CA2_MiniProject

August 23, 2023

Data Analysis 2023 - Coursework 2 (50%)

1 Analysing gravitational wave signals

1.1 Deadline Thursday week 11, 2pm.

Instructions This coursework assesses learning outcomes from **all Chapters** of the course, but in particular **Chapters 6, 7, 8 and 9**. It is worth 50% of the module. Please ensure you have read the Chapter 8 Jupyter Notebook before starting this coursework.

These assessments are equivalent to an exam: - Submit your work via Turn-It-In on Learning Central. Note that you will need to upload your final notebook exported as a html file. **Don't forget to click `run all` before you export it**. You can constantly update this document until the deadline. - The breakdown of the assessment criteria is provided in Learning Central under Assessment. - Don't worry about how your code looks - marks are not given for pretty code, but rather for the approach used in solving the problem, your reasoning, explanation and answer. - Please also take note of the University's policy on **plagiarism**, which is outlined in your student handbook.

Tips

- Explain all your reasoning for each step. A *significant fraction* of the marks are given for explanations and discussion, as they evidence understanding of the analysis.

- Some of these steps will take a while to run and compile. It's a good idea to add in print statements to your code throughout eg `print('this step is done')` to make sure that your bit of code has finished.
- Add the import packages statements at the top of your Jupyter notebook. We will use the `pandas` package to read in the data, with eg `dataIn=pd.read_csv('filename.csv')`.
- You may need to do some additional research into this subject. You may find it useful to look at the following publication from the LIGO consortium. https://arxiv.org/pdf/1608.01940.pdf

---

Gravitational waves are disturbances in the curvature of spacetime, generated by accelerated masses, that propagate as waves outward from their source at the speed of light. They are predicted in General Relativity and other theories of gravity and since 2017, they have now been observed!

In this exercise we will analyse some mock gravitational wave data from two unknown astrophysical objects merging together and coelescing. We will use a Monte Carlo Markov Chain (MCMC)

to compare a scaled model that predicts how the wave changes depending on the total mass of the merging objects and their distance from us to the observed waveform. This will allow us to determine the nature of the orbiting objects that merged to form the gravitational wave using MCMC, whether for instance they could be originating from merging white dwarfs, neutron stars or black holes.

The mock or simulated waveforms measure the strain as two compact, dense astrophysical objects coalesce. The strain describes the amplitude of the wave. The system is parameterised by the masses of the merging objects, M_1 and M_2, and their distance from the observer D.

Other useful parameters and equations relevant for this assessment are given in the Chapter 8 Jupyter Notebook.

---

1. How do astronomers know that the gravitational waves from the GW150914 event were due to two black holes merging?

   [2 marks]

2. Describe the different parts of the waveform produced due to gravitational waves from a merging black hole event.

   [3 marks]

3. The file `gravitationalwaveevents.csv` contains the properties of previously observed gravitational waves. Ultimately, we will aim to compare your merger event results with the properties from previously observed gravitational waves. Parameters obtained for these systems are known as posteriors as they are derived using Bayes Theorem (ie posterior = likelihood $\times$ prior). The errors in these values are the credible intervals from the posterior distribution derived for each parameter. Plot the **total** mass (given in solar masses, $1\mathrm{M}_\odot = 1.99 \times 10^{30}$ kg) of the merging sources against their distance (given in units of Mpc, $1\,\mathrm{pc} = 3 \times 10^{16}$m).

   [10 marks]

Answer:

Your answer here

1.2 Part B - The data

Now it is your turn to look at observations and measure the mass and distance of the merging system. We first need to access the observational data measured with the gravitational wave detectors (the waveform observed when two compact, dense astrophysical objects coalesce), and format it correctly.

1. Read in the datafile of the observed waveform `Observedwaveform.csv`. These files store the strain as a function of "GPS time" for the merger of two bodies.

   [5 marks]

2. The GPS time of the merger for your waveform is 1205951542.153363. Your data will need to be shifted so that the merger occurs at time = 0 secs. This is required for when we compare

model waveforms with our data as the model waveforms are simulated with the merger at t=0s.

[4 marks]

3. We need to estimate the average noise and its standard deviation in our data. This requires careful thought about where the noise can be seen in the waveform.

   [6 marks]

Answer:

Your answer here

1.3 Part C - Interpolate reference model to match the observed data time sampling

In this part of the question we will learn to match the time samples between a reference waveform (eg one generated by a model) and a mock data waveform, so they can be directly compared. We need to do this so that in Part D onwards we can compare our observations (Observedwaveform.csv) to our expectations (a reference waveform with different values of M and D).

The reference waveform we will use assumes $M = 40M_{sun}$, $D = 1$Mpc and $q = M_2/M_1 = 1$ and is named `reference_Mtot40Msun_Dist1Mpc.csv`.

You will also find some "mock data" for the same mass and distance in the file `mockData_Mtot40Msun_Dist1Mpc.csv` (which has $t = 0$ at the merger, just like the reference waveform).

Now that we have some mock data and a reference waveform, we need to do one more fix. Currently the data waveforms and our reference waveforms have different sampling on the x axis - ie they have different values of x (time). The reference waveforms have approx 20,000+ time steps, yet our data has less than hundreds of data points in the same time range! We need to try and match the x times up so that for each value of x we can compare the y values from our observations (the observed strain) with the y values from the reference waveform.

We need to only consider the times when we have observed data, so we will trim our other datasets.

1. Open the mock data file using the `pandas` package. Our data waveform starts at some time $t_{\rm min}$. Find out what this is. Next, take your observed data waveform and output data for $t > t_{\rm min}$ and $t < 0$ (ie only keep information for times ≤ 0 (before the merger), or for times where there is data). Verify, by plotting, that your new observed waveform only has data in this restricted time range.

   [5 marks]

2. Open the reference file using the `pandas` package. We want to convert our reference waveform to have the same time sampling, ie the same number of x data points as our data (in this question, our mock data). We need to interpolate the reference waveform to match the time samples of the data. To do this use the following code:

   assuming `ref_x[index]` and `ref_y[index]` are the reference data (time and strain respectively) and `data_x` is the observed data you wish to match the x axis for (this would be

mock_x for this question):

```
from scipy.interpolate import interp1d

# get interpolation object using a reference waveform with ref_x (time) and ref_y (strain).
interp_fn = interp1d(ref_x[index],ref_y[index],bounds_error=False)

# now interpolate the data waveform
interp_strain = interp_fn(data_x)

# plot
plt.plot(data_x,interp_strain)
```

Briefly verify that this works.

[10 marks]

Hints:

- *One can use the following code example to pull out bits of data* `index = np.where((data > 5)&(data < 10))[0]`. *This type of statement returns a list of indices (*`index`*) where the conditions in the bracket have been met.* `data[index]` *pulls out* `data` *that satisfy the conditions in the brackets above.*

Answer:

Your answer here

1.4 Part D - Using model waveforms to estimate the total mass and distance to the system "a by-eye estimate")

In this part of the question we will attempt to produce a waveform for any mass and distance values using the reference waveform with $M = 40M_{sun}$, $D = 1$Mpc and $q = M_2/M_1 = 1$ and scaling it by any new mass and distance.

The reference waveform/template we will use is the same as before: `reference_Mtot40Msun_Dist1Mpc.csv`.

You will need to follow the steps below when answering this question:

1. Write a function in python to produce the time t and strain h of a general waveform with $q = 1$, total mass M and distance D from the interpolation object you created above, using the equations for how the waveform strain and time depends on mass and distance from the Chapter 8 Jupyter Notebook.

   [10 marks]

2. Test your function works by substituting in values of $M = 70\,M_{sun}$ and $D = 5$Mpc, and compare your resulting waveform with the mock data in `mockData_Mtot70Msun_Dist5Mpc.csv`. Comment on your result.

   [6 marks]

3. Use your function to scale the reference waveform ($M = 40M_{sun}$, $D = 1$Mpc) to make an initial rough estimate "by eye" of the total mass and distance that "best" fits your data (e.g. to within +/- 5 Msun, +/- 100 Mpc).

   [8 marks]

Hints:

- As you are creating a function that returns h and t for the reference waveform scaled by M and D as per the Chapter 8 equations, you will need to do the interpolation inside the function.

Answer

Your answer here

1.5 Part E - Estimating the total mass using MCMC

Now that we know how to make the scaled reference (ie 40Msun,1Mpc template file) and the observed data have the same time sampling, we can use MCMC to find out the mass and distance of the system that merged together to create the waveform (the data) we see. You have two options in this question, each option is worth a different amount of marks.

1. Choose **one** out of the following options and create an MCMC to sample the parameter(s).

   Option 1. Use MCMC to sample the **total mass** and find the "best value".

   [20 marks]

   OR

   Option 2. Use MCMC to sample the **total mass and distance** to find the "best values".

   [40 marks]

2. Display the results in an appropriate manner and comment on your findings, as well as your results from the MCMC. Has your MCMC converged?

   [20 marks]

3. Report the median and 90% credible limits on your value of M and comment on your values. Compare the waveform generated from your MCMC result with the observed waveform.

   [15 marks]

You may assume that: - the noise is described by a Gaussian distribution, - the total mass of the system is in the range [20,100] M_{sun}. - Think carefully about what the likelihood function will be in this case (see Chapters 6-9) since we are trying to find out how good our model is matching the data.

Hints:

- *You should work with "log(Likelihood)" to avoid numerical errors - note this will affect both your posterior and the step in the MCMC chain where we usually write* $p_{\text{proposed}}/p_{\text{current}}$
- *The step size between samples of the MCMC is quite important. A suggested value is* $0.1\,M_{sun}$

- *The initial guess of your mass is also very important. You may find yourself getting into a local minimum rather than your code finding the true minimum. You could always start close to the estimate from Part D3.*
- *Test your MCMC on a small number of samples (e.g. 10-100) before trying it with a larger number (e.g.* 10^5 *or* 10^6*)*
- *At the end, ask yourself if you need to include every sample?*
- *Depending on your step size, this part can take a long time to run. Suggest that you move all your plotting routines to a different code cell to save you re-running everything 10000s of times when you just want to change a plot command.*
- *To find out how long it will take for a Jupyter notebook to compile the MCMC code cell, add the following snippet to your code before you go into your MCMC loop (where Nsteps is the number of steps your MCMC is using):*

```
def time_spent_waiting(n):     from datetime import datetime,timedelta
preddur=[n*0.01,n*0.02]     print('predicted duration: {:.2f}-{:.2f}
mins'.format(preddur[0]/60.,preddur[1]/60.))     return
```

Answer:

Your answer here

1.6 Part F - Putting it all together

If you are unable to get the MCMC in Part E working, please use your mass and distance estimate from the by-eye fit in part D3.

If your MCMC is working, please use your mass from the MCMC and by eye estimate of distance from Part D3 (option 1) **or** your mass and distance estimate from the MCMC (option 2).

1. Estimate the chirp mass for your system and the individual masses of your merging bodies, describing your reasoning. Comment on your individual masses.

   [**5 marks**]

2. Plot your MCMC derived properties alongside the previously discovered gravitational wave systems.

   [**5 marks**]

3. Estimate the period from your observed waveform around the peak amplitude of the wave.

   [**12 marks**]

4. Assuming that the objects are not spinning, and that their orbits are Keplerian and essentially circular, use your period to estimate the orbital separation in **km** of the two bodies around peak amplitude. Think carefully about how the orbital period is related to the period of your gravitational wave.

   [**10 marks**]

5. Comment on what your analysis suggests are the best astrophysical candidates for the merging objects?

   [4 marks]

Answer:

Your answer here

1.7 Part G - Understanding, Presentation and Interpretation

An additional 20 marks will be awarded for evidence of understanding and knowledge via (for example) explanations, plots, comments on your results and well formatted and well explained results. Marks are also available for additional investigations carried out on your analysis above. These marks are available for those data analysis reports that show evidence of work that is *very high quality* or *outstanding* as per the decile descriptions in the Assessment Criteria for modules.

[20 marks]