Astrophysical Insights through Gravitational Wave Data Analysis: Data Preprocessing

Shufan Dong[1]

[1]Class of 2026, Bronx High School of Science, NY, USA

Abstract

This paper presents a distinct approach to the analysis of gravitational wave (GW) data, integrating astrophysical theories and computational programming. The study focuses on the preprocessing, noise filtering, and visualization of GW signals, introducing advanced data analysis methods to extract meaningful information and features from the raw GW data. The methodology involves downloading GW data from the GW Open Science Center and processing GW data, handling missing values, applying noise filtration, normalizing data, and employing various plotting techniques to inspect the GW data. Although machine learning (ML) is not applied in this paper, the data preprocessing methods discussed are crucial for future ML applications in GW astronomy.

1 Introduction

GW astronomy has revolutionized our understanding of the universe, offering insights into cosmic events such as black hole mergers and neutron star collisions. The implementation of ML techniques has further enhanced the capability to analyze and interpret GW data. This paper integrates astrophysical data analysis with data analysis methodologies to preprocess, filter, and visualize GW signals, preparing the data for future ML applications.

2 Environment Setup

2.1 Import Libraries

We import essential libraries that are essential for data handling and visualization:

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import requests, os
from scipy.signal import butter, filtfilt, spectrogram
from sklearn.preprocessing import StandardScaler

import warnings
warnings.filterwarnings('ignore')
```

Figure 1: General libraries imported.

3 Data Acquisition and Setup

3.1 Setting GPS Time and Detector

For this study, we focus on a specific GW event (GW150914, the first confirmed observation of GWs from colliding black holes).

```
# Set GPS time:
t_start = 1126259462.4
t_end = 1126259462.4 # For specific events, make t_end the same as t_start

# Choose detector (H1, L1, or V1)
detector = 'H1'
```

Figure 2: Locating GPS time for Binary Black Holes merger (BBH) event GW150914 and choosing the Hanford (H1) detector.

3.2 Importing TimeSeries Package

We ensure that we can successfully import TimeSeries from gwpy by installing the other required packages necessary for this installation.

```
try:
    from gwpy.timeseries import TimeSeries
except:
    ! pip install -q "gwpy==3.0.8"
    ! pip install -q "matplotlib==3.9.0"
    ! pip install -q "astropy==6.1.0"
    from gwpy.timeseries import TimeSeries
```

Figure 3: Importing TimeSeries from gwpy.

3.3 Downloading and Reading Data

The GW data is downloaded and read into a TimeSeries object.

```
from gwosc.locate import get_urls
url = get_urls(detector, t_start, t_end)[-1]

# If an event is chosen, then its info will be shown in url
print('Downloading: ' , url)
fn = os.path.basename(url)
with open(fn,'wb') as strainfile:
    straindata = requests.get(url)
    strainfile.write(straindata.content)
```

Figure 4: Downloading and reading the GW data with the TimeSeries package imported in the last subsection.

4 Data Extraction and Handling Missing Values

4.1 Extracting Data

The timestamps and strain values are extracted and stored in a pandas DataFrame.

```
# Extract time and strain vals
timestamps = strain.times.value
strain_values = strain.value

# Store data in pd df
data = pd.DataFrame({
    'time': timestamps,
    'strain': strain_values
})
```

Figure 5: Extracting the time and strain features from the raw GW data file.

4.2 Handling Missing Values

Any missing values in the dataset are dropped to ensure clean data.

```
# Drop rows with missing vals
data = data.dropna()

print("\nMissing vals after cleaning:")
print(data.isnull().sum())
```

Figure 6: Dropping any NaN values from the dataset.

5 Data Noise Filtering and Normalization

5.1 Band-Pass Filtering

Noise filtering is crucial in GW data analysis due to the presence of various noise sources that can distract us from the true signal. One common method is band-pass filtering, which allows signals within a specific frequency range to pass through while reducing the significance of signals outside this range.

> Purpose: The goal of band-pass filtering is to isolate the frequency range where GW signals are expected to be prominent, thus reducing the impact of noise outside the frequency ranges.

Application: the low cutoff frequency (20 Hz) and high cutoff frequency (500 Hz) are chosen based on the expected characteristics of a BBH event.
Importance: Applying a band-pass filter helps in enhancing the signal-to-noise ratio (SNR) of the GW data, increasing the exposure of the actual signal.

```python
# Band-pass filter function
def butter_bandpass(lowcut, highcut, fs, order=5):
    nyq = 0.5 * fs
    low = lowcut / nyq
    high = highcut / nyq
    b, a = butter(order, [low, high], btype='band')
    return b, a

def bandpass_filter(data, lowcut, highcut, fs, order=5):
    b, a = butter_bandpass(lowcut, highcut, fs, order=order)
    y = filtfilt(b, a, data)
    return y

# Filter params
lowcut = 20  # Low cutoff frequency (Hz)
highcut = 500  # High cutoff frequency (Hz)

# Band-pass filter strain data
data['strain'] = bandpass_filter(data['strain'], lowcut, highcut, 4096)
```

Figure 7: butter_bandpass function designs a band-pass filter with specified low and high cutoff frequencies, while bandpass_filter function applies the designed filter to the GW data, removing noise outside the specified frequency range.

5.2 Data Normalization

Normalization is another crucial preprocessing step that adjusts the GW data to a common scale, making it easier to analyze and compare.

Purpose: Normalization ensures that the strain data have a mean of zero and a standard deviation of one. This is particularly important for the future application of ML algorithms that are sensitive to the scale of the data.
Importance: Standardizing the strain data is essential for ensuring that all features contribute equally to the analysis and for improving the performance of ML models.

```
# Normalize strain data
scaler = StandardScaler()
data['strain'] = scaler.fit_transform(data[['strain']])
```

Figure 8: StandardScaler function standardizes the features so that they're easier for ML algorithms to analyze.

6 Data Inspection

6.1 Initial Data Inspection

We briefly look at the data after it's being preprocessed.

```
First few rows of data:
           time    strain
0  1.126257e+09 -2.509170
1  1.126257e+09  0.070279
2  1.126257e+09  2.209691
3  1.126257e+09  3.618610
4  1.126257e+09  4.256309

Col headers:
Index(['time', 'strain'], dtype='object')

Summary stats:
               time        strain
count  1.677722e+07  1.677722e+07
mean   1.126259e+09 -1.758737e-17
std    1.182413e+03  1.000000e+00
min    1.126257e+09 -3.686864e+00
25%    1.126258e+09 -7.088868e-01
50%    1.126259e+09  1.167451e-03
75%    1.126260e+09  7.087773e-01
max    1.126262e+09  4.284804e+00

Missing vals in each col:
time      0
strain    0
dtype: int64
Sampling frequency: 4096.0 Hz Hz
```

Figure 9: Characteristics and features of the preprocessed GW data.

7 Data Visualization

Visualization is a key part of data analysis, providing intuitive insights into the structure and characteristics of the GW data. Below, we explain the purpose and significance of each plot used in this section.

7.1 Time Series Plot

We visualize how the strain data changes over time.

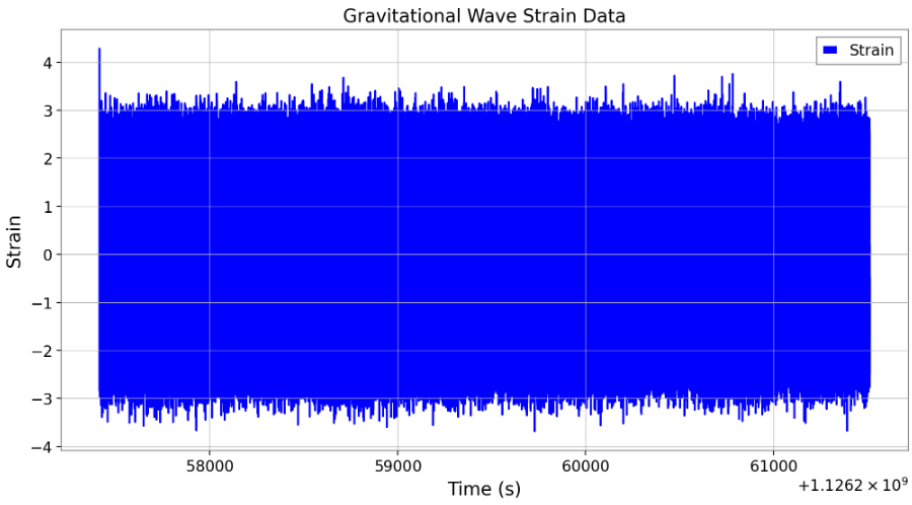

Figure 10: Graph of time-series plot (strain data versus time).

In the plot, peaks and troughs may correspond to significant events such as black hole mergers or neutron star collisions, and it is useful for initial data inspection, allowing us to identify the presence of potential GW events.

7.2 Spectrogram

We visualize how the frequency content of the strain data changes over time.

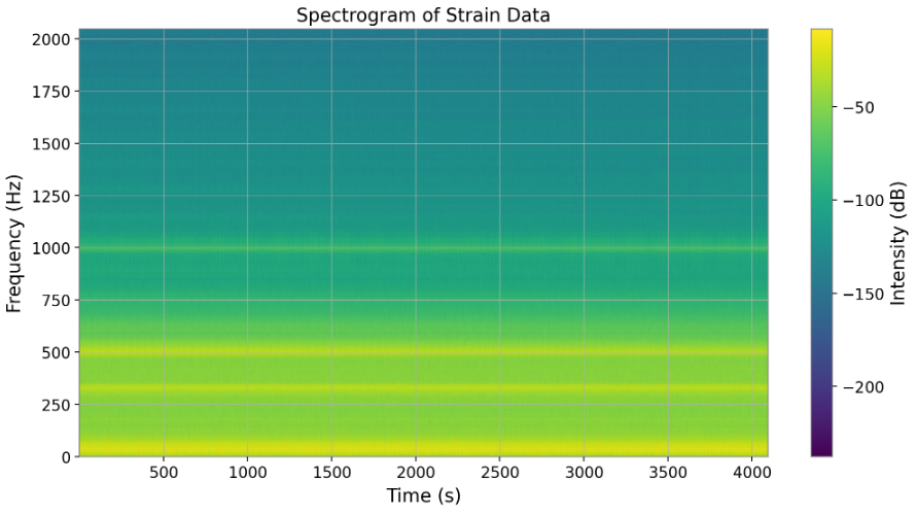

Figure 11: Graph of spectrograms (strain data's frequency versus time).

This plot helps identify transient events and their frequency components, which are crucial for distinguishing between noises and actual GW signals. Additionally, spectrograms provides a detailed view of how the signal's frequency content evolves, and spectrogram data can be used as 2D GW data for the implementation of certain ML models.

7.3 Histogram

We visualize the distribution of strain values.

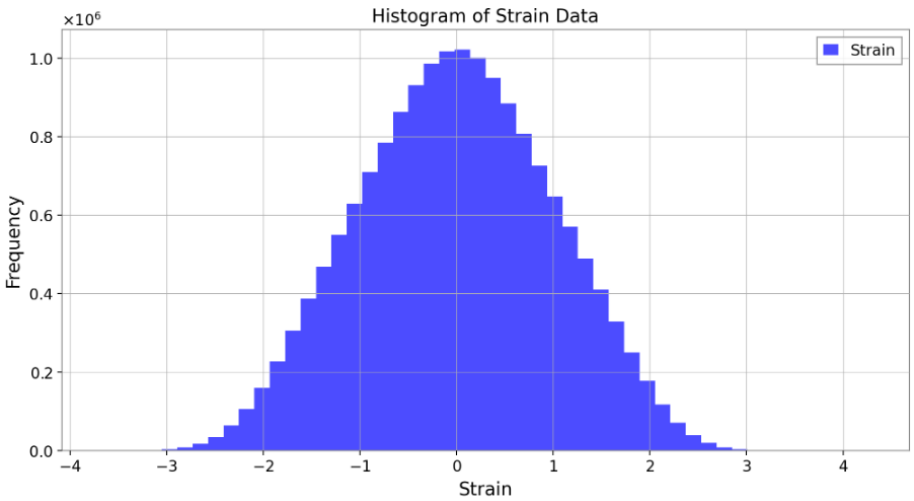

Figure 12: Graph of Histogram (frequency distribution of strain data).

This plot provides an overview of the data's spread, central tendency, and outliers. This is useful for identifying any anomalies or patterns in the data. Besides this, understanding the distribution of the strain values is crucial for subsequent statistical analysis and for ensuring that the GW data meets the expectations of various ML algorithms.

8 Conclusion

In this paper, we have demonstrated the integration of astrophysical data analysis with programming techniques to preprocess, filter, and visualize GW data. Band-pass filtering effectively reduces noise, enhancing the SNR. Normalization ensures that the data is on a standard scale, improving statistical analysis and future ML model performance. The visualizations (time-series plot, spectrogram, and histogram) provide critical insights into the GW data, enabling the plain identification of GW events and the assessment of data quality. Although ML is not applied in this paper, the preprocessing methods discussed are essential for preparing the data for future ML applications in GW analysis.

References

[1] Abbott, B. P., et al. "Observation of gravitational waves from a binary black hole merger." Physical Review Letters 116.6 (2016): 061102.

[2] Abbott, B. P., et al. "GW170817: Observation of gravitational waves from a binary neutron star inspiral." Physical Review Letters 119.16 (2017): 161101.

[3] Amaro-Seoane, P., et al. "Laser Interferometer Space Antenna." arXiv preprint arXiv:1702.00786 (2017).

[4] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo During the First Half of the Third Observing Run." arXiv preprint arXiv:2010.14527 (2020).

[5] Chatziioannou, K., et al. "The last gravitational wave in the window: An improved waveform model for binary black hole inspirals." Physical Review D 95.10 (2017): 104027.

[6] Martynov, D. V., et al. "Sensitivity of the Advanced LIGO detectors at the beginning of gravitational wave astronomy." Physical Review D 93.11 (2016): 112004.

[7] Fairhurst, S. "Source localization with an advanced gravitational wave detector network." Classical and Quantum Gravity 28.10 (2011): 105021.

[8] Finn, L. S., and Chernoff, D. F. "Observing binary inspiral in gravitational radiation: One interferometer." Physical Review D 47.6 (1993): 2198.

[9] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs." Physical Review X 9.3 (2019): 031040.

[10] Schutz, B. F. "Networks of gravitational wave detectors and three figures of merit." Classical and Quantum Gravity 28.12 (2011): 125023.

[11] Cutler, C., and Thorne, K. S. "An overview of gravitational-wave sources." General Relativity and Gravitation 39.5 (2002): 151-165.

[12] Singer, L. P., et al. "The first two years of electromagnetic follow-up with advanced LIGO and Virgo." The Astrophysical Journal 795.2 (2014): 105.

[13] Abbott, B. P., et al. "Properties of the binary black hole merger GW150914." Physical Review Letters 116.24 (2016): 241102.

[14] Abbott, B. P., et al. "Tests of general relativity with GW150914." Physical Review Letters 116.22 (2016): 221101.

[15] Abbott, B. P., et al. "Astrophysical implications of the binary black hole merger GW150914." The Astrophysical Journal Letters 818.2 (2016): L22.

[16] Abbott, B. P., et al. "Multi-messenger observations of a binary neutron star merger." The Astrophysical Journal Letters 848.2 (2017): L12.

[17] Acernese, F., et al. "Status of Virgo." Classical and Quantum Gravity 24.19 (2007): S381.

[18] Aasi, J., et al. "Advanced LIGO." Classical and Quantum Gravity 32.7 (2015): 074001.

[19] Acernese, F., et al. "Advanced Virgo: a second-generation interferometric gravitational wave detector." Classical and Quantum Gravity 32.2 (2015): 024001.

[20] Abbott, B. P., et al. "Improved analysis of GW150914 using a fully spin-precessing waveform model." Physical Review X 6.4 (2016): 041014.

[21] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Half of the Third Observing Run." arXiv preprint arXiv:2111.03606 (2021).

[22] Abbott, B. P., et al. "A gravitational-wave standard siren measurement of the Hubble constant." Nature 551.7678 (2019): 85-88.

[23] Abbott, B. P., et al. "Multi-messenger observations of a binary neutron star merger." The Astrophysical Journal Letters 848.2 (2017): L12.

[24] Abbott, B. P., et al. "Observation of gravitational waves from a binary black hole merger." Physical Review Letters 116.6 (2016): 061102.

[25] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-1: A gravitational-wave transient catalog of compact binary mergers observed by LIGO and Virgo during the first and second observing runs." Physical Review X 9.3 (2019): 031040.

[26] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Half of the Third Observing Run." arXiv preprint arXiv:2111.03606 (2021).

[27] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo During the First Half of the Third Observing Run." arXiv preprint arXiv:2010.14527 (2020).

[28] The LIGO Scientific Collaboration, the Virgo Collaboration, et al. "GWTC-3: Compact Binary Coalescences Observed by LIGO and Virgo During the Second Half of the Third Observing Run." arXiv preprint arXiv:2111.03606 (2021).