PyCBC Live

Rapid Detection of Gravitational Waves from Compact Binary Mergers

Naoki Kita

June 8, 2018 @ando Lab

Contents

- Introduction
- Methodology
- Sensitivity of the analysis
- Multidetector sky localization
- Other applications
- Conclusions

Introduction

- The era of GW multi-messenger astronomy started. (Ex. GW170817 and GRB 170817A)
- This paper presents "PyCBC Live", a new method to detect GW events.
- This technique enabled the initial discovery of GW170104 and GW170608.
- How to analyze now and in the future.
- The important word "low latency"

What is “PyCBC” ?

The software package used to explore astrophysical sources of gravitational waves.

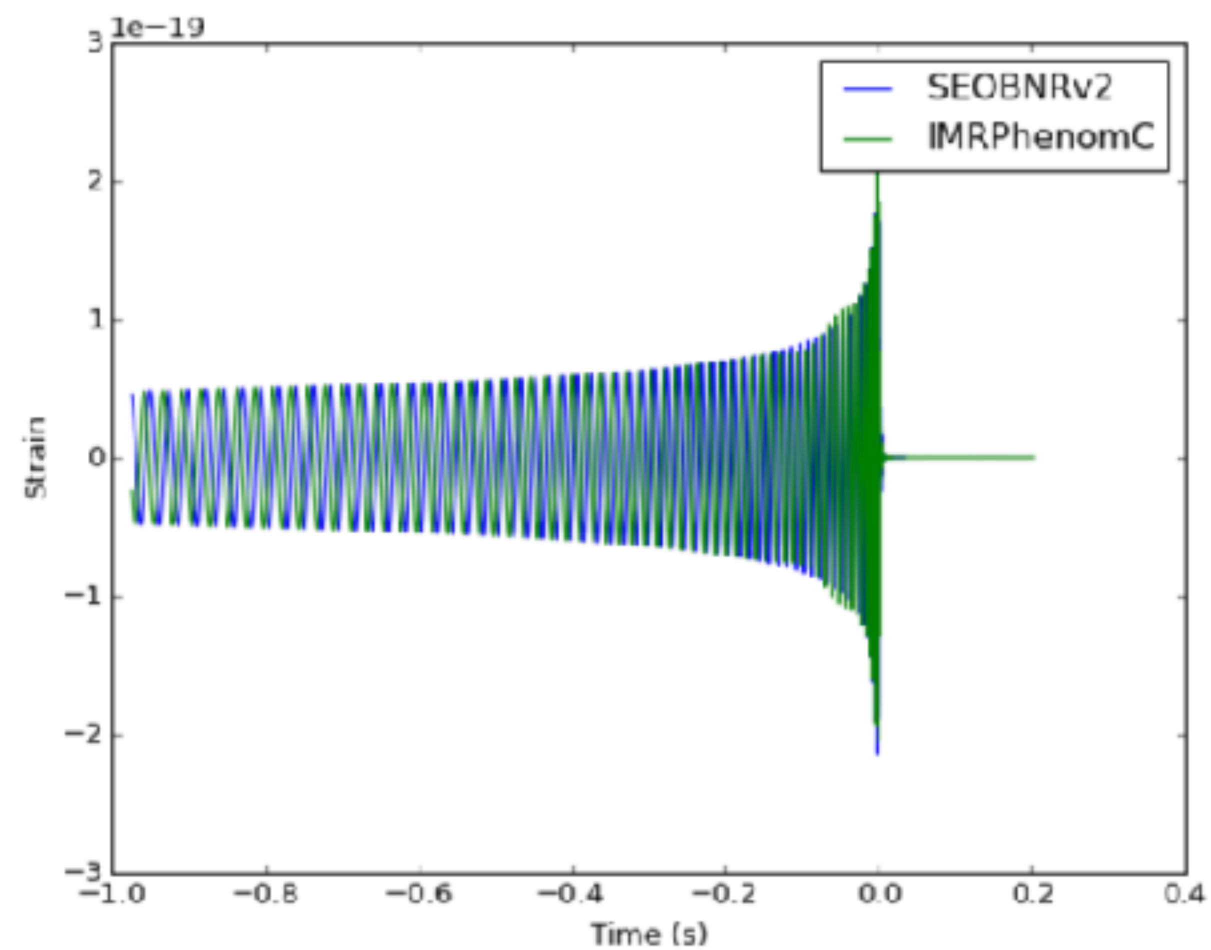
Plotting Time Domain Waveforms

```
import pylab
from pycbc.waveform import get_td_waveform

for apx in ['SEOBNRv2', 'IMRPhenomC']:
    hp, hc = get_td_waveform(approximant=apx,
                             mass1=10,
                             mass2=10,
                             spin1z=0.9,
                             delta_t=1.0/4096,
                             f_lower=40)

    pylab.plot(hp.sample_times, hp, label=apx)

pylab.ylabel('Strain')
pylab.xlabel('Time (s)')
pylab.legend()
pylab.show()
```



https://ahnitz.github.io/pycbc/waveform.html

What is "PyCBC" ?

Plotting frequency evolution of TD waveform

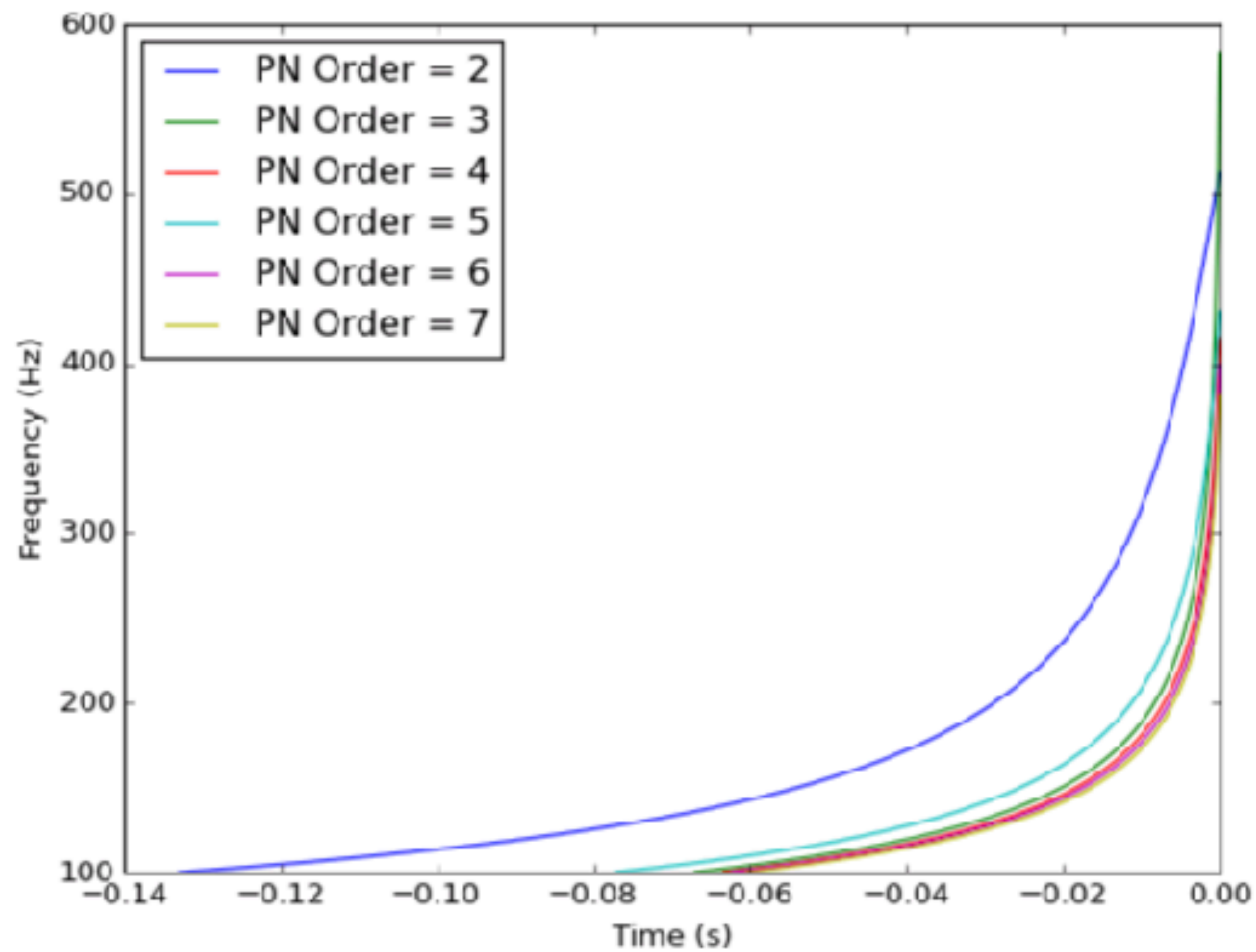
```
import pylab
from pycbc import waveform

for phase_order in [2, 3, 4, 5, 6, 7]:
    hp, hc = waveform.get_td_waveform(approximant='SpinTaylorT4',
                                     mass1=10, mass2=10,
                                     phase_order=phase_order,
                                     delta_t=1.0/4096,
                                     f_lower=100)

    hp, hc = hp.trim_zeros(), hc.trim_zeros()
    amp = waveform.utils.amplitude_from_polarizations(hp, hc)
    f = waveform.utils.frequency_from_polarizations(hp, hc)

    pylab.plot(f.sample_times, f, label="PN Order = %s" % phase_order)

pylab.ylabel('Frequency (Hz)')
pylab.xlabel('Time (s)')
pylab.legend(loc='upper left')
pylab.show()
```



https://ahnitz.github.io/pycbc/waveform.html

Methodology

- The analysis relies on matched filtering which extracts signals from stationary colored Gaussian noise.
- PyCBC Live is based on the deep offline analysis.
- Latency is determined by
  1:The production, aggregation and transfer of the strain data
  2:The sky localization of candidates
  3:The generation and distribution of the resulting alerts
- The appropriate latency time scale is O(10s)

Methodology

1. **Extracting Gravitational-wave Signals**

2. **Selecting and Ranking Gravitational-wave Candidate Event**

3. **Background Estimation**

4. **Architecture and Computational Considerations**

Extracting Gravitational-wave Signals

- This step is to generate a matched-filter SNR time series for each template.
- Finite impulse response(FIR) filters are used to guarantee a fixed latency.
- The main part of the analysis is “analysis stride”, which determine the pace at analyzing.
- The maximum latency is 20s, with an average latency of only 16s, in the O2 LIGO / Virgo observing run.

Extracting Gravitational-wave Signals

New data

High passing

Resampling

Ideal frequency reponse

15Hz, primally to reduce the dynamic range of the data

by FIR

The computationally-intensive part is performed in single-precision floating point arithmetic.

16384Hz → 2048Hz

We can discard the the higher frequency data!

Extracting Gravitational-wave Signals

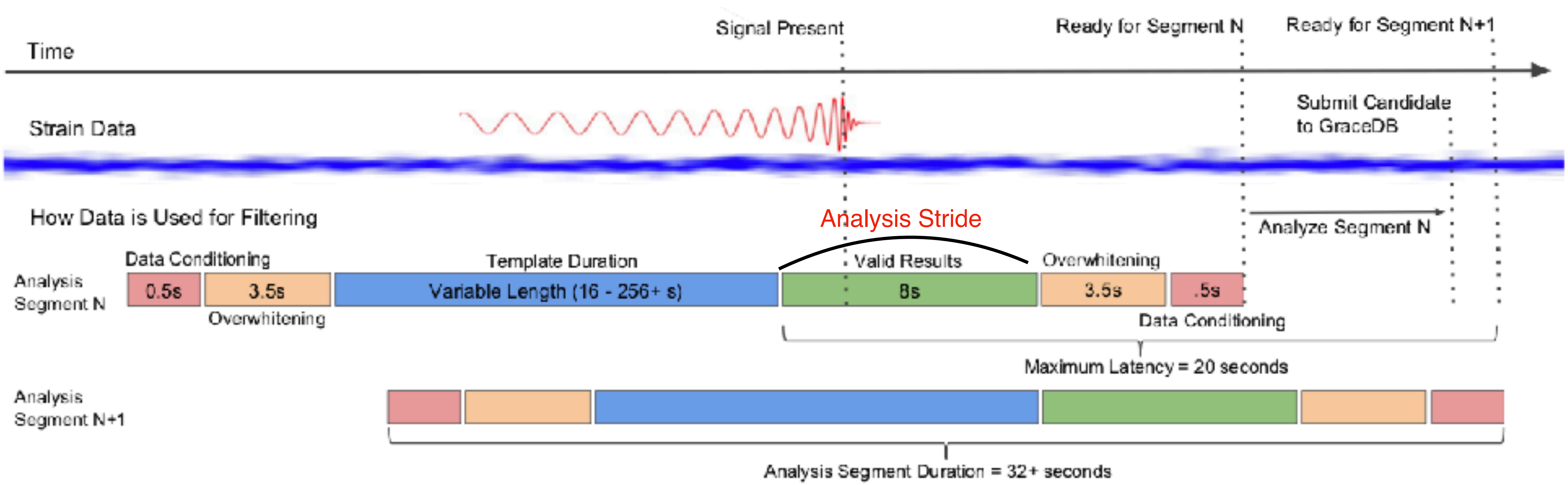


Fig1. A diagram of how data is processed by the PyBCB Live

Extracting Gravitational-wave Signals

- The matched filter SNR is

$$\rho^2 \equiv \frac{\| \langle s|h \rangle \|^2}{\langle h|h \rangle}$$

cf. inner product

$$\langle a|b \rangle = 4 \int_0^\infty \frac{\tilde{a}(f)\tilde{b}^*(f)}{S_n(f)} df$$

s : data
h : template waveform
$S_n(f)$: the estimated one-side PSD of the noise around the time of event

- The data is divided into 4s intervals and converted to the frequency domain.

Extracting Gravitational-wave Signals

- The SNR time series by inverse FFT

$$\rho^2(t) = \frac{4}{\langle h|h\rangle} \int_0^\infty \frac{\tilde{s}(f)\tilde{h}^*(f)}{S_n(f)} e^{2\pi i f t} df$$

Frequency-domain $\Rightarrow$ Time-domain

- Lower frequency cut off accumulates at least 99.5% of signal power relative to much lower cutoff.
- Overwhitened data is produced for each analysis segment by constructing an FIR filter from $1/S_n$.

Selecting and Ranking Gravitational-wave Candidate Event

- How are candidate events identified?
- How are they ranked based on their signal and astrophysical consistency?

Using the SNR time series, we identify the paek SNR in each analysis stride

And

it defines a single-detector trigger

But the number of triggers is very large…

Selecting and Ranking Gravitational-wave Candidate Event

To control the triggers and minimize the loss of sensitivity, set the threshold on the peak SNR at ～5.5.

For low-latency

Due to time low-latency analysis can't access to the full data.
And
we make additional cuts based on

- A continuous local estimation of detector's sensitivity
- The variability of the PSD over time

The measurement of an unreasonably large SNR

➡ Limitting bad time

Selecting and Ranking Gravitational-wave Candidate Event

And we apply the robust signal consistency test.
A χ^2 signal consistency test is

$$\chi_r^2 = \frac{1}{2p-2} \sum_{i=1}^{i=p} \| \langle s|h_i \rangle - \langle h_i|h_i \rangle \|^2 ,$$

Template h is divided into p frequency bands.

With this parameter, we can classify as

- Astrophysical signals
- Non-Gaussian transient noise

we can use it for the few SNR peaks above threshold.

Selecting and Ranking Gravitational-wave Candidate Event

So we have to use re-weighted SNR.

$$\tilde{\rho} = \begin{cases} \rho & \text{for } \chi_r^2 <= 1 \\ \rho\left[\frac{1}{2}\left(1 + \left(\chi_r^2\right)^3\right)\right]^{-1/6} & \text{for } \chi_r^2 > 1 \end{cases}$$

and new statistics is

$$\tilde{\rho}_c^2 = \tilde{\rho}_H^2 + \tilde{\rho}_L^2 + 2\ln\left(p^S(\vec{\theta})\right)$$

The astrophysical probability of a trigger

$\vec{\theta}$ includes

- Relative amplitudes
- Phase
- Time difference between observatories

Background Estimation

- We need to determine

  If a candidate is significant enough for consideration by astronomers

→ we calculate the False Alarm Rate(FAR) by resampling

$$IFAR = \frac{T_{buffer}^2}{T_{shift}}$$

For the low-latency analysis,

only 5 hours of past data is kept.

Background Estimation

The two cases when the background estimate is not stable

- A loud coincident event
- A broadband disturbance in the PSD

Low-latency analysis has fewer data quality and optimal removal time was not processed.

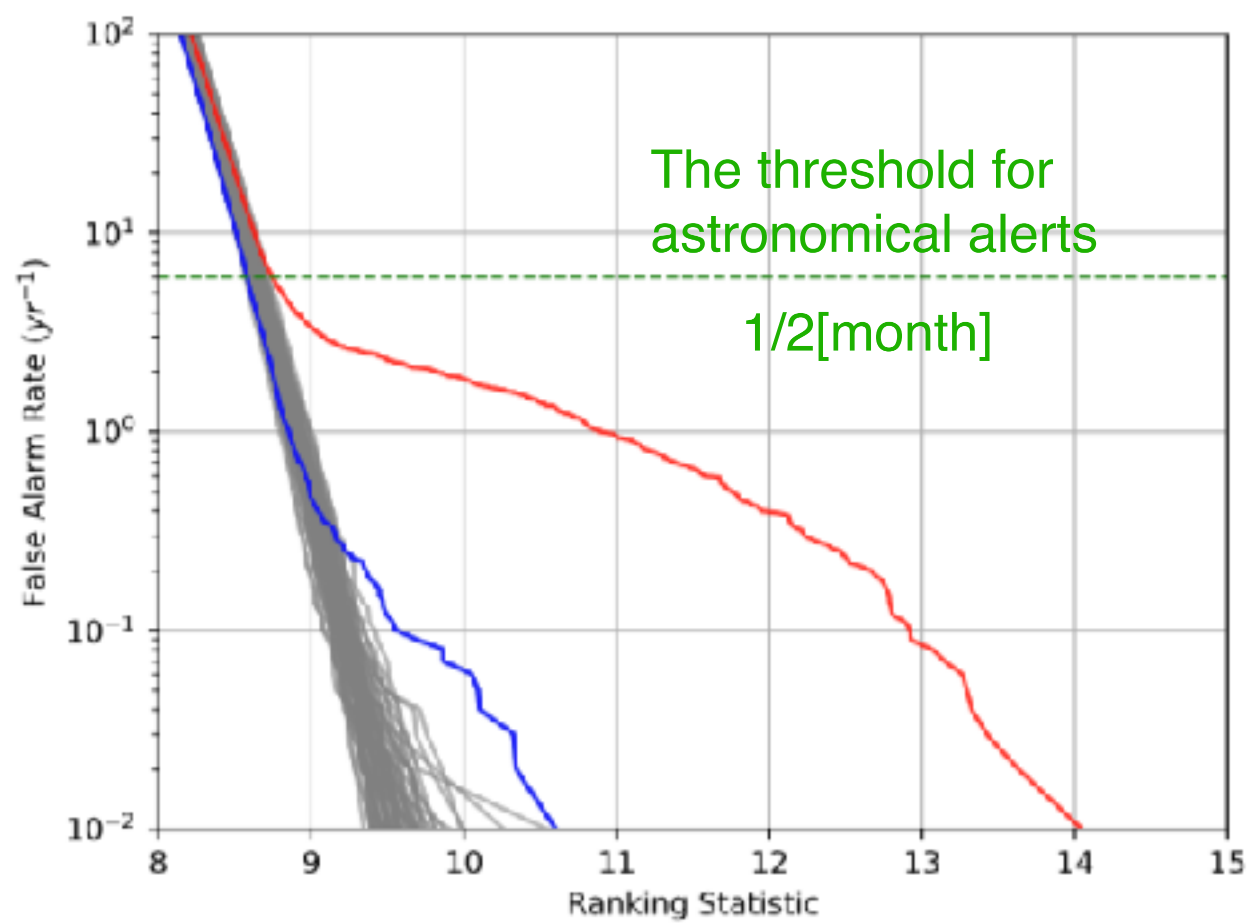


Fig2. Variability of the relationship between the ranking static and FAR during O1

Architecture and Computational Considerations

PyCBC Live has the high level architecture of analysis.

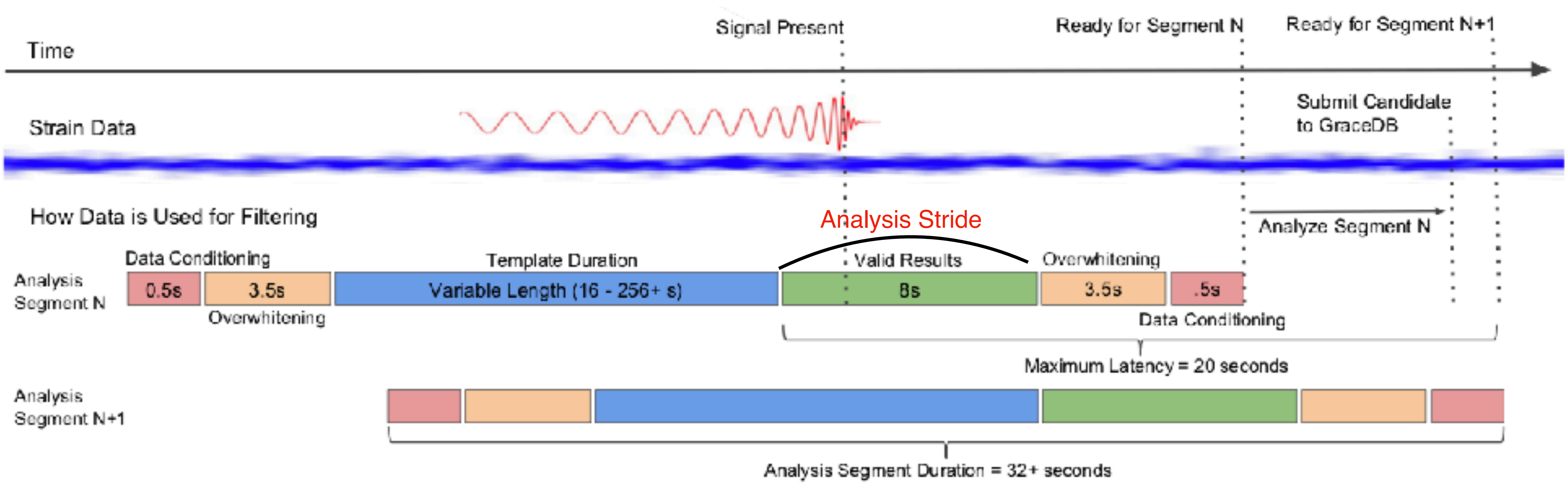
- To ensure the analysis completes in a time shorter than the “analysis stride”, we adopt the parallelization system.

Message Passing Interface

- In the early advanced-detector era, banks contain $O(10^5)$ template waveforms.

- Uploading Candidates to the GW Candidate Database(GraceDB) is conducted if it passes nominal FAR 1/day

Architecture and Computational Considerations



Architecture and Computational Considerations

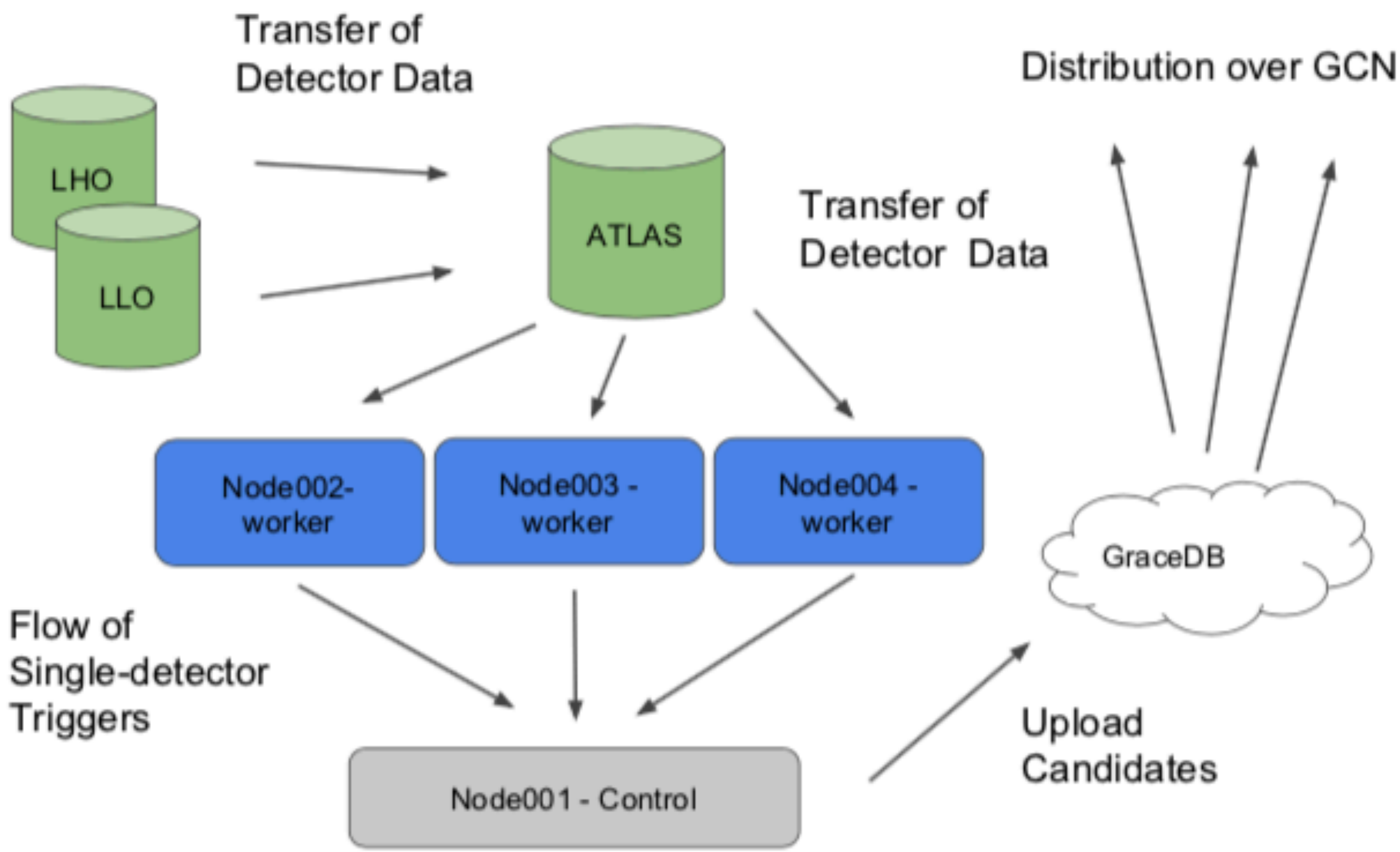


Fig3. The high level overview of how data is flowing through the PyCBC Live analysis

Architecture and Computational Considerations

- To ensure that the analysis keeps up with the incoming data, it is configured to only take this time.
- The computational cost is

$$C \propto \frac{\bar{T}\ln(\bar{T})}{T_{stride}}$$

$\bar{T}$: the template duration averaged over the bank
T_{stride} : the analysis stride

$\bar{T}$ is primary determined by the shape of the PSD during observation.

- There is a balance between the latency of the analysis and the computational cost.

Sensitivity of the analysis

The sensitivity is estimated by

- Simulating an astrophysical population of sources
- Adding the signals to a data set
- Observing how many signals are detected

Table1. Relative sensitivity of the PyCBC Live low-latency analysis and the PyCBC-based offline analysis

Source Category	Relative Sensitivity
BNS	0.87 ± 0.05
NSBH	0.98 ± 0.04
BBH	1.05 ± 0.09

Sensitivity of the analysis

Waveform Model

Effective-one-body

Inspiral-merger-ringdown

Post-Newtonian

Inspiral-only

Compact Binary Mergers

Binary Black holes

Newtron Star-Black hole Binaries

Binary Neutron Stars

Sensitivity of the analysis

Compare the sensitivity of “PyCBC Live” and “PyCBC-based offline”.

We want to minimize the difference!

Live → all together

Offline → treated background separately for templates

FAR(a False Alarm Rate) is 1/2[month]

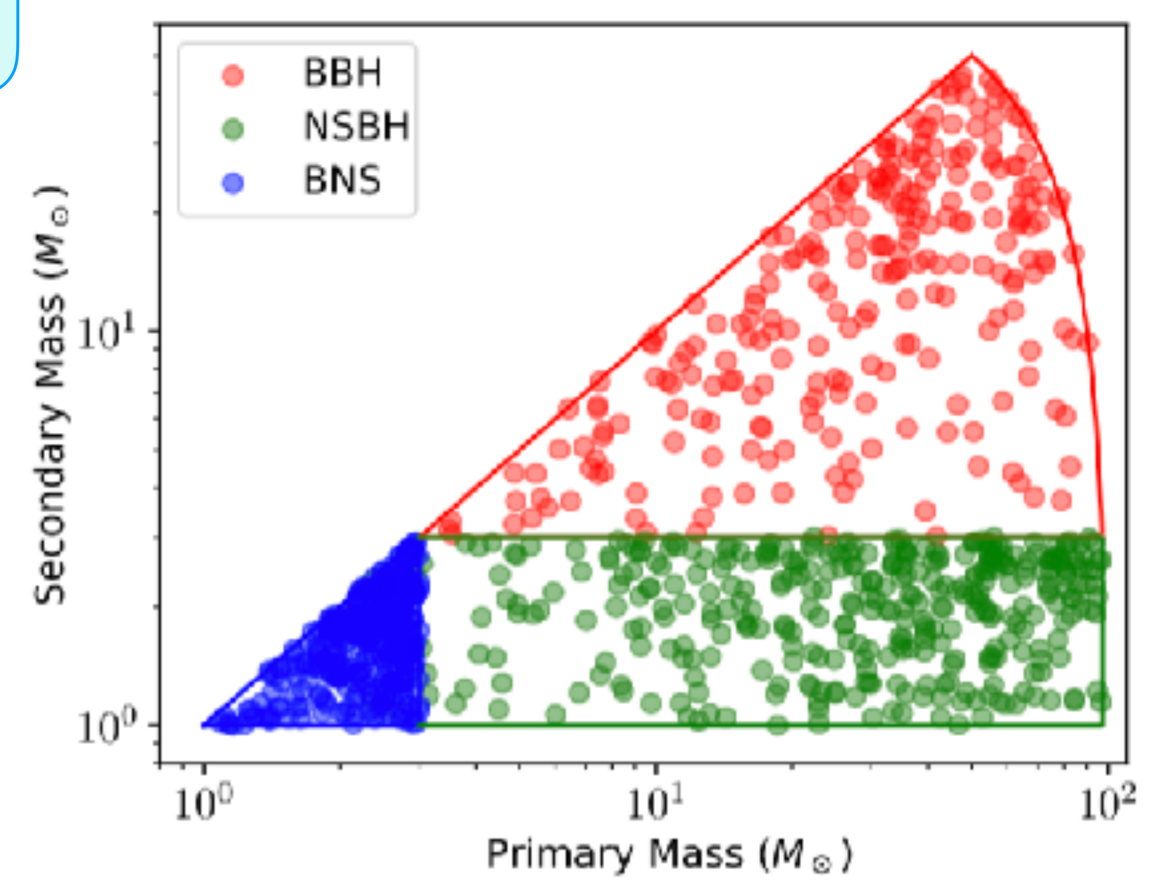


Fig4. Binary component masses of the samples of the offline analysis injection set (O(1000))

Multidetector sky localization

- LIGO, Virgo and a 3rd instrument allows for the more accurate Sky Localization. (Ex. GW170817)

$10 \sim 10^2 \ \mathrm{deg}^2$

- The 3rd detector doesn't need to detect candidates at all.

LIGO

Virgo

3rd Detector

Need

Matched-filtering separately

Combine the results in variable way

Multidetector sky localization

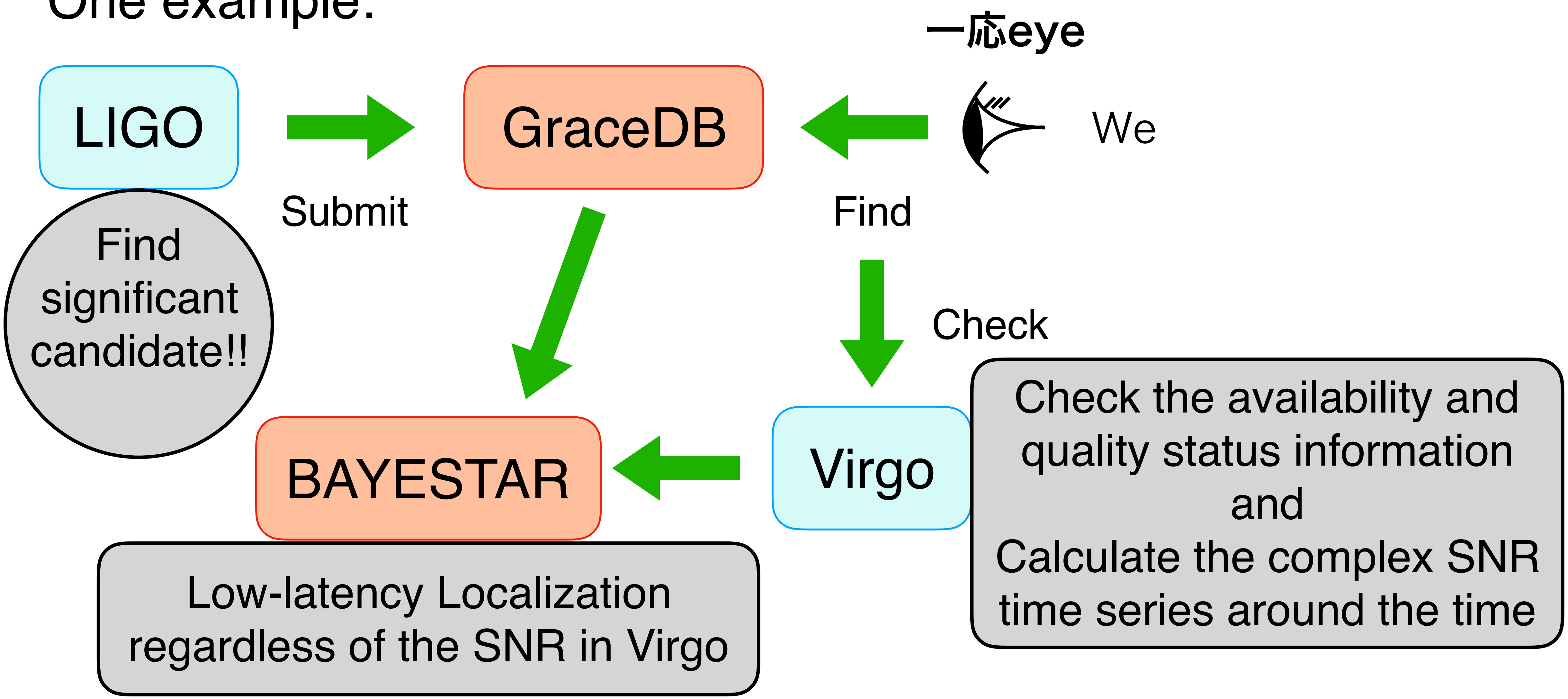
The rapid Localization is performed by the BAYESTAR.

- A candidate's complex SNR time series
- A local estimate of the noise PSD from any number of detectors

regardless of which detectors reported a trigger for the candidate

Multidetector sky localization

One example:



We use the same template for LIGO and Virgo.

Multidetector sky localization

Test 3000 BNS, NSBH and BBH signals.

The detector’ noise
→analytic PSD models

[Term]
| : O1 + early O2
| : Late O2
| : Full LIGO + Virgo

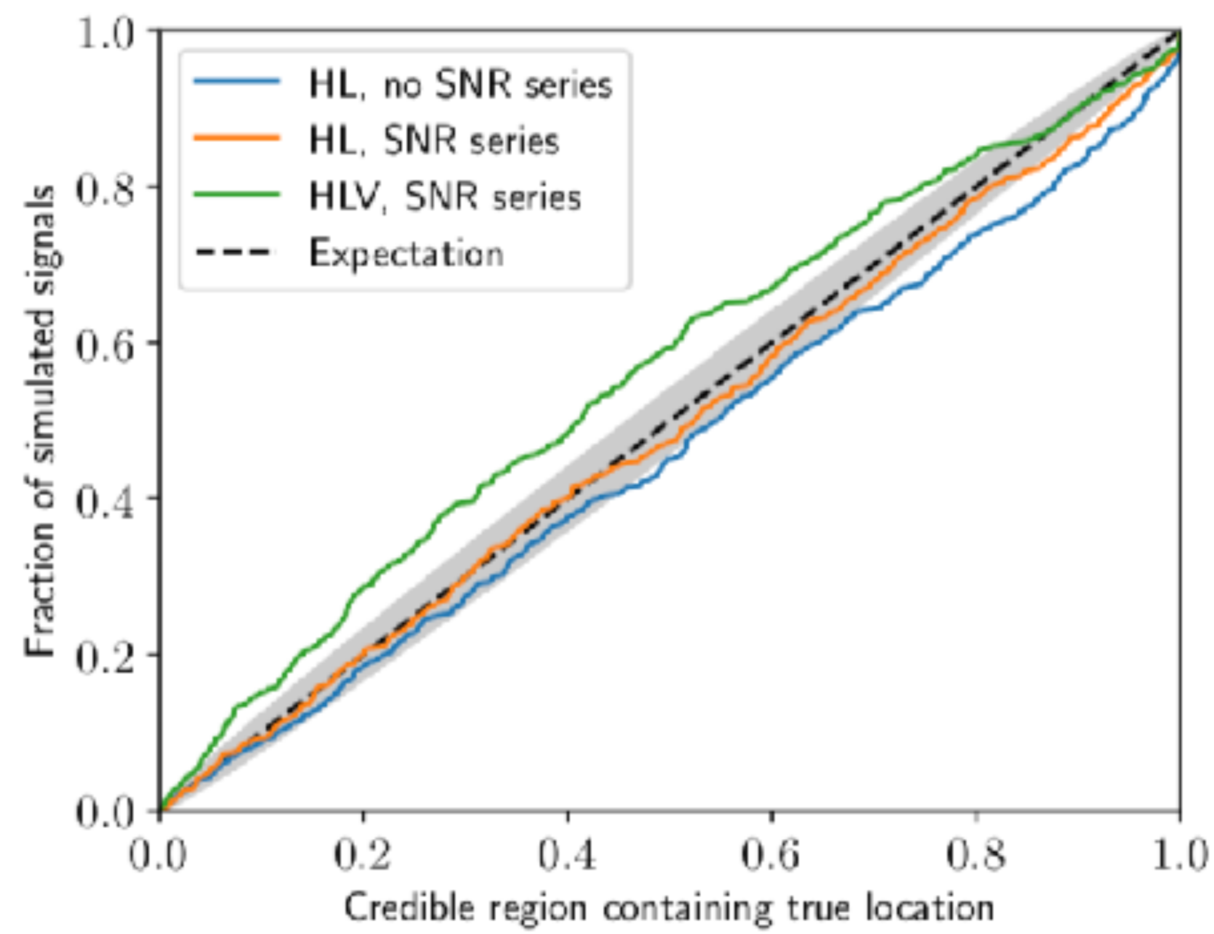


Fig5.Relativity of the rapid sky localization of simulated signals detected by PyCBC Live and localized by BAYESTAR

Multidetector sky localization

Test 3000 BNS, NSBH and BBH signals.

The detector' noise
→analytic PSD models

[Term]
| : O1 + early O2
| : Late O2
| : Full LIGO + Virgo

Because of the BAYESTAR settings

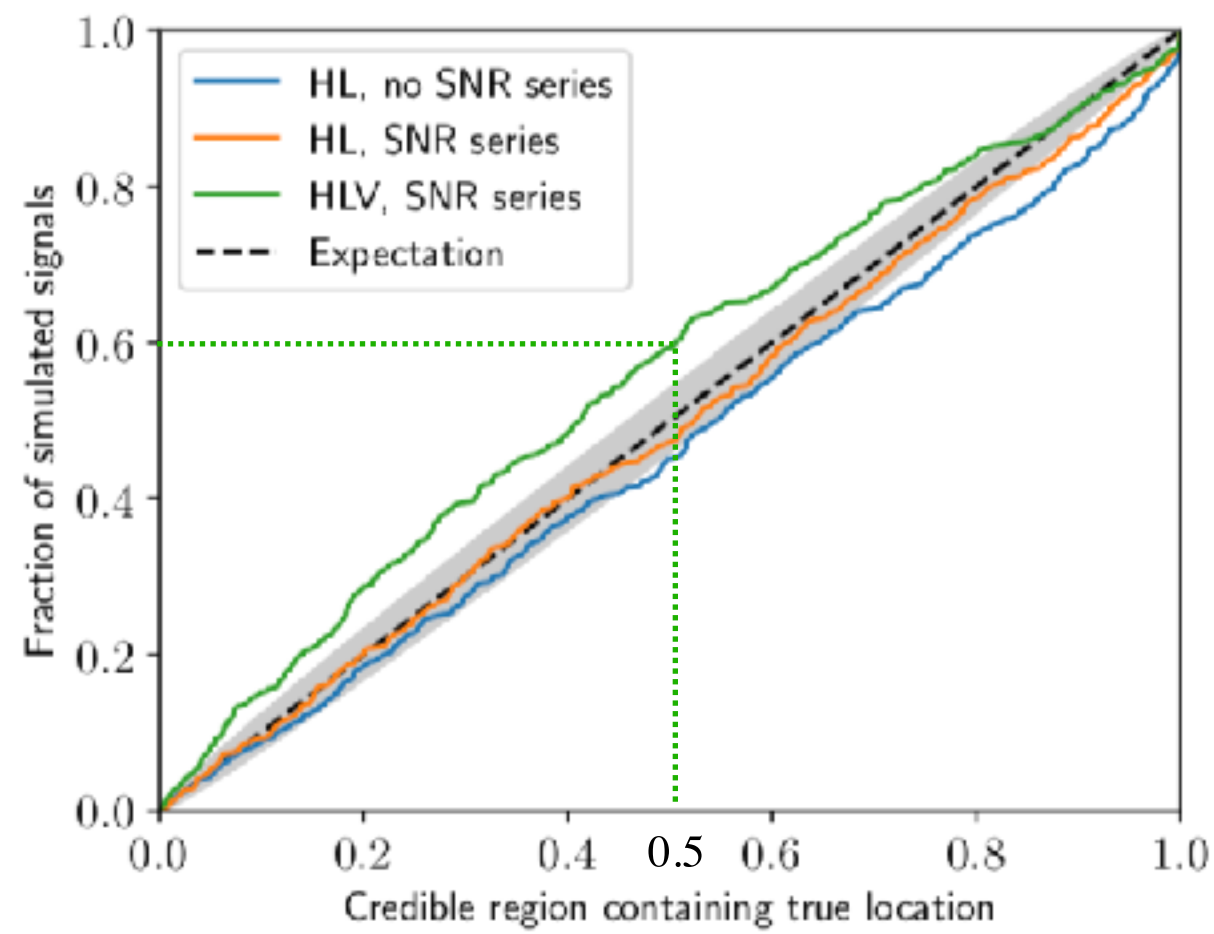


Fig5.Relativity of the rapid sky localization of simulated signals detected by PyCBC Live and localized by BAYESTAR

Other applications

1. Single Detector Search

2. Data Monitoring

Single Detector Search

Why is the detection based on a single detector difficult?
→ the background incurred from a detector's non-Gaussian

noise transients

With other detectors for GW or electromagnetic observations, Sky Localization is established.

Then…single detector search is no used?

Single Detector Search

Why is the detection based on a single detector difficult?
→ the background incurred from a detector's non-Gaussian

noise transients

With other detectors for GW or electromagnetic observations, Sky Localization is established.

Then…single detector search is no used?

It is useful with the tools, especially
"the ranking of triggers"

Single Detector Search

A few additional cuts make it possible to reject noise!!

- Candidates which are consistent with very short duration transient noise are excluded.
- Signal consistency test is more stringently applied and a cut directly on the reduced $\chi^2 < 4$ is used.

FAR(a False Alarm Rate) is
less than 1/1[month]

Data Monitoring

- The PyCBC Live is also used for Data Monitoring tool.
- The pPyCBC Live is running continuously and regularly monitors the triggers.

Time, SNR, template mass and spins, along with signal consistency tests

- Especially, GW17014 and GW170608

Conclusions

- PyCBC Live is a effective analysis designed to rapidly detect GW.
- PyCBC Live operation during O2 shows the utility of standard frequency-domain matched filtering.
- Due to the existence of new observatories, KAGRA, LIGO India and so on, a new similar technique like PyCBC Live will be investigated in the future.


Thank you for Listening!!
kabekin.com