical Preconditioning and Optimization," pp. 638–645, June 2008.
- [3] T. Limpanuparb and J. Milthorpe, "Associated Legendre Polynomials and Spherical Harmonics Computation for Chemistry Applications," Oct. 2014. arXiv:1410.1748 [physics].
- [4] E. W. Weisstein, "Spherical Harmonic." Publisher: Wolfram Research, Inc.
- [5] G. B. Arfken, H. J. Weber, and F. E. Harris, *Mathematical Methods for Physicists: A Comprehensive Guide*. Amsterdam ; Boston: Academic Press Inc, 7e édition ed., Mar. 2012.
- [6] P. Dyke, "Fourier Series," in *An Introduction to Laplace Transforms and Fourier Series* (P. Dyke, ed.), Springer Undergraduate Mathematics Series, pp. 83–122, London: Springer, 2014.
- [7] E. W. Weisstein, "Laplace Series." Publisher: Wolfram Research, Inc.