Gravitational Wave Event Detection: Numerical Features in Gravitational Wave Data

Shufan Dong[1]

[1]Class of 2026, Bronx High School of Science, NY, USA.
dongs1@bxscience.edu

Abstract

This paper presents methodologies for analyzing gravitational wave (GW) data, focusing on time-domain features, event detection, event parameter estimation, and basic statistical analysis. Detailed explanations and Python codes are provided for calculating time-domain features, detecting events, and estimating parameters, followed by summarizing event parameters. The results are contextualized within the framework of ongoing advancements and research in GW astronomy as observed by LIGO, Virgo, and KAGRA collaborations.

Contents

1 Introduction

GW astronomy has opened new platforms for understanding the universe, with instruments like LIGO, Virgo, and KAGRA detecting numerous events such as black holes and neutron star mergers. This research aims to provide tools for the detailed analysis of GW data by focusing on time-domain feature extraction, event detection, and parameter estimation. You can refer to this paper [40] to learn about the preprocessing of these GW data and the libraries imported. The methodologies are illustrated using Python codes, fostering an accessible approach to data analysis in this field, and while this paper does not introduce the use of machine learning (ML) for event detection, it will be applied later for more precise and accurate detection of events.

2 Time-Domain Features

The function `calc_and_print_time_domain_features` is designed to extract and print key time-domain features from GW data.

```
def calc_and_print_time_domain_features(data, strain_column, fs):
    peak_amplitude = np.max(data[strain_column])
    min_amplitude = np.min(data[strain_column])
    print(f"Peak Amplitude ({strain_column}): {peak_amplitude}")
    print(f"Min Amplitude ({strain_column}): {min_amplitude}")

    threshold = 0.5 * peak_amplitude
    significant_signal = data[strain_column].abs() > threshold
    signal_duration = significant_signal.sum() * (1/fs)
    print(f"Signal Duration ({strain_column}): {signal_duration}s")

    signal_power = np.mean(data[strain_column]**2)
    noise_power = np.mean(data[data[strain_column].abs() <= threshold][strain_column]**2)
    snr = 10 * np.log10(signal_power / noise_power)
    print(f"Signal-to-Noise Ratio (SNR) ({strain_column}): {snr} dB\n")

# Calc features for strain data
print("Calc features for strain data: ")
calc_and_print_time_domain_features(data, 'strain', fs)
```

Figure 1: The function accepts three parameters: data (a DataFrame containing the signal and time data), strain_column (the strain data column), and fs (the sampling frequency), and the function calculates the peak and minimum amplitudes of the specified strain column. For computation purposes, a threshold is set at 50% of the peak amplitude, and the duration of significant signals exceeding this threshold is calculated and printed. As a result, the function calculates and prints the signal power, noise power, and Signal-to-Noise Ratio (SNR).

Time-domain features in GW data are crucial because they provide direct insights into the dynamics of astrophysical sources and the propagation of GWs. By analyzing these features, we can extract critical information about the nature and behavior of compact objects, such as black holes and neutron stars, and the environments in which they reside.

For instance, the shape and structure of the GW signal in the time domain can reveal the mass, spin, and orbital dynamics of a binary merger event. Features such as chirps, where the frequency and amplitude of the wave increase as the objects spiral closer, are particularly informative. Also, detecting short-lived, transient signals helps identify specific events like black hole mergers and neutron star collisions, and each of them has a unique, discoverable signature in the time domain.

Time-domain analysis allows for the identification of noises, which is essential for improving the signal-to-noise ratio (SNR) and ensuring the accuracy of the detected signals.

```
Calc features for strain data:
Peak Amplitude (strain): 4.284804453733104
Min Amplitude (strain): -3.686864089465157
Signal Duration (strain): 92.680908203125s
Signal-to-Noise Ratio (SNR) (strain): 0.4926804961051281 dB
```

Figure 2: The output of the function prints the peak amplitude, minimum amplitude, signal duration, and SNR.

3 Basic Event Detection & Parameter Estimation

The `calc_threshold` function calculates a threshold for event detection based on the standard deviation of the noise in the strain data.

```python
def calc_threshold(data, strain_column, factor=3):
    noise_std = np.std(data[strain_column])
    threshold = factor * noise_std
    return threshold
```

Figure 3: This function calculates a threshold based on the standard deviation of the strain data. The threshold is set to a multiple of this standard deviation and returned. A threshold of approximately 3 is calculated.

The `detect_events` function identifies events in the strain data based on the calculated threshold.

```python
def detect_events(data, strain_column, threshold):
    events = []
    event_start = None

    for i, strain in enumerate(data[strain_column]):
        if abs(strain) > threshold:
            if event_start is None:
                event_start = i
        else:
            if event_start is not None:
                event_end = i
                events.append((event_start, event_end))
                event_start = None

    # Check if an event is ongoing at end of data
    if event_start is not None:
        events.append((event_start, len(data[strain_column]) - 1))
    return events
```

Figure 4: This function identifies events where the absolute strain exceeds the calculated threshold, and it iterates through the strain data, marking the start and end of events. In the end, detected events are stored as start and end indices in a list.

Event detection is the process of identifying important signals within the GW data that correspond to astrophysical phenomena. Rapid detection enables follow-up observations with electromagnetic (EM) and other observatories, providing critical support to multi-messenger astronomy.

The `estimate_event_params` function calculates parameters for each detected event.

```python
def estimate_event_params(data, strain_column, events, fs):
    time_column = 'time'

    event_params = []
    for event in events:
        start_idx, end_idx = event
        event_data = data[strain_column].iloc[start_idx:end_idx]
        peak_amplitude = np.max(np.abs(event_data))
        duration = (end_idx - start_idx) / fs
        event_params.append({
            'start_time': data[time_column].iloc[start_idx],
            'end_time': data[time_column].iloc[end_idx - 1],
            'peak_amplitude': peak_amplitude,
            'duration': duration
        })
    return event_params
```

Figure 5: This function calculates parameters such as start time (GPS time), end time (GPS time), peak amplitude, and duration for each detected event. For each event, the function extracts relevant data and calculates the required parameters, storing them in an array.

Parameter estimation conveys the importance of determining the physical parameters of the GW source, such as masses, spins, distances, and orbital characteristics. Accurate parameter estimation is vital for interpreting GW observations and understanding the underlying physics.

High-precision parameter estimation allows for stringent tests of general relativity and other gravitational theories. Detailed parameter estimation helps expound the population properties of compact objects, their formation channels, and their role in the cosmos.

```
Event Params:
{'start_time': 1126257415.0007324, 'end_time': 1126257415.001709, 'peak_amplitude': 4.284804453733104, 'duration': 0.001220703125}
{'start_time': 1126257418.3012695, 'end_time': 1126257418.3015137, 'peak_amplitude': 3.10656720831358, 'duration': 0.00048828125}
{'start_time': 1126257418.4667969, 'end_time': 1126257418.467041, 'peak_amplitude': 3.1396660570114685, 'duration': 0.00048828125}
{'start_time': 1126257423.3283691, 'end_time': 1126257423.3283691, 'peak_amplitude': 3.0493328722819033, 'duration': 0.000244140625}
{'start_time': 1126257423.4399414, 'end_time': 1126257423.4401855, 'peak_amplitude': 3.139458428439673, 'duration': 0.00048828125}
{'start_time': 1126257423.4418945, 'end_time': 1126257423.4421387, 'peak_amplitude': 3.1423561734113865, 'duration': 0.00048828125}
{'start_time': 1126257424.4831543, 'end_time': 1126257424.4833984, 'peak_amplitude': 3.0957766543715897, 'duration': 0.00048828125}
{'start_time': 1126257424.4851074, 'end_time': 1126257424.4855957, 'peak_amplitude': 3.1352130176788724, 'duration': 0.000732421875}
{'start_time': 1126257424.4865723, 'end_time': 1126257424.4873047, 'peak_amplitude': 3.198057213381449з, 'duration': 0.0009765625}
{'start_time': 1126257426.8842773, 'end_time': 1126257426.8845215, 'peak_amplitude': 3.186333462298193, 'duration': 0.00048828125}
```

Figure 6: These are the event parameters of the first 10 events detected.

4 Basic Statistical Analysis

The `summarize_event_params` function summarizes the parameters of detected events.

```
def summarize_event_params(event_params):
    if not event_params:  # Check if event_params array is empty
        return {
            'num_events': 0,
            'average_duration': 0,
            'max_duration': 0,
            'average_peak_amplitude': 0,
            'max_peak_amplitude': 0
        }

    durations = [param['duration'] for param in event_params]
    peak_amplitudes = [param['peak_amplitude'] for param in event_params]

    summary = {
        'num_events': len(event_params),
        'average_duration': np.mean(durations),
        'max_duration': np.max(durations),
        'average_peak_amplitude': np.mean(peak_amplitudes),
        'max_peak_amplitude': np.max(peak_amplitudes)
    }
    return summary
```

Figure 7: This function summarizes detected event parameters, and if no events are detected, it returns a summary with zeros. For detected events, it calculates and returns the number of events, average duration, maximum duration, average peak amplitude, and maximum peak amplitude.

```
Summary of Event Params:
{'num_events': 1645, 'average_duration': 0.0005074266242401216, 'max_duration': 0.004150390625, 'average_peak_amplitude': 3.127237008782134, 'max_peak_amplitude': 4.284804453733104}
```

Figure 8: This is the summary of the detected events and their corresponding parameters, including total number of events detected, average duration, maximum duration, average peak amplitude, and maximum peak amplitude.

5 Discussion and Conclusion

This paper has detailed the methodologies and Python codes for analyzing time-domain features and detecting events in GW data. The presented functions and their explanations provide a solid framework for handling GW data, enhancing our ability to extract meaningful insights from GW observations. As GW astronomy advances with new observing runs and enhanced detector sensitivities, these tools will become essential in exploring the cosmos and understanding the underlying physics of these extraordinary events. Later, the implementation of ML will be attempted, and the application of ML may produce expectedly better performance than merely pure Python codes.

References

[1] Aasi, J., et al. (2015). Advanced LIGO. Classical and Quantum Gravity, 32(7), 074001.

[2] Abbott, B. P., et al. (2016). Observation of gravitational waves from a binary black hole merger. Physical Review Letters, 116(6), 061102.

[3] Abbott, B. P., et al. (2016). Properties of the binary black hole merger GW150914. Physical Review Letters, 116(24), 241102.

[4] Abbott, B. P., et al. (2016). Tests of general relativity with GW150914. Physical Review Letters, 116(22), 221101.

[5] Abbott, B. P., et al. (2016). Astrophysical implications of the binary black hole merger GW150914. The Astrophysical Journal Letters, 818(2), L22.

[6] Abbott, B. P., et al. (2017). GW170817: Observation of gravitational waves from a binary neutron star inspiral. Physical Review Letters, 119(16), 161101.

[7] Abbott, B. P., et al. (2017). Multi-messenger observations of a binary neutron star merger. The Astrophysical Journal Letters, 848(2), L12.

[8] Abbott, B. P., et al. (2019). A gravitational-wave standard siren measurement of the Hubble constant. Nature, 551(7678), 85-88.

[9] Abbott, B. P., et al. (2020). GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo During the First Half of the Third Observing Run. arXiv preprint arXiv:2010.14527.

[10] Abbott, B. P., et al. (2018). 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), 031040.

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

[12] Acernese, F., et al. (2015). Advanced Virgo: a second-generation interferometric gravitational wave detector. Classical and Quantum Gravity, 32(2), 024001.

[13] Acernese, F., et al. (2007). Status of Virgo. Classical and Quantum Gravity, 24(19), S381.

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

[15] Chatziioannou, K., et al. (2017). The last gravitational wave in the window: An improved waveform model for binary black hole inspirals. Physical Review D, 95(10), 104027.

[16] Cutler, C., & Thorne, K. S. (2002). An overview of gravitational-wave sources. General Relativity and Gravitation, 39(5), 151-165.

[17] Fairhurst, S. (2011). Source localization with an advanced gravitational wave detector network. Classical and Quantum Gravity, 28(10), 105021.

[18] Finn, L. S., & Chernoff, D. F. (1993). Observing binary inspiral in gravitational radiation: One interferometer. Physical Review D, 47(6), 2198.

[19] LIGO: The Laser Interferometer Gravitational-Wave Observatory. arXiv. 2007;0711.3041.

[20] Magee, R., et al. (2021). First demonstration of early warning gravitational wave alerts. Astrophys J Lett. 910(2).

[21] Martynov, D. V., et al. (2016). Sensitivity of the Advanced LIGO detectors at the beginning of gravitational wave astronomy. Physical Review D, 93(11), 112004.

[22] Parameter estimation with gravitational waves. Rev Mod Phys. 2022;94:025001.

[23] Polarization-based tests of gravity with the stochastic gravitational-wave background. Phys Rev X. 2017;7:041058.

[24] Sachdev, S., et al. (2020). An early-warning system for electromagnetic follow-up of gravitational-wave events. Astrophys J Lett. 905(2).

[25] Schutz, B. F. (2011). Networks of gravitational wave detectors and three figures of merit. Classical and Quantum Gravity, 28(12), 125023.

[26] Singer, L. P., et al. (2014). The first two years of electromagnetic follow-up with advanced LIGO and Virgo. The Astrophysical Journal, 795(2), 105.

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

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

[29] What is LIGO? LIGO Lab, Caltech. Caltech. 2023.

[30] Abbott, R., et al. (2021). Population properties of compact objects from the second LIGO-Virgo gravitational-wave transient catalog. Astrophys J Lett. 913:7.

[31] Abbott, R., et al. (2021). Tests of general relativity with binary black holes from the second LIGO-Virgo gravitational-wave transient catalog. Phys Rev D. 103:122002.

[32] Abbott, R., et al. (2021). Upper limits on the isotropic gravitational-wave background from Advanced LIGO's and Advanced Virgo's third observing run. arXiv. 2101.12130.

[33] Abbott, R., et al. (2021). Searches for continuous gravitational waves from young supernova remnants in the early third observing run of Advanced LIGO and Virgo. arXiv. 2101.12130.

[34] Abbott, B. P., et al. (2016). Improved analysis of GW150914 using a fully spin-precessing waveform model. Physical Review X, 6(4), 041014.

[35] Abbott, B. P., et al. (2017). Multi-messenger observations of a binary neutron star merger. The Astrophysical Journal Letters, 848(2), L12.

[36] Finn, L. S., & Chernoff, D. F. (1993). Observing binary inspiral in gravitational radiation: One interferometer. Physical Review D, 47(6), 2198.

[37] Fortifying gravitational-wave tests of general relativity against astrophysical assumptions. Phys Rev D. 2023;108:124060.

[38] GW170817: Observation of gravitational waves from a binary neutron star inspiral. arXiv. 2017;1710.05832.

[39] Machine learning for gravitational-wave astronomy: Methods and applications for high-dimensional laser interferometry data. Academic Commons. 2021.

[40] Dong, S. (2024). Astrophysical Insights Through Gravitational Wave Data Analysis: Data Preprocessing. viXra preprint viXra:2407.0026.