Application

Proposal title

Astronomy Using Unevenly Sampled Data

Organization

OpenAstronomy(StingraySoftware)

Summary

Analyzing unevenly sampled data cannot be done with methods of the Fourier domain. We use the Lomb Scargle method to account for the unevenness of the data. I will be implementing the Lomb Scargle cross spectrum and power spectrum from scratch. Then I will be creating wrapper classes and helper functions that will provide a sensible API to the user. The Lomb Scargle cross spectrum implementation can be optimized by using JAX. Reason for choosing to do the power spectrum from scratch is that it allows us to get a raw PSD to which we can apply stingray's methods of normalisation and astropy's implementation does not have that option, and their normalisations are different from the ones used in stingray, and we can use the

$$\mathcal{O}(N \log N)$$

algorithm proposed by Press and Rybicki. There would also be a need of helper functions to create objects of these classes automatically from various input types similar to the existing PSDs. Extra parameter of maximum frequency is to be given as it cannot be inferred from the data as equal to 1/time between two data points. Further the frequency resolution cannot be inferred due to the same reason. The cross spectrum class must be compatible with statistic functions already existing in stingray such as the time lag, phase lag and coherence (which exist only for cross spectra) (if this turns out to not be possible then create new similar statistic functions).

Contributor Information

- Blog : https://pupperemeritus.github.io/gsoc_blog/
- Name : Sri Guru Datta Pisupati
- Email : pupperemeritus9@protonmail.com
- Time-zone : +05:30 (IST)
- Slack : Sri Guru Datta Pisupati
- Github : pupperemeritus
- PR Links :

  - notebooks pull request 64
  - stingray pull request 708
  - stingray pull request 710

Background

I am a second year undergraduate student of Chaitanya Bharathi Institute of Technology studying Artificial Intelligence and Data Science. I have a lot of hobbies out of which some of them include astrophotography, Guitar(Rock and Pop Trinity Grade 8) and chess. Furthermore, I originally wanted to study a course in Computational Astronomy/Astrophysics as it is the confluence of all my interests but could not do it due to lack of availability of such a course in my country. I am passionate about using, propagating and contributing to open source software. I also like to do astrophotography in my free time, though I cannot really produce great images as I am limited by a Bortle 9 sky and lack of a tracker.

Relevant Experience

- Created a discord bot that uses AI to determine the mood of chat messages
- Participated in my college's Hacktober fest and came 5th in my panel.

Interest in Open Astronomy

I've always been interested in astronomy. I also delved into astrophotography recently. Furthermore, I originally wanted to study astrophysics/astronomy along with computer science as my major and minor but the institutes in my country neither offer such flexibility nor the courses in one institute. Not only that, but I have done a few courses on Coursera on astronomy such as Data Driven Astronomy and Analyzing The Universe. Furthermore, I am passionate about using and propagating open source software, and I feel this is a multi-faceted way of giving back to the community that I like so much. Likewise, I feel my skill set is best suited for OpenAstronomy compared to any other project as I have a decent amount of domain as well as programming and software design knowledge. I am always looking to learn more. I view research work and work that aids in other's research as a noble pursuit. Furthermore, I hope OpenAstronomy will not only be my journey into the world of FOSS but also to research, and it will help me gain experience for higher studies.

Detailed Description

The project involves time series analysis of unevenly sampled data. The goal is to create the classes that generates Lomb-Scargle Power spectrum and Crossspectrum of the data respectively and match these implementations with the structure of existing functionalities of Powerspectrum and Crosspectrum classes as well as the normalisations performed by them.

Time-lag, phase lag and coherence of light curves are also to be found out in the case of cross spectra.

The implementation of cross spectrum is to be done from scratch in Python as it only exists in MATLAB and FORTRAN courtesy of Dr. Jeffrey D. Scargle (Studies in astronomical time series analysis. III-Fourier transforms, autocorrelation functions, and cross-correlation functions of unevenly spaced data).

The implementation of power spectrum may not be required since a compatible implementation exists in scipy.signal.LombScargle. We can refer to the original paper by Jeffrey D. Scargle Studies in astronomical time series analysis. II-Statistical aspects of Spectral Analysis of Unevenly Spaced Data. But it is useful to implement it from scratch as we can use the optimizations introduced by William H. Press and George B. Rybicki as shown below.

A deep dive into Lomb Scargle Method and various optimizations

Referring Review paper by Jacob T. VanderPlas

The Lomb Scargle method uses the mean of the sampling intervals to account for the uneven sampling intervals. Lomb Scargle method is equivalently interpreted as a Fourier method as well as a least squares method. The generalized periodogram form ensures time shift invariance.

The least squares approach works by minimizing the chiˆ2 test statistic of a function

$$y(t; f) = A_f \sin(2\pi f(t - \phi_f))$$

where amplitude

$$A_f$$

and phase

$$\phi_f$$

vary with frequency.

$$\chi^2 = \sum_n (y_n - y(t_n; f))^2$$

The periodogram using this approach is given by

$$P(f) = \frac{\chi^2 - \chi^2(f)}{2}$$

where

$$\chi^2(f)$$

is the

$$\chi^2$$

test statistic for the given frequency and

$$\chi^2$$

is the non varying reference model.

The periodogram can also be derived by extending the classical Schuster periodogram by generalizing it to unevenly sampled data which comes from the fourier domain of methods.

Given by

$$P(f) = \frac{1}{2}[\frac{(\sum_n g_n \cos(2\pi f[t_n - \tau]))^2}{\sum_n \cos^2(2\pi f[t_n - \tau])} + \frac{(\sum_n g_n \sin(2\pi f[t_n - \tau]))^2}{\sum_n \sin^2(2\pi f[t_n - \tau])}]$$

where

$$\tau = \frac{1}{4\pi f} \tan^{-1} \frac{\sum_n \sin(4\pi f_n)}{\sum_n \cos(4\pi f_n)}$$

We can modify the least squares approach to account for Gaussian errors. Where

$$\chi^2$$

test statistic is modified as follows

$$\chi^2(f) = \sum_n \frac{(y_n - y_{model}(t_n; f))^2}{\sigma_n^2}$$

Another important modification made is to add an offset term to the sinusoidal model at each frequency in the least squares approach.

$$y_{model}(t; f) = y_0(f) + A_f \sin(2\pi f(t - \tau))$$

The

$$y_{model}$$

can be further generalized by adding K - 1 terms to it.

$$y_{model}(t; f) = A_f^0 + \sum K_{k=1} A_f^k \sin(2\pi k f(t - \phi_f^k))$$

These implementations are

$$\mathcal{O}(N^2)$$

The above can be further optimized by using the optimizations introduced by William H. Press and George B. Rybicki which brings down the overall time complexity of the algorithm to

$$\mathcal{O}(N \log N)$$

The optimizations basically include the following:

- Simplifications of the following trigonometric sums if we define

$$S_y = \sum_j y_j \sin(\omega t_j)$$

$$C_y = \sum_j y_j \cos(\omega t_j)$$

$$S_2 = \sum_j \sin(2\omega t_j)$$

$$C_2 = \sum_j \cos(2\omega t_j)$$

- Then we can replace

$$\sum_j y_j \cos(\omega(t_j - \tau)) = C_y \cos(\omega\tau) + S_y \sin(\omega\tau)$$

$$\sum_j y_j \sin(\omega(t_j - \tau)) = S_y \sin(\omega\tau) - C_y \sin(\omega\tau)$$

$$\sum_j \cos^2(\omega(t_j - \tau)) = 0.5(N + C_2 \cos(2\omega\tau) + S_2 \sin(2\omega\tau))$$

$$\sum_j \sin^2(\omega(t_j - \tau)) = 0.5(N - C_2 \cos(2\omega\tau) - S_2 \sin(2\omega\tau))$$

- Where

  $$\omega\ is\ 2\pi f$$

  and N is the number of samples.
- If the

  $$t_j$$

  's were evenly spaced then the above calculations could be performed by FFTs
- Therefore, we extirpolate the values at evenly spaced

  $$t_j$$

  's by using extirpolation weight which is the same as interpolation weight

$$g(t) = \sum_n w_n(t) g(\hat{t}_n)$$

where

$$w_n(t)$$

are interpolation weights and

$$\hat{t}_n$$

is the extirpolated evenly spaced points

- Our sum of interest

$$\sum_n y_n g(t_n) \approx \sum_j y_j [\sum_k (w_k(t_j) g(\hat{t}_k))] = \sum_k \sum_j h_j w_k(t_j) \equiv \sum_k \hat{h}_k g(\hat{t}_k)$$

where

$$\hat{h}_k = \sum_j h_j w_k(t_j)$$

The following are the practical considerations when implementing the method

- Frequency limits and grid spacing must be chosen carefully. The low frequency limit is often chosen to be 0 for the sake of convenience but can be set to
$$\frac{1}{T}$$
where T is the total span of the time series. The high frequency limit can be calculated by finding the true Nyquist frequency or pseudo Nyquist limit based on careful scrutiny of the window function.
- The interaction of the signal, convolution caused by survey window and noise in the data, the largest peak in the periodogram may correspond to an alias of the true frequency
- The peak may correspond to a higher harmonic of the fundamental frequency which itself is subject to the above aliasing
- The cross spectrum is computed using the Lomb Scargle Fourier Transform.

$$P_{XY} = FT_x(\omega) FT_y^*(\omega)$$

$$FT_x(\omega) = F_0 \sum AX_n \cos(\omega t'_n) + iBX_n \sin(\omega t'_n)$$

Which can be simplified by taking

$$X_i = 1$$

$$FT_x(\omega) = F_0 \sum A\cos(\omega t'_n) + iB\sin(\omega t'_n)$$

We can compute this quite quickly using FFTs

My approach

Proposed Structure

Filename	Description
utils.py	helper functions
LombScargle.py	Implementation of the power spectrum and cross spectrum classes

Additions to utils.py

Helper functions like ACF, CCF that work with unevenly sampled data.

Lomb Scargle Cross Spectrum class

Wraps the Lomb Scargle cross spectrum in our own implementation. Handles input of two light curves, eventlists, timearray and light curve iterable. Needs to perform data validity checks and normalisation(leahy/absolute rms/fractional rms).

Class parameters

Parameter	Description	Default
data1	light curve or eventlist	None
data2	light curve or eventlist	None
power_type	real or complex power	all
norm	Normalisation methods(frac RMS, abs RMS, leahy)	Fractional RMS
dt	time resolution only needed for eventlist objects	Automatically selected sensible value
gti	time intervals which have good data (if not none overrides common gtis from both light curves)	Common gti from both light curves
skip_checks	flag to skip checks	False
maxfreq	maximum frequency to check for	Automatically chosen sensible value
df	frequency resolution	Automatically chosen sensible value

Class attributes

Attribute	Description
freq	array of mid bin frequencies that the lsft samples
power	array of cross spectra (complex numbers)
power error	uncertainties in power
m	the number of averages cross spectra amplitudes in each bin
n	the number of data points/time bins in one segment of the light curves
k	the rebinning scheme if object has been rebinned otherwise is set to 1
nphots1	the total number of photons in light curve 1
nphots2	the total number of photons in light curve 2

NOTE : This is a rough outline and the attributes and parameters may change depending on further research at the time of implementation

Lomb Scargle Power Spectrum

Using our own implementation of Press and Rybicki's fast Lomb Scargle power spectrum. Handles input of one light curve. Helper functions that allow creation of PSD from timearray, eventlists and light curve iterable must be created. And must perform validity checks and normalisation(leahy/absolute rms/fractional rms).

Class parameters

Parameter	Description	Default
data	light curve or eventlist	None
power_type	real or all powers	all
norm	Normalisation methods(frac RMS, abs RMS, leahy)	Fractional RMS
dt	time resolution only needed for eventlist objects	Automatically chosen sensible value
gti	time intervals which have good data (if not None overrides existing gti in light curve)	None
skip_checks	flag to skip checks	False
df	frequency resolution	Automatically chosen sensible value

Parameter	Description	Default
maxfreq	maximum frequency to check for	Automatically chosen sensible value

Class attributes

Attribute	Description
freq	array of mid bin frequencies that the lsft samples
power	array of cross spectra (complex numbers)
power error	uncertainties in power
m	the number of averages cross spectra amplitudes in each bin
n	the number of data points/time bins in one segment of the light curves
k	the rebinning scheme if object has been rebinned otherwise is set to 1
nphots	the total number of photons in light curve

NOTE : This is a rough outline and the attributes and parameters may change depending on further research at the time of implementation

Deliverables

1. Implementation of Lomb Scargle Cross Spectrum along with helper functions, statistical functions, user facing wrapper class, tests and documentation
2. Implementation of Lomb Scargle Power Spectrum along with helper functions, statistical functions, user facing wrapper class, tests and documentation

Description and Timeline

Period	Description
May 4th - 28th	Getting to know mentors, reading documentation, setup development environment, contributing fixes to smaller issues to get familiar with the codebase and getting up to speed with working on the project.
May 29th - June 4th	Implementing the Lomb Scargle Crossspectrum
June 5th - June 19th	Implementing the Lomb Scargle Crossspectrum class

Period	Description
June 19th - July 2nd	Implementing Lomb Scargle Power Spectrum class
July 3rd - July 16th	Implementing the helper functions
July 14th	Midterm evaluation deadline
July 31st - August 13th	JAX optimization, tests and documentation : tests for the wrapper classes as well as the core implementation , documentation for the API and how to use the class
August 13th - August 21st	Buffer period and improving code quality
August 21st - August 28th	Final evaluation week
August 28th - September 4th	Mentor final evaluation week
September 5th - September 24th	Buffer period, comprehensive tests and documentation

GSoC

I have not participated previously in GSoC. I am not applying to any other projects as this project has piqued my interest the most and aligns the most with my knowledge and skill set.

Schedule availability

I don't really have any plans for holidays during the time of the program. There are barely any holidays between my 4th and 5th semesters which would be around 10-15 days in late August early September. I would be willing to work during these days as well if need be. I would have slightly reduced output from 13th July to Mid/Late August due to semester end exams. My window of maximum productivity would be from May 29th ~ June 25th. I would not like to put behind any work for the buffer period as it clashes with my semester end exams. I would most likely be to work at full productivity from roughly 11th August onward assuming exam schedule would be similar to current semester. Likewise, I would try and cover the work of later weeks early if any unforeseen circumstances arise.