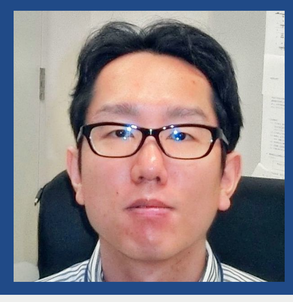
Prototype Implementation of a Web-Based Gravitational Wave Signal Analyzer: SNEGRAF

Satoshi Eguchi, Shota Shibagaki, Kazuhiro Hayama, Kei Kotake

Fukuoka University



Abstract

A direct detection of gravitational waves is one of the most exciting frontiers for modern astronomy and astrophysics. Gravitational wave signals combined with "classical" electro-magnetic observations, known as multi-messenger astronomy, promise newer and deeper insights about the cosmic evolution of astrophysical objects such as neutron starts and black holes. To this end, we have been developing an original data processing pipeline for KAGRA, a Japanese gravitational wave telescope, for optimal detections of supernova events. As a part of our project, we have just released a web application named "SuperNova Event Gravitational-wave-display in Fukuoka (SNEGRAF)" in this autumn. SNEGRAF accepts the users' theoretical waveforms as a plain text file consisting of a time series of h_+ and $h_\times$ (the plus and cross mode of gravitational waves, respectively), then displays the input, a corresponding spectrogram, and power spectrum together with KAGRA sensitivity curve and the signal-to-noise ratio; we adopt Google Visualization API for the interactive visualization of the input waveforms. However, it is a time-consuming task to draw more than $\sim 10^5$ data points directly with JavaScript, although the number can be typical for a supernova hunt by assuming a typical duration of the event and sampling rate of the detectors; a combination of recursive decimations of the original in the server-side program and an appropriate selection of them depending on the time duration requested by the user in a web browser achieves an acceptable latency. In this poster, we present the current design, implementation and optimization algorithms of SNEGRAF, and its future perspectives.

1. What Are Gravitational Waves?

In the framework of general relativity, a mass curves the space-time around it, and the curvature is observed as gravity. An accelerated motion of a mass generates a disturbance of space-time, which propagates in a vacuum in the form of waves; these waves are referred to as "gravitational waves." Gravitational waves can penetrate even a very dense material, and carry the information of the space-time around a massive but compact astronomical object such as a neutron star and black hole. The first direct detection of a gravitational wave is known as GW150914, where a merger of two stellar-mass black holes took place.

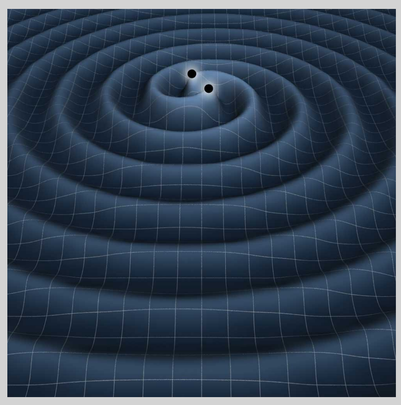


Figure 1: Image credit: NASA

2. Multi-Messenger Astronomy and Our Goals

Multi-messenger astronomy, which utilizes observations of gravitational waves and neutrinos combined with those in multiple wavelengths, attracts a lot of attention recently since it promises a deeper understanding of the innermost part of a high energy astrophysical phenomenon. At Fukuoka University, we assembled a team to promote multi-messenger astronomy focusing on the physics of supernovae in this April. Goals of our mission are:

- developing an original data processing pipeline for KAGRA, a Japanese gravitational wave telescope, to detect a supernova event at optimal efficiency,
- providing data visualization and analysis software for the KAGRA observations to the world.

3. SNEGRAF

As the first step of our software releases, we have just made a web application named "SuperNova Event Gravitational-wave-display in Fukuoka (SNEGRAF; Fig. 2)" public in this October. SNEGRAF accepts a time series of h_+ and $h_\times$ (two individual modes of a gravitational wave) in a character-separated-value (CSV) format as an input, and displays the input waveforms, a corresponding spectrogram, and power spectrum together with the signal-to-noise ratio of the input signal and the analytic KAGRA sensitivity curve. To try SNEGRAF, please visit `https://nibiru.sci.fukuoka-u.ac.jp/snegraf/` or scan the QR code on the top of this poster.

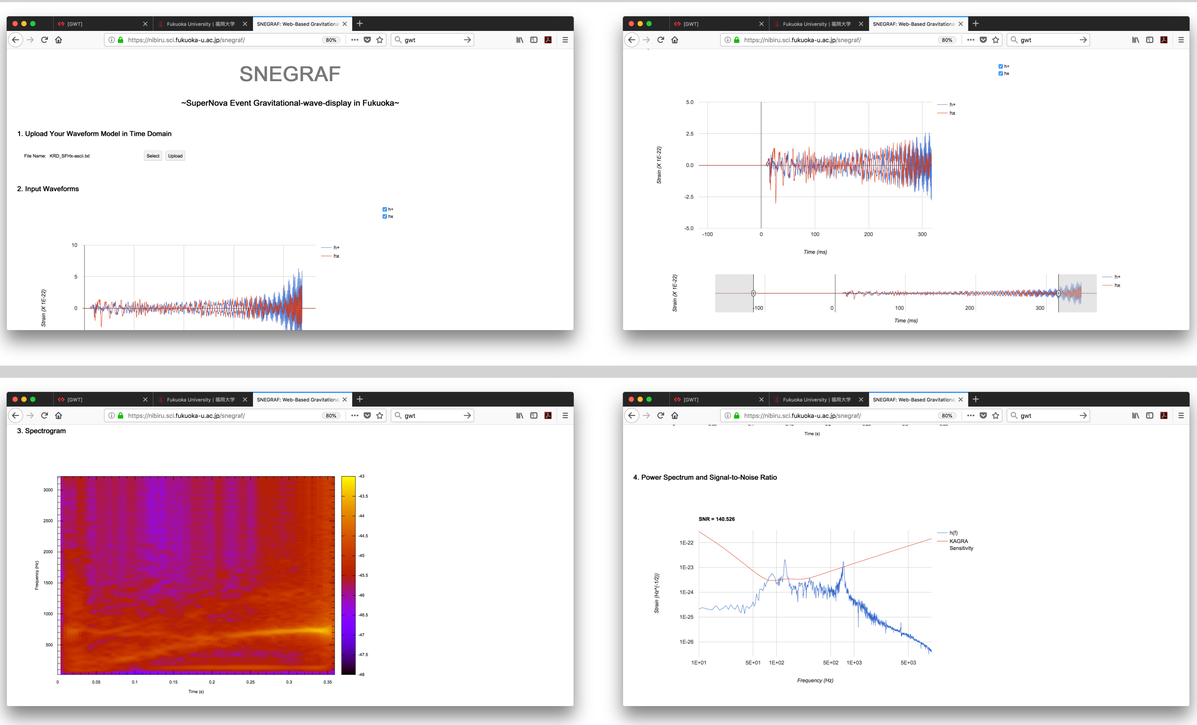


Figure 2: Screenshots of SNEGRAF. From left to right and top to bottom, the banner, waveform viewer, spectrogram, and power spectrum, respectively.

Column	Content
1st	Time (sec)
2nd	h_+
3rd	$h_\times$

Table 1: Details of an input file format for SNEGRAF. A pipe (`|`), comma (`,`), tab (`\t`), and white space are acceptable for a column separator. A hash (#) is regarded as a beginning of comments.

4. Inside of SNEGRAF

SNEGRAF is a simple Ajax application hosted on a Java servlet. Since we have quite limited human resources and utilize existing software libraries written in either C/C++ or Java, we adopt GWT (previously known as Google Web Toolkit), which generates both server-side and client-side codes from a single Java source file, for an application framework. Google Visualization API and its GWT binding (GWT Charts) are used for an interactive visualization of input waveforms.

The file uploading functionality is implemented with File API in HTML5. A text file uploaded by a user is transfered to the servlet as is as an argument of type `String` during a remote procedure call (RPC). Then the string is parsed into arrays of type `double` to hold $(t, h_+, h_\times)$ in each row in the servlet, and "resampled and decimated hierarchically." The servlet invokes a Python script to compute a spectrogram, which is converted to a scalable vector graphics (SVG) file by `gnuplot` and encoded into a Base64 string. A power spectrum is calculated with a Java implementation of fast Fourier transform (FFT), accompanied by an evaluation of the signal-to-noise ratio (SNR) based on the analytic KAGRA sensitivity curve. At the end, the waveform arrays, the Base64 encoded spectrogram, the array for the power spectrum, and the SNR are packed into a single object in JavaScript object notation (JSON), and returned to the web client as the result of the RPC. The waveforms and power spectrum are plotted with Google Visualization API on the client.

5. Data Reduction Algorithm

By assuming a typical sampling rate of a gravitational wave detector, our programs should be able to handle $N \sim 10^5$ data points on the fly. However, this is a very heavy task for a JavaScript application like SNEGRAF currently. To achieve this goal even on a low-powered CPU, we applied "hierarchical decimation technique" to SNEGRAF. The basic idea of this method is to apply a decimation by a factor of 2 recursively on server side, and to select an adequate result depending on the time duration requested by a user on client side.

1. Find the integer m satisfying $2^m < N \leq 2^{m+1}$ and resample the original waveform evenly into new 2^m points by linear interpolation. This takes $O\left(m2^m\right)$ time.
2. Decimate the resampled waveform by 2. This yields new $N_{m-1} = 2^{m-1}$ data points and takes $O\left(2^{m-1}\right)$ time.
3. Apply the 2nd step recursively until the number of new data points N_i is less than $N_{\mathrm{disp,th}}\,(= 2048)$. At this step, the total amount of data points is exactly less than $N + N/2 + N/4 + \cdots = 2N$.
4. On client side, calculate the number of data points $N_{\mathrm{disp},i}$ which fall inside the user specified time range for each decimated data. The total processing time is $O\left(m^2\right)$.
5. On the client, find the largest i such that $N_{\mathrm{disp},i} \leq N_{\mathrm{disp,th}}$ with a binary search algorithm, and plot the data.

When N is $\sim 10^5 \simeq 2^{17}$, there are just $\simeq 150$ lookups of the arrays and $\leq N_{\mathrm{disp,th}}$ drawings on the client with just consuming twice as much as the initial memory space. The processing time on client side is reduced by two orders of magnitude thanks to this algorithm, and SNEGRAF quickly responds to the user's operations even on a low-powered computer with an Intel Atom CPU.

6. Future Work

- A spectrogram and power spectrum displayed on the current version are "static". We have a plan to make them interactive (e.g., the time range on the input waveform viewer will link to that selected on a spectrogram).
- To display a sky map, which is a heat map representing the likelihood of the source direction.

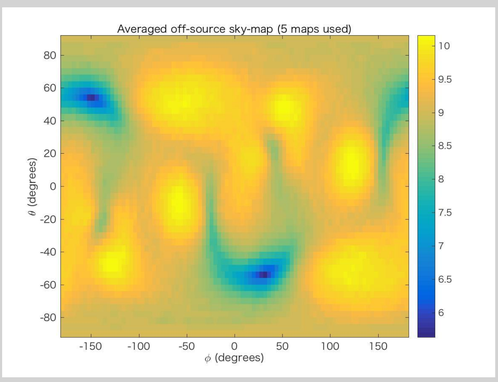


Figure 3: An example of a sky map.

Acknowledgements

This work is supported by JSPS KAKENHI Grant Number 17H06364.