Optical Spectra Interpolation and Inference

Yuping Zheng, Henry Schmitz, Nicole Sullivan, Juan Velasco, Angelina Gallego

April 2022

Abstract

The objective of the project "Optical Spectra and Interpolation" is to reverse engineer the primary information of a binary neutron star merger based on the provided spectra. This primary information consists of mass, flux, and star type. We interpolated over the simulated data from a 4-dimensional grid to attempt to infer the four parameters, namely mass of Lanthanide-free ejecta (m_{LF}), mass of Lanthanide-rich ejecta (m_{LR}), half-opening angle (Φ_{ej}) and lastly viewing angle (Θ_{obs}). The main technique we use is Gaussian Process Regression (GPR) and potentially deep learning techniques.

Key Words: Electromagnetic Transmission, Gaussian Process, Light Curve, Neutron Star Mergers, Nuclear Multi Messenger Astronomy Framework, Spectra, Wavelength Dependent Opacity

1 Introduction

The detection of the binary neutron star (NS) merger GW170817 (Figure 1) and the observation of the electromagnetic counterparts known as a *kilonova* (KN) opens up a new era of Multi-Messenger Astronomy (MMA) [3]. When using MMA we aim to "... track high-energy particles pelting Earth, particles passing through space and ripples in the fabric of space-time to answer questions about some of the most extreme events in the universe."(news.wisc.edu) In this case these extreme events are our KN. KN are of great value by providing insight into the properties of NSs and the evolution of the universe. A large amount of theoretical work on modeling KN considers two forms of ejecta, including their properties like mass, velocity, and composition to further link to the NS properties. These ejecta are dynamical and wind ejecta when working with NS mergers. Wind ejecta is the results of remnant accretion disk winds which can be driven by neutrino energy and dynamical ejecta results from the "tidal stripping of the NSs and from NS-NS contact interface." [3]

In this project, we are particularly interested in a high-dimensional models of KN for predicting spectra. Such a high-dimensional model uses Spectral Energy Distributions (SEDs) simulated using the time-dependent multi-dimensional Monte Carlo radiative transfer code POSSIS (Polarization Spectral Sythesis In Supernovae) [1]. This particular model is controlled by four parameters: mass of the dynamical ejecta (m_{ej}^{dyn}), mass of the post-merger disk-wind ejecta (m_{ej}^{pm}), half-opening angle of the lanthanide-rich dynamical-ejecta component (Φ_{ej}), and the observer viewing angle (Θ_{obs}). See [3] for the details of this model.

Scientists aim to estimate these four parameters via the following two steps: 1) interpolating the four-dimensional parameter grid using a Gaussian process, and 2) inferring the parameters through Bayesian analysis. For the interpolation step, we can also potentially use machine learning technique to get the likelihoods that support Bayesian learning.

2 Methodology

After setting up a database (see Appendix), our first though was to interpolate the parameter space and then infer the best fit parameters for *all* days and wavelengths. We started the analysis with a simple case, only to focus on the model on day 2.5 and wavelength 5100 Å. For the interpolation task, we utilized the Scikit-learn package for the Gaussian Process Regression (GPR) and performed the Bayesian inference through the EMCEE package. Note here we decided not to spend a lot of time tuning the prior of MCMC so the inferred parameter is without guaranteed correctness. After achieving one-day-one-wavelength interpolation-inference, we expanded the similar process to multiple wavelength independently but still on day 2.5. The interpolation took much longer than the single wavelength case, so we decided to use SVD to decrease the model dimension and also attempted a deep learning technique to achieve the speedup of the interpolation. While we are skeptical about its correctness, the deep learning technique is indeed much faster than the GPR method. Some efforts were spent on customizing the model class using the George package for GPR, but we stuck with Scikit-learn in the end.

The next process in this project was to interpolate spectra over multiple days and wavelengths and eventually infer the event's parameters. We started by using a table of this information and restricting it to each day of observed spectra. At first, we considered doing this process with files for each day, but ended up just using SQL queries. For the singular value decomposition (SVD) model, we used a fifth of the wavelengths over the full range within the simulations. We also focused on days $\{1.5, 2.5, \dots, 10.5\}$ since these corresponded to the days of the observed spectra from the event. For the SVD model, there were two main parts to set up all of the matrices. In part 1, we had to get all parameter combinations and put them into a matrix, then normalize these columns from 0 to 1. For part 2, we gathered brightness matrices for each of the ten days; each column corresponded to one wavelength and each row corresponded to a unique combination of the four parameters. Each brightness column was normalized from 0 to 1, and all necessary information was stored in the SVD model's dictionary for the corresponding day. With our data fully processed and properly stored, we found the SVD of each day's brightness matrix of simulated spectra. Most of the ideas from this point on are taken from homework seven, where we performed Principal Component Analysis (PCA) on the spectra of the same event. This reduced the dimensionality of our data while hopefully retaining its variation. The number of components we chose for PCA was 3, which was the same as the homework. Once we calculated the SVD for each day, we stored the number of components, full principal components decomposition of the brightness matrix along with its standard deviation, and the right singular vectors in the SVD model. Now, we just had to create and fit Gaussian processes for each day's component in the SVD model. To do this, we used the Rational Quadratic Kernel from Scikit-learn. With all of the GPRs created, we could finally begin interpolating the predicted spectra for each day. To create the predicted data, we supplied the GPR models with values for each parameter and interpolated over the wavelengths measured in the observed data using scipy's interpolate method. To standardize the spectra between the predictions and observations for comparison during MCMC, we divided all brightnesses by the maximum brightness over all days.

3 Results

Applying the first part of our methodology, we were able to interpolate and infer parameters for a particular day and wavelength, however the corresponding prediction was not very accurate, especially in the deep learning model.

As for the result of the 30 GPRs, we generated reasonable-looking 10-day spectra given any set of the four parameters within the simulated parameter space. Using the GPRs from the SVD model, we attempted to run MCMC against the observed spectra to find the best parameters given our model, except it would take around 13 hours to finish running 1500 iterations, so either we needed to filter the wavelengths from the observed data, modify the code for getting the interpolated data to run faster, or some combination of these for a reasonable run time. We successfully interpolated spectra given values for m_{LF}, m_{LR}, Φ_{ej}, and Θ_{obs}, which can be seen in Figures 4a and 4b. However, we fell short at inferring these values for the observed data due to lengthy run time of MCMC. We are confident that if we had a more powerful computer, less data, or maybe a more intricate machine learning model we may have been able to get the inference part working but we are satisfied with our interpolation.

With respect our application of PCA, it is evident from Figure 5 that we need to keep more coefficients (rather than 3) to improve our approximation of spectral curves. In order to not increase computation time considerably, we had to do a trade off, decreasing the number of GPs per day. So instead of having one GP per coefficient, we have one GP per 20 coefficients. This set of 10 GPs was tested on the grid; it fits the modeled data, as expected. Next, we tested it outside the grid to see if the GP actually worked, however we didn't have a reference other than the target, so although the GP can actually compute values off the grid, our guesses sadly weren't predictions of the merger in question.

4 Discussion and Conclusion

We successfully interpolated the discrete parameter space using Gaussian process regression and the inference on a day and a wavelength. For the simple one-day-one-wavelength model, we simply used the complete feature matrix, which is the combination of all the possible four parameters. As for the complicated multiple-day-multiple-wavelength model, we used PCA via the SVD to decrease the size of the matrix to interpolate the parameter space within a reasonable amount of time. However, due to the time limitation, we don't have a satisfied parameter prediction, even for the simple model of a day and a wavelength. And for PCA method described above, there should be at least 20 coefficients preserved, otherwise it might degrade the performance of the model. If we include more days and wavelength, the computational cost increases, such that we were not able to perform inference over the multiple days and wavelengths. To be able to finish this time-consuming inference, there are two tracks we can dive into: 1) optimize the algorithms, and 2) utilize a parallelism technique, either using multiple CPU cores or the GPU to accelerate the process.

5 Appendix

Data pre-processing

The data for this research fell into two categories: 1) real data (from the GW170817 KN) and 2) theoretical data, generated from simulations of different combinations of the four relevant parameters (dynamical ejecta, wind ejecta, viewing angle, half opening angle). The real data were already provided for us in our repo (Figure 2), as one file per day (10 files altogether). These files would be pre-processed in a similar fashion to the second and third steps of pre-processing used for the theoretical data (described next). Obtaining the theoretical data was a bit more involved, for two reasons: (A) the data were not already contained in our repo, so that data first needed to be extracted/produced from another repo and (B) there was a much larger volume of theoretical data than real data (as it could be produced for an infinite number of combinations of parameters), so Dr. Mattia Bulla's Python script [2] output each simulation as a separate file. Pre-processing the theoretical data was broken down into 3 steps: (1) save the data locally as separate files, (2) reshape and combine the separate files and their parameter values into one massive file, and finally (3) push that file to a cloud database, Google Cloud Platform (GCP).

To (1) extract the theoretical spectra, a Python script written by Bulla was edited to generate spectra files and save them locally (see `https://github.umn.edu/umn-csci-8581-S22/repo-InterpInfer/blob/main/make_spectra.py` or the edits made to Bulla's file; note that files are stored in a dictionary and written to a csv, which was different than Bulla's lightcurve routine). At this point, each file contained 500 wavelengths x 100 timepoints and the combination of parameter values used to produce that spectra was a part of the file's name. Therefore, in the next step (2), the parameter values for that simulation needed to be pulled from the title of the file and saved as their own columns, and the file itself needed to be pivoted from wide to long in order to create a single flat table in the format the pandas and Scikit-learn libraries would expect (ie to allow the team to focus on interpolation and inference in the following sections rather than data wrangling). Lastly, (3) the flat file would need to be stored somewhere centrally, as it was quite large (17 GB), and it would be unwieldy to share via email, slack or other similar methods. Having it stored somewhere centrally would also ensure that there was one single source of truth for the project, and that team members weren't accessing different versions of the data created, processed or sent at different times (which can often happen even with a project where only one person is working on things!). Google Cloud Platform (GCP) was selected as the central cloud database host for this project, as it has a free sandbox space that includes up to 1TB of computing (querying) free each month, and it has a well-developed Python package for calling the BigQuery API, as well as an intuitive GUI (if the google-cloud-bigquery package proved a pain point). The routine to perform steps (2) & (3) was parallelized (see lines 90-157 of `https://github.umn.edu/umn-csci-8581-S22/repo-InterpInfer/blob/main/process_spectra.R`)and run over 9 cores (local laptop). Note that prior to step (3), an intermediate step was run to create the requisite datasets in GCP, in which tables could then be created (for the intermediate step, see section B in `https://github.umn.edu/umn-csci-8581-S22/repo-InterpInfer/blob/main/Presentation_One.ipynb`)That's because GCP's structure is: project > datasets> tables, so a dataset must exist as a container for tables. See a screenshot of the team's structure below, where project = interp-infer > datasets = lightcurves, spectra > tables = bolometric, photometric, GW170817, theoretical, $theoretica_(va)$.

To store the data in GCP, an Owner-level JSON key was generated that authorized the user to write to and pull from the interp-infer GCP project. That JSON key was then shared with the team so members could query data directly into their Python scripts or Jupyter notebooks. Alternatively, team members could also simply write a SQL query in the console (pictured above) and click “Save results” to download the query as a csv locally. As long as the SQL for the query was saved, that same data pull could be reproduced by anyone else with access to the GCP project.

Figures

Figure 1: Image of the neutron star merger that we were doing the interpolation and inference of.

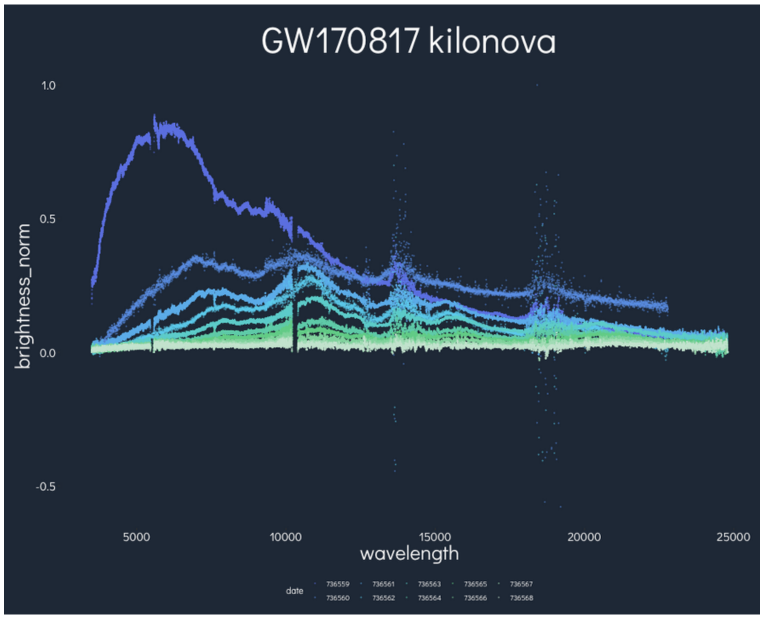


Figure 2: 10-day spectra of the GW170817 NS merger.

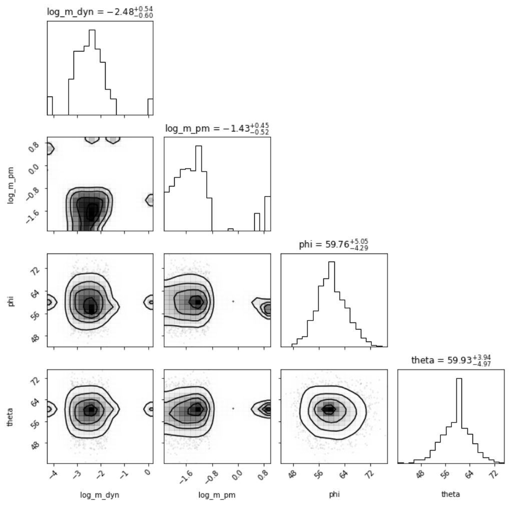
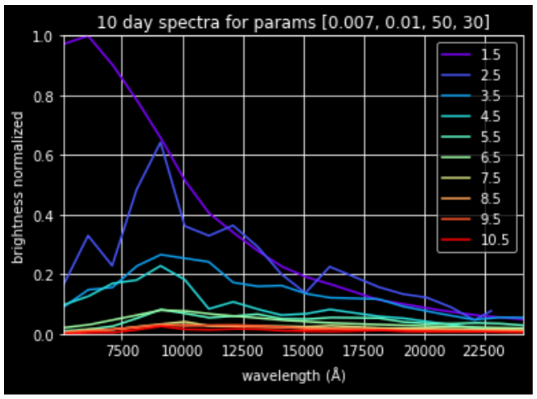
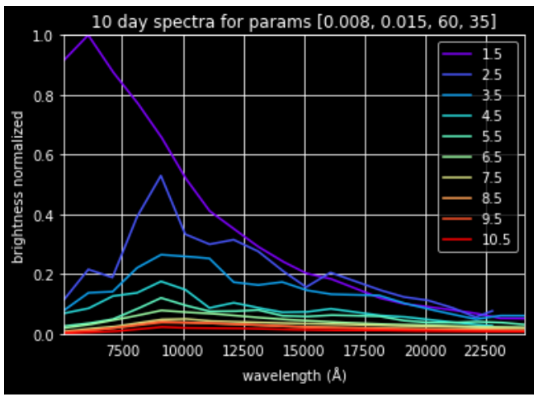


Figure 3: **Bayesian Inference Result.** The inference is conducted on the model of Day 2.5 at wavelength Å. For the sake of shorter runtime, we constrained the prior fairly tight so MCMC would converge quickly. The other inference result from the multi-wavelength model using GPR and also tensorflow is similar so we only present this one here.



(a) Interpolation using parameters $\{0.007, 0.01, 50, 30\}$



(b) Interpolation using parameters $\{0.008, 0.015, 60, 35\}$

Figure 4: Some examples of interpolated spectra from the multiple-day-multiple-wavelength SVD model.

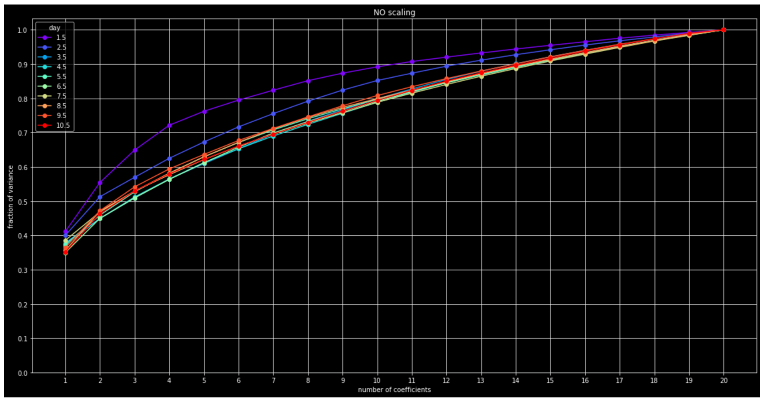


Figure 5: Variance vs. Coefficient graph shows we need to improve our approximation of spectral curves.

References

[1] M. Bulla. possis: predicting spectra, light curves, and polarization for multidimensional models of supernovae and kilonovae. *Monthly Notices of the Royal Astronomical Society*, 489(4):5037–5045, sep 2019.

[2] M. Bulla. kilonova_models. `https://github.com/mbulla/kilonova_models.git`, 2022.

[3] M. W. Coughlin, T. Dietrich, Z. Doctor, D. Kasen, S. Coughlin, A. Jerkstrand, G. Leloudas, O. McBrien, B. D. Metzger, R. O'Shaughnessy, and S. J. Smartt. Constraints on the neutron star equation of state from AT2017gfo using radiative transfer simulations. *Monthly Notices of the Royal Astronomical Society*, 480(3):3871–3878, aug 2018.