Draft version March 26, 2024
Typeset using LaTeX **modern** style in AASTeX631

starry 2

Lionel J. Garcia,[1] Soichiro Hattori,[1] and Daniel Foreman-Mackey[1]

[1] *Center for Computational Astrophysics, Flatiron Institute, New York, NY, USA*

ABSTRACT

Keywords: exoplanet detection methods, stellar activity, time series analysis, gaussian processes regression, computational methods, GPU computing

INTRODUCTION

tldr: Modeling occultation light curves serve a wide variety of science cases, especially in exoplanetary science. The latest models account for non-uniform surfaces of stars (starry), but given the large dataset available, could be optimized further. We present several optimizations that improve on the starry model.

Stellar light curves and radial velocity models provided an important tool to study exoplanetary systems. At the origin of these models lies the first principles of orbital mechanics, the Kepler equations, and the algorithms used to solve them. On top of that, models like Agol, of occultation light curves extended our capability to study these systems in greater details, using transits. In practice, computing this model and making inference based on these observables has been enabled by the implementation of these models with optimized frameworks and inference tools (pymc3), providing robust implementation of sampling algorithms and other inferrence tools. A good example of this synergy is the development of exoplanet and starry in c++ backed by Pymc3 which enabled major discoveries based on large datasets. With the recent lunch of JWST, we crumble under data. For example, transmission spectra on hundreds of channels, multi-planet systems and a large number of free parameters with their priors. At the same time, machine learning continue to grow, and novel framework appear and lead to application in physics. One of this framework is jax, developped by google, and gaining a huge popularity on the astronomy community. JAX is a hp machine-learning library that pre-compile code to LAX for paralilisation. It allows complex model to be written direclty in python, where usually a lower-level langage like C is preferable, and parallelised on CPU, GPU or any lax-compatible device. Not only it's good because of the higher level syntax of python that made it popular in science. But also because of the capability to run expansive models on GPU, a trend observed in other field of ML and DL where scaling is required.

In this paper, we describe an implementation of the starry formalism in jax, allowing for the inference of a large number of parameters on complex models, exploiting the latest dev in ML machinery.

With *starry*, every light curve computation at a given time t involves a rotation of the spherical harmonics basis, from the rest-frame of the star (Figure 1, left) to its sky-projected orientation (Figure 1, right), with a final rotation relative to the position of the occulting body (see Figure 2. from Luger et al. 2019).

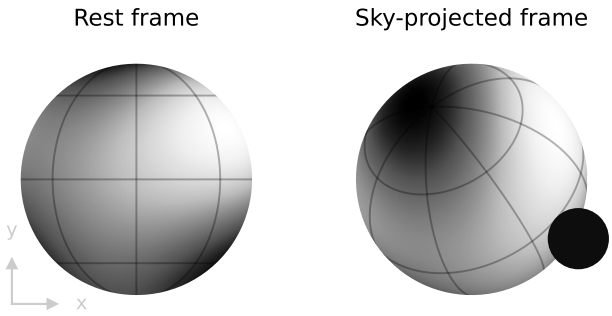

Figure 1. Rotation of the spherical harmonics basis to place the star in the sky-projected frame of the observer.

The rotation of spherical harmonics involves the computation of Wigner-D-matrices, commonly obtained through robust recursion relations in both degree l and order m, but leading to costly computations. Hence, knowing how to decompose the full spherical harmonics rotation using pre-computed rotations is essential to achieve optimal performance in the *starry* framework. Figure 2 shows the 6 elementary rotations currently used in the `starry` implementation.

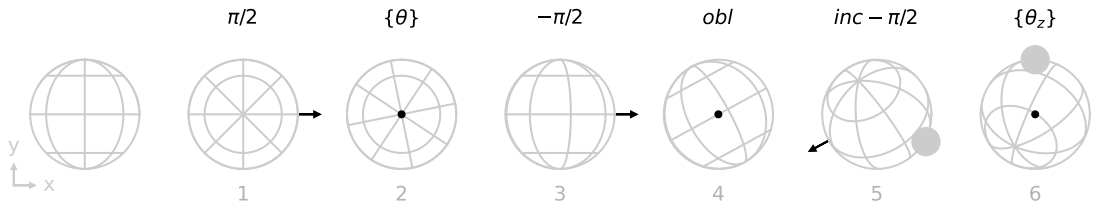

Figure 2. Consecutive rotations of the spherical harmonics basis in the `starry` implementation. Each schematic shows a sphere rotated from a previous orientation around an axis displayed as a black vector and an angle shown on top.

In this figure, rotations 1 to 3 correspond to a rotation of the star around its rotation axis, and rotations 3 to 5 place the star on a sky-projected frame. Finally, rotation

6 aligns the spherical harmonics map to the occulting body on the y axis in order to apply Green's theorem in the appropriate basis (Figure 2 from Luger et al. 2019). The first thing to notice is that rotations 1 to 3 could be simply reduced to a rotation of angle θ about the y-axis, turning steps 1 to 3 into one. However, rotations around the pole, i.e. with a rotation axis along z, have simpler expressions that can be implemented separately. In addition, pre-computing these matrices at lower cost is particularly important for steps 2 and 6, involving a potentially large set of angles $\{\theta\}$ and $\{\theta_z\}$, defined over times $\{t\}$, for which to compute the full light curve. This explains the current decomposition of the complete rotation in six separate steps.

In `jaxoplanet`, we compute the Wigner-D-matrices by employing the Risbo recursion relations (Risbo 1996) implemented in `JAX` as part of the `s2fft` Python package (Price & McEwen 2023). In addition, we merge rotations 3, 4 and 5 (see Figure 3) into a single compound rotation of axis

$$\boldsymbol{v} = \frac{1}{\sqrt{1 - \cos^2\left(\frac{inc}{2}\right)\cos^2\left(\frac{obl}{2}\right)}} \begin{pmatrix} \sin\left(\frac{inc}{2}\right)\cos\left(\frac{obl}{2}\right) \\ \sin\left(\frac{inc}{2}\right)\sin\left(\frac{obl}{2}\right) \\ -\cos\left(\frac{inc}{2}\right)\sin\left(\frac{obl}{2}\right) \end{pmatrix}, \tag{1}$$

and angle

$$\omega = 2\cos^{-1}\left(\cos\left(\frac{inc}{2}\right)\cos\left(\frac{obl}{2}\right)\right). \tag{2}$$

This way, the complete rotation reduces to the four separate steps shown in Figure 3.

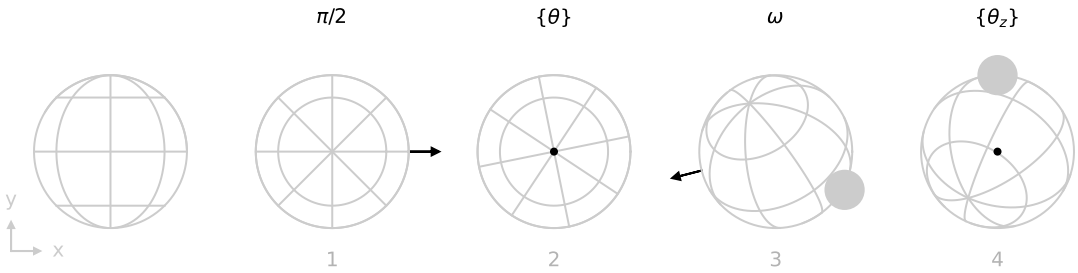

Figure 3. Consecutive rotations of the spherical harmonics basis in the `jaxoplanet` implementation. Each schematic shows a sphere rotated from a previous orientation around an axis displayed as a black vector and an angle shown on top.

2.3. *Wigner-D matrices*

2.4. *Numerical integration of the solution vectors*

3. PERFORMANCE

4. CASE STUDY: ...

REFERENCES

Luger, R., Agol, E., Foreman-Mackey, D., et al. 2019, The Astronomical Journal, 157, 64, doi: 10.3847/1538-3881/aae8e5

Price, M. A., & McEwen, J. D. 2023, Journal of Computational Physics, submitted

Risbo, T. 1996, Journal of Geodesy, 70, 383, doi: 10.1007/BF01090814