NRGW summer school, 2022.07.25~29

# 중력파 데이터 분석


김영민 (울산과학기술원)

Email: ymkim715@gmail.com
ymkim715@unist.ac.kr

# 목차

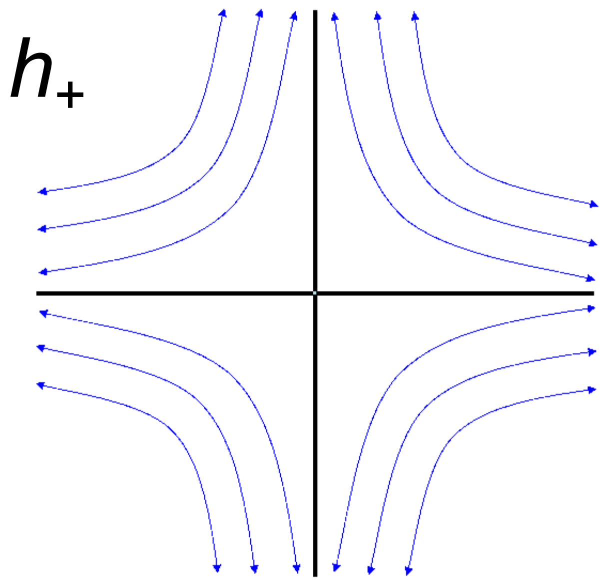
1. What is a Gravitational Wave ?
   - Astrophysical sources, detection sensitivity
2. Gravitational Wave Open Data
   - Data Product
3. Data analysis
   - Nyquist frequency, Window function
   - Visualization, Q-transform
   - Signal Search
   - Data Quality, Detector Characterization
   - Parameter Estimation
4. 겨울학교 문제 풀이

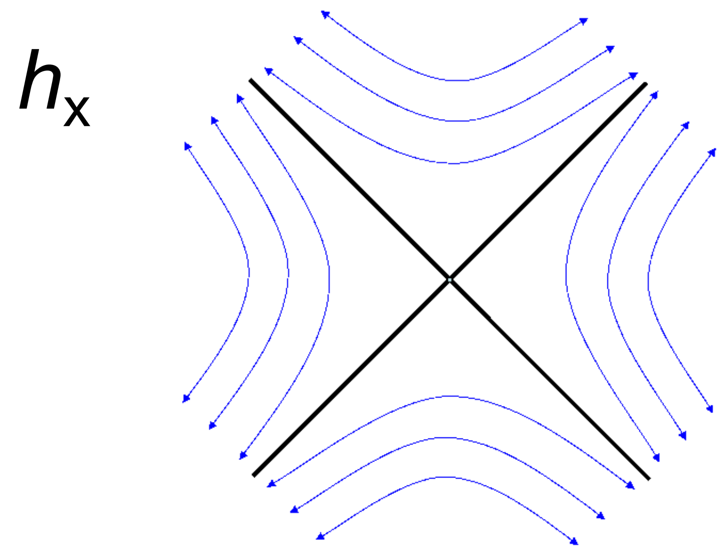
# Gravitational Waves

**Gravitational waves are *propagating solutions* to the Einstein Field Equations of General Relativity → solutions *h*(*r*,*t*) → dynamic space-time!**

$$h_{\mu\nu} \approx \frac{1}{r}\frac{G}{c^4}\ddot{I}_{\mu\nu}$$

**transverse to the propagation direction of the gravitational wave**




**Physically, *h* is a *strain*: ΔL/L**

In D. Reitze's presentation in LIGO ODW #1, 2018

# Gravitational Waves

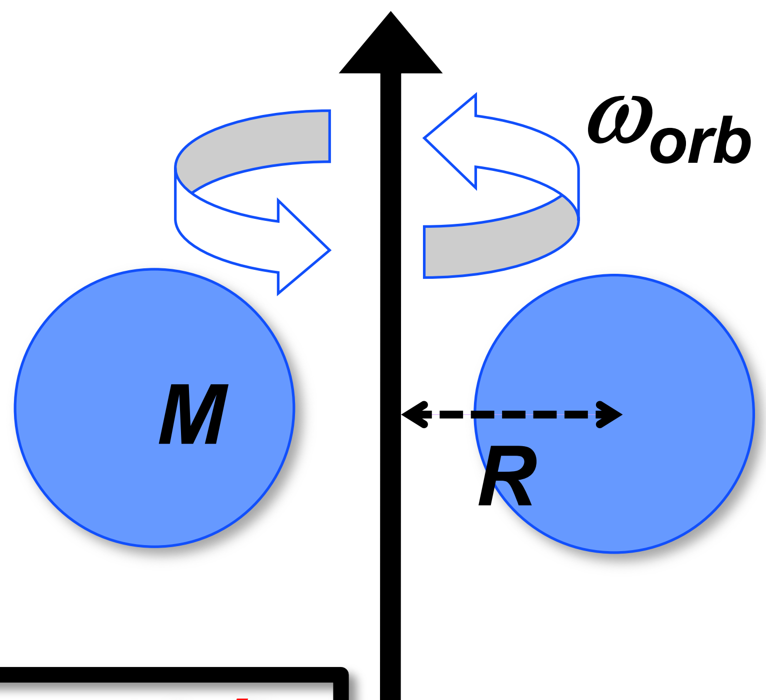
**Gravitational waves are *propagating solutions* to the Einstein Field Equations of General Relativity → solutions *h*(*r*,*t*) → dynamic space-time!**

$$h_{\mu\nu} \approx \frac{1}{r}\frac{G}{c^4}\ddot{I}_{\mu\nu}$$

**transverse to the propagation direction of the gravitational wave**

**$10M_{\odot}$ Binary Black Hole System**

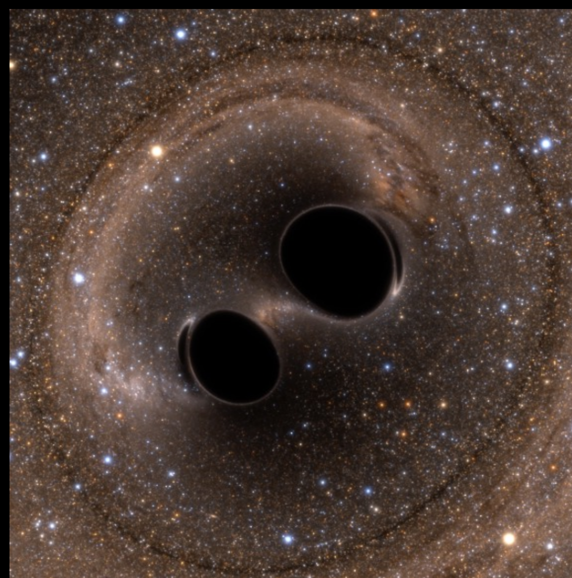
$$h \approx \frac{8GMR^2\omega_{orb}^2}{rc^4} \sim 10^{-21}$$




**Physically, *h* is a *strain*: ΔL/L**

In D. Reitze's presentation in LIGO ODW #1, 2018

# Astrophysical GW Sources



Credit: Bohn, Hébert, Throwe, SXS

***Coalescing Binary Systems***

- Black hole – black hole
- Black hole – neutron star
- Neutron star – neutron star

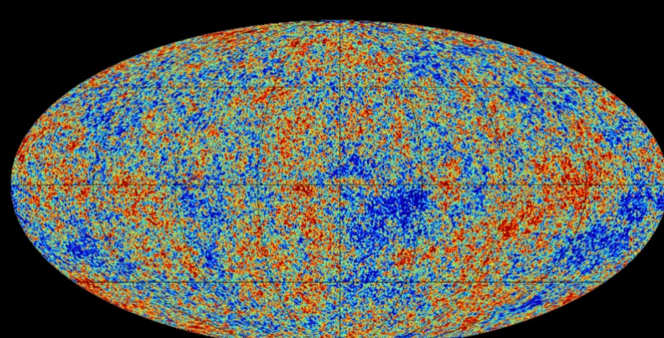
(modeled waveform)

Credit: Chandra X-ray Observatory

***Transient 'Burst' Sources***

- asymmetric core collapse supernovae
- cosmic strings
- ???

(Unmodeled waveform)



Credit: Planck Collaboration

***Stochastic Background***

- residue of the Big Bang
- incoherent sum of unresolved 'point' sources

(stochastic, incoherent noise background)

Credit: Casey Reed, Penn State

***Continuous Sources***

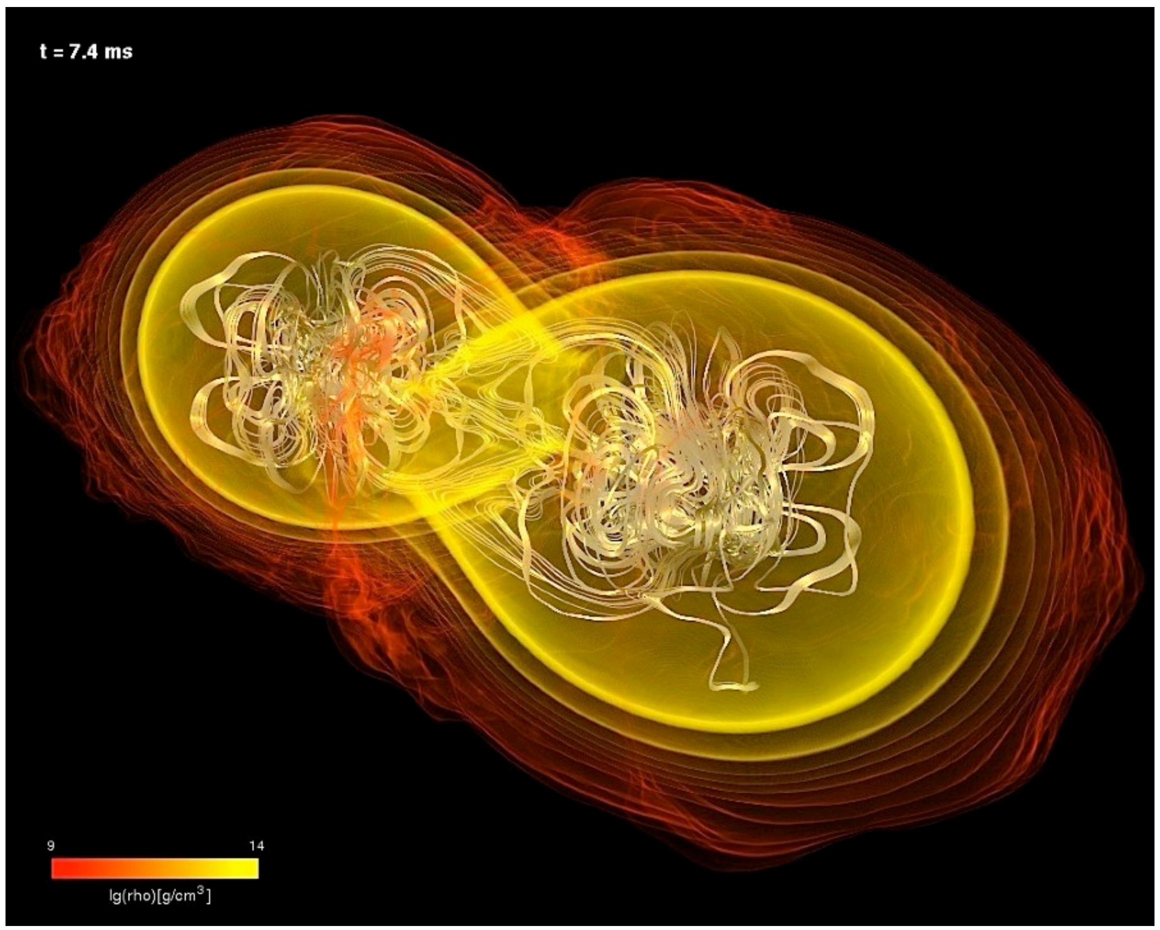
- Spinning neutron stars

(monotone waveform)

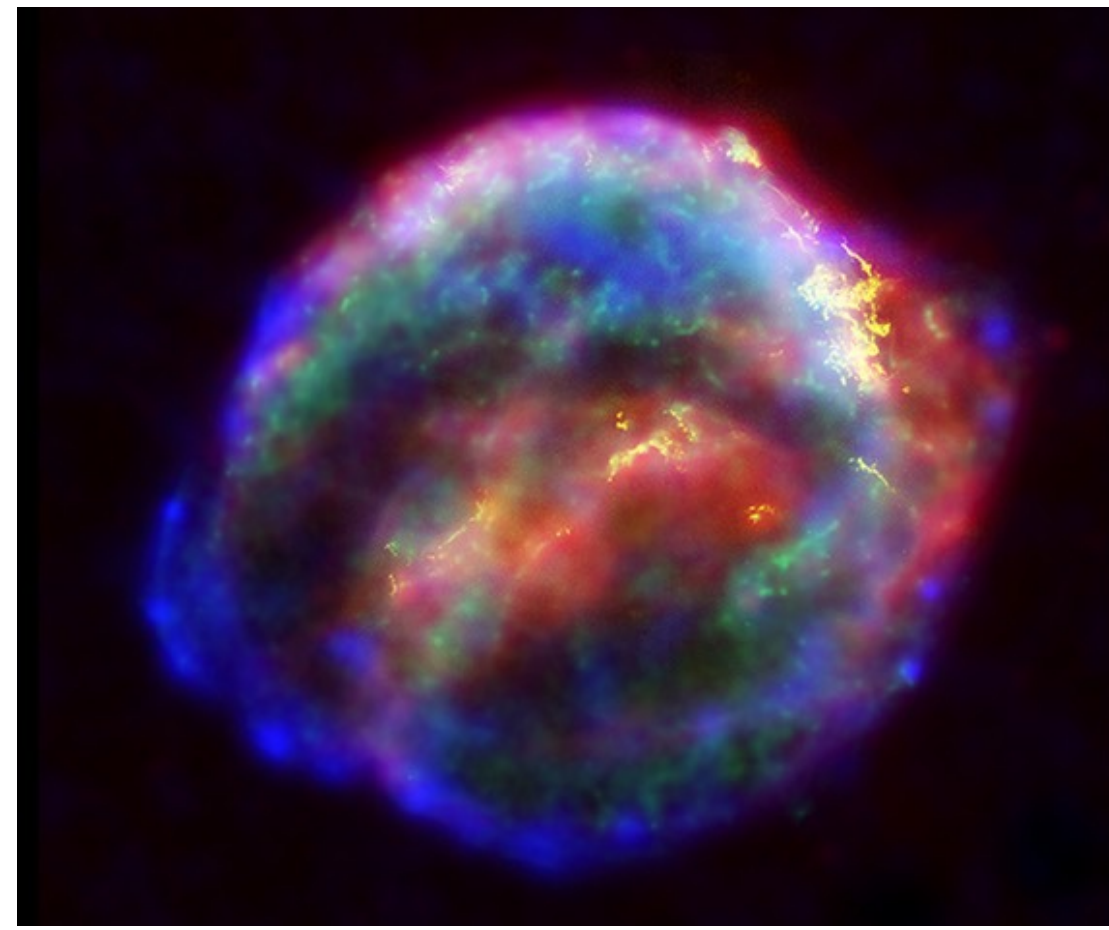
In D. Reitze's presentation in LIGO ODW #1, 2018

# The Astrophysical Sources of GWs (1)

## Transient sources




Credit: Albert Einstein Institute (AEI)



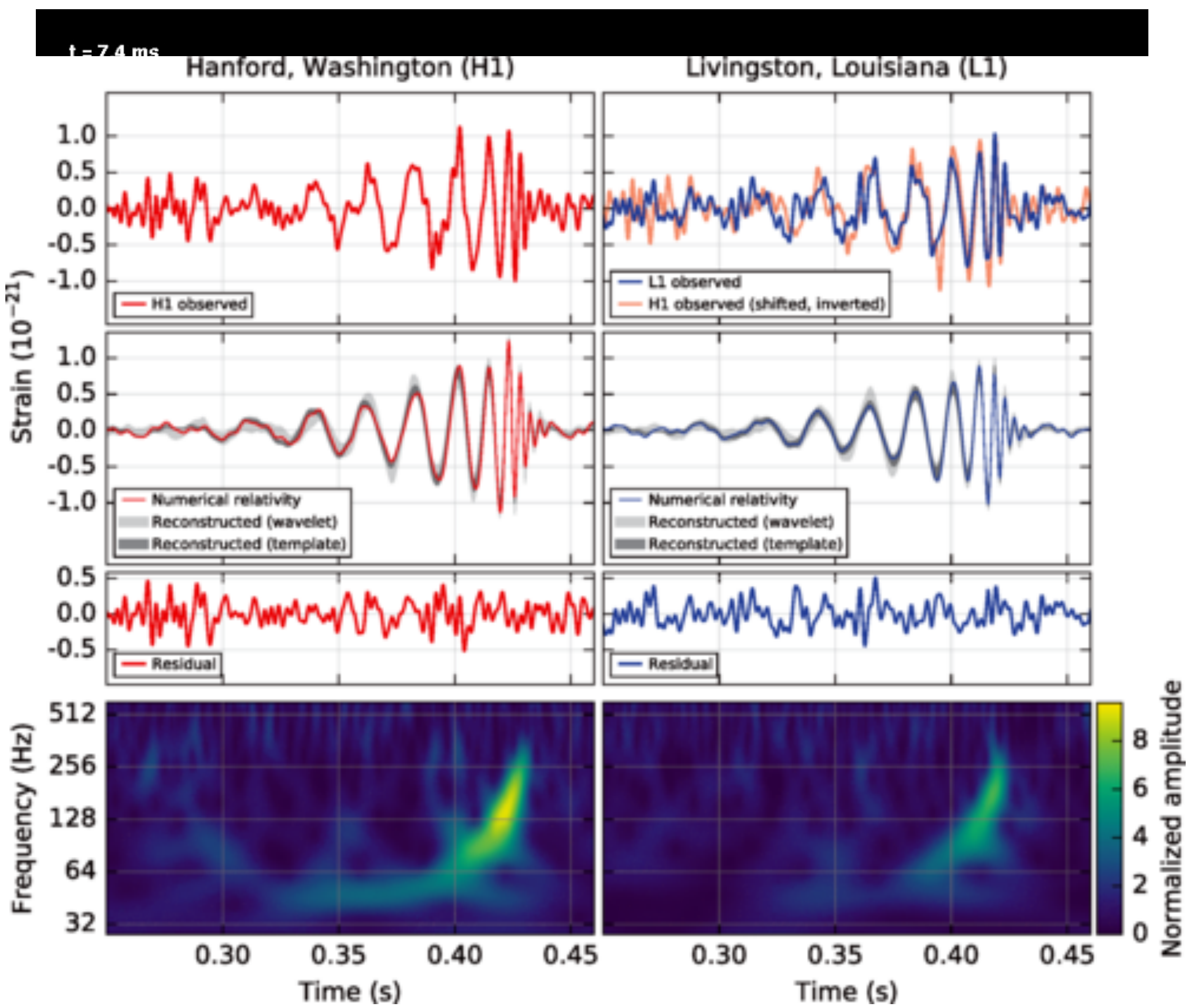
Credit: Chandra X-ray Observatory

Compact Binary Coalescence
(modeled waveform)

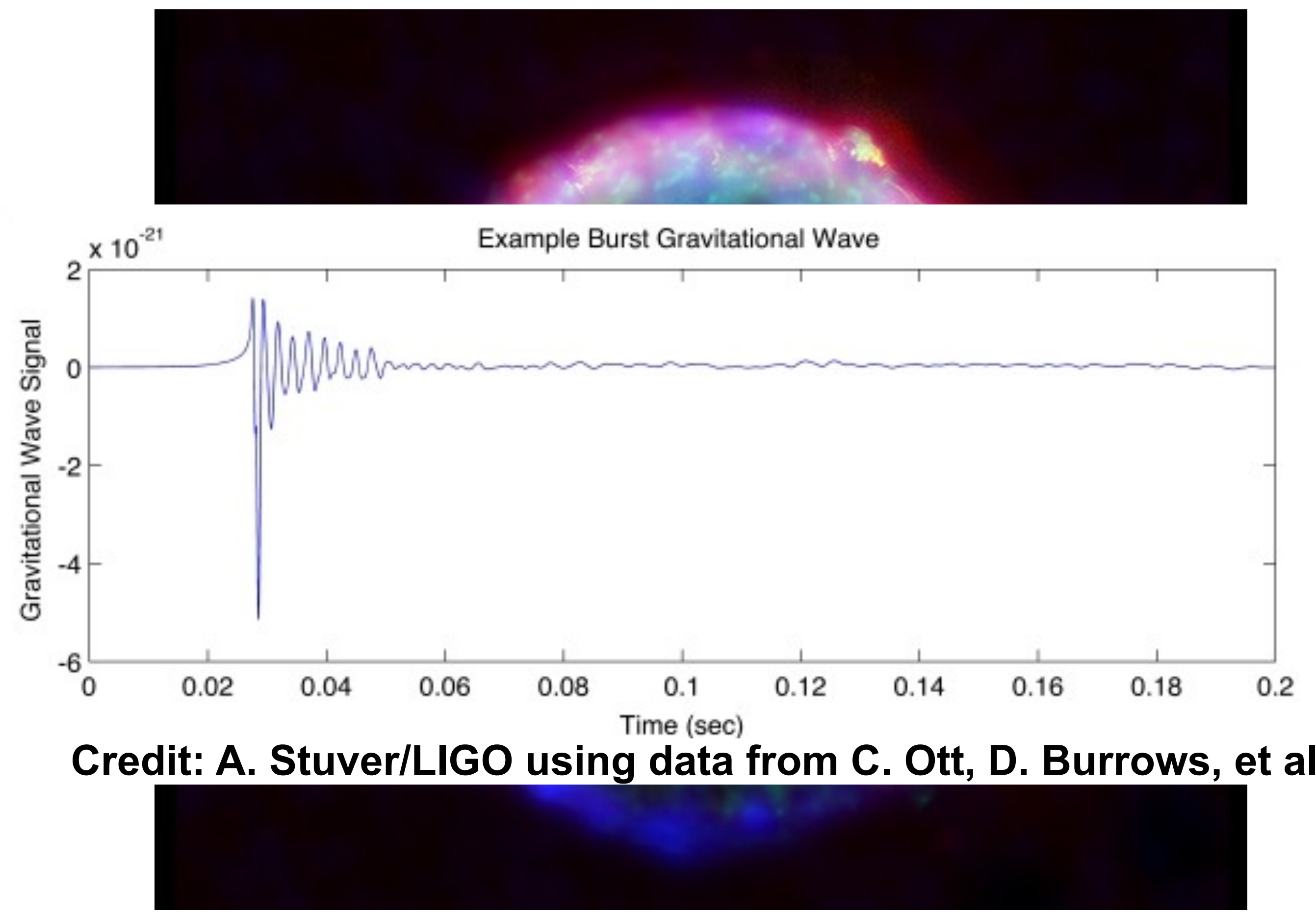
Burst sources
(un-modeled waveform)

# The Astrophysical Sources of GWs (1)

## Transient sources



PhysRevLett.116.061102



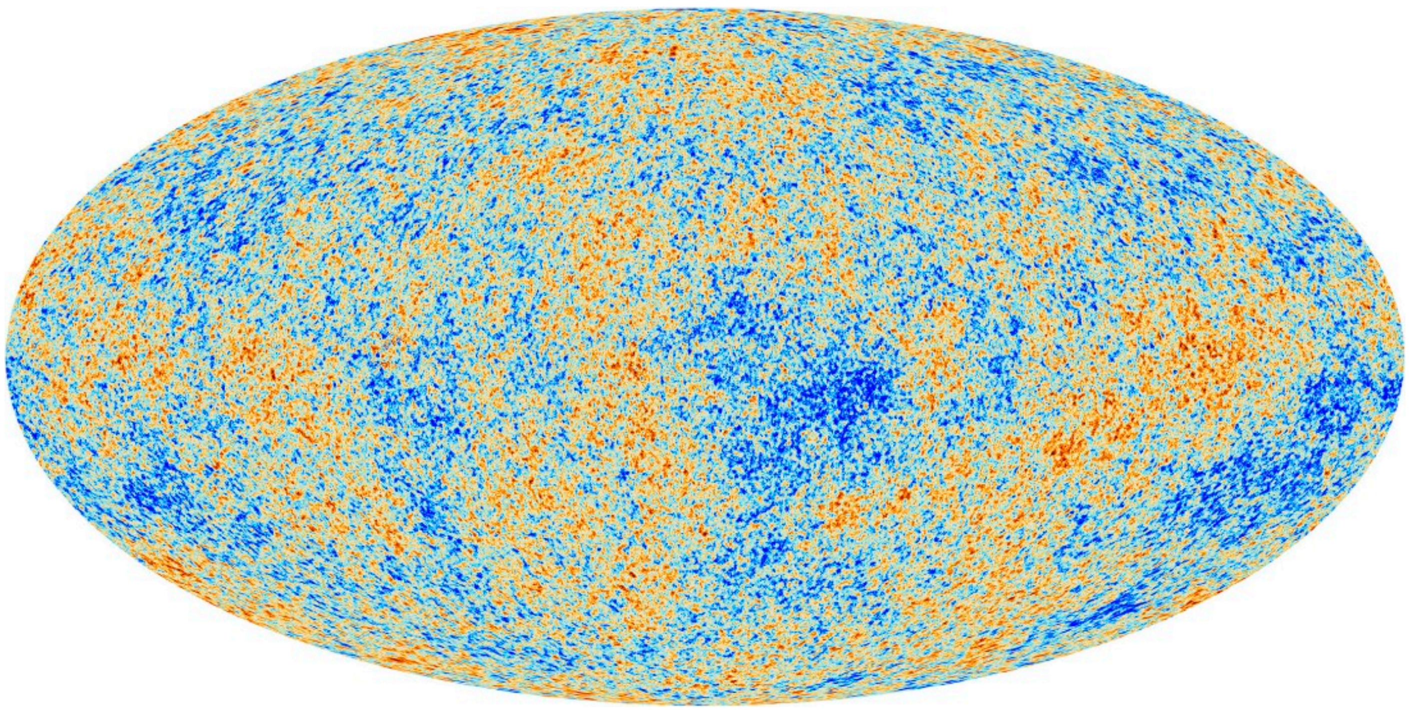
Credit: Chandra X-ray Observatory

Compact Binary Coalescence
(modeled waveform)

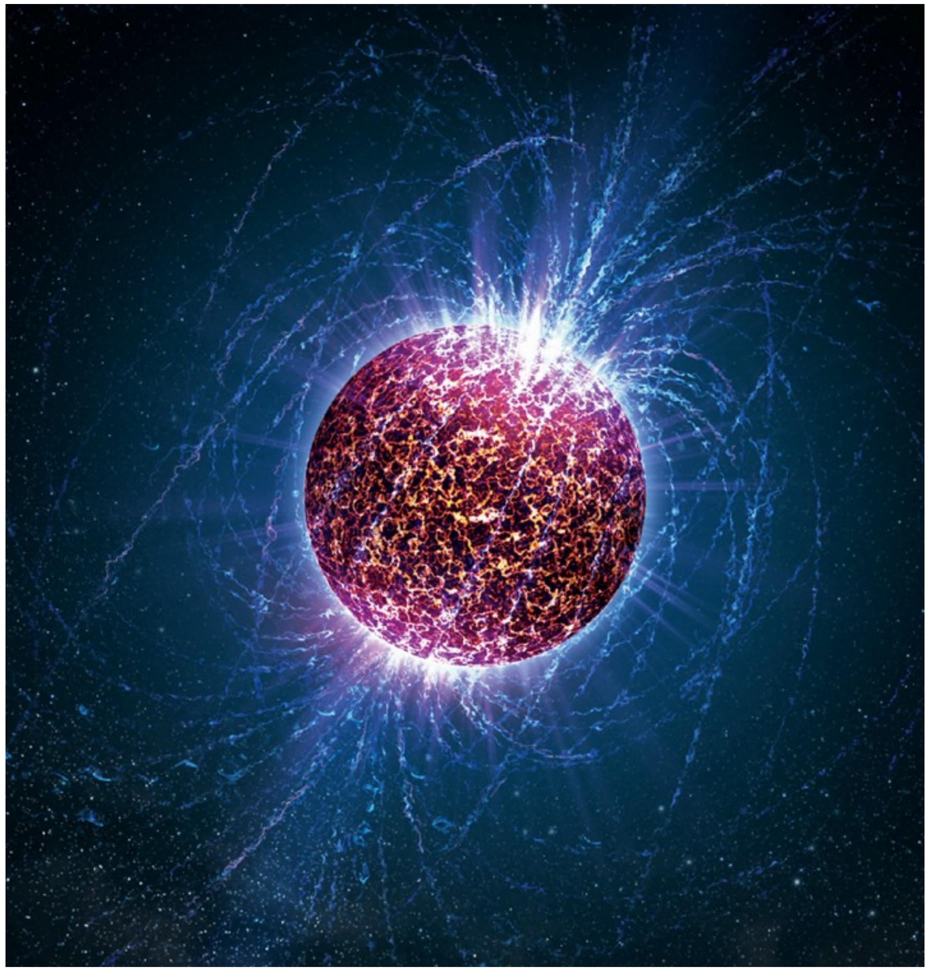
Burst sources
(un-modeled waveform)

# The Astrophysical Sources of GWs (2)

## non-Transient sources



Credit: Plank Collaboration



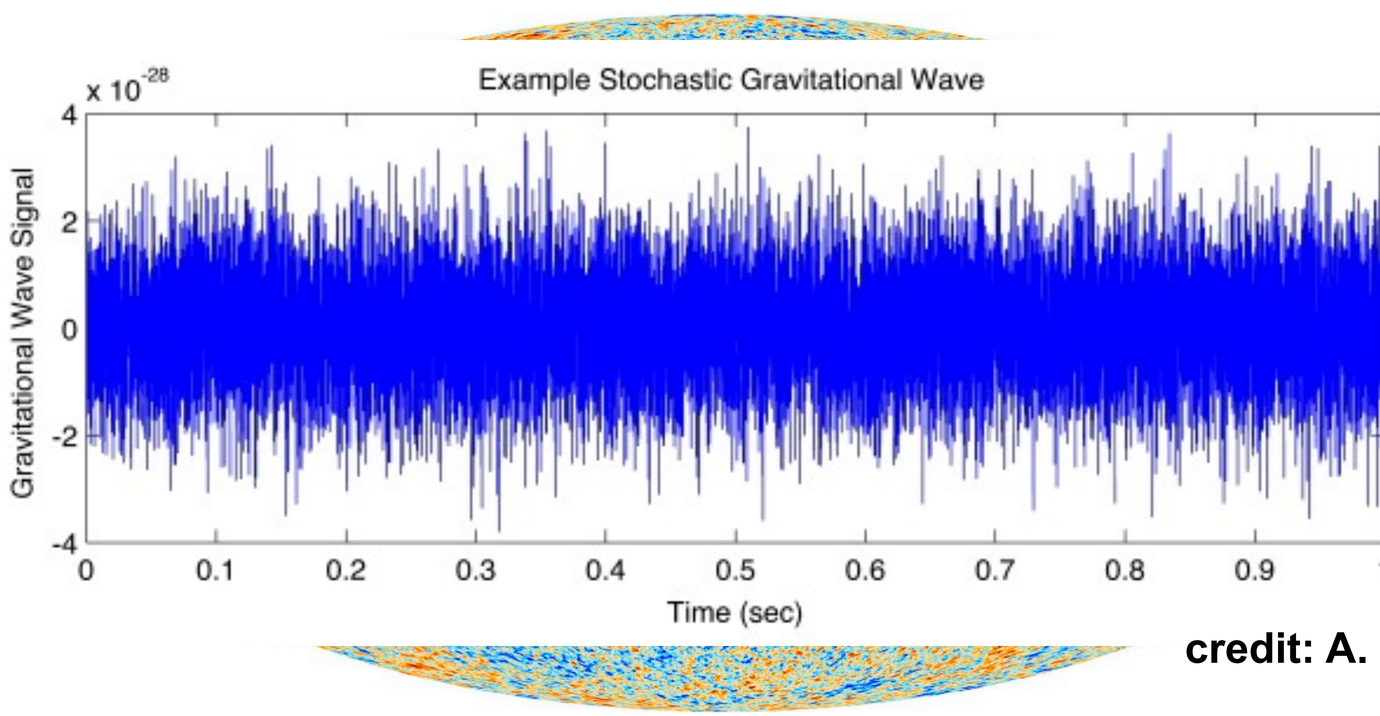
Credit: Casey Reed, Penn State

Stochastic Background

Continuous Sources

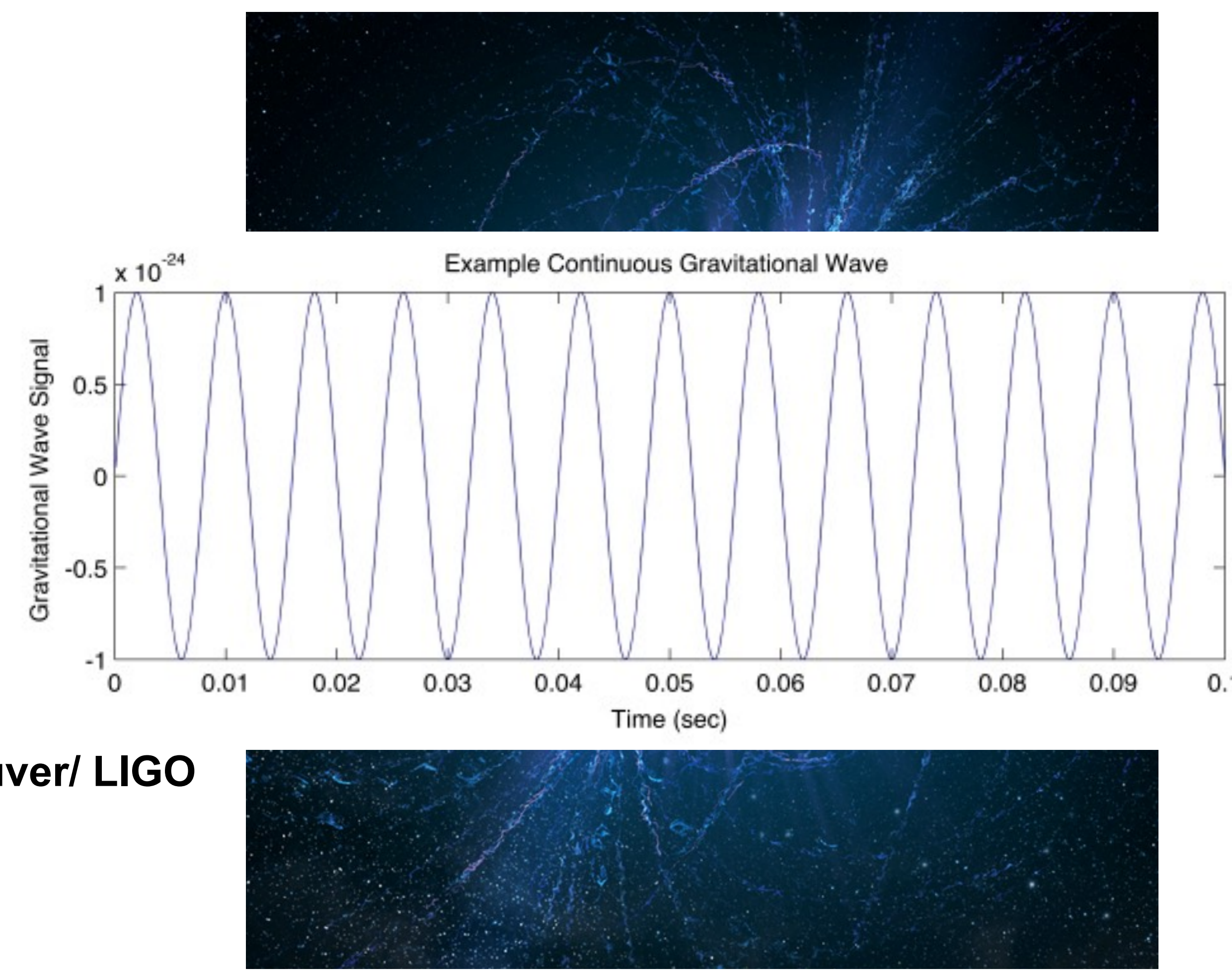
# The Astrophysical Sources of GWs (2)

## non-Transient sources




credit: A. Stuver/ LIGO

Credit: Plank Collaboration

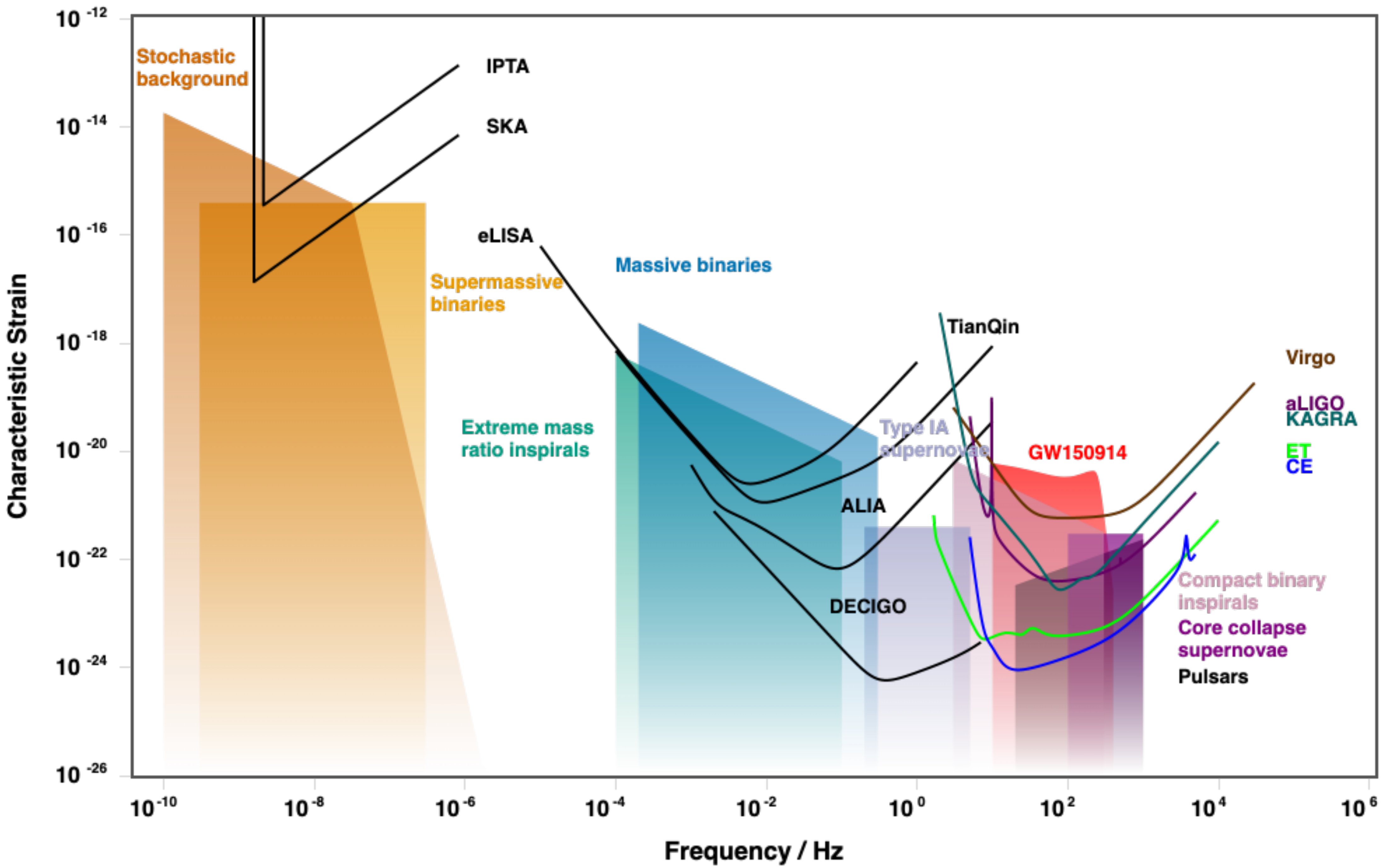



Credit: Casey Reed, Penn State

## Stochastic Background

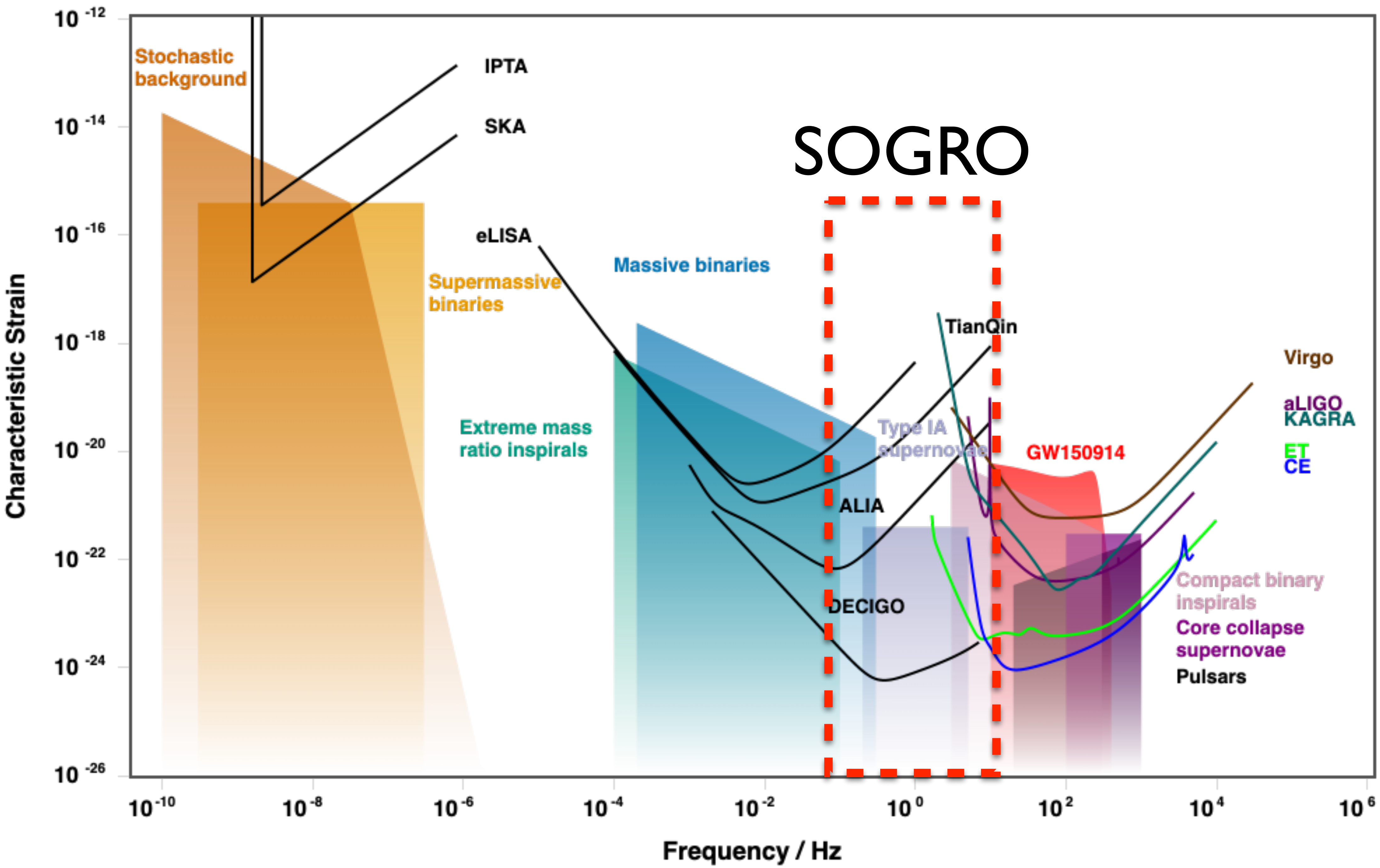
## Continuous Sources

# Detector Bands



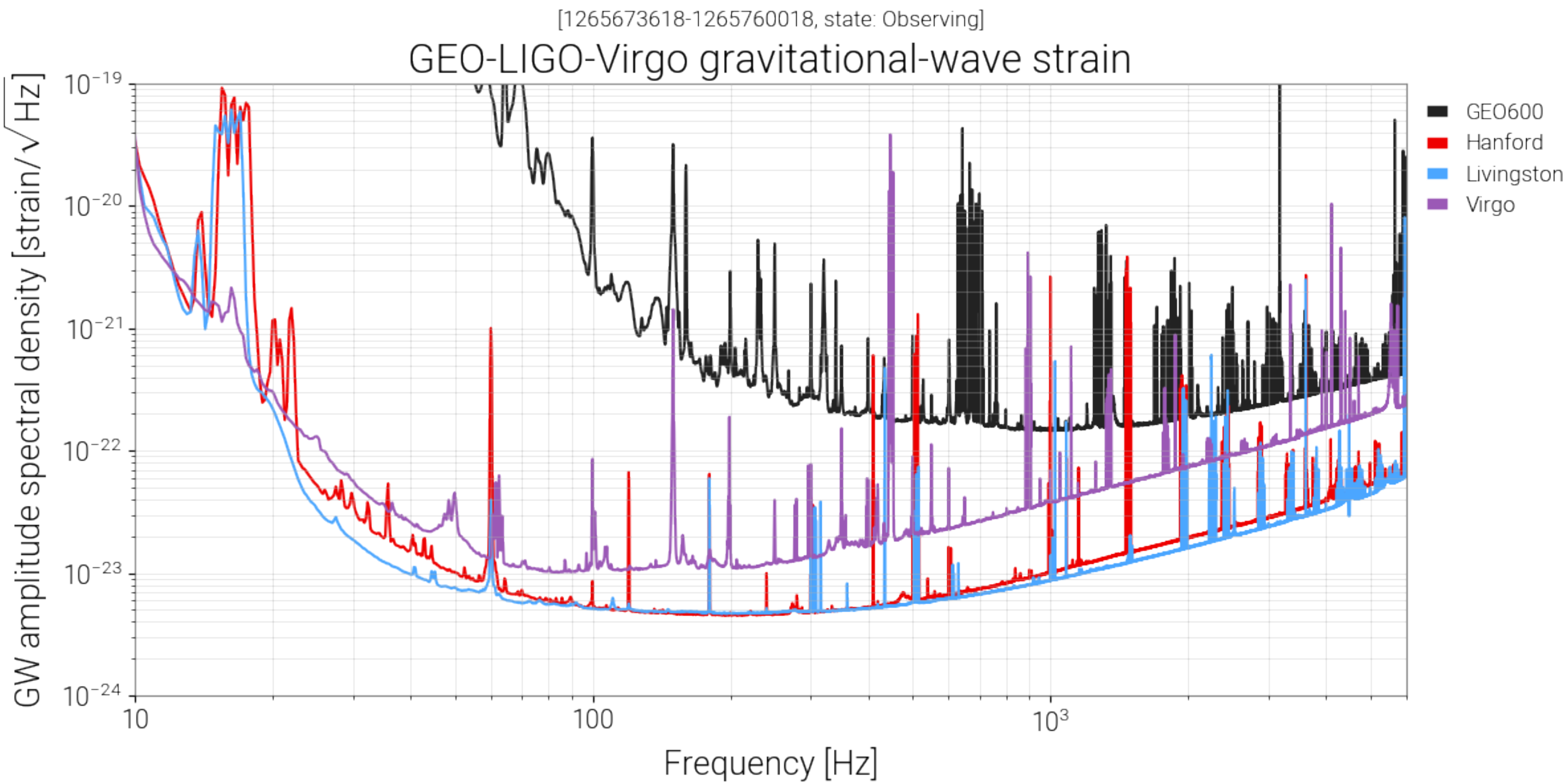

http://gwplotter.com/ by Moore, Cole, and Berry

# Detector Bands

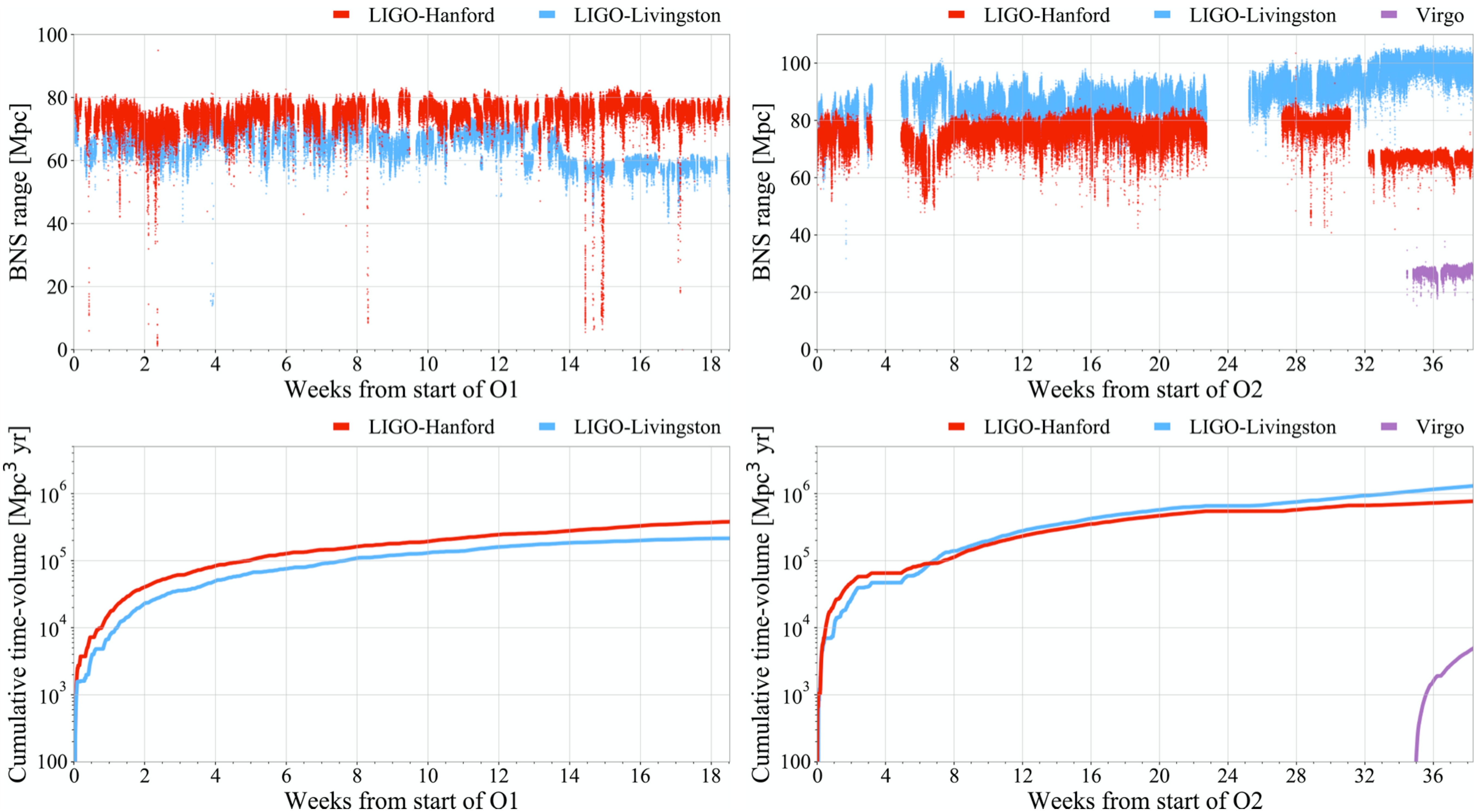



http://gwplotter.com/ by Moore, Cole, and Berry

# Detector Sensitivity

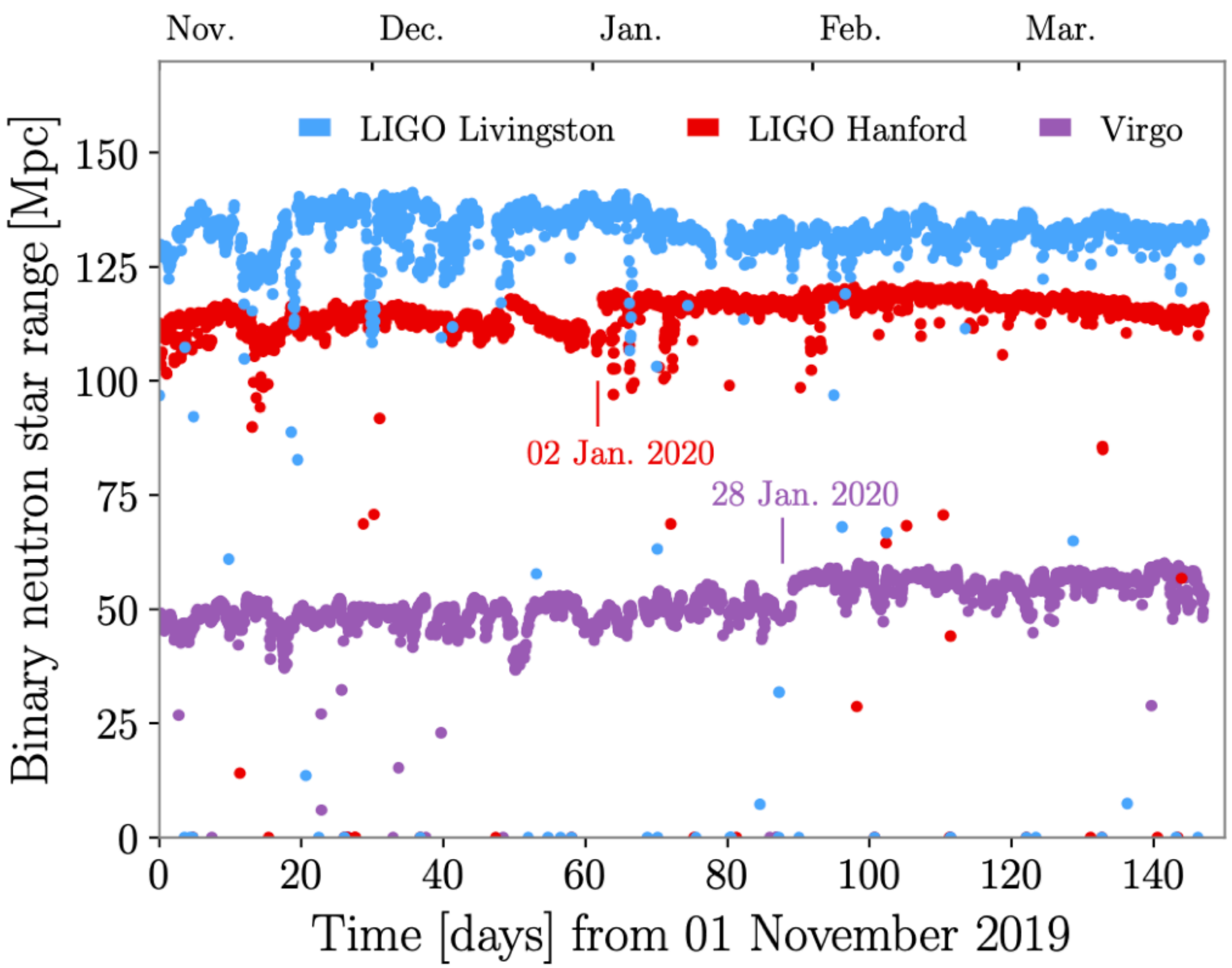


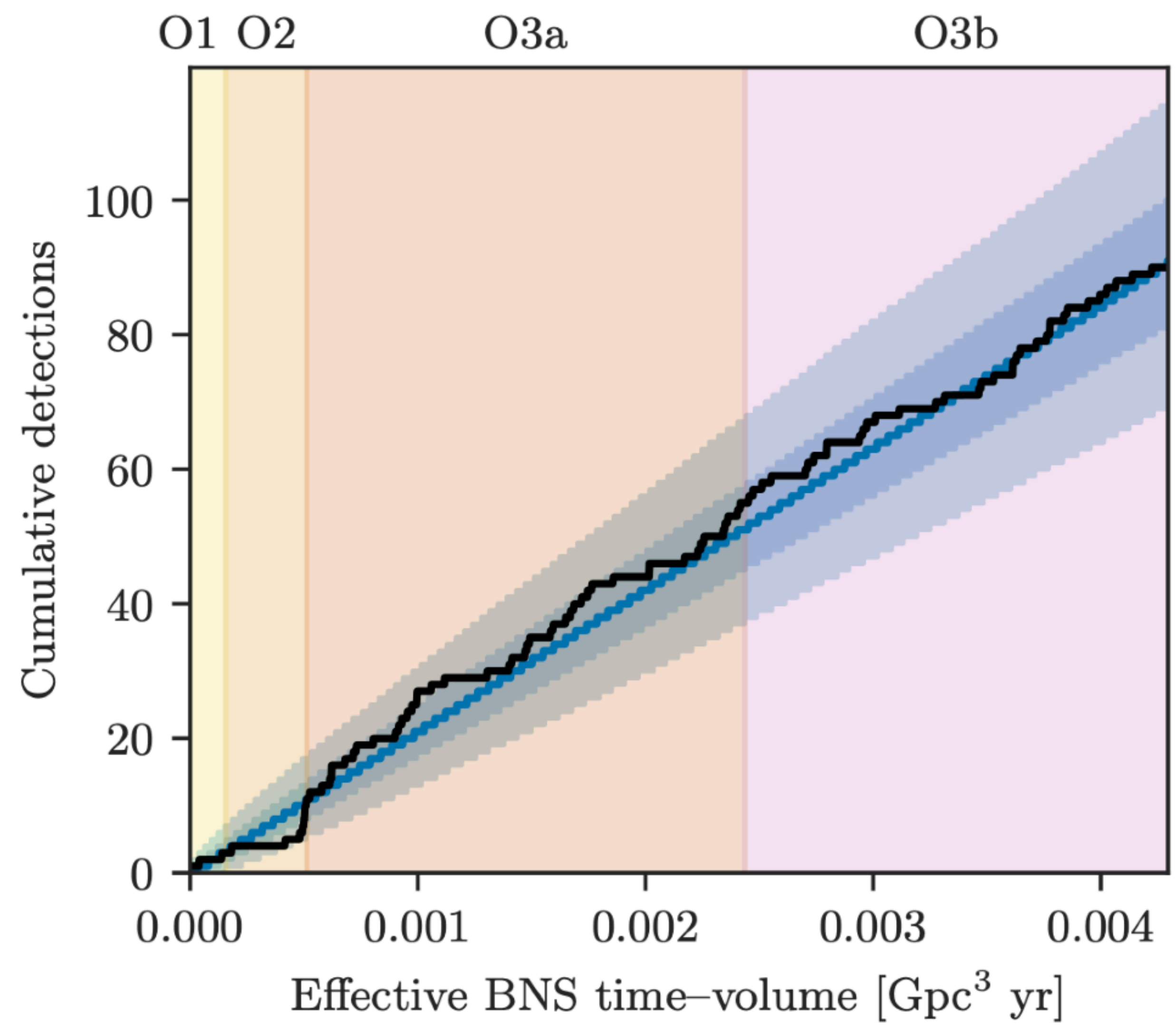
# BNS range and cumulative Time-Volume



# GW events

1. 90 Confident Events : p_astro > 0.5

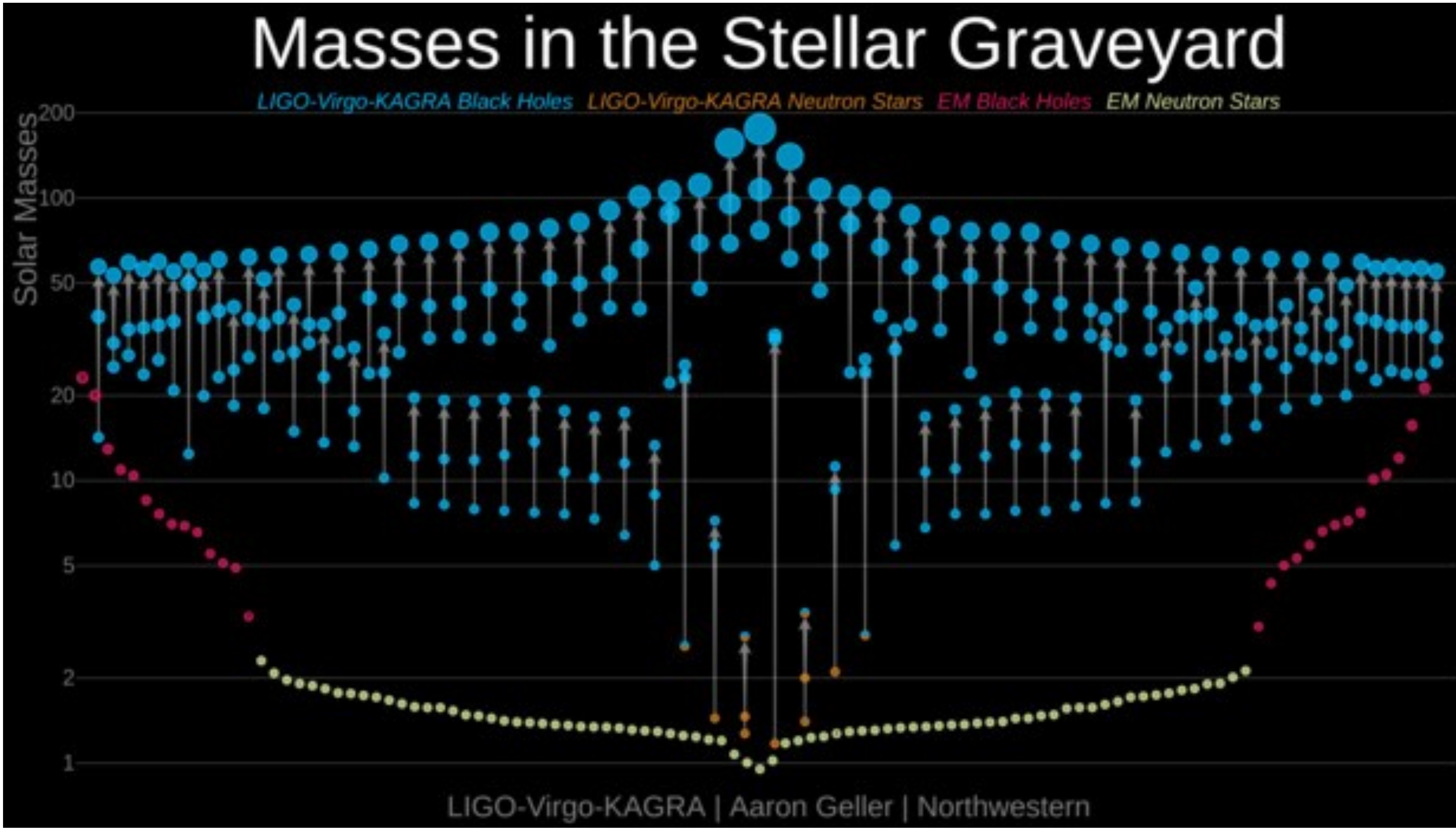



O3

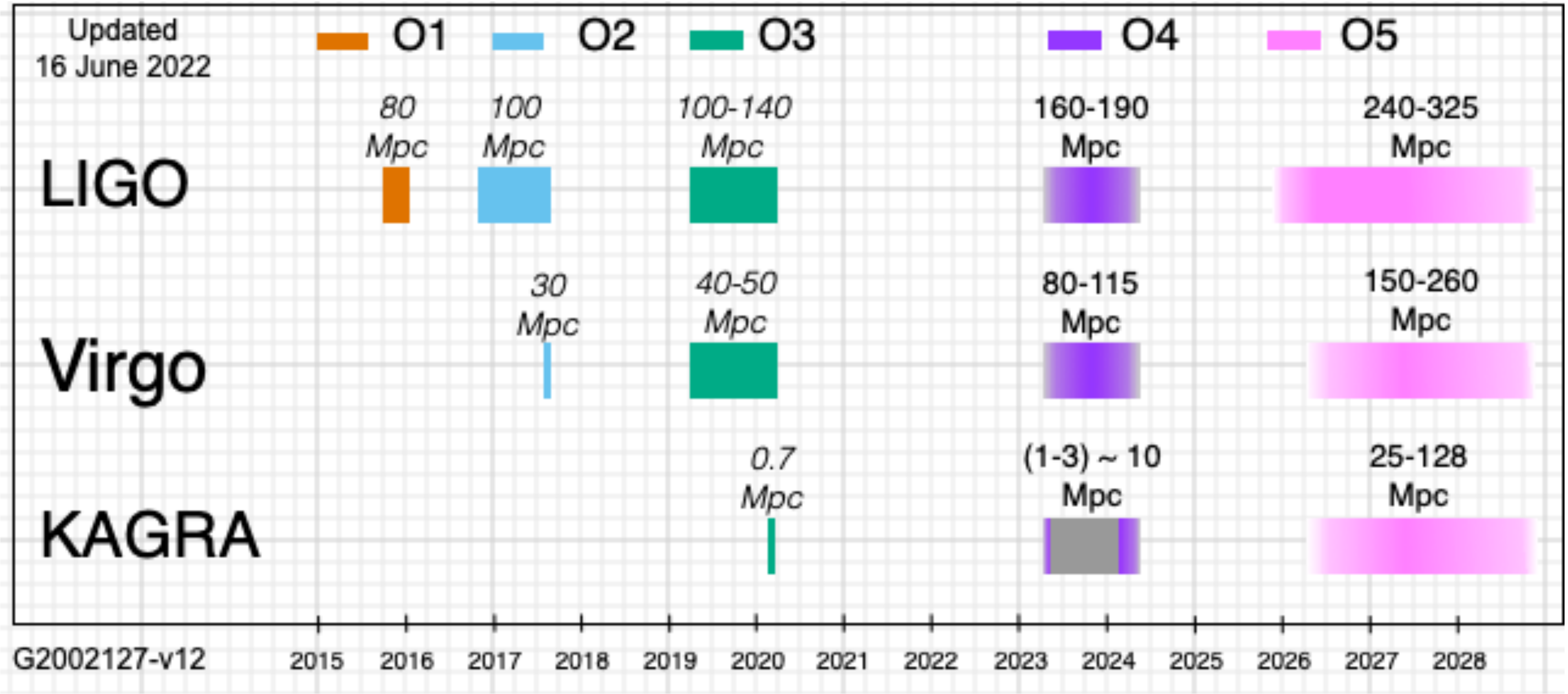
arxiv:2111.03606

# GWTC-3



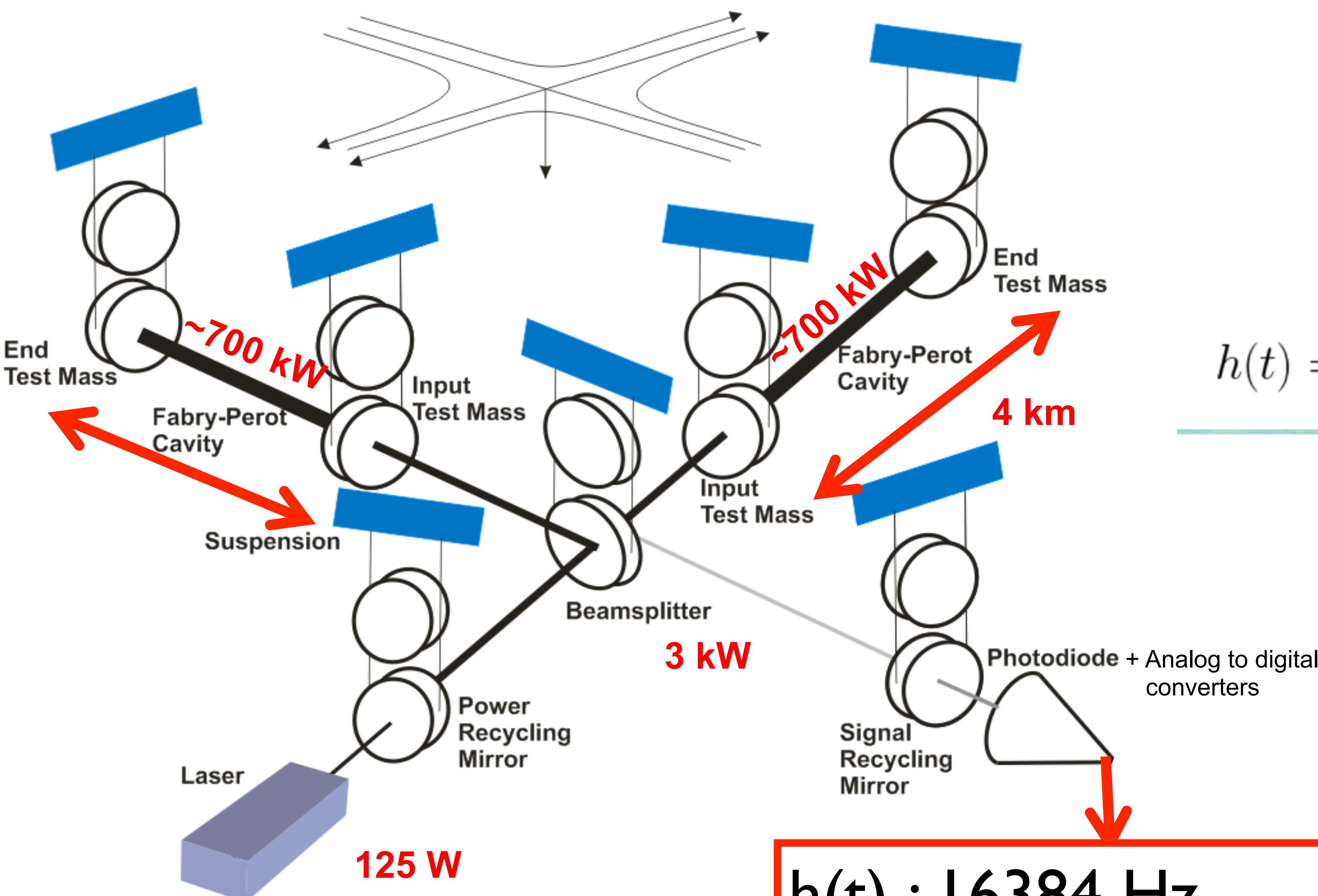
Masses in the Stellar Graveyard
LIGO-Virgo-KAGRA Black Holes
LIGO-Virgo-KAGRA Neutron Stars
EM Black Holes
EM Neutron Stars
Solar Masses
200
100
50
20
10
5
2
1
LIGO-Virgo-KAGRA | Aaron Geller | Northwestern

# Observing plan




https://observing.docs.ligo.org/plan/

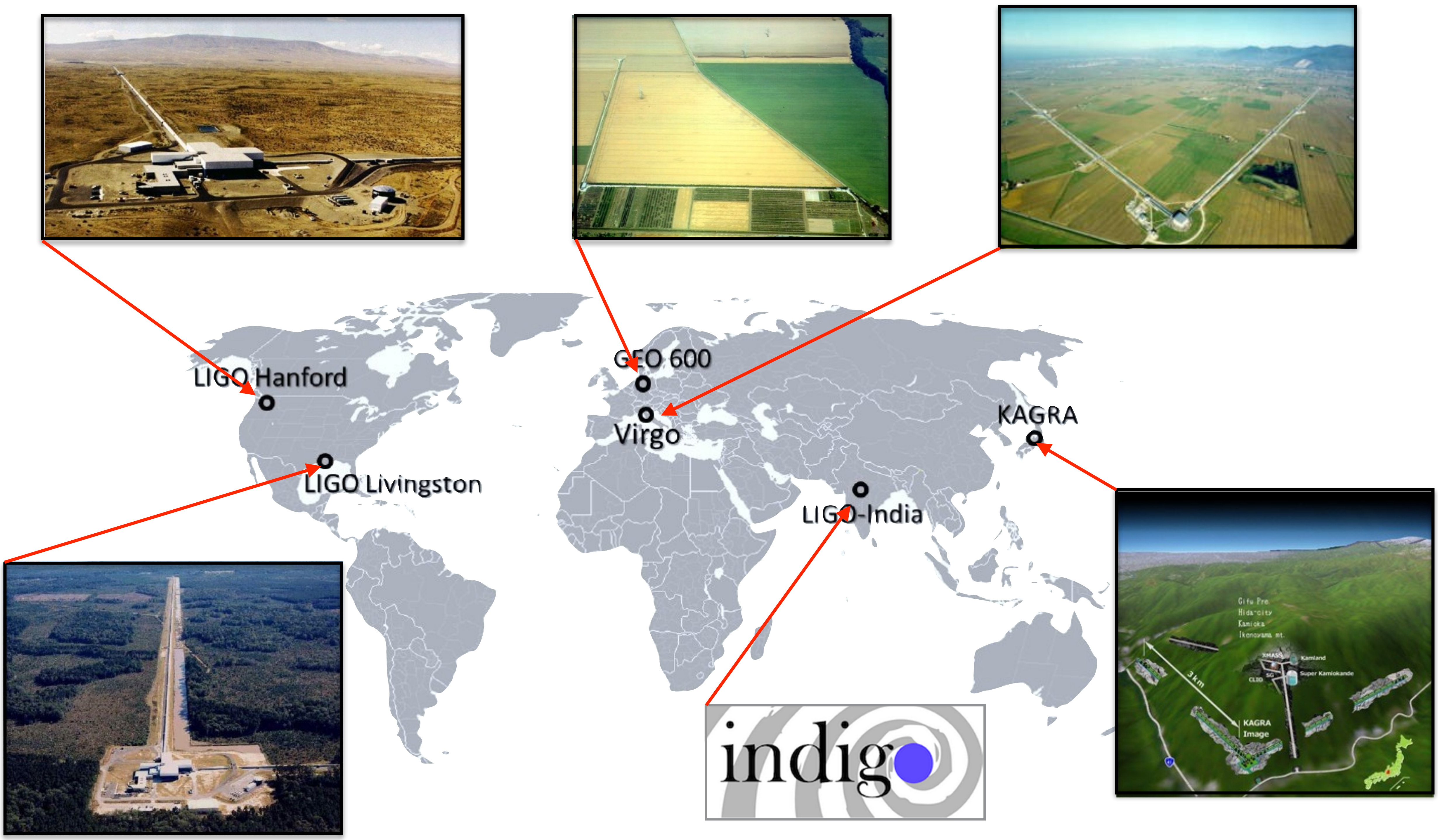
# Laser Interferometer



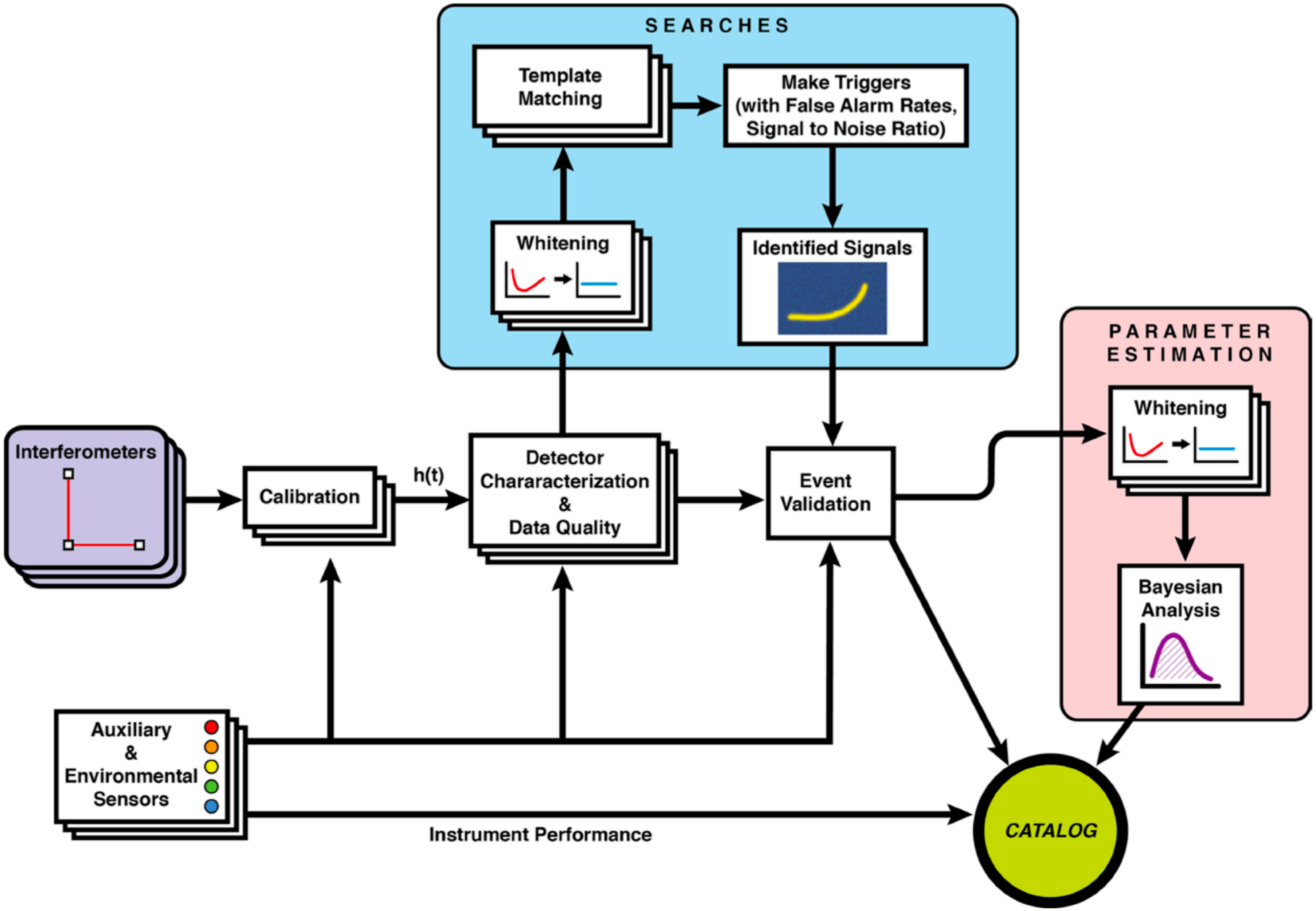

$$h(t) = \frac{\delta L_x(t) - \delta L_y(t)}{L}$$


h(t) : 16384 Hz
GWOSC h(t) : 4096 Hz

# Gravitational-Wave Detection Network



# Data Flow


SEARCHES
Template Matching
Make Triggers (with False Alarm Rates, Signal to Noise Ratio)
Whitening
Identified Signals
PARAMETER ESTIMATION
Whitening
Interferometers
Calibration
h(t)
Detector Chararacterization & Data Quality
Event Validation
Bayesian Analysis
Auxiliary & Environmental Sensors
Instrument Performance
CATALOG

# GW Open Science Center

https://www.gw-opensience.org/about/

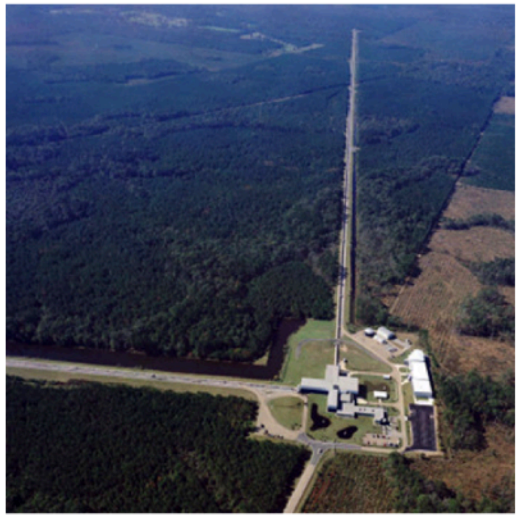
HEP - INSPIRE-HEP | LLO-summary | aLIGO LLO Logbook | LSC and LSC/Virg... | KAGRA - JGW Wiki | KAGRA/Subgroup... | K1 Summary | KGWG Wiki HOME | NIMS ResNote Wiki | set_parameter | Fwd: https://arxiv....

## Gravitational Wave Open Science Center

Data | Software | Online Tools | About GWOSC

**The Gravitational Wave Open Science Center provides data from gravitational-wave observatories, along with access to tutorials and software tools.**

LIGO Hanford Observatory, Washington
(Credits: C. Gray)



LIGO Livingston Observatory, Louisiana
(Credits: J. Giaime)

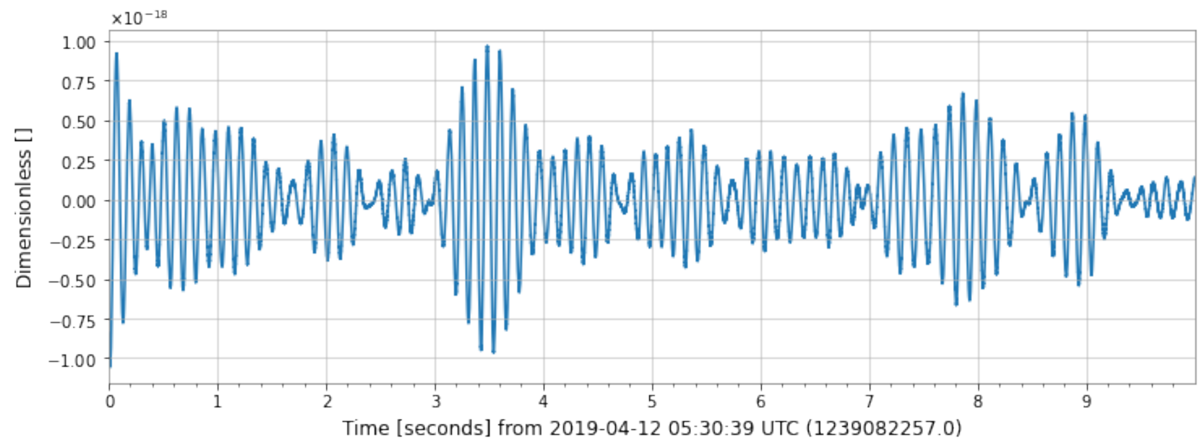
Virgo detector, Italy
(Credits: Virgo Collaboration)

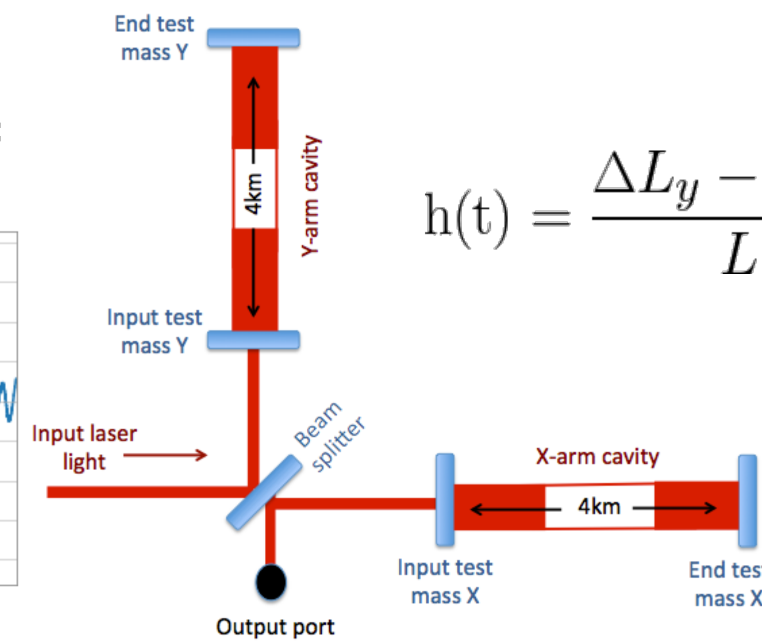
**GW200105 and GW200115 event data available!**

**New Event Portal Query Page!**

# Open Data Products

**Strain, h(t)**

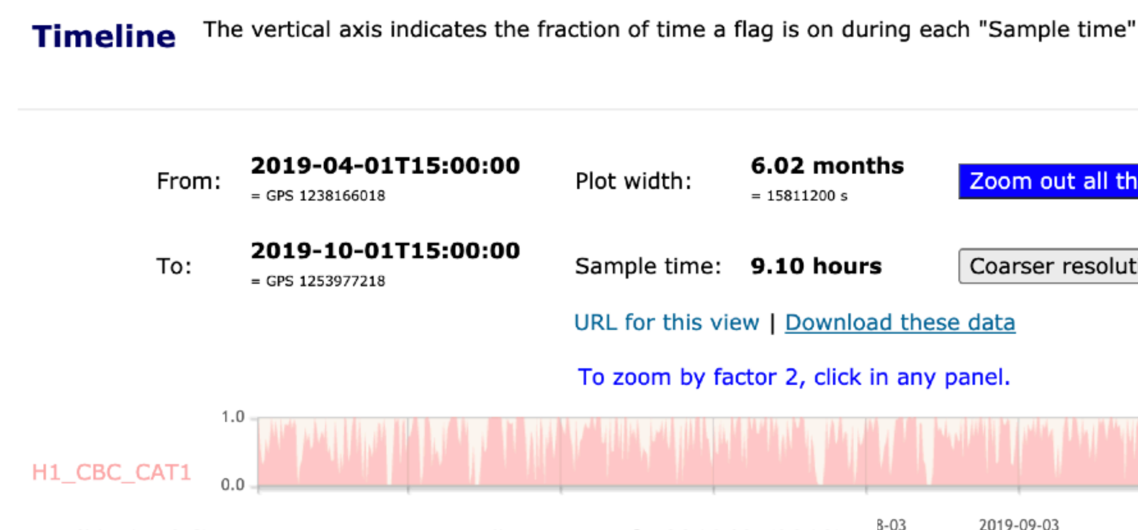


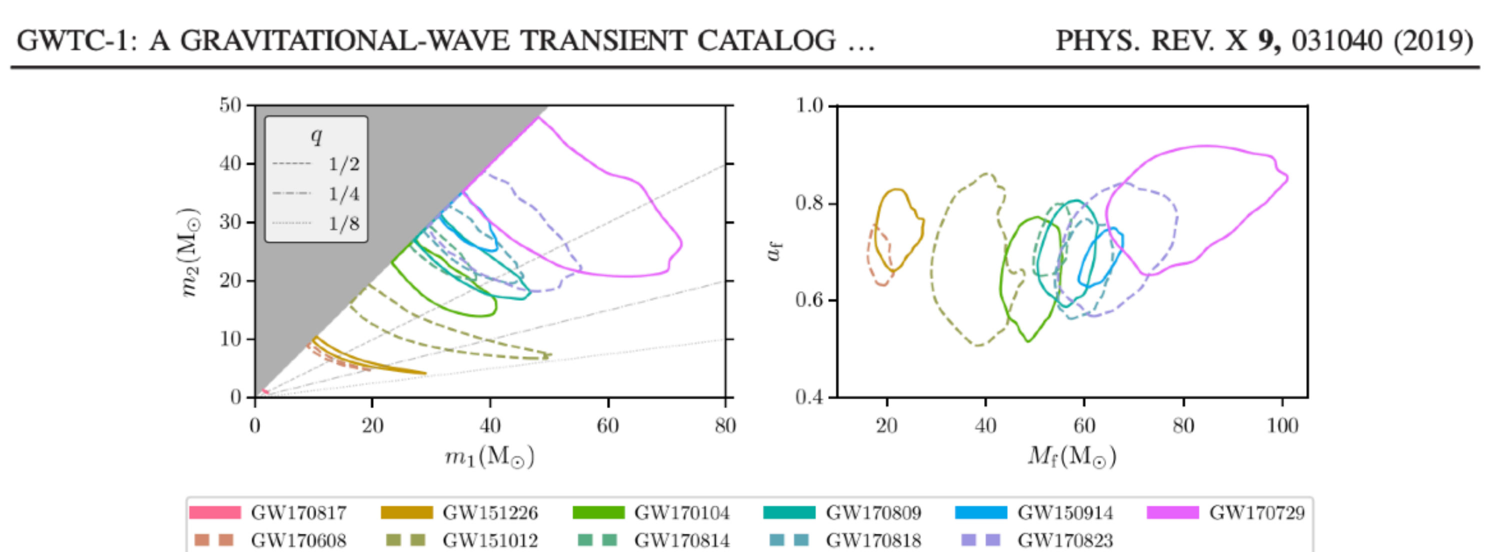

$$h(t) = \frac{\Delta L_y - \Delta L_x}{L}$$

**Segments (Timelines)**

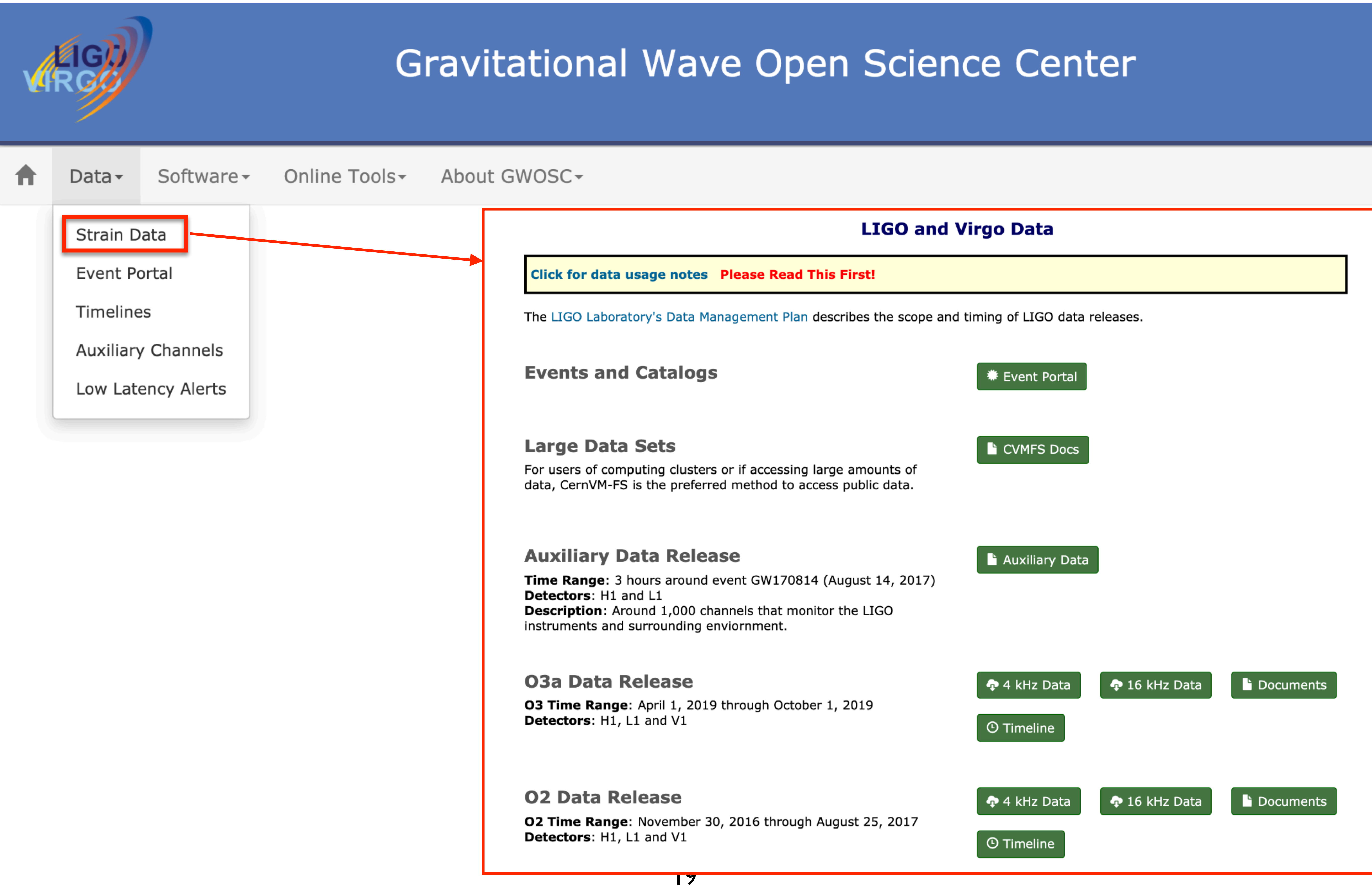
| | | |
|---|---|---|
| 1238166018 | 1238170549 | 4531 |
| 1238170954 | 1238172929 | 1975 |
| 1238172987 | 1238196793 | 23806 |
| 1238198080 | 1238215142 | 17062 |
| 1238227174 | 1238232306 | 5132 |
| 1238235751 | 1238239667 | 3916 |
| 1238245131 | 1238252455 | 7324 |
| 1238290519 | 1238292971 | 2452 |
| 1238292998 | 1238308337 | 15339 |
| 1238318337 | 1238344470 | 26133 |
| 1238350866 | 1238357139 | 6273 |
| 1238368387 | 1238374369 | 5982 |




**Analysis Results**




Gravitational Wave Open Science Center
Data
Software
Online Tools
About GWOSC
Strain Data
Event Portal
Timelines
Auxiliary Channels
Low Latency Alerts
The Gravitational Wave Open Science Center provides dat
observatories, along with access to tutorials an
LIGO Hanford Observatory, Washington
(Credits: C. Gray)
LIGO Livingston Observatory, Louisiana
(Credits: J. Giaime)



LIGO VIRGO
Gravitational Wave Open Science Center
Data
Software
Online Tools
About GWOSC
Strain Data
Event Portal
Timelines
Auxiliary Channels
Low Latency Alerts
LIGO and Virgo Data
Click for data usage notes
Please Read This First!
The LIGO Laboratory's Data Management Plan describes the scope and timing of LIGO data releases.
Events and Catalogs
Event Portal
Large Data Sets
For users of computing clusters or if accessing large amounts of data, CernVM-FS is the preferred method to access public data.
CVMFS Docs
Auxiliary Data Release
Time Range: 3 hours around event GW170814 (August 14, 2017)
Detectors: H1 and L1
Description: Around 1,000 channels that monitor the LIGO instruments and surrounding enviornment.
Auxiliary Data
O3a Data Release
O3 Time Range: April 1, 2019 through October 1, 2019
Detectors: H1, L1 and V1
4 kHz Data
16 kHz Data
Documents
Timeline
O2 Data Release
O2 Time Range: November 30, 2016 through August 25, 2017
Detectors: H1, L1 and V1
4 kHz Data
16 kHz Data
Documents
Timeline

# Gravitational Wave Open Science Center

Data ▾ Software ▾ Online Tools ▾ About GWOSC ▾

- Strain Data
- Event Portal
- Timelines
- Auxiliary Channels
- Low Latency Alerts

## LIGO and Virgo Data

**Click for data usage notes** **Please Read This First!**

The LIGO Laboratory's Data Management Plan describes the scope and timing of LIGO data releases.

**Events and Catalogs** Event Portal

**Large Data Sets** CVMFS Docs

For users of computing clusters or if accessing large amounts of data, CernVM-FS is the preferred method to access public data.

## Accessing Data Using CernVM-FS

### Overview

It is now possible to access the larger bulk data sets from Observation Runs using CernVM-FS. This distributed file system will allow you to mount the data locally on your computer if you have one of the supported platforms (a number of Linux distributions and MacOS X 10.11 or beyond). Once you have installed and configured CernVM-FS, you will be able to access data from these observation runs as files in subdirectories on your computer.

### CernVM-FS Installation Instructions

Instructions for installing CernVM-FS are available from the IGWN | Computing website located **here**.

Please take special note that the section on **IGWN proprietary data (ligo.osgstorage.org)** only applies to the LIGO/Virgo/KAGRA Collaboration members and not to the general public.

**Return to the Data Page to learn more about Bulk Data Releases..**

# Gravitational Wave Open Science Center

Data | Software | Online Tools | About GWOSC

Strain Data

## O3a Data Release

**O3 Time Range**: April 1, 2019 through October 1, 2019
**Detectors**: H1, L1 and V1

4 kHz Data | 16 kHz Data | Documents | Timeline

## O2 Data Release

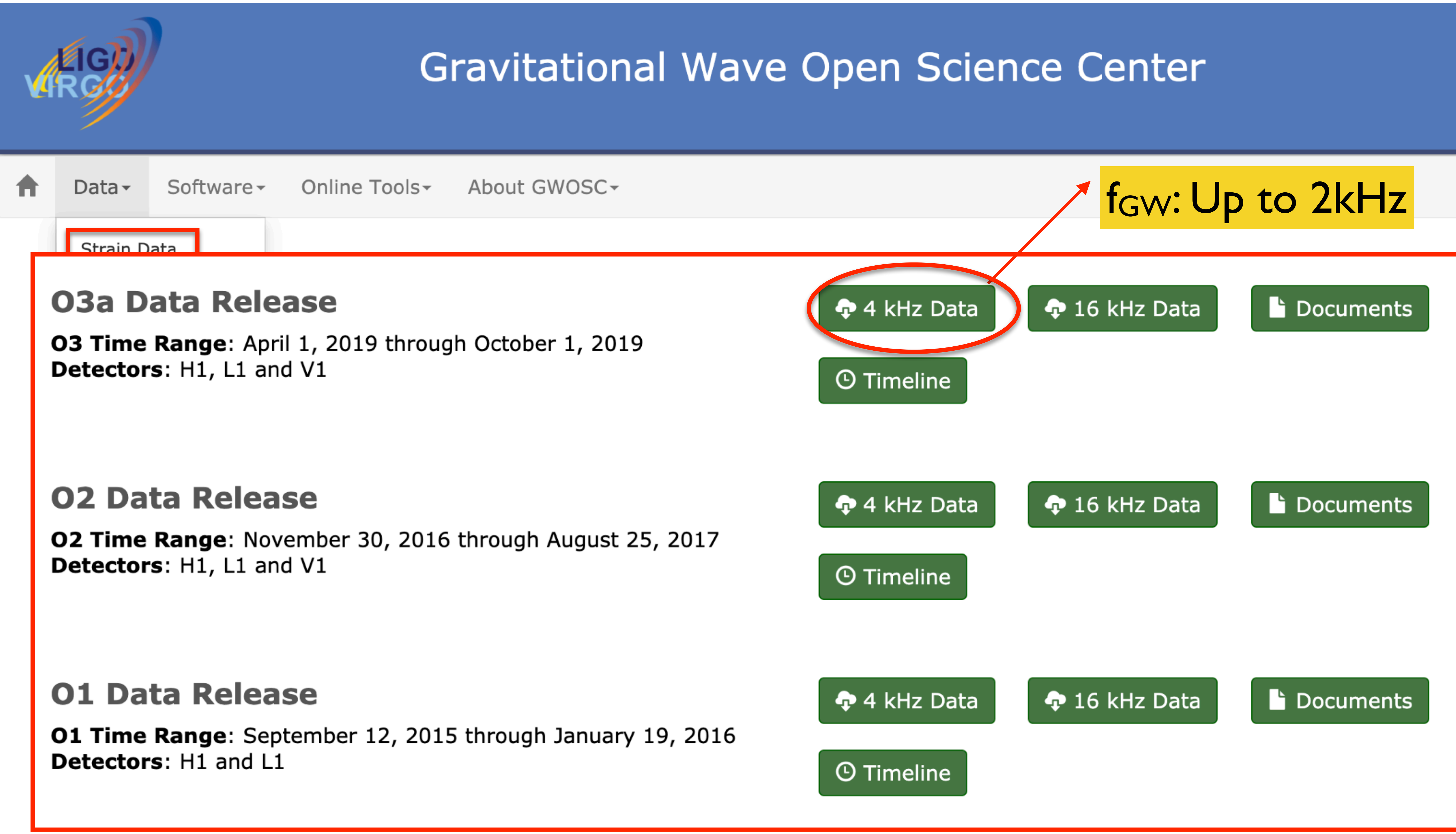
**O2 Time Range**: November 30, 2016 through August 25, 2017
**Detectors**: H1, L1 and V1

4 kHz Data | 16 kHz Data | Documents | Timeline

## O1 Data Release

**O1 Time Range**: September 12, 2015 through January 19, 2016
**Detectors**: H1 and L1

4 kHz Data | 16 kHz Data | Documents | Timeline


Gravitational Wave Open Science Center
LIGO
VIRGO
Data
Software
Online Tools
About GWOSC
Strain Data
O3a Data Release
O3 Time Range: April 1, 2019 through October 1, 2019
Detectors: H1, L1 and V1
4 kHz Data
16 kHz Data
Documents
Timeline
O2 Data Release
O2 Time Range: November 30, 2016 through August 25, 2017
Detectors: H1, L1 and V1
4 kHz Data
16 kHz Data
Documents
Timeline
O1 Data Release
O1 Time Range: September 12, 2015 through January 19, 2016
Detectors: H1 and L1
4 kHz Data
16 kHz Data
Documents
Timeline
fGW: Up to 2kHz

# The O1 Data Release

**Click for data usage notes** **Please Read This First!**

**Run Overview**

- O1 dates: 2015 Sep 12th 0:00 UTC (GPS 1126051217) to 2016 Jan 19 16:00 UTC (GPS 1137254417)
- Data is available from two detectors, H1 and L1 (Virgo data was not collected during O1)
- The O1 data set is available at the original 16 KHz and the downsampled 4KHz sample rates.
- This is the first observing run of Advanced LIGO
- We released three events from this run, two confirmed (and one possible) binary black hole mergers

**Get O1 Data**

- Data in the 24 hours around GW150914: H1 | L1
- Query to the 4KHz O1 strain data archive
- Download the md5 checksums for the 4KHz O1 data: All 4KHz HDF5 files | All 4KHz GWF files
- Query to the 16KHz O1 strain data archive
- Download the md5 checksums for the 16KHz O1 data: All 16KHz HDF5 files | All 16KHz GWF files
- Find when data is available
- Query for the livetime, data quality, injections
- Instructions for accessing data on your local file system using **CernVM-FS**

**New to O1 16KHz GWF Files**

The O1 16KHz GWF files (ending with extension .gwf) have new channel names that differ from the standard names used in S5, S6 and O1 4KH z GWF files.

### Hardware Injections

**The O1 data set contains simulated astrophysical signals, known as hardware injections, used for testing and calibration.** For an example, see the Find a Hardware Injection Tutorial. For complete documentation, see:

- O1 Hardware Injections Page

Segment lists of times that do NOT have hardware injections; each line of the file is GPS start time, GPS end time, and the difference of those.

- NO_CBC_HW segment lists: H1 | L1
- NO_BURST_HW segment lists: H1 | L1
- The NO_DETCHAR_HW segment lists: H1 | L1
- The NO_CW_HW segment lists: H1 | L1

### Instrumental Spectral Lines

A power spectral density of LIGO data typically shows a number of spectral lines. Many of these are associated with known instrumental resonances. The O1 instrumental spectral lines page gives an explanation, and catalog, of these instrumental lines.

### Representative PSDs

- H1 Representative Sensitivity Plots
- L1 Representative Sensitivity Plots
- Plots of daily sensitivity available on the archived detector status pages

### Technical Details

GWOSC 4KHz strain data have been **repackaged** and **downsampled** from 16384 Hz to 4096 Hz. Advanced LIGO data are not calibrated or valid below 10 Hz or above 5 kHz, and the data sampled at 4096 Hz are not valid above 2 kHz. In most searches for astrophysical sources, data below 20 Hz are not used because the noise is too high. More detailed information about the data set can be seen on the **Technical Details** page.

### Acknowledge

To acknowledge the use of O1 data, see the Acknowledgement Page.
Data Set DOI: **https://doi.org/10.7935/K57P8W9D**

---

**Revision History**

- **Aug 14, 2018:** Added links to "Representative PSDs"

# GWTC-1 Documentation

A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs, 2015-2017.

### Description

This catalog is described in: GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs. Parameter values appear in Tables I, II, and III.

The table of marginal triggers includes triggers from O1 and O2 that passed the initial threshold of a FAR less than one per thirty days in at least one of the two matched-filter searches, but were not assigned a probability of astrophysical origin of more than 50% by either pipeline.

### Strain Data

## Confident Detections

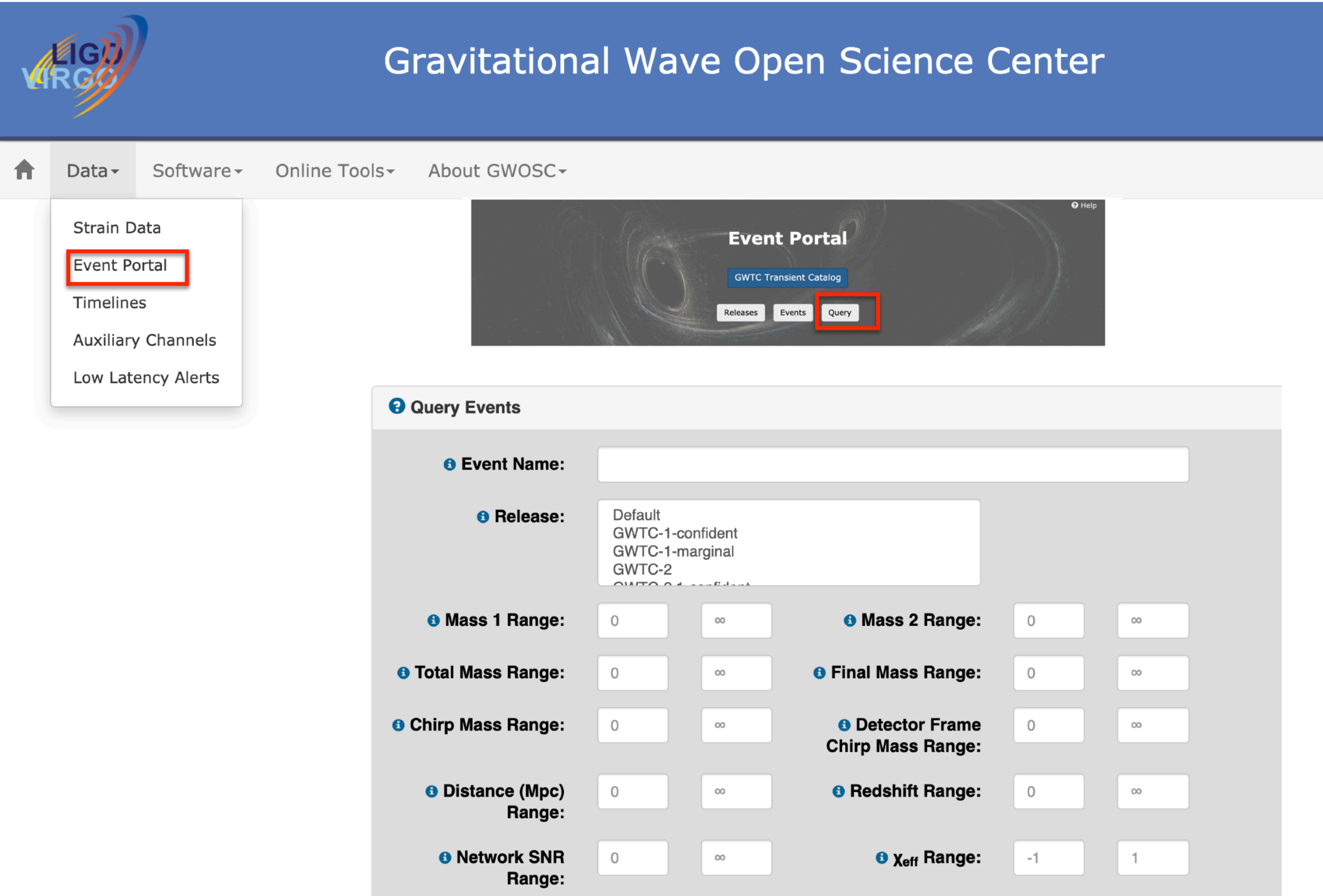
## Marginal Triggers

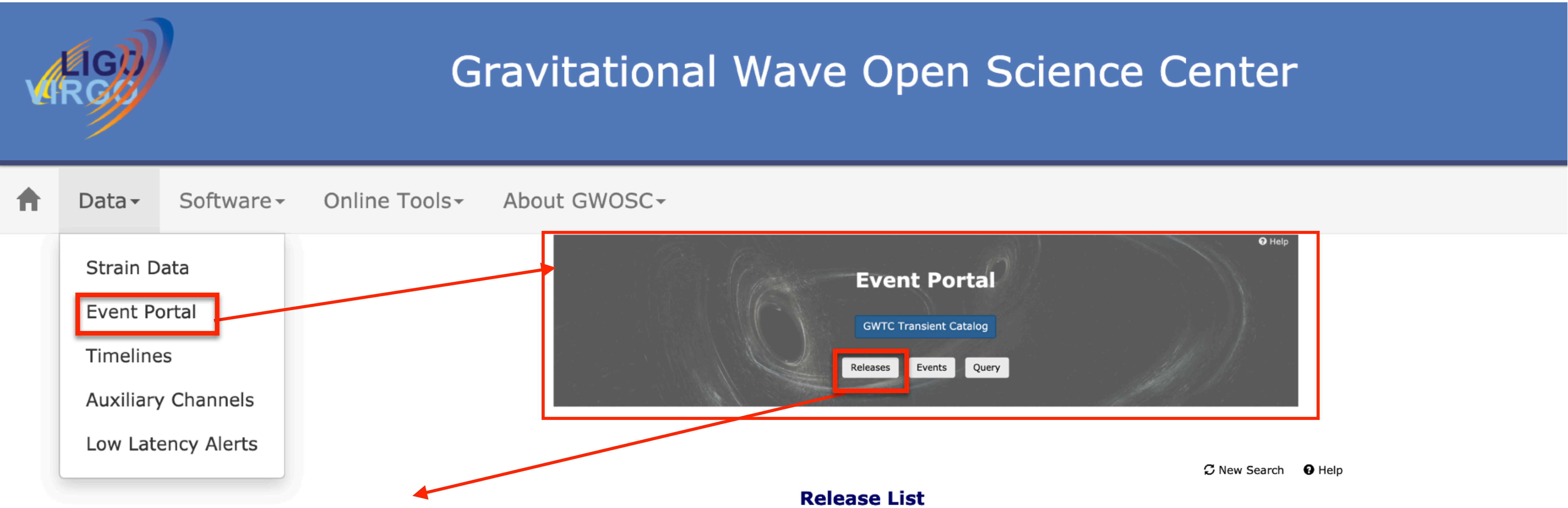
### Associated Data Products

- **Catalog Paper and Figures:** P1800307
- **Parameter Estimation Samples:** P1800370
- **Skymaps (source localization):** P1800381
- **PSDs (noise model power spectra):** P1900011
- **Calibration uncertainty:** P1900040
- **Glitch Subtraction for GW170817:** T1700406
- **Rates and Populations:** P1800324
- **Search Pipeline Trigger Data:** P1900392
- **Audio Files:** GWTC-1 Audio | Sounds of Spacetime
- **GCN Notices:** GCN Notices
- **GCN Circulars:** GCN Circulars

  GW150914 | GW151012 | GW151226 | GW170104 | GW170608

  GW170809 | GW170814 | GW170817 | GW170823


Gravitational Wave Open Science Center
Data
Software
Online Tools
About GWOSC
Strain Data
Event Portal
Timelines
Auxiliary Channels
Low Latency Alerts
Help
Event Portal
GWTC Transient Catalog
Releases
Events
Query
Query Events
Event Name:
Release:
Default
GWTC-1-confident
GWTC-1-marginal
GWTC-2
Mass 1 Range:
Mass 2 Range:
Total Mass Range:
Final Mass Range:
Chirp Mass Range:
Detector Frame Chirp Mass Range:
Distance (Mpc) Range:
Redshift Range:
Network SNR Range:
χeff Range:
0
∞
-1
1



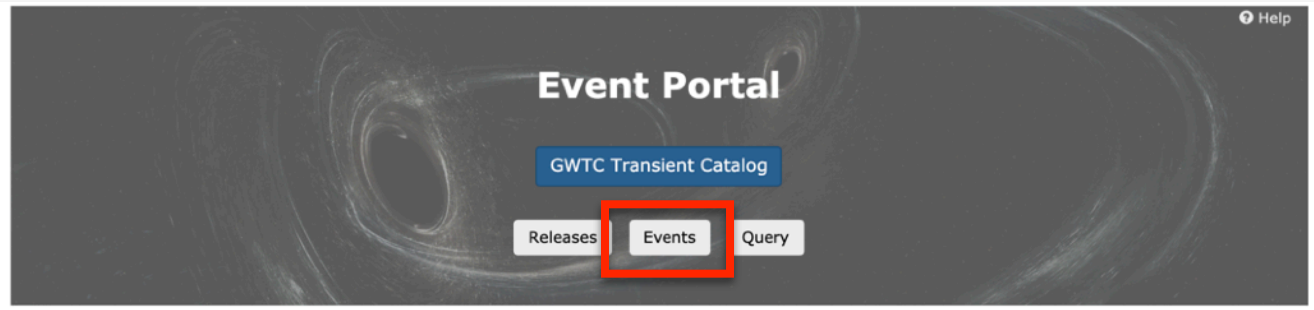

## Release List

| Release Name | Description |
|---|---|
| **GWTC-1-confident** | Confident detections from "GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs." Additional data products, including PE samples and skymaps, are linked from the documentation at https://doi.org/10.7935/82H3-HH23 |
| **GWTC-1-marginal** | Marginal triggers from "GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs." Additional data products are linked from the documentation at https://doi.org/10.7935/82H3-HH23 |
| **GWTC-2** | Events from the O3A observation run of LIGO and Virgo, as described in the GWTC-2 catalog paper. These events are also included in a cumulative list of all GWTC events published to date. Details and additional data products are linked from the documentation page. |
| **Initial_LIGO_Virgo** | Event data releases from initial LIGO and Virgo, 2005 - 2010. No astrophysical detections were made during this period. |
| **O1_O2-Preliminary** | Notable events in O1 and O2 initially published before the GWTC-1 catalog. These data releases may contain preliminary versions of data quality segments and calibration. For additional documentation released at the time of publication, see the "reference" link for each event. Updated information for these events may be found in the GWTC-1 catalog. |
| **O3_Discovery_Papers** | Notable events in O3 initially published outside of main catalogs. Associated data releases may contain preliminary versions of data quality segments and calibration. See documentation page for additional notes. |
| **O3_IMBH_marginal** | O3 IMBH marginal candidates associated with O3 observation run for LIGO and Virgo |

# Gravitational Wave Open Science Center

Data ▾ Software ▾ Online Tools ▾ About GWOSC ▾

- Strain Data
- Event Portal
- Timelines
- Auxiliary Channels
- Low Latency Alerts




## Event List

**GWTC-1-confident**

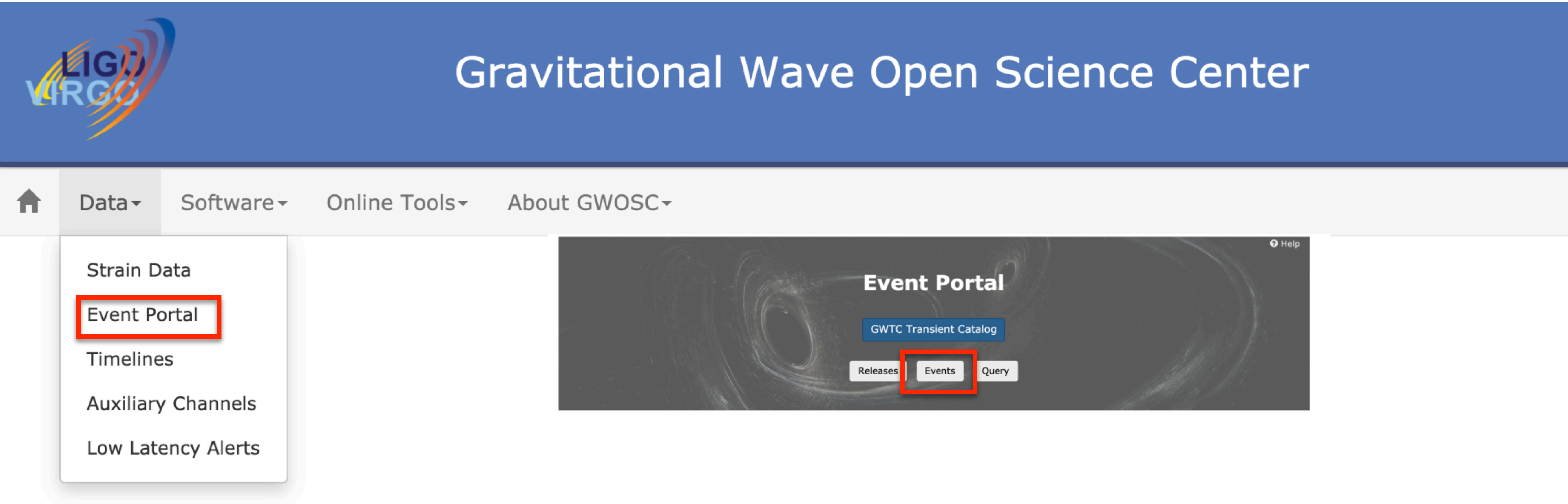
Confident detections from "GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs." Additional data products, including PE samples and skymaps, are linked from the documentation at https://doi.org/10.7935/82H3-HH23

- Toggle columns on/off with widget at right
- Click an event name for more information

SORT: GPS ↓

| Name | Version | Release | GPS ↓ | Mass 1 ($M_\odot$) | Mass 2 ($M_\odot$) | Network SNR | Distance (Mpc) | $\chi_{eff}$ | Chirp Mass ($M_\odot$) | False Alarm Rate ($yr^{-1}$) |
|---|---|---|---|---|---|---|---|---|---|---|
| GW170823 | v1 | GWTC-1-confident | 1187529256.5 | $39.5^{+11.2}_{-6.7}$ | $29.0^{+6.7}_{-7.8}$ | 11.5 | $1940^{+970}_{-900}$ | $0.09^{+0.22}_{-0.26}$ | $29.2^{+4.6}_{-3.6}$ | ≤ 1.0e-07 |
| GW170818 | v1 | GWTC-1-confident | 1187058327.1 | $35.4^{+7.5}_{-4.7}$ | $26.7^{+4.3}_{-5.2}$ | 11.3 | $1060^{+420}_{-380}$ | $-0.09^{+0.18}_{-0.21}$ | $26.5^{+2.1}_{-1.7}$ | 4.2e-05 |
| GW170817 | v3 | GWTC-1-confident | 1187008882.4 | $1.46^{+0.12}_{-0.10}$ | $1.27^{+0.09}_{-0.09}$ | 33.0 | $40^{+7}_{-15}$ | $0.00^{+0.02}_{-0.01}$ | $1.186^{+0.001}_{-0.001}$ | ≤ 1.0e-07 |
| GW170814 | v3 | GWTC-1-confident | 1186741861.5 | $30.6^{+5.6}_{-3.0}$ | $25.2^{+2.8}_{-4.0}$ | 15.9 | $600^{+150}_{-220}$ | $0.07^{+0.12}_{-0.12}$ | $24.1^{+1.4}_{-1.1}$ | ≤ 1.0e-07 |
| GW170809 | v1 | GWTC-1-confident | 1186302519.8 | $35.0^{+8.3}_{-5.9}$ | $23.8^{+5.1}_{-5.2}$ | 12.4 | $1030^{+320}_{-390}$ | $0.08^{+0.17}_{-0.17}$ | $24.9^{+2.1}_{-1.7}$ | ≤ 1.0e-07 |
| GW170729 | v1 | GWTC-1-confident | 1185389807.3 | $50.2^{+16.2}_{-10.2}$ | $34.0^{+9.1}_{-10.1}$ | 10.2 | $2840^{+1400}_{-1360}$ | $0.37^{+0.21}_{-0.25}$ | $35.4^{+6.5}_{-4.8}$ | 0.02 |
| GW170608 | v3 | GWTC-1-confident | 1180922494.5 | $11.0^{+5.5}_{-1.7}$ | $7.6^{+1.4}_{-2.2}$ | 14.9 | $320^{+120}_{-110}$ | $0.03^{+0.19}_{-0.07}$ | $7.9^{+0.2}_{-0.2}$ | ≤ 1.0e-07 |
| GW170104 | v2 | GWTC-1-confident | 1167559936.6 | $30.8^{+7.3}_{-5.6}$ | $20.0^{+4.9}_{-4.6}$ | 13.0 | $990^{+440}_{-430}$ | $-0.04^{+0.17}_{-0.21}$ | $21.4^{+2.2}_{-1.8}$ | ≤ 1.0e-07 |
| GW151226 | v2 | GWTC-1-confident | 1135136350.6 | $13.7^{+8.8}_{-3.2}$ | $7.7^{+2.2}_{-2.5}$ | 13.1 | $450^{+180}_{-190}$ | $0.18^{+0.20}_{-0.12}$ | $8.9^{+0.3}_{-0.3}$ | ≤ 1.0e-07 |
| GW151012 | v3 | GWTC-1-confident | 1128678900.4 | $23.2^{+14.9}_{-5.5}$ | $13.6^{+4.1}_{-4.8}$ | 10.0 | $1080^{+550}_{-490}$ | $0.05^{+0.31}_{-0.20}$ | $15.2^{+2.1}_{-1.2}$ | 7.9e-03 |
| GW150914 | v3 | GWTC-1-confident | 1126259462.4 | $35.6^{+4.7}_{-3.1}$ | $30.6^{+3.0}_{-4.4}$ | 24.4 | $440^{+150}_{-170}$ | $-0.01^{+0.12}_{-0.13}$ | $28.6^{+1.7}_{-1.5}$ | ≤ 1.0e-07 |

Download table as: JSON - ASCII - CSV




New Search  Help

## Event List

**GWTC-1-confident**
Confident detections from "GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs." Additional data products, including PE samples and skymaps, are linked from the documentation at https://doi.org/10.7935/82H3-HH23

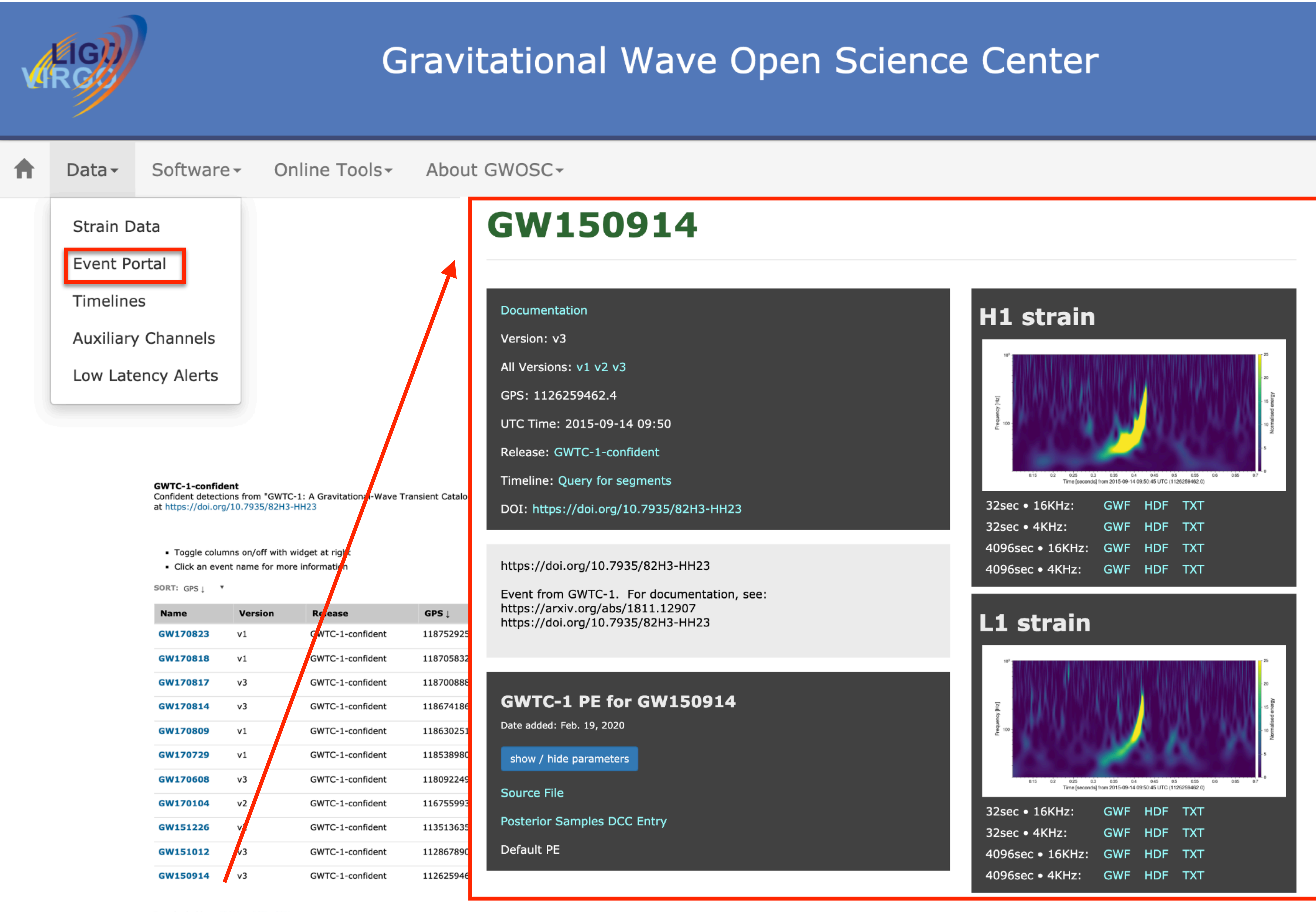
- Toggle columns on/off with widget at right
- Click an event name for more information

SORT: GPS ↓

| Name | Version | Release | GPS ↓ | Mass 1 ($M_\odot$) | Mass 2 ($M_\odot$) | Network SNR | Distance (Mpc) | $\chi_{eff}$ | Chirp Mass ($M_\odot$) | |
|---|---|---|---|---|---|---|---|---|---|---|
| GW170823 | v1 | GWTC-1-confident | 1187529256.5 | $39.5^{+11.2}_{-6.7}$ | $29.0^{+6.7}_{-7.8}$ | 11.5 | $1940^{+970}_{-900}$ | $0.09^{+0.22}_{-0.26}$ | $29.2^{+4.6}_{-3.6}$ | |
| GW170818 | v1 | GWTC-1-confident | 1187058327.1 | $35.4^{+7.5}_{-4.7}$ | $26.7^{+4.3}_{-5.2}$ | 11.3 | $1060^{+420}_{-380}$ | $-0.09^{+0.18}_{-0.21}$ | $26.5^{+2.1}_{-1.7}$ | |
| GW170817 | v3 | GWTC-1-confident | 1187008882.4 | $1.46^{+0.12}_{-0.10}$ | $1.27^{+0.09}_{-0.09}$ | 33.0 | $40^{+7}_{-15}$ | $0.00^{+0.02}_{-0.01}$ | $1.186^{+0.001}_{-0.001}$ | |
| GW170814 | v3 | GWTC-1-confident | 1186741861.5 | $30.6^{+5.6}_{-3.0}$ | $25.2^{+2.8}_{-4.0}$ | 15.9 | $600^{+150}_{-220}$ | $0.07^{+0.12}_{-0.12}$ | $24.1^{+1.4}_{-1.1}$ | |
| GW170809 | v1 | GWTC-1-confident | 1186302519.8 | $35.0^{+8.3}_{-5.9}$ | $23.8^{+5.1}_{-5.2}$ | 12.4 | $1030^{+320}_{-390}$ | $0.08^{+0.17}_{-0.17}$ | $24.9^{+2.1}_{-1.7}$ | |
| GW170729 | v1 | GWTC-1-confident | 1185389807.3 | $50.2^{+16.2}_{-10.2}$ | $34.0^{+9.1}_{-10.1}$ | 10.2 | $2840^{+1400}_{-1360}$ | $0.37^{+0.21}_{-0.25}$ | $35.4^{+6.5}_{-4.8}$ | |
| GW170608 | v3 | GWTC-1-confident | 1180922494.5 | $11.0^{+5.5}_{-1.7}$ | $7.6^{+1.4}_{-2.2}$ | 14.9 | $320^{+120}_{-110}$ | $0.03^{+0.19}_{-0.07}$ | $7.9^{+0.2}_{-0.2}$ | |
| GW170104 | v2 | GWTC-1-confident | 1167559936.6 | $30.8^{+7.3}_{-5.6}$ | $20.0^{+4.9}_{-4.6}$ | 13.0 | $990^{+440}_{-430}$ | $-0.04^{+0.17}_{-0.21}$ | $21.4^{+2.2}_{-1.8}$ | |
| GW151226 | v2 | GWTC-1-confident | 1135136350.6 | $13.7^{+8.8}_{-3.2}$ | $7.7^{+2.2}_{-2.5}$ | 13.1 | $450^{+180}_{-190}$ | $0.18^{+0.20}_{-0.12}$ | $8.9^{+0.3}_{-0.3}$ | |
| GW151012 | v3 | GWTC-1-confident | 1128678900.4 | $23.2^{+14.9}_{-5.5}$ | $13.6^{+4.1}_{-4.8}$ | 10.0 | $1080^{+550}_{-490}$ | $0.05^{+0.31}_{-0.20}$ | $15.2^{+2.1}_{-1.2}$ | 7.9e-03 |
| GW150914 | v3 | GWTC-1-confident | 1126259462.4 | $35.6^{+4.7}_{-3.1}$ | $30.6^{+3.0}_{-4.4}$ | 24.4 | $440^{+150}_{-170}$ | $-0.01^{+0.12}_{-0.13}$ | $28.6^{+1.7}_{-1.5}$ | ≤ 1.0e-07 |

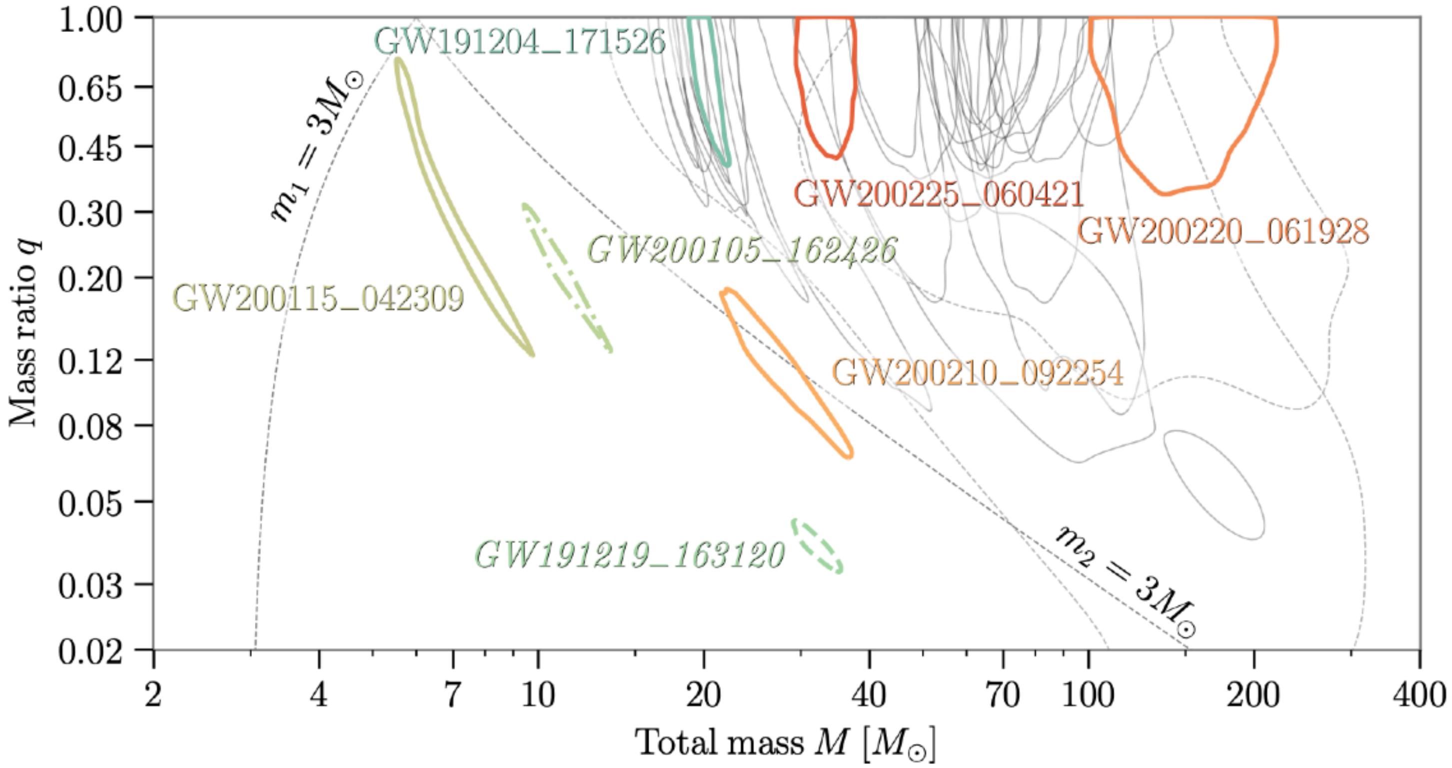
- [x] Version
- [x] Release
- [x] GPS
- [x] Mass 1 (M⊙)
- [x] Mass 2 (M⊙)
- [x] Network SNR
- [x] Distance (Mpc)
- [x] χeff
- [ ] Total Mass (M⊙)
- [x] Chirp Mass (M⊙)
- [ ] Detector Frame Chirp Mass (M⊙)
- [ ] Redshift
- [x] False Alarm Rate (yr-1)
- [ ] Final Mass (M⊙)

Download table as: JSON - ASCII - CSV

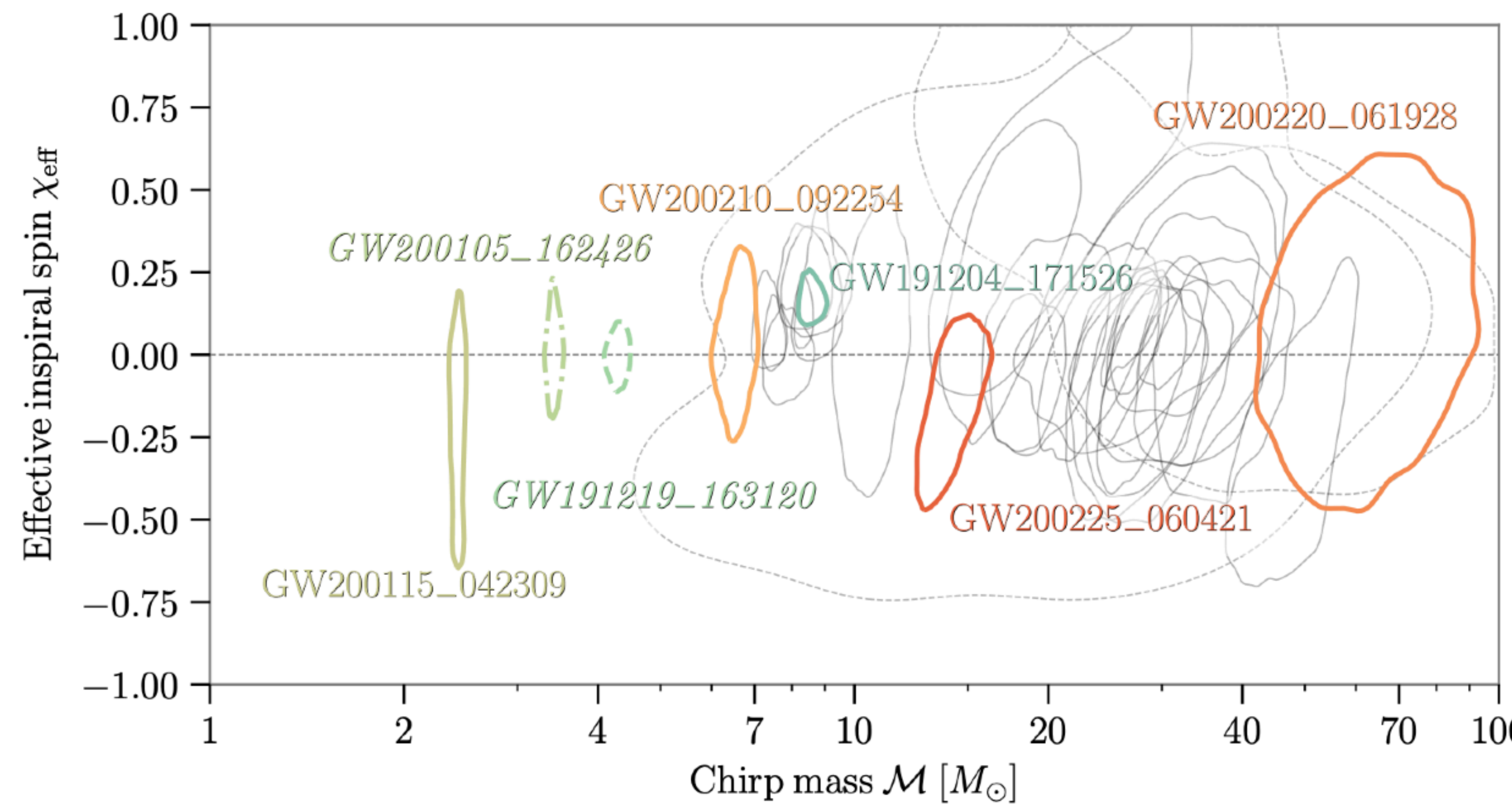


Gravitational Wave Open Science Center
Data
Software
Online Tools
About GWOSC
Strain Data
Event Portal
Timelines
Auxiliary Channels
Low Latency Alerts
GWTC-1-confident
Confident detections from "GWTC-1: A Gravitational-Wave Transient Catalog
at https://doi.org/10.7935/82H3-HH23
Toggle columns on/off with widget at right
Click an event name for more information
SORT: GPS ↓
Name Version Release GPS ↓
GW170823 v1 GWTC-1-confident 118752925
GW170818 v1 GWTC-1-confident 118705832
GW170817 v3 GWTC-1-confident 118700888
GW170814 v3 GWTC-1-confident 118674186
GW170809 v1 GWTC-1-confident 118630251
GW170729 v1 GWTC-1-confident 118538980
GW170608 v3 GWTC-1-confident 118092249
GW170104 v2 GWTC-1-confident 116755993
GW151226 v2 GWTC-1-confident 113513635
GW151012 v3 GWTC-1-confident 112867890
GW150914 v3 GWTC-1-confident 112625946
Download table as: JSON - ASCII - CSV
GW150914
Documentation
Version: v3
All Versions: v1 v2 v3
GPS: 1126259462.4
UTC Time: 2015-09-14 09:50
Release: GWTC-1-confident
Timeline: Query for segments
DOI: https://doi.org/10.7935/82H3-HH23
https://doi.org/10.7935/82H3-HH23
Event from GWTC-1. For documentation, see:
https://arxiv.org/abs/1811.12907
https://doi.org/10.7935/82H3-HH23
GWTC-1 PE for GW150914
Date added: Feb. 19, 2020
show / hide parameters
Source File
Posterior Samples DCC Entry
Default PE
H1 strain
32sec • 16KHz: GWF HDF TXT
32sec • 4KHz: GWF HDF TXT
4096sec • 16KHz: GWF HDF TXT
4096sec • 4KHz: GWF HDF TXT
L1 strain
32sec • 16KHz: GWF HDF TXT
32sec • 4KHz: GWF HDF TXT
4096sec • 16KHz: GWF HDF TXT
4096sec • 4KHz: GWF HDF TXT

# Posterior samples




O3b candidates with p_astro > 0.5




arxiv:2111.03606

# Accessing Large Data

## Accessing Data Using CernVM-FS

### Overview

It is now possible to access the larger bulk data sets from Observation Runs using CernVM-FS. This distributed file system will allow you to mount the data locally on your computer if you have one of the supported platforms (a number of Linux distributions and MacOS X 10.11 or beyond). Once you have installed and configured CernVM-FS, you will be able to access data from these observation runs as files in subdirectories on your computer.

### CernVM-FS Installation Instructions

Instructions for installing CernVM-FS are available from the IGWN | Computing website located **here**.

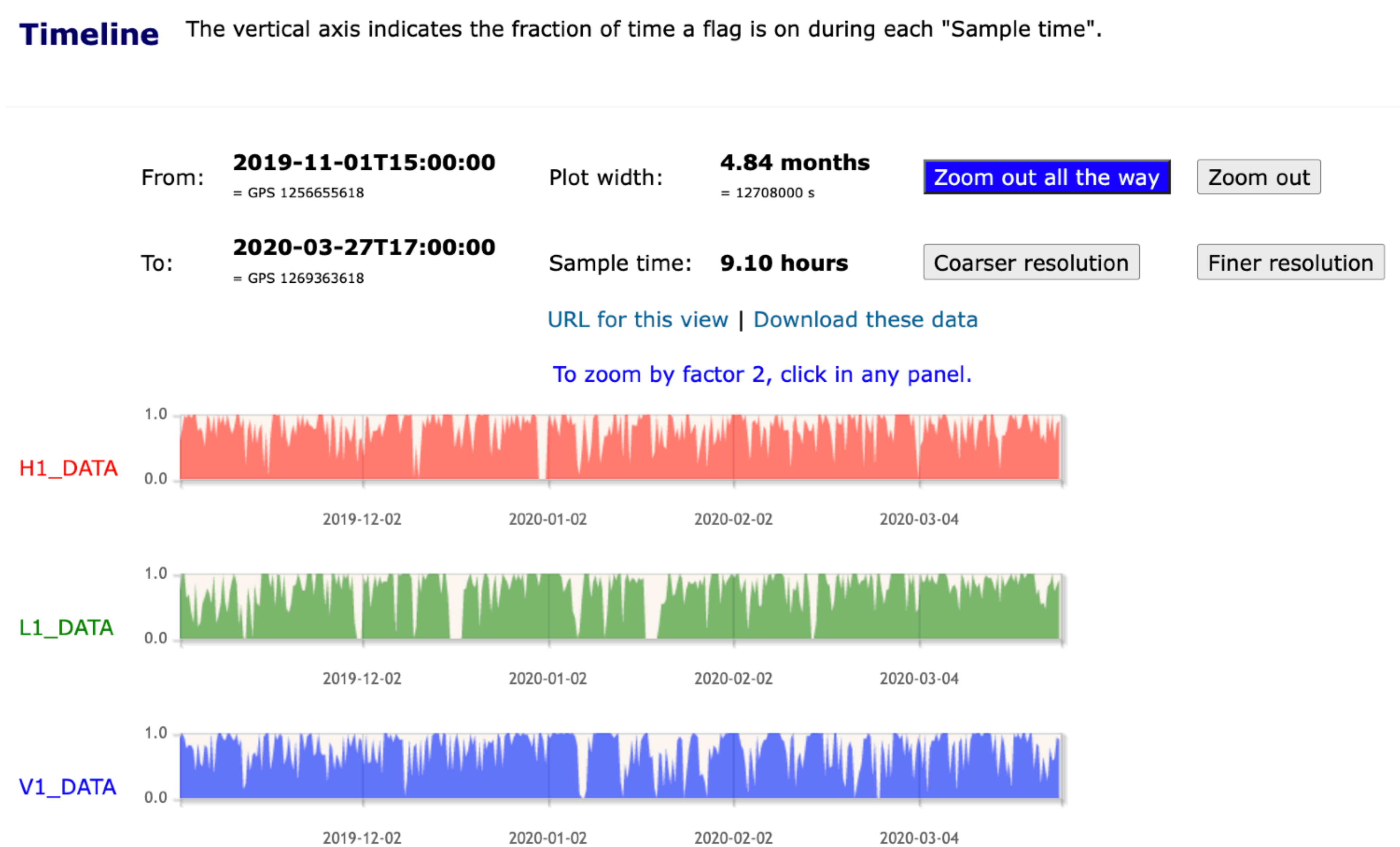
> Please take special note that the section on **IGWN proprietary data (ligo.osgstorage.org)** only applies to the LIGO/Virgo/KAGRA Collaboration members and not to the general public.

**Return to the Data Page to learn more about Bulk Data Releases..**

https://www.gw-openscience.org/cvmfs/

# Segment Information

1. segment DB : https://segments.ligo.org
   - query an available segment to segment DB
2. Public segment information in GWOSC (www.gw-openscience.org)



# Gravitational Wave Open Science Center

Data ▾ Software ▾ Online Tools ▾ About GWOSC ▾

- Strain Data
- Event Portal
- Timelines
- Auxiliary Channels
- Low Latency Alerts

## Timeline

The Timeline App provides information on times when data are available, as well as

### Timeline Queries

- Use the **Run Timeline Query Form** to request any of the Run Timeline or Segment Lists.
- Use the **Event Portal** to access individual Events and request any of the Event Timeline or Seg

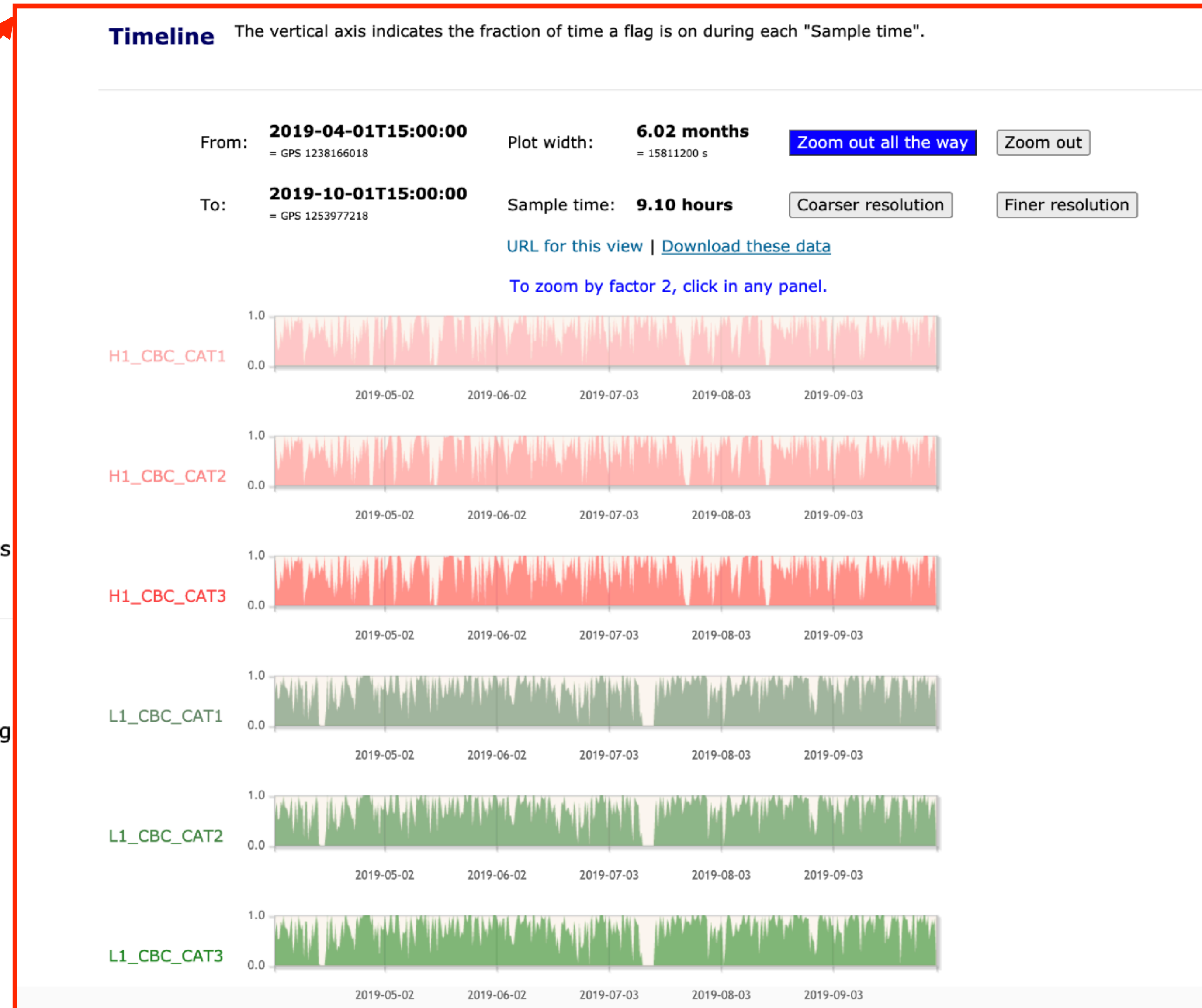
### Timeline Examples

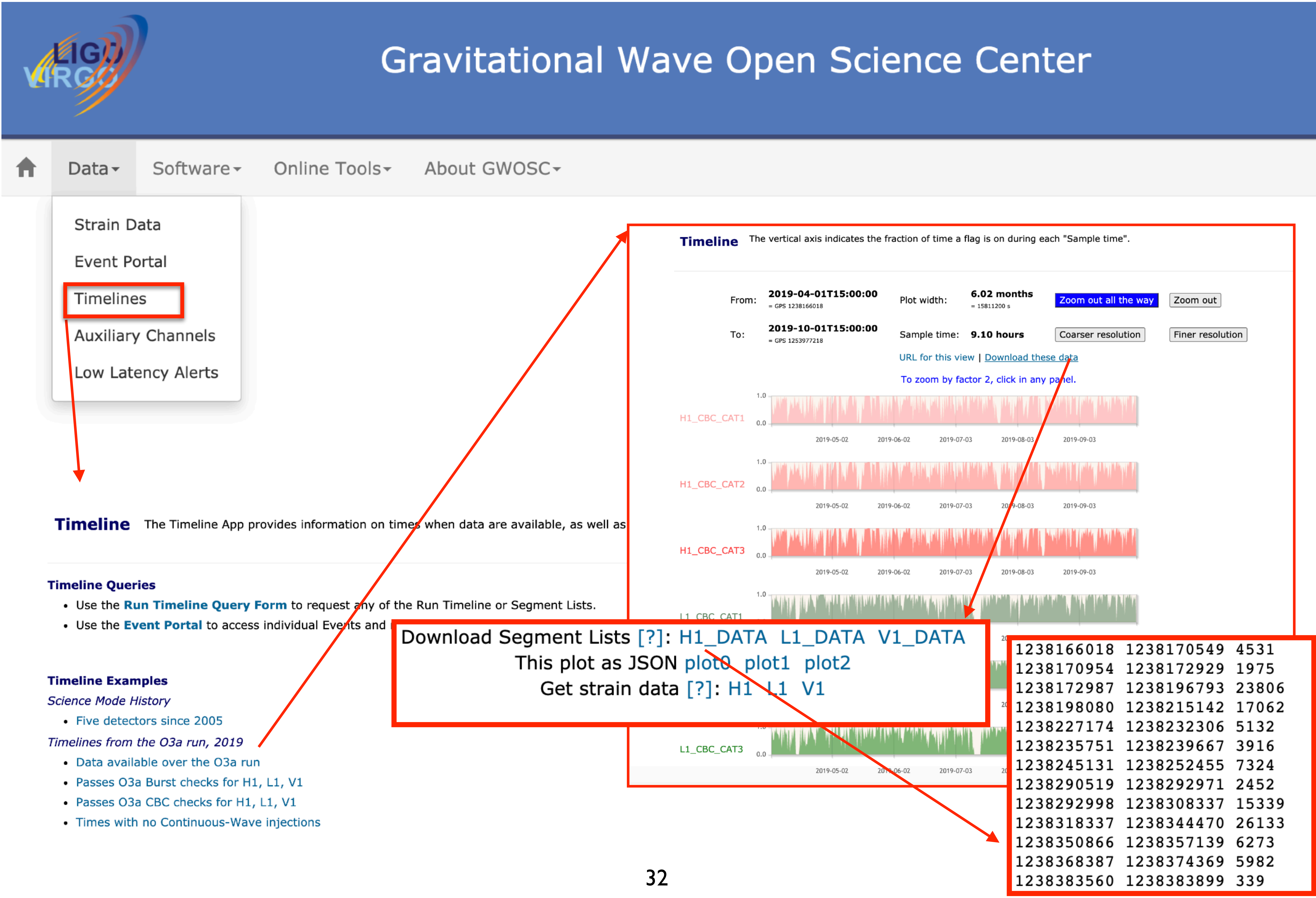
*Science Mode History*

- Five detectors since 2005

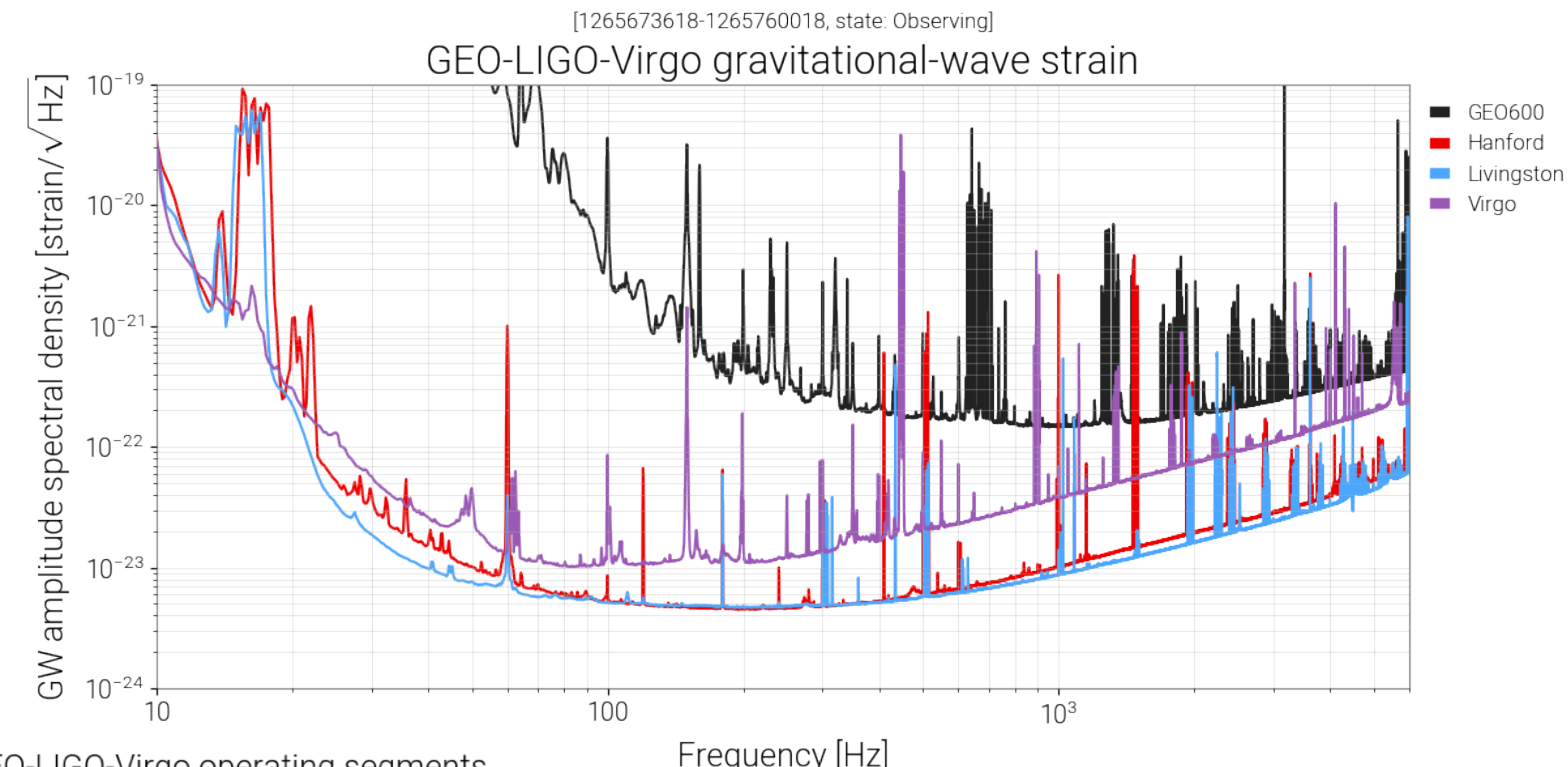
*Timelines from the O3a run, 2019*

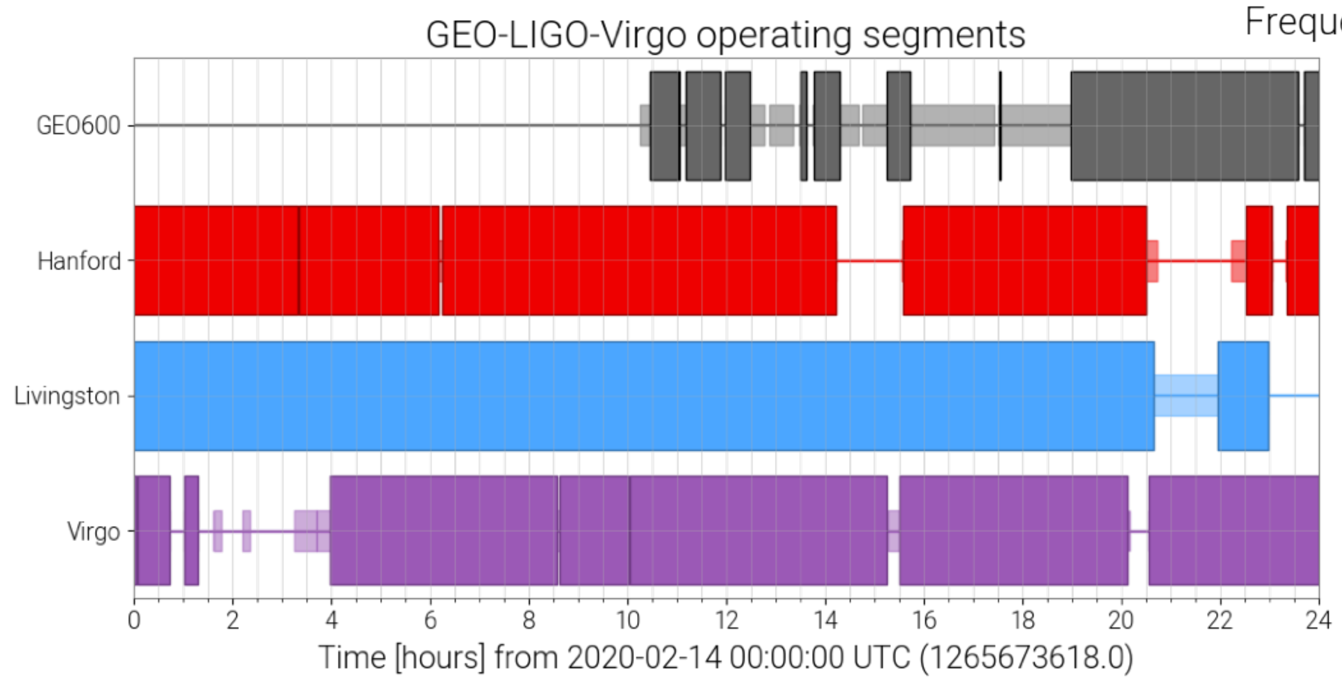
- Data available over the O3a run
- Passes O3a Burst checks for H1, L1, V1
- Passes O3a CBC checks for H1, L1, V1
- Times with no Continuous-Wave injections

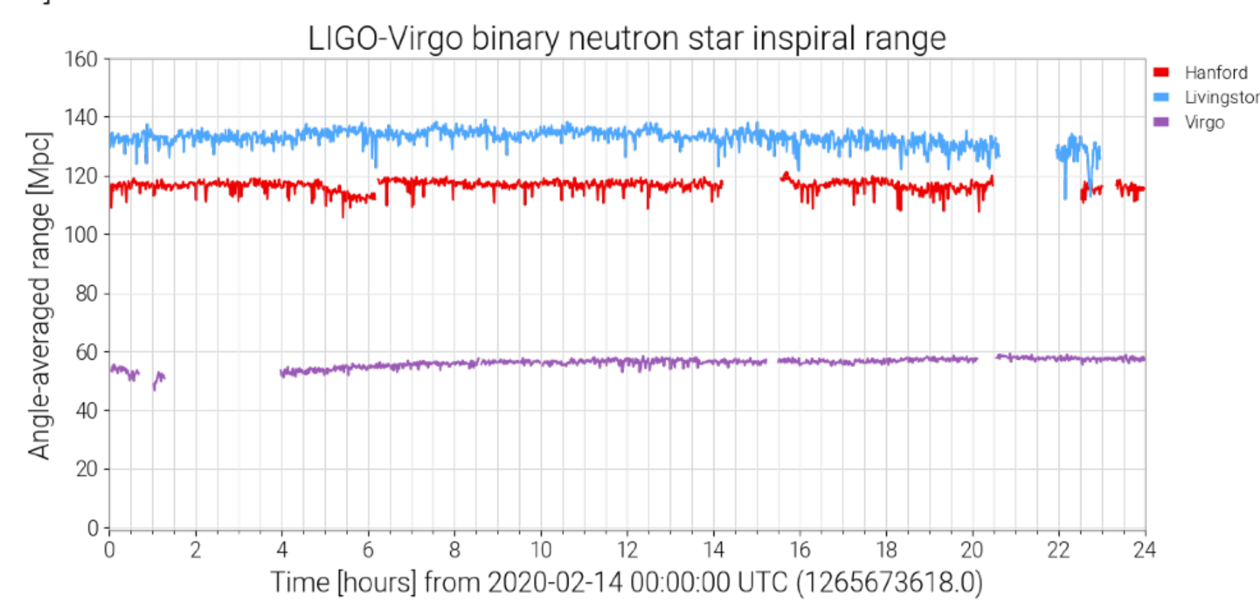



Gravitational Wave Open Science Center
LIGO
VIRGO
Data
Software
Online Tools
About GWOSC
Strain Data
Event Portal
Timelines
Auxiliary Channels
Low Latency Alerts
Timeline The Timeline App provides information on times when data are available, as well as
Timeline Queries
Use the Run Timeline Query Form to request any of the Run Timeline or Segment Lists.
Use the Event Portal to access individual Events and
Timeline Examples
Science Mode History
Five detectors since 2005
Timelines from the O3a run, 2019
Data available over the O3a run
Passes O3a Burst checks for H1, L1, V1
Passes O3a CBC checks for H1, L1, V1
Times with no Continuous-Wave injections
Timeline The vertical axis indicates the fraction of time a flag is on during each "Sample time".
From: 2019-04-01T15:00:00
= GPS 1238166018
To: 2019-10-01T15:00:00
= GPS 1253977218
Plot width: 6.02 months
= 15811200 s
Sample time: 9.10 hours
Zoom out all the way
Zoom out
Coarser resolution
Finer resolution
URL for this view | Download these data
To zoom by factor 2, click in any panel.
H1_CBC_CAT1
H1_CBC_CAT2
H1_CBC_CAT3
L1_CBC_CAT1
L1_CBC_CAT3
1.0
0.0
2019-05-02
2019-06-02
2019-07-03
2019-08-03
2019-09-03
Download Segment Lists [?]: H1_DATA L1_DATA V1_DATA
This plot as JSON plot0 plot1 plot2
Get strain data [?]: H1 L1 V1
1238166018 1238170549 4531
1238170954 1238172929 1975
1238172987 1238196793 23806
1238198080 1238215142 17062
1238227174 1238232306 5132
1238235751 1238239667 3916
1238245131 1238252455 7324
1238290519 1238292971 2452
1238292998 1238308337 15339
1238318337 1238344470 26133
1238350866 1238357139 6273
1238368387 1238374369 5982
1238383560 1238383899 339

# Detector Status in GWOSC




https://www.gw-openscience.org/detector_status/day/20200214/

# Gravitational Wave Open Science Center

Data ▾ Software ▾ Online Tools ▾ About GWOSC ▾

- Strain Data
- Event Portal
- Timelines
- Auxiliary Channels
- Low Latency Alerts

## Auxiliary Channel Three Hour Release

**Data Set**

A large number of sensors are used to record the state of the LIGO instruments and their enviornment. This data release contains sensor data recorded in around 500 channels at each LIGO site. These data represent three hours of time centered on GW170814 (GPS 1186736512 — 1186747264). Strain data from the same period are available in the O2 Data Release.

**Download Data**

The data are available as down-sampled HDF5 files [19 GB], or full sample rate GWF files [68 GB]:

HDF5 Data | GWF Data

500 ch.

~3hr

Data may also be accessed from a network data server (NDS2) using the NDS2 client or GWpy:

```
from gwpy.timeseries import TimeSeries
data = TimeSeries.fetch('L1:LSC-DARM_OUT_DQ', start=1186741850, end=1186741870, host='losc-nds.ligo.org')
```

See the NDS2 Example Code for details.

**Example Software**

Example software is available in an associated git repo. To work with GWF files, see the software page.

**Channel Descriptions**

This data set is designed to be used for subtracting noise sources - especially controls noise - from LIGO data. Channels included in this set are those most likley to include a coupling to the gravitational wave strain channel, and so are possible sources of noise.

The Channel List shows each channel with a few properties:

- Channel name
- Desired sample rate: The sampling rate in the down-sampled, HDF5 data
- Notes: A brief note explaining the meaning of the data in the channel
- Calibration: Where available, a calibration factor is included. Most channels are not calibrated.
- Units: Where available, the units corresponding to the calibration factor

In some cases, data for a given channel may not be available. These are marked "invalid" in the HDF5 files, and the corresponding channel may be absent from the GWF files. Unavailable channels may correspond to sensors that are not present at a particular site or not operational at a particular time.

# Gravitational Wave Open Science Center

Data ▾ | Software ▾ | Online Tools ▾ | About GWOSC ▾

- Strain Data
- Event Portal
- Timelines
- Auxiliary Channels
- Low Latency Alerts

## LIGO/Virgo Public Alerts

**Announcement**

**From GCN Circular 24045:**

Our third observing run ("O3") began as scheduled on 2019 April 1 at 15:00 UTC. At that time the LIGO Hanford, LIGO Livingston, and Virgo Observatories transitioned from engineering and commissioning to observing. All three detectors are operating at good sensitivity and stability. We are analyzing data in low latency and processing candidate transient events automatically.

As of April 2 20:00 UTC, we have configured our low-latency analysis pipeline to send public alerts for significant gravitational-wave transient candidates that are detected in coincidence across two or more gravitational-wave detectors.

Automated Preliminary GCN Notices will be sent immediately without any human intervention. Shortly afterward, they will be vetted by an LSC/Virgo rapid response team and either confirmed with an Initial GCN Notice and Circular, or withdrawn with a Retraction.

Retraction notices may be issued more frequently over the next few weeks as our understanding of the instrumental background improves.

For further information about vetting procedures, analysis methodology, and the contents of LIGO/Virgo public alerts, refer to the LIGO/Virgo Public Alerts User Guide: https://emfollow.docs.ligo.org/userguide/

This marks the beginning of the era of public alerts for the field of gravitational-wave astronomy.

**Resources**

- LIGO/Virgo Alerts User Guide
- Gravitational Wave Candidate Event Database (GraceDB)
- GCN: The Gamma-ray Coordinates Network
- GW Events iPhone app
- Press release on start of O3

# GraceDB

GraceDB Public Alerts Latest Search Documentation Login

Please log in to view full database contents.

Test and MDC events and superevents are not included in the search results by default. See the query help (link below) for information on how to search for events and superevents in those categories.

**Query:** far < 3.171e-14

**Search for:** Superevent

**Get neighbors:** (Events only)

Search

Query help

Show 25 entries

Search:

| UID | Labels | FAR | Preferred Event | GW Events | $t_0$ | Submitted: | Submitted By: |
|---|---|---|---|---|---|---|---|
| S191129u | DQOK EM_READY ADVOK EMBRIGHT_READY SKYMAP_READY PASTRO_READY EM_Selected GCN_PRELIM_SENT PE_READY | 2.650027704713234e-35 | G355916 | G379081 G378825 G377982 G377655 G370621 G360429 G360428 G355991 G355990 G355989 G355988 G355987 G355986 G355985 G355984 G355983 G355982 G355981 G355980 G355979 G355978 G355977 G355976 G355975 G355974 G355973 G355972 G355971 G355970 G355969 G355968 G355967 G355966 G355965 G355964 G355963 G355962 G355961 G355960 G355959 G355958 G355957 G355956 G355955 G355954 G355953 G355952 G355951 G355950 G355949 G355948 G355947 G355946 G355945 G355944 G355943 G355942 G355941 G355940 G355939 G355938 G355937 G355936 G355935 G355934 G355933 G355932 G355931 G355930 G355929 G355928 G355927 G355926 G355925 G355924 G355923 G355922 G355921 G355920 G355919 G355918 G355917 G355916 G355915 G355914 G355913 G355912 G355911 G355910 G355909 G355908 G355907 G355906 G355905 G355904 G355903 G355902 G355901 G355900 G355899 G355898 G355897 G355896 G355895 G355894 G355893 G355892 | 1259070047.197 | 2019-11-29 13:41:25 UTC | LIGO/Virgo EM Follow-Up |
| S190814bv | ADVOK DQOK SKYMAP_READY PASTRO_READY EMBRIGHT_READY GCN_PRELIM_SENT PE_READY | 2.032625014217899e-33 | G347305 | G378036 G376876 G360582 G347305 G347304 G347296 G347295 G347294 G347293 G347292 G347291 G347290 G347289 G347288 G347287 G347286 G347285 G347284 G347283 G347282 G347281 G347280 G347279 G347278 G347277 G347276 G347275 G347274 G347273 G347272 G347271 G347270 G347269 G347268 G347267 G347266 G347265 G347264 G347263 G347262 G347261 G347260 G347259 G347258 G347257 G347256 G347255 G347254 G347253 G347252 G347251 G347250 G347249 | 1249852257.013 | 2019-08-14 21:11:18 UTC | LIGO/Virgo EM Follow-Up |

# GWOSC- find_datasets

```
! pip install -q 'gwosc==0.5.4'
```

```
import gwosc
```

```
from gwosc.datasets import find_datasets
from gwosc import datasets

#-- List all available catalogs
print("List of available catalogs")
print(find_datasets(type="catalog"))
print("")

#-- Print all the GW events from the GWTC-1 catalog
gwtc1 = datasets.find_datasets(type='events', catalog='GWTC-1-confident')
print('GWTC-1 events:', gwtc1)
print("")
```

GW ODW #5,
https://www.gw-openscience.org/odw/odw2022/

Tutorial (git hub) : https://github.com/gw-odw/odw-2022

```
List of available catalogs
['GWTC-1-confident', 'GWTC-1-marginal', 'GWTC-2',
'Initial_LIGO_Virgo', 'O1_O2-Preliminary', 'O3_Discovery_Papers',
'O3_IMBH_marginal']

GWTC-1 events: ['GW150914-v3', 'GW151012-v3', 'GW151226-v2',
'GW170104-v2', 'GW170608-v3', 'GW170729-v1', 'GW170809-v1', 'GW170814-
v3', 'GW170817-v3', 'GW170818-v1', 'GW170823-v1']
```

# GWOSC- find_datasets

```
#-- Print all the large strain data sets from LIGO/Virgo observing runs
runs = find_datasets(type='run')
print('Large data sets:', runs)
```

```
Large data sets: ['BKGW170608_16KHZ_R1', 'O1', 'O1_16KHZ',
'O2_16KHZ_R1', 'O2_4KHZ_R1', 'O3a_16KHZ_R1', 'O3a_4KHZ_R1',
'S5', 'S6', 'oldhistory']
```

```
print(find_datasets())
```

```
['151008-v1', '151012.2-v1', '151116-v1', '161202-v1', '161217-v1', '170208-v1', '170219-v1', '170405-v1',
'170412-v1', '170423-v1', '170616-v1', '170630-v1', '170705-v1', '170720-v1', '190924_232654-v1',
'191223_014159-v1', '191225_215715-v1', '200114_020818-v1', '200214_224526-v1', 'BKGW170608_16KHZ_R1',
'GRB051103-v1', 'GW150914-v1', 'GW150914-v2', 'GW150914-v3', 'GW151012-v1', 'GW151012-v2', 'GW151012-v3',
'GW151226-v1', 'GW151226-v2', 'GW170104-v1', 'GW170104-v2', 'GW170608-v1', 'GW170608-v2', 'GW170608-v3',
'GW170729-v1', 'GW170809-v1', 'GW170814-v1', 'GW170814-v2', 'GW170814-v3', 'GW170817-v1', 'GW170817-v2',
'GW170817-v3', 'GW170818-v1', 'GW170823-v1', 'GW190408_181802-v1', 'GW190412-v1', 'GW190412-v2', 'GW190412-v3',
'GW190413_052954-v1', 'GW190413_134308-v1', 'GW190421_213856-v1', 'GW190424_180648-v1', 'GW190425-v1',
'GW190425-v2', 'GW190426_152155-v1', 'GW190503_185404-v1', 'GW190512_180714-v1', 'GW190513_205428-v1',
'GW190514_065416-v1', 'GW190517_055101-v1', 'GW190519_153544-v1', 'GW190521-v1', 'GW190521-v2', 'GW190521-v3',
'GW190521_074359-v1', 'GW190527_092055-v1', 'GW190602_175927-v1', 'GW190620_030421-v1', 'GW190630_185205-v1',
'GW190701_203306-v1', 'GW190706_222641-v1', 'GW190707_093326-v1', 'GW190708_232457-v1', 'GW190719_215514-v1',
'GW190720_000836-v1', 'GW190727_060333-v1', 'GW190728_064510-v1', 'GW190731_140936-v1', 'GW190803_022701-v1',
'GW190814-v1', 'GW190814-v2', 'GW190828_063405-v1', 'GW190828_065509-v1', 'GW190909_114149-v1',
'GW190910_112807-v1', 'GW190915_235702-v1', 'GW190924_021846-v1', 'GW190929_012149-v1', 'GW190930_133541-v1',
'GW200105-v1', 'GW200115-v1', 'GWTC-1-confident', 'GWTC-1-marginal', 'GWTC-2', 'Initial_LIGO_Virgo', 'O1',
'O1_16KHZ', 'O1_O2-Preliminary', 'O2_16KHZ_R1', 'O2_4KHZ_R1', 'O3_Discovery_Papers', 'O3_IMBH_marginal',
'O3a_16KHZ_R1', 'O3a_4KHZ_R1', 'S5', 'S6', 'blind_injection-v1', 'oldhistory']
```

# GWOSC

```python
from gwosc.datasets import event_gps
gps = event_gps('GW190425')
print(gps)
```

```
1240215503.0
```

```python
from gwosc.datasets import run_segment
print(run_segment('O1'))
```

```
(1126051217, 1137254417)
```

```python
from gwosc.locate import get_event_urls
urls = get_event_urls('GW150914')
print(urls)
```

```
['https://www.gw-openscience.org/eventapi/json/GWTC-1-confident/GW150914/v3/H-H1_GWOSC_4KHZ_R1-1126259447-32.hdf5',
'https://www.gw-openscience.org/eventapi/json/GWTC-1-confident/GW150914/v3/H-H1_GWOSC_4KHZ_R1-1126257415-4096.hdf5
'https://www.gw-openscience.org/eventapi/json/GWTC-1-confident/GW150914/v3/L-L1_GWOSC_4KHZ_R1-1126259447-32.hdf5',
'https://www.gw-openscience.org/eventapi/json/GWTC-1-confident/GW150914/v3/L-L1_GWOSC_4KHZ_R1-1126257415-4096.hdf5
```

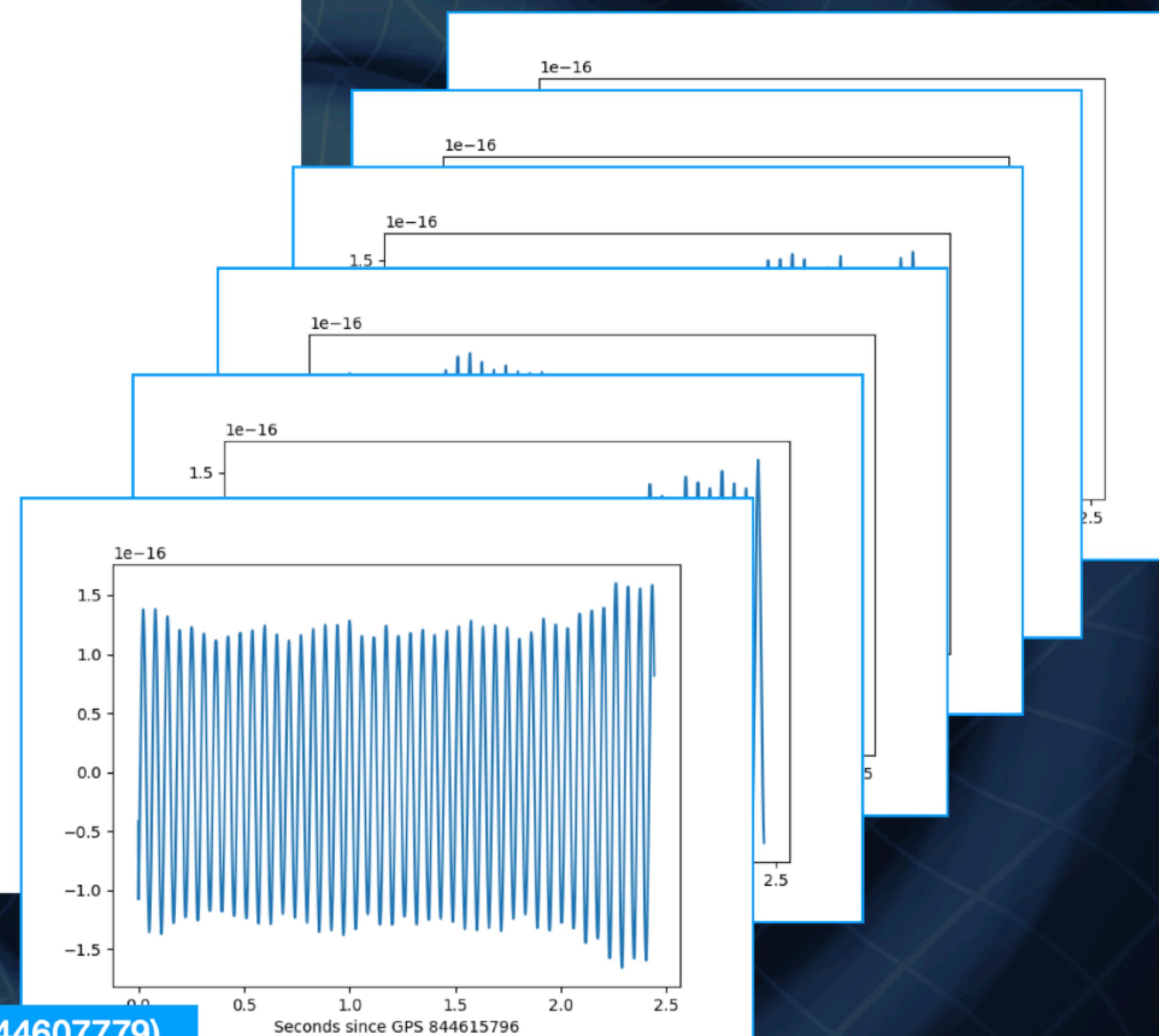
```python
urls = get_event_urls('GW150914', duration=32, detector='L1')
print(urls)
```

```
['https://www.gw-openscience.org/eventapi/json/GWTC-1-confident/GW150914/v3/L-L1_GWOSC_4KHZ_R1-1126259447-32.hdf5']
```

# Readligo.py

https://www.gw-openscience.org/s/sample_code/use_readligo.py

```python
import numpy as np
import matplotlib.pyplot as plt
import readligo as rl

start = 844605900
stop =  844615900

segList = rl.getsegs(start, stop, 'H1', flag='BURST_CAT2')
print (segList)
#----------------------------------------
# Plot a few seconds of each "good" segment
#----------------------------------------
N = 10000
for (begin, end) in segList:
    # -- Use the getstrain() method to load the data
    strain, meta, dq = rl.getstrain(begin, end, 'H1')

    # -- Make a plot
    plt.figure()
    ts = meta['dt']
    rel_time = np.arange(0, end-begin, meta['dt'])
    plt.plot(rel_time[0:N], strain[0:N])
    plt.xlabel('Seconds since GPS ' + str(begin) )
plt.show()
```




**Output:**

SegmentList( [(844605900, 844606294), (844606594, 844606649), (844606759, 844607779), (844608784, 844609581), (844612612, 844615722), (844615796, 844615900)] )

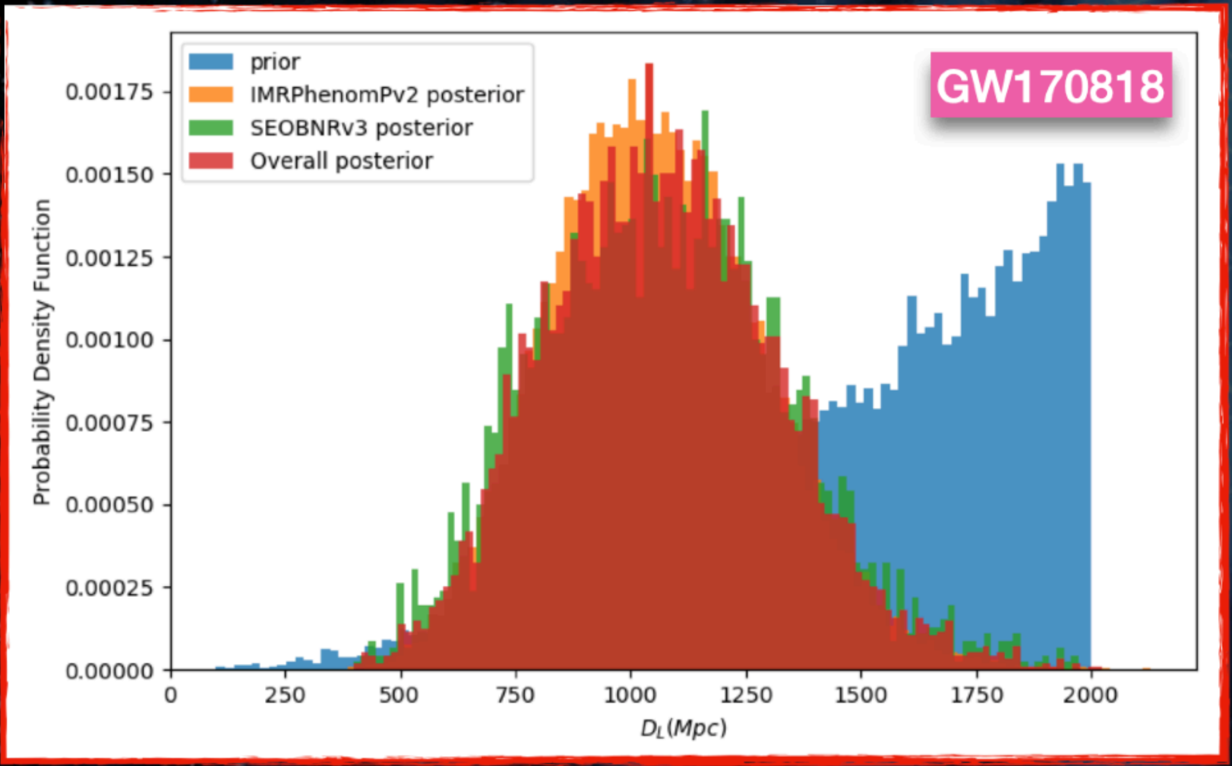
# Other analysis products

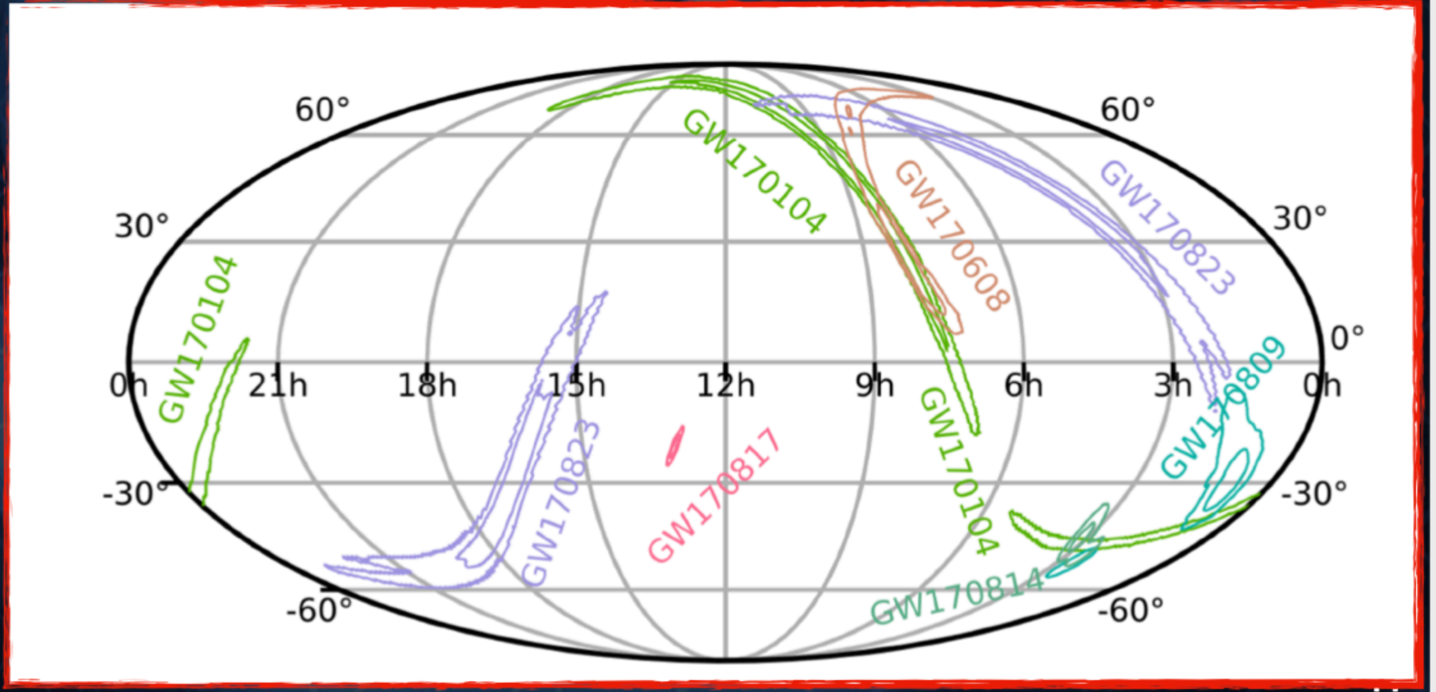
- The event portal contains links to:
  - ✓ Posterior samples
  - ✓ Confidence intervals
  - ✓ Skymaps

GWTC-2 documentation page

**Data Products and Publications**

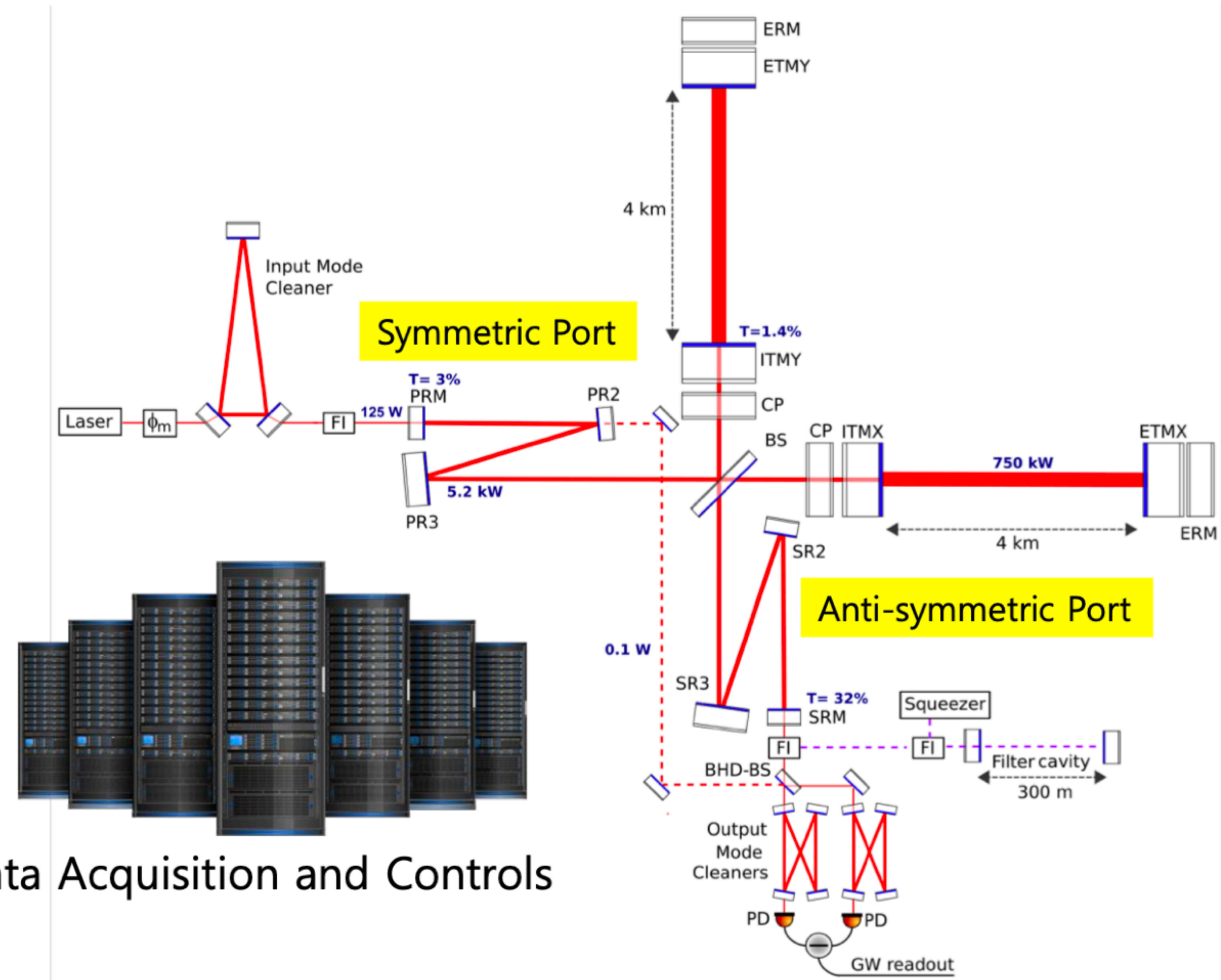
- **Catalog Paper and Figures:** P2000061
- **Strain Data:** Event Portal
- **Parameter Estimation Samples & Skymaps:** P2000223
- **Tests of General Relativity:** P2000091
- **Population Properties:** P2000077
- **Search Sensitivity:** P2000217
- **Glitch Models:** P2000289
- **Low-Latency Alerts:** GraceDB




A. Trovato, ODW#4, 10th May 2021

# GW data w/ sampling rates



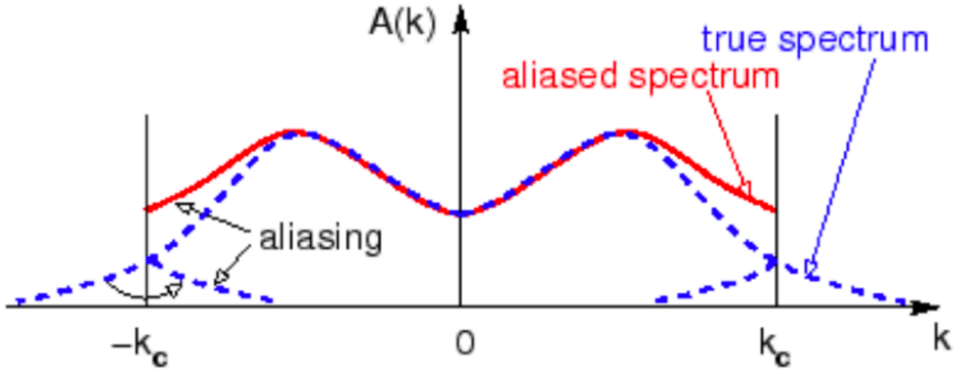

Data Acquisition and Controls

- Sampling rate
  - 16384Hz:LIGO
  - 20kHz:Virgo
  - 16384Hz:KAGRA
- Valid range
  - 10Hz~5kHz : LIGO
  - 10Hz~8kHz : Virgo
  - 10Hz~5kHz : KAGRA

이형원교수님
지난 여름학교 강의중에서

# Nyquist Fruency

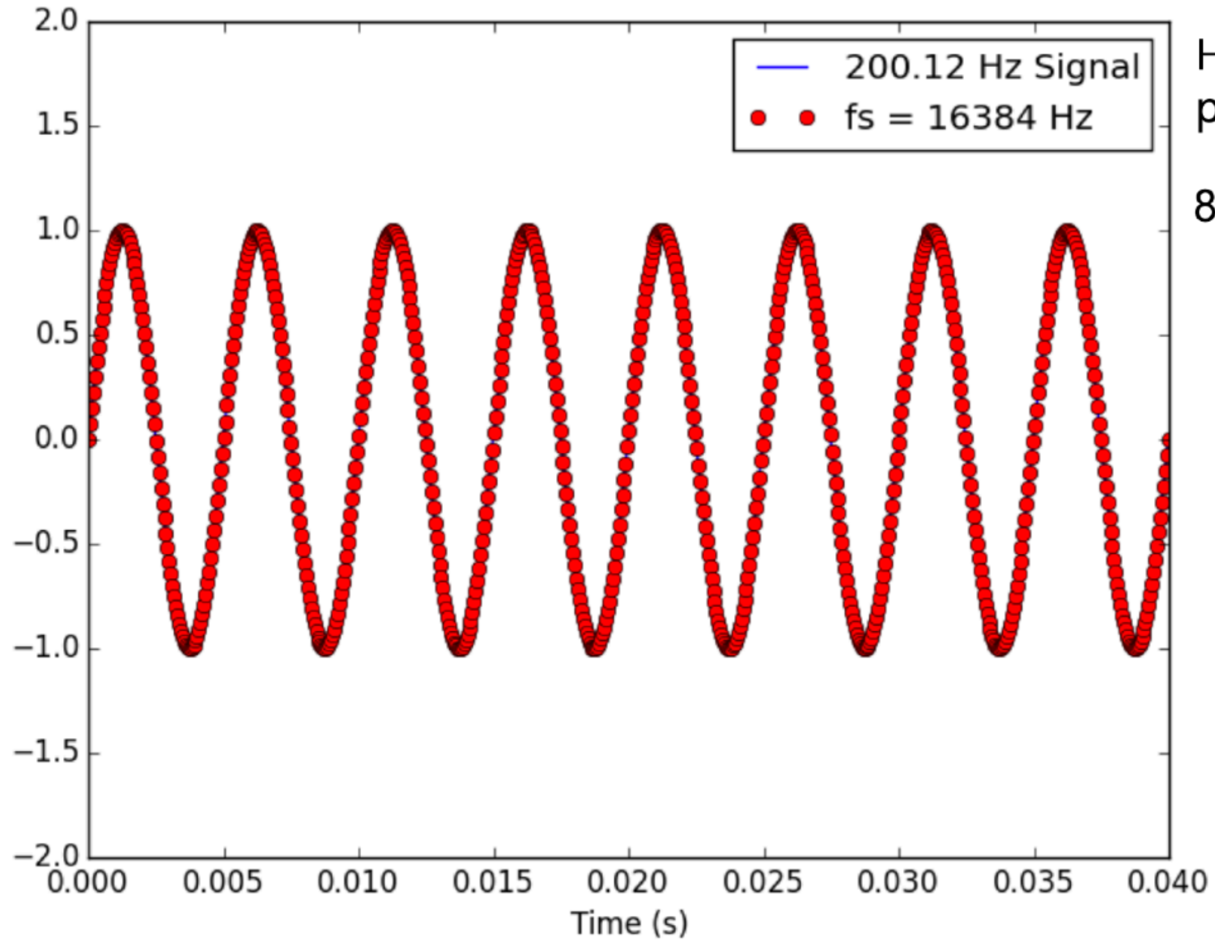
- Nyquist Frequency = $f_s$ / 2
  - Discretely sampled data with sampling rate fs can represent a continuous signal which only has frequency content below the Nyquist frequency
- Data can only contain frequency content below the *Nyquist frequency*
- Higher frequency signals will be lost or "aliased" to lower frequencies




이형원교수님
지난 여름학교 강의중에서

# Example - Nyquist frequency

## Discrete Time Samples




How many samples per cycle?

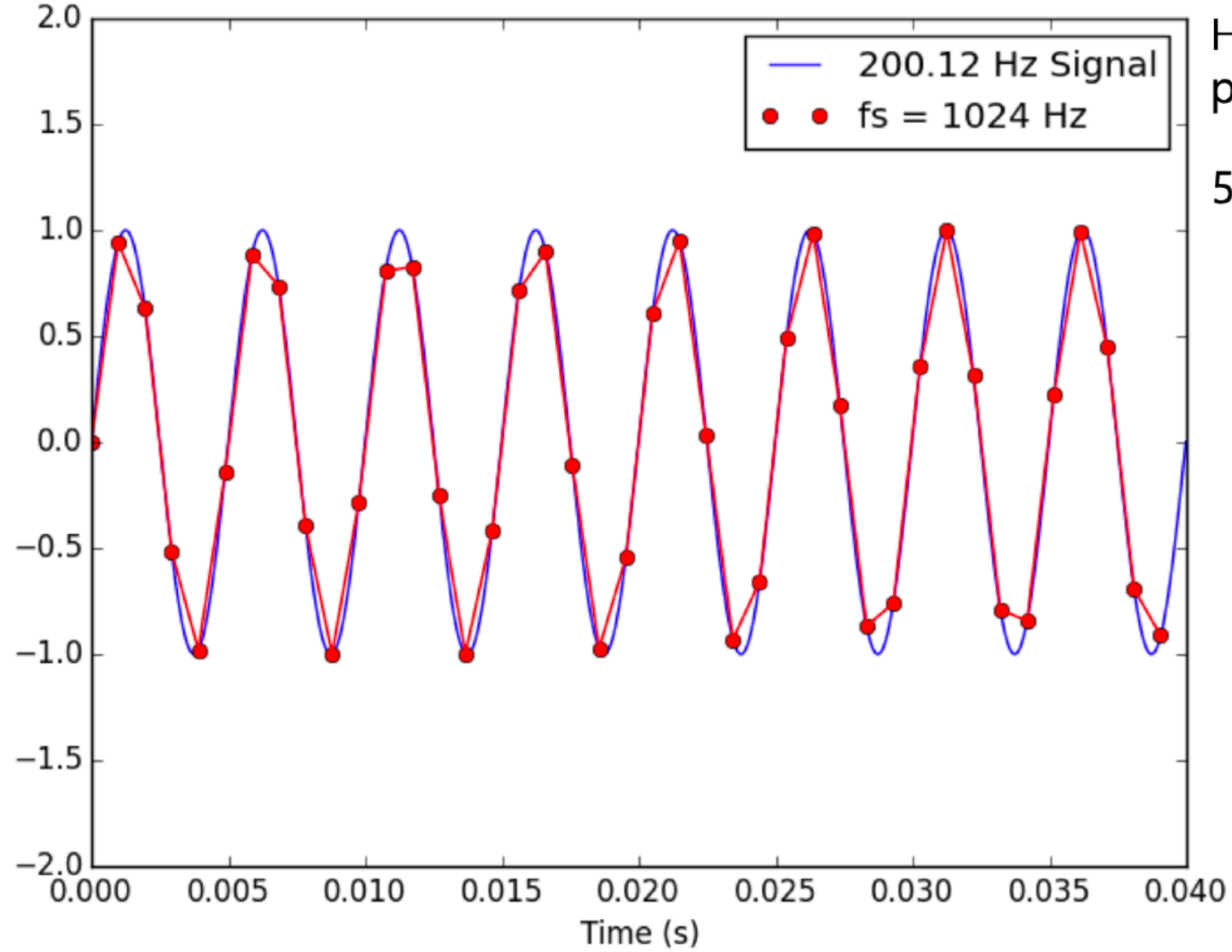
81.87 times



이형원교수님
지난 여름학교 강의중에서

# Example - Nyquist frequency

## Discrete Time Samples



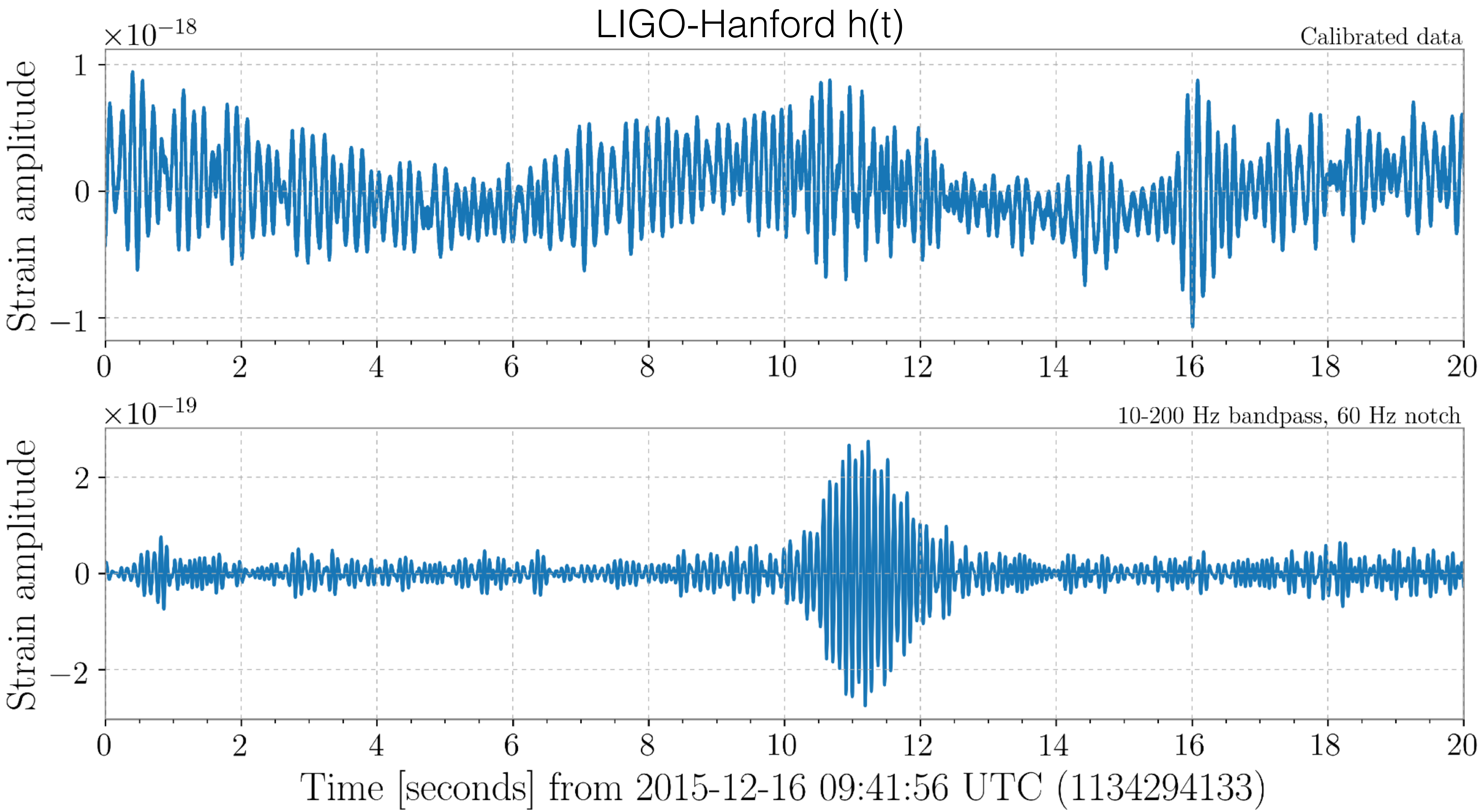

How many samples per cycle?

5.12 times



이형원교수님
지난 여름학교 강의중에서

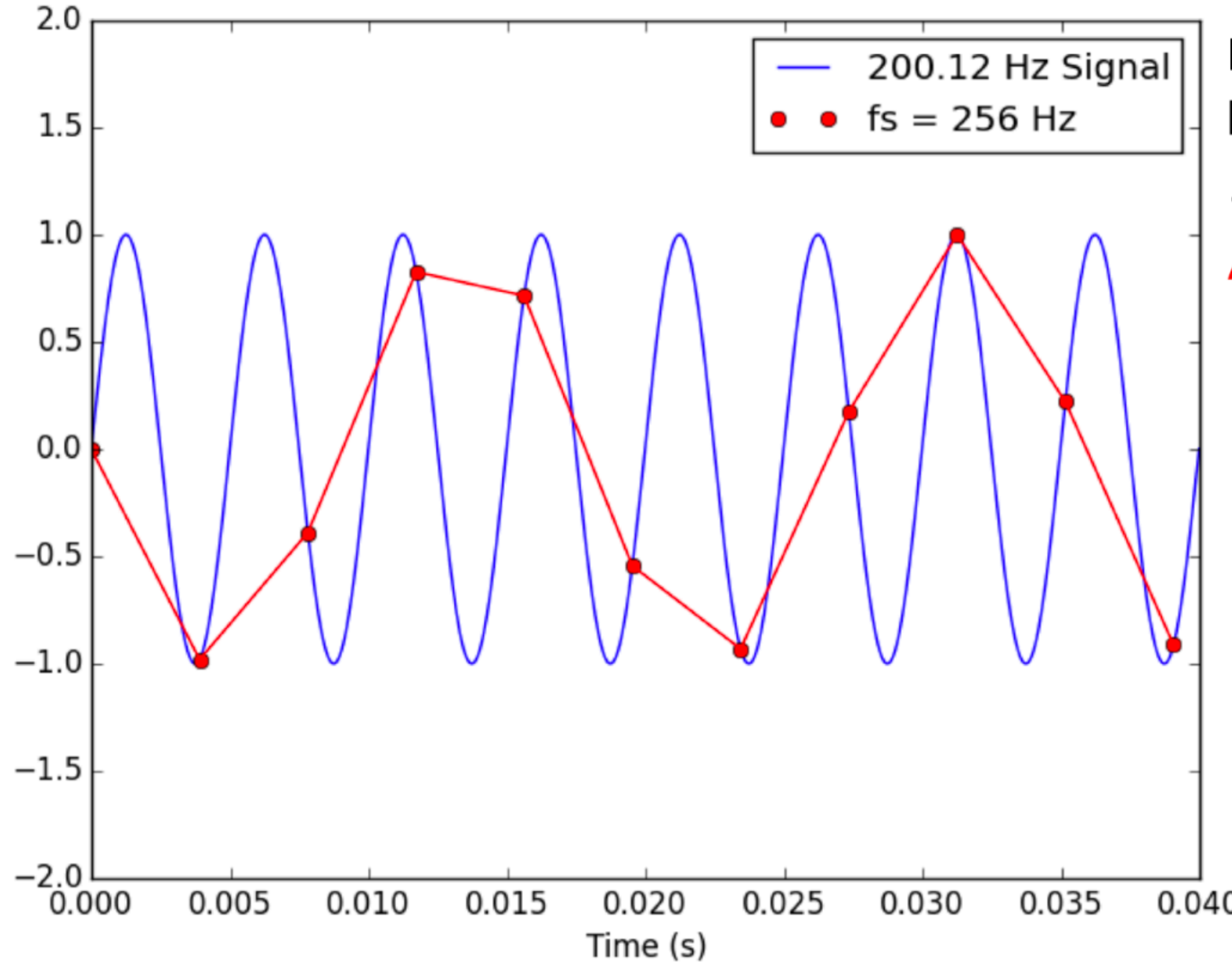
# What does LIGO data look like?




Made with GWpy by Duncan Macleod

# Example - Nyquist frequency

## Discrete Time Samples




How many samples per cycle?

1.28 times
Aliasing occurs



이형원교수님
지난 여름학교 강의중에서

# Possible properties of noise

**Stationary : statistical properties are independent of time**

Ergodic process: time averages are equivalent to ensemble averages

**Gaussian : A random variable follows Gaussian distribution**

For a single random variable, $$p(x) = \frac{1}{\sqrt{2\pi\sigma_x^2}} \exp\left[-\frac{1}{2}\frac{(x-\mu_x)^2}{\sigma_x^2}\right]$$

More generally, a *set* of random variables (e.g. a time series) is Gaussian if the joint probability distribution is governed by a covariance matrix

$$C_{xij} := \langle x_i x_j \rangle - \langle x_i \rangle \langle x_j \rangle$$

such that

$$p(x_1, x_2, \ldots, x_N) = \frac{1}{(2\pi)^{N/2}\sqrt{\det C_x}} \exp\left[-\frac{1}{2}\sum_{i,j=0}^{N-1} C_{xij}^{-1}(x_i - \mu_{xi})(x_j - \mu_{xj})\right]$$

**White : Signal power is uniformly distributed over frequency**

$\Rightarrow$ Data samples are uncorrelated

이형원교수님
지난 여름학교 강의중에서

# In Frequency Domain

Fourier Transform is used to represent data in the frequency domain

**Fourier transform**

$$\tilde{x}(f) = \int_{-\infty}^{\infty} dt\, x(t) e^{-i2\pi f t}$$

$$\Rightarrow \qquad x(t) = \int_{-\infty}^{\infty} df\, \tilde{x}(f) e^{i2\pi f t}$$

$|\tilde{x}(f)|^2$ can be interpreted as energy spectral density

Efficient way to calculate complete discrete Fourier Transform:
Fast Fourier Transform (FFT)

이형원교수님
지난 여름학교 강의중에서

# Power Spectral Density

**Parseval's theorem:**

$$\int_{-\infty}^{\infty} dt\, |x(t)|^2 = \int_{-\infty}^{\infty} df\, |\widetilde{x}(f)|^2$$

$\Rightarrow$ Total energy in the data can be calculated in either time domain or frequency domain

$|\widetilde{x}(f)|^2$ can be interpreted as energy spectral density

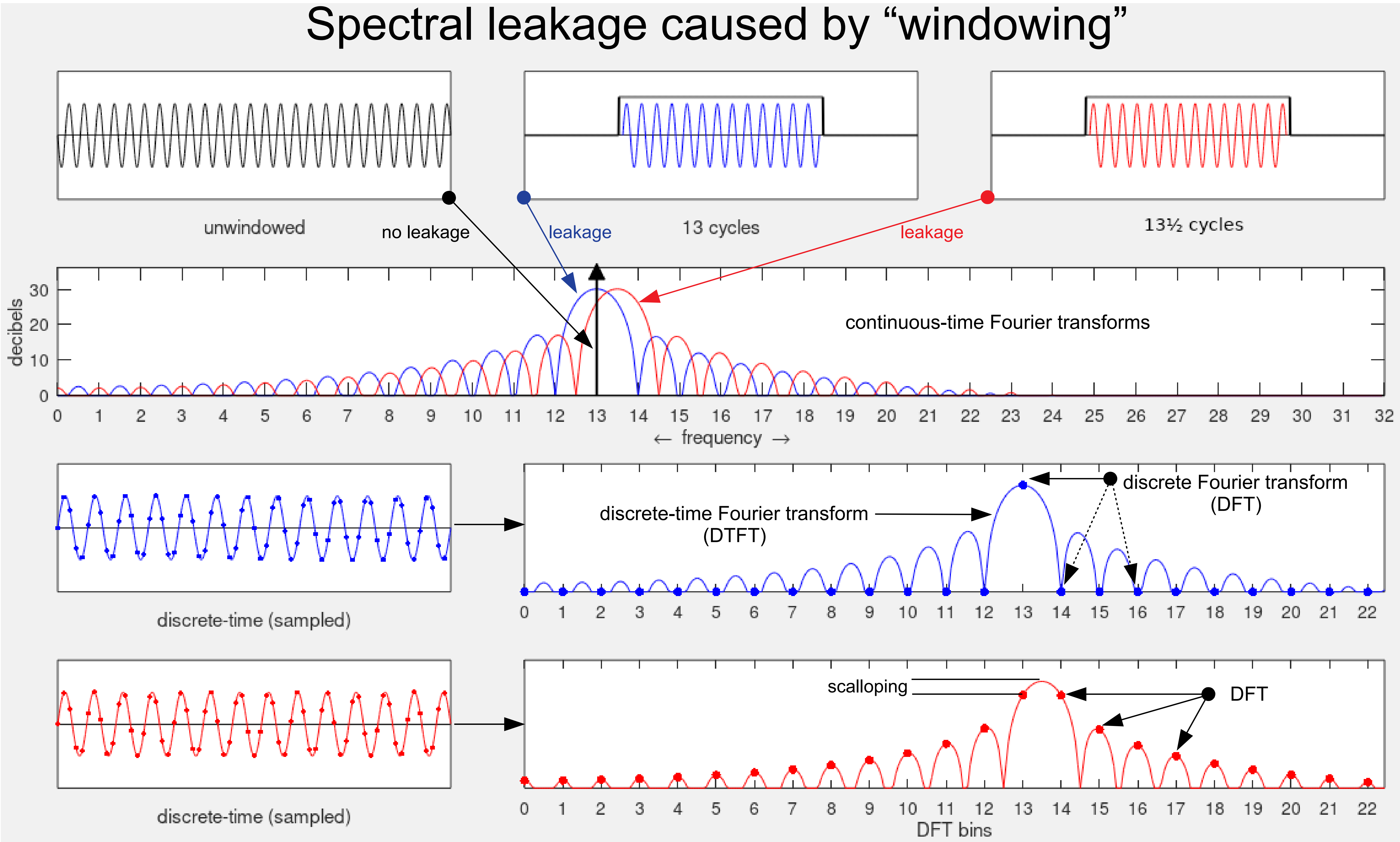
**When noise (or signal) has infinite extent in time domain, can still define the power spectral density (PSD)**

$$\lim_{T\to\infty} \frac{1}{T} \left|\widetilde{x}_T(f)\right|^2 \qquad \langle n(f) n^*(f')\rangle = \delta(f - f') S_n(f)$$

**Watch out for one-sided vs. two-sided PSDs**

Slide from P. Shawhan

# Spectral Power leakage



# Window function

A mathematical function that is zero-valued outside of some chosen interval, normally symmetric around the middle of the interval, usally near a maximum in the middle, and usually tapering away from the middle.
- https://en.wikipedia.org/wiki/Window_function

## Rectangular window [ edit ]

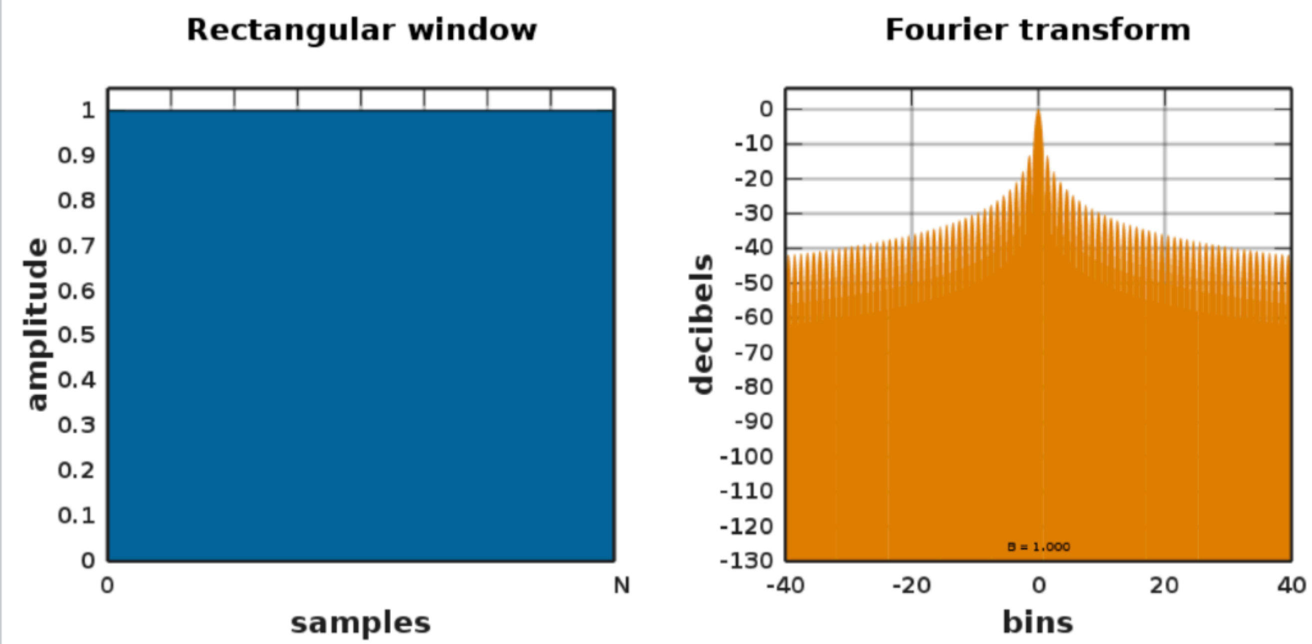
The rectangular window (sometimes known as the **boxcar** or **Dirichlet window**) is the simplest window, equivalent to replacing all but *N* values of a data sequence by zeros, making it appear as though the waveform suddenly turns on and off:

$$w[n] = 1.$$

Other windows are designed to moderate these sudden changes, which reduces scalloping loss and improves dynamic range, as described above (§ Spectral analysis).

The rectangular window is the 1st order *B*-spline window as well as the 0th power power-of-sine window.

The rectangular window provides the minimum mean square error estimate of the Discrete-time Fourier transform, at the cost of other issues discussed.



Rectangular window

https://en.wikipedia.org/wiki/List_of_window_functions

# Hann window

The customary cosine-sum windows for case $K = 1$ have the form:

$$w[n] = a_0 - \underbrace{(1 - a_0)}_{a_1} \cdot \cos\left(\frac{2\pi n}{N}\right), \quad 0 \le n \le N,$$

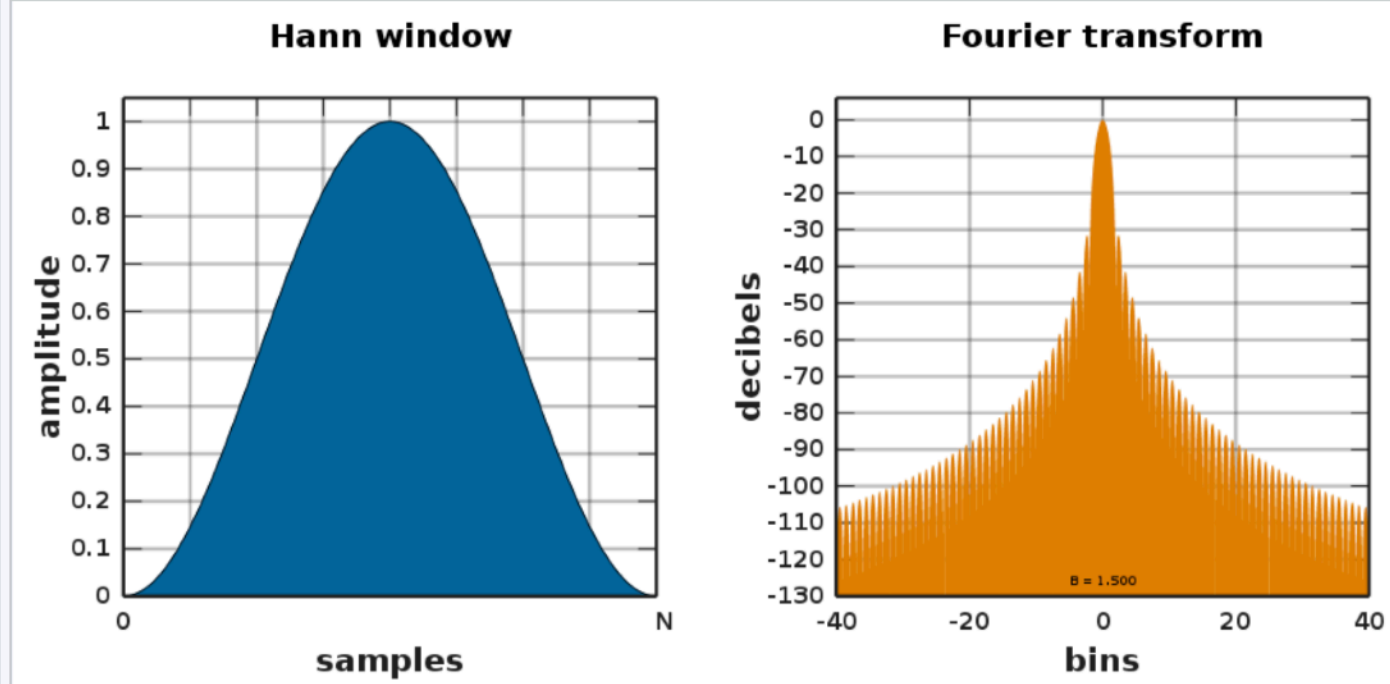
which is easily (and often) confused with its zero-phase version:

$$\begin{aligned} w_0(n) &= w\left[n + \frac{N}{2}\right] \\ &= a_0 + a_1 \cdot \cos\left(\frac{2\pi n}{N}\right), \quad -\frac{N}{2} \le n \le \frac{N}{2}. \end{aligned}$$

Setting $a_0 = 0.5$ produces a **Hann window:**

$$w[n] = 0.5\left[1 - \cos\left(\frac{2\pi n}{N}\right)\right] = \sin^2\left(\frac{\pi n}{N}\right),$$ [15]

named after Julius von Hann, and sometimes erroneously referred to




Hann window

https://en.wikipedia.org/wiki/List_of_window_functions

# Tukey Window

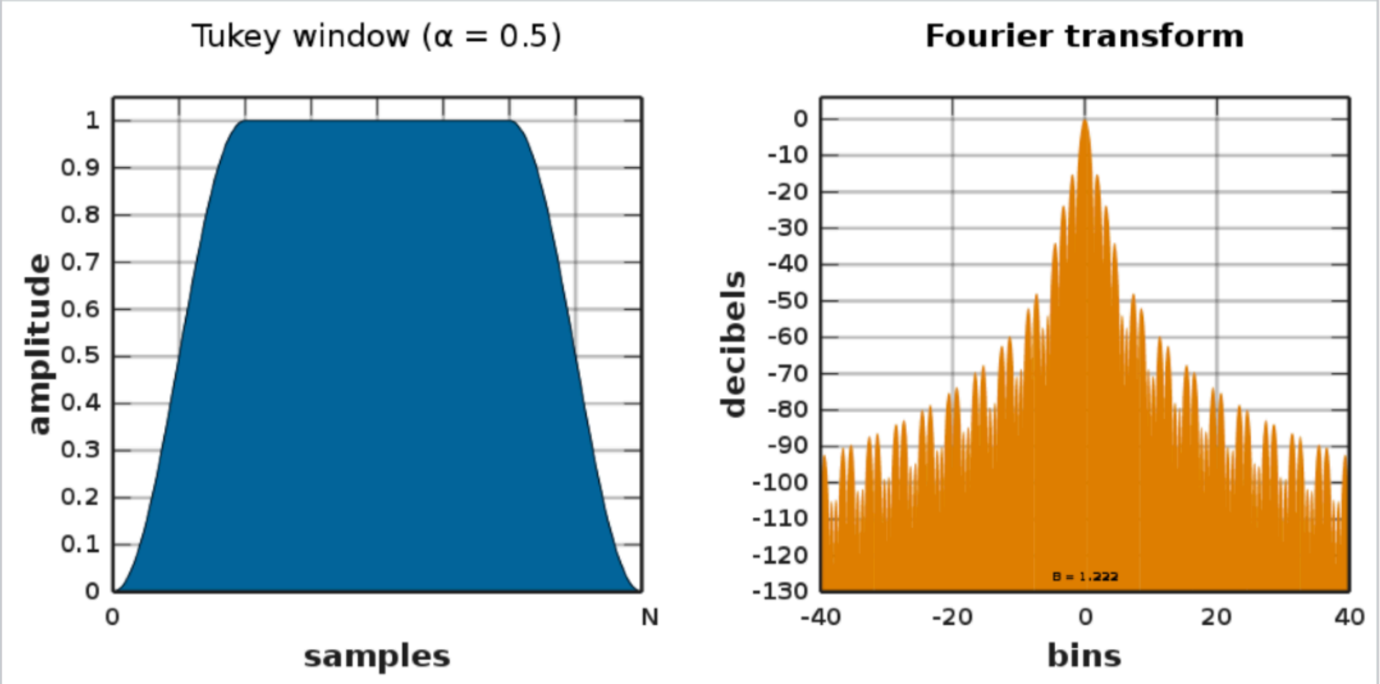
**Tukey window** [ edit ]

The Tukey window, also known as the *cosine-tapered window*, can be regarded as a cosine lobe of width $N\alpha/2$ (spanning $N\alpha/2 + 1$ observations) that is convolved with a rectangular window of width $N(1 - \alpha/2)$.

$$\left.\begin{array}{ll} w[n] = \frac{1}{2}\left[1 - \cos\left(\frac{2\pi n}{\alpha N}\right)\right], & 0 \le n < \frac{\alpha N}{2} \\ w[n] = 1, & \frac{\alpha N}{2} \le n \le \frac{N}{2} \\ w[N-n] = w[n], & 0 \le n \le \frac{N}{2} \end{array}\right\}$$

[38][B][C]

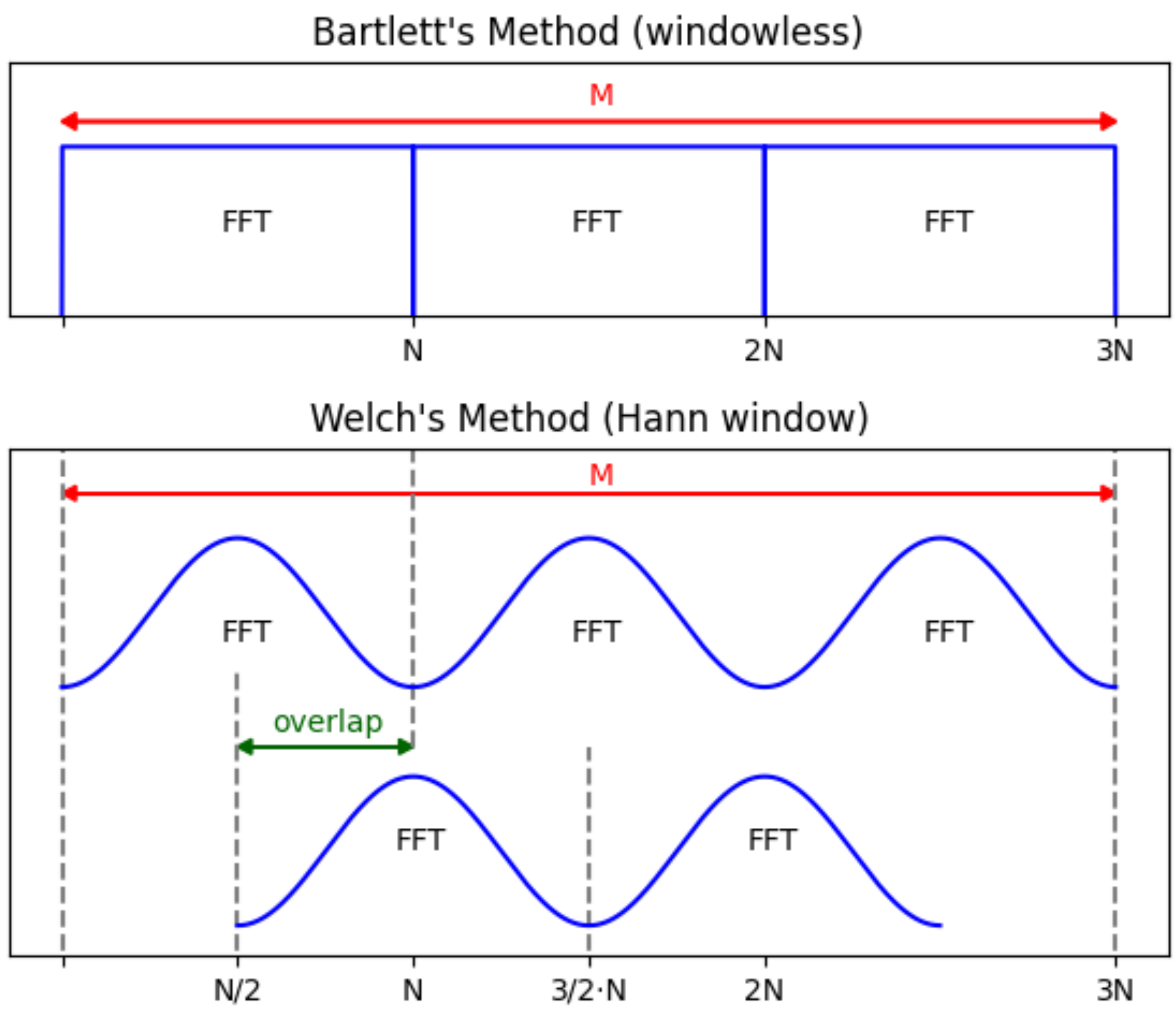
At $\alpha = 0$ it becomes rectangular, and at $\alpha = 1$ it becomes a Hann window.



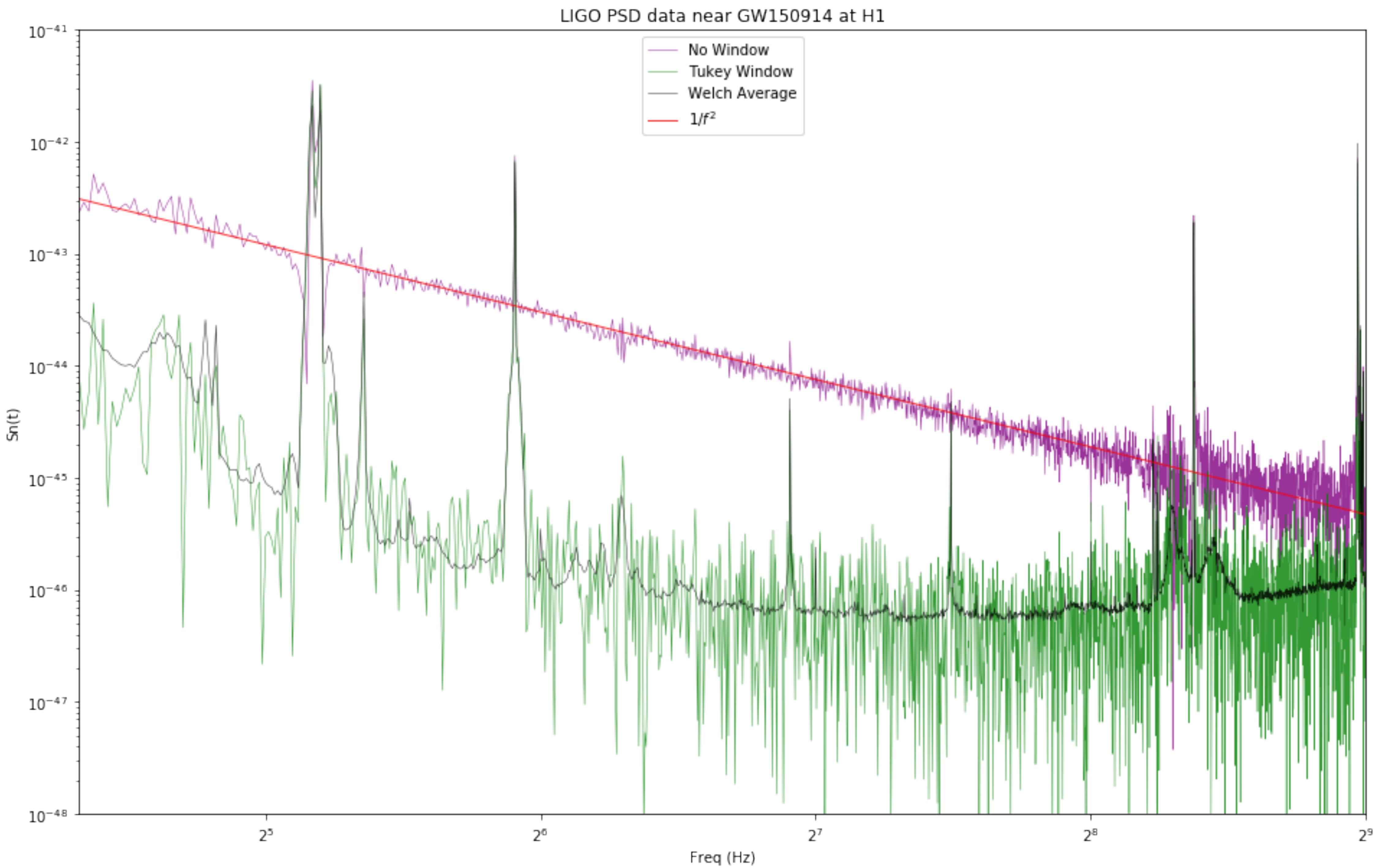

Tukey window, $\alpha = 0.5$

https://en.wikipedia.org/wiki/List_of_window_functions

# PSD - averaging

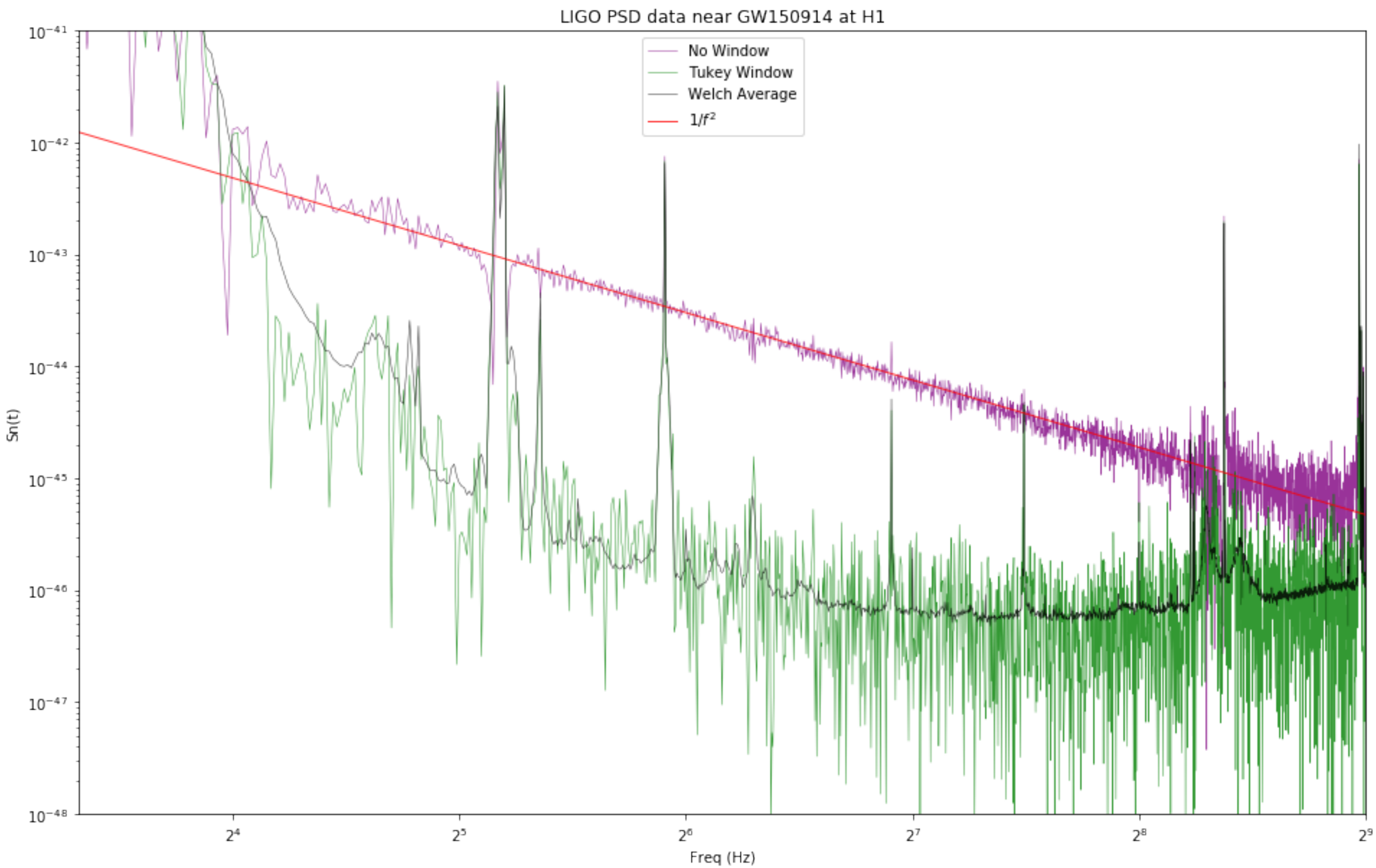


# PSD w/ or w/o window



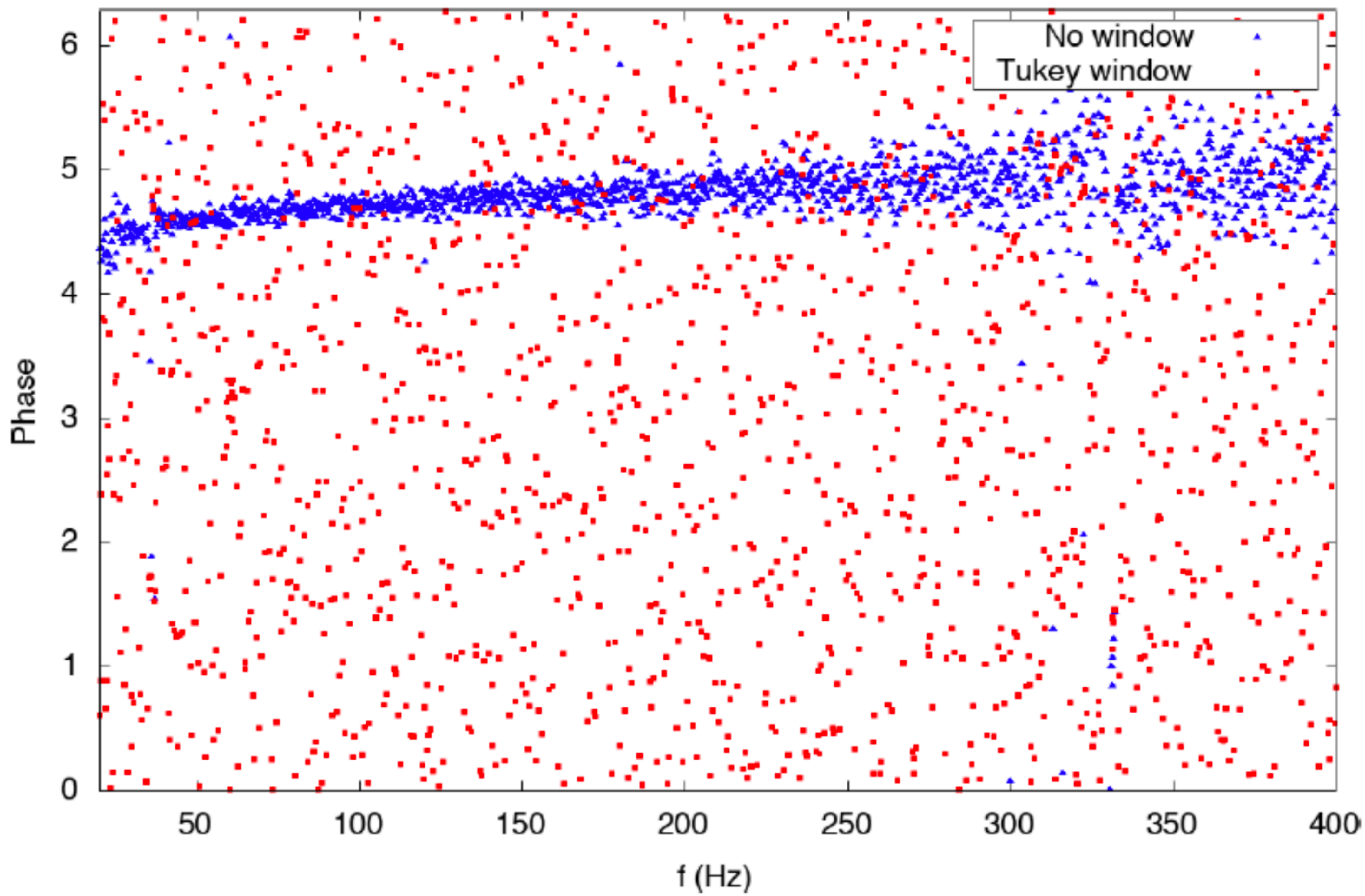

CQG 37, 055002 (2020)

# PSD w/ or w/o window



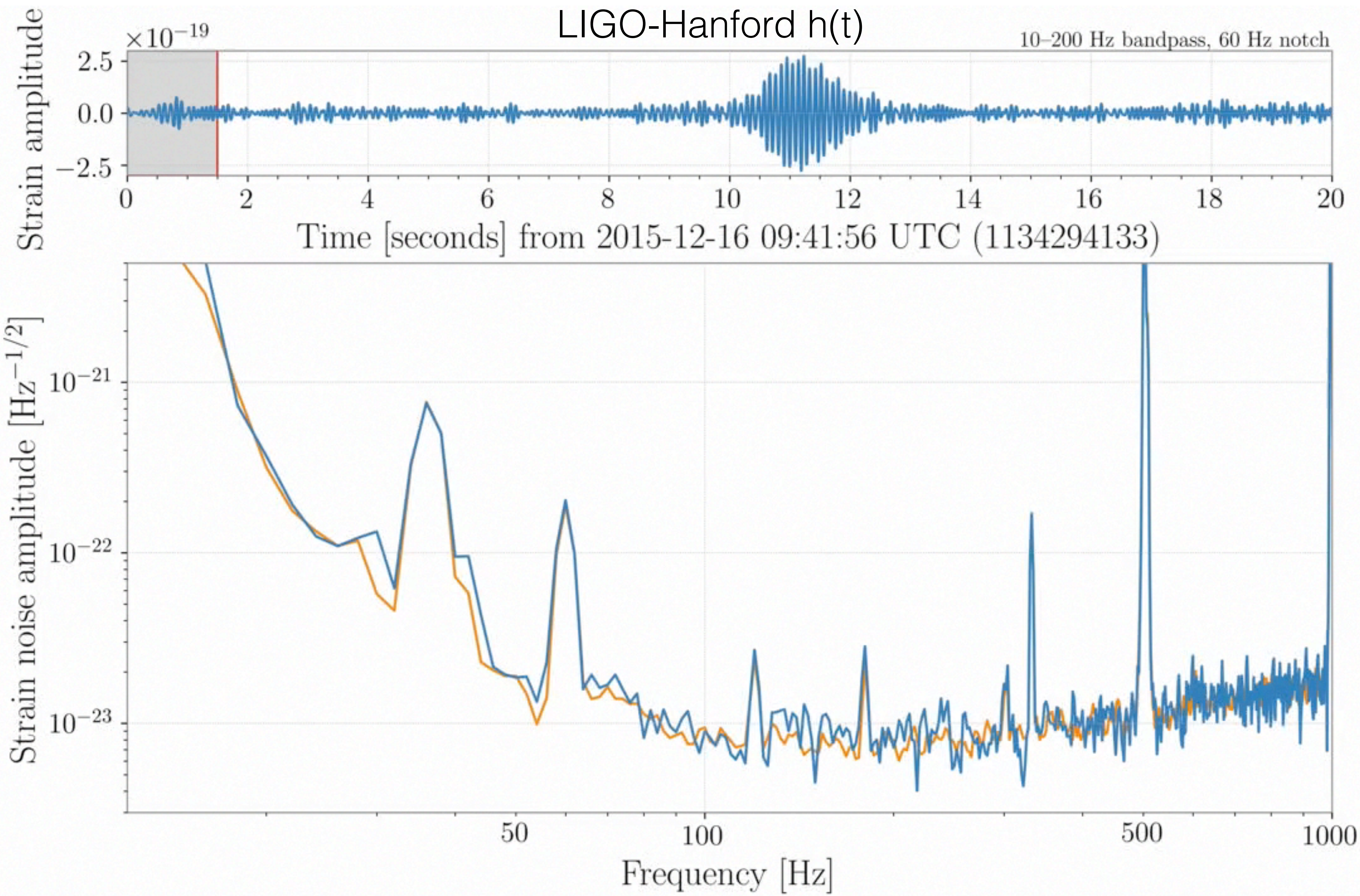

CQG 37, 055002 (2020)

# Phase w/ or w/o window




CQG 37, 055002 (2020)

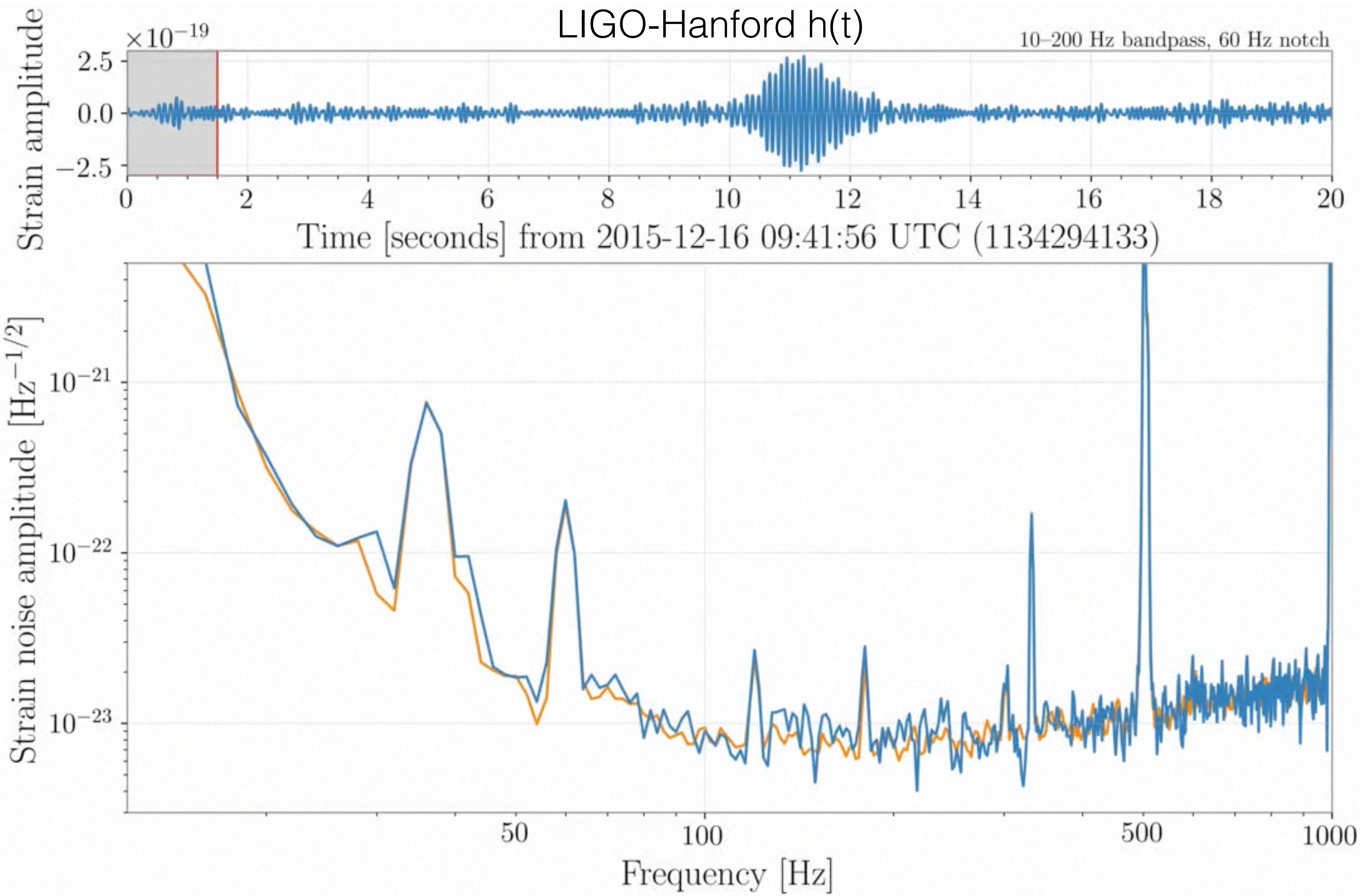
# LIGO data in the frequency domain




Made with GWpy by Duncan Macleod. Code: https://git.io/gwpy-ligo-scattering-animation

0.5 second FFT; 5 averages covering 1.5 seconds; 50% overlap

# LIGO data in the frequency domain

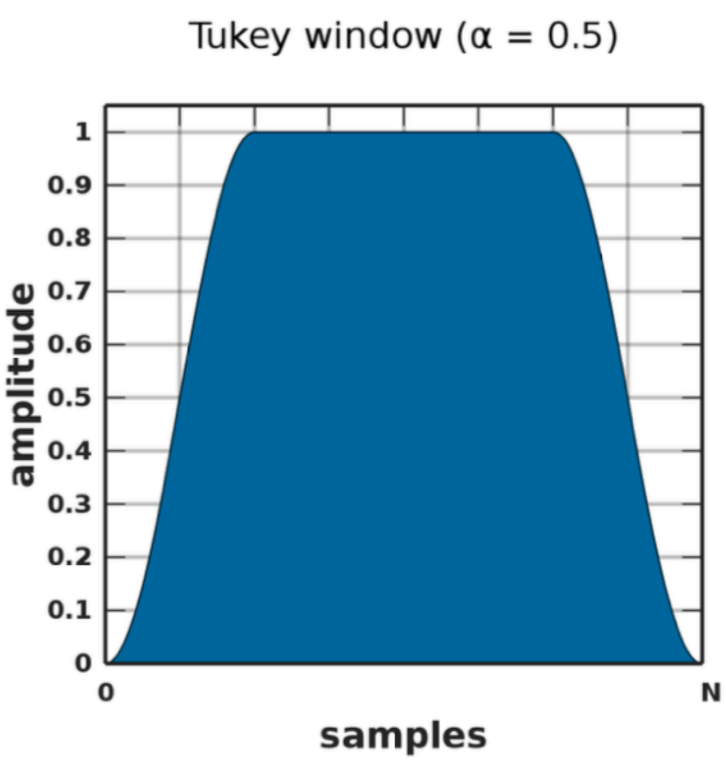


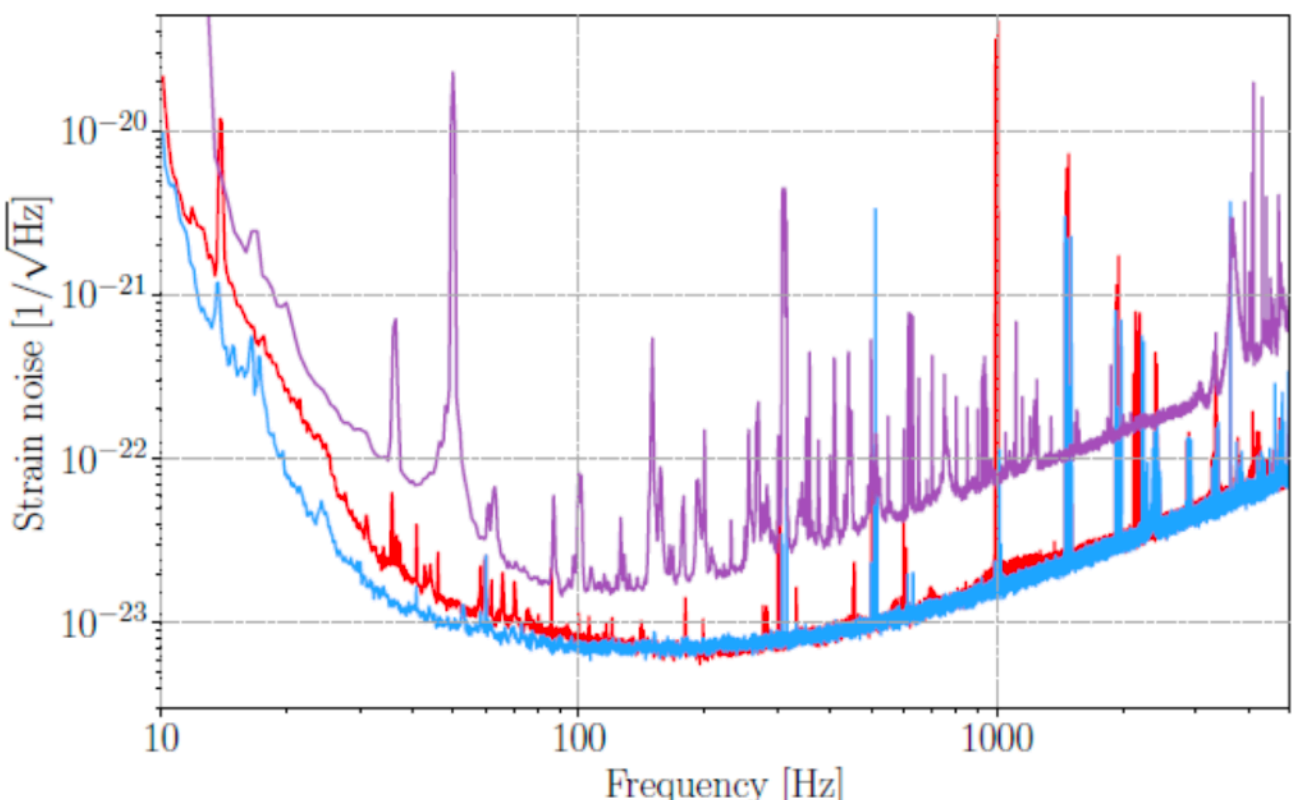

Made with GWpy by Duncan Macleod. Code: https://git.io/gwpy-ligo-scattering-animation

0.5 second FFT; 5 averages covering 1.5 seconds; 50% overlap

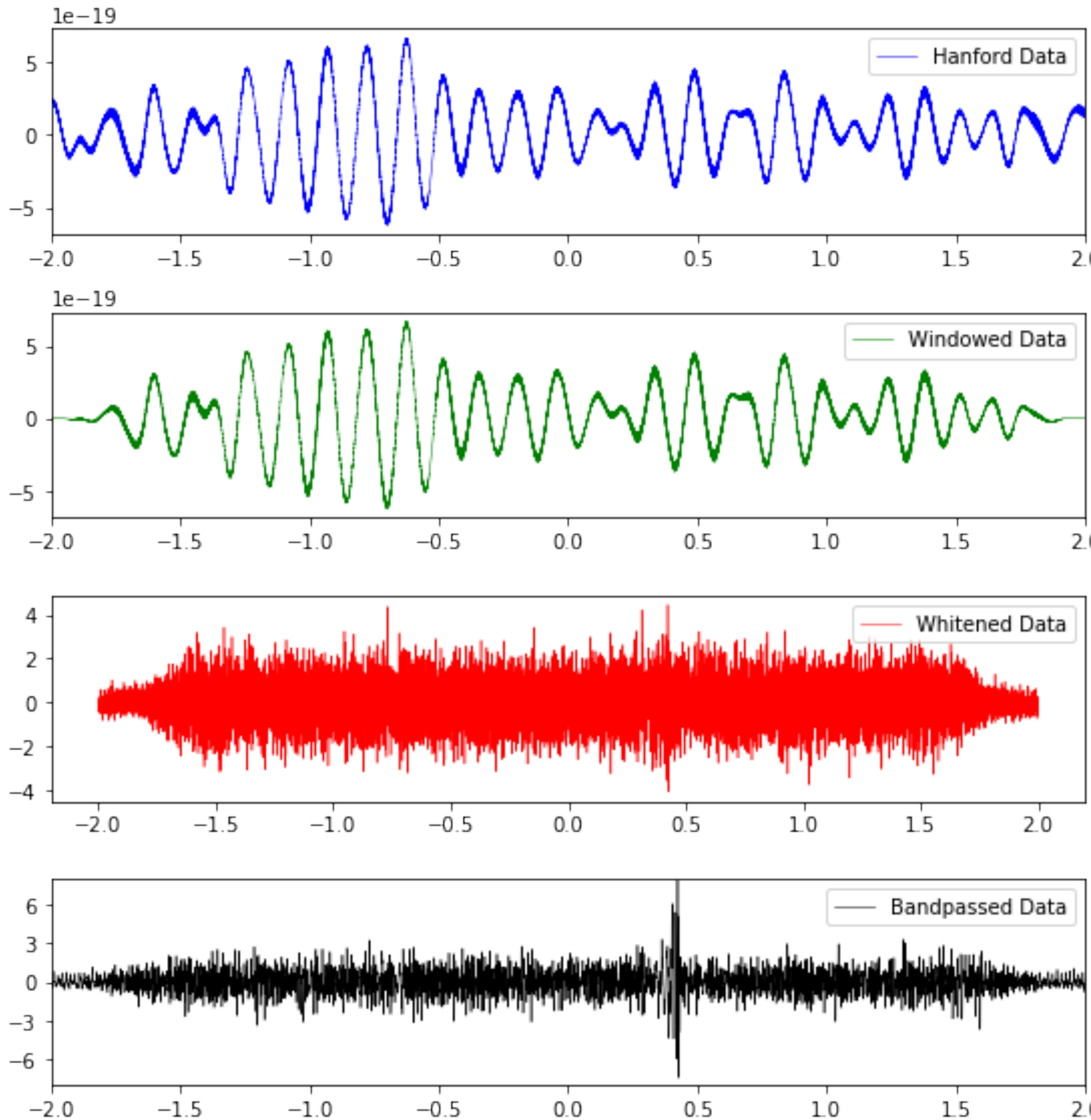
# Whitening

$$d(t) \xrightarrow{\text{FFT}} \tilde{d}(f) \xrightarrow{\text{Whiten}} \tilde{d}_w(f) = \frac{\tilde{d}(f)}{S_n^{1/2}(f)} \xrightarrow{\text{iFFT}} d_w(t)$$



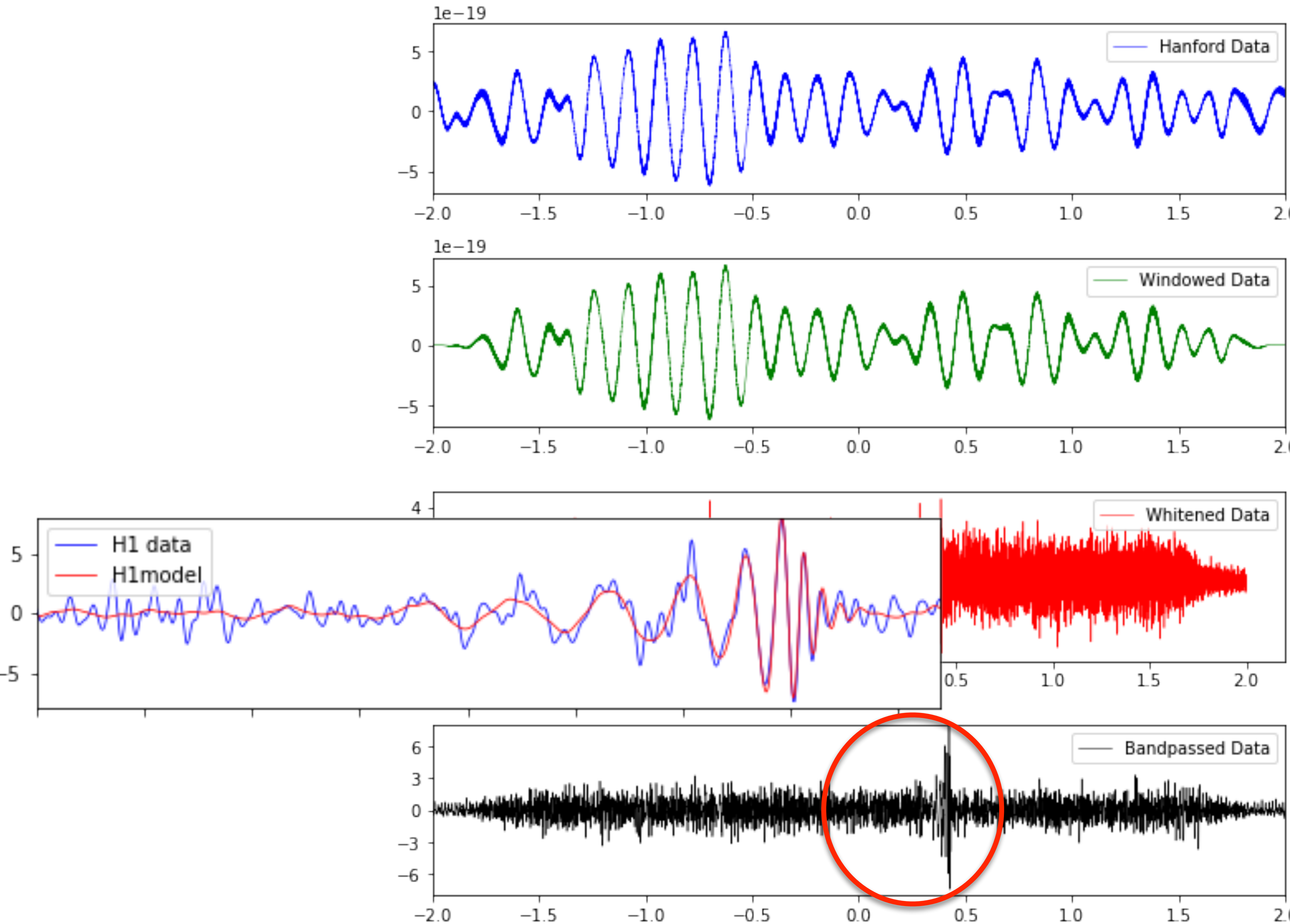



# whitening after applying window function




CQG 37, 055002 (2020)

# whitening after applying window function




CQG 37, 055002 (2020)

# GW data in time domain

```python
from gwosc.datasets import event_gps
gps = event_gps('GW190412')
print(gps)
```

```
1239082262.2
```

'G1' - GEO600

'H1' - LIGO-Hanford

'L1' - LIGO-Livingston

'V1' - (Advanced) Virgo

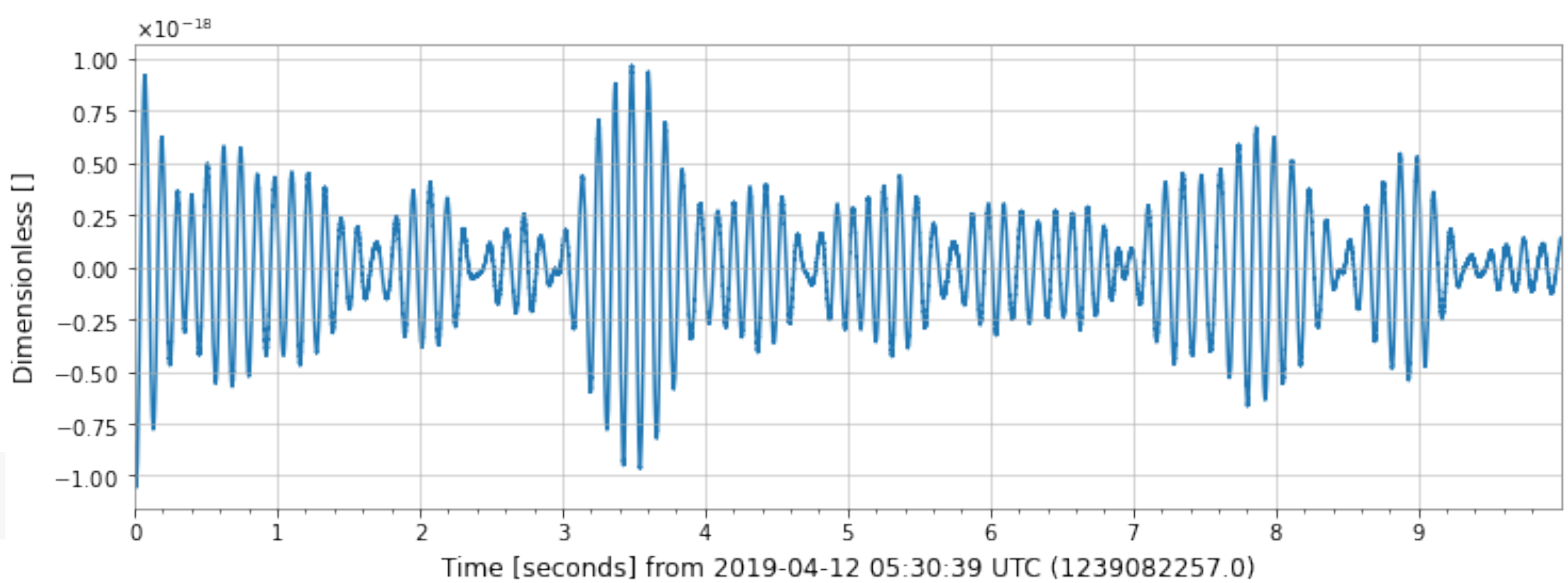
```python
segment = (int(gps)-5, int(gps)+5)
print(segment)
```

```
(1239082257, 1239082267)
```

```python
from gwpy.timeseries import TimeSeries
ldata = TimeSeries.fetch_open_data('L1', *segment, verbose=True)
print(ldata)
```

```
Fetched 1 URLs from www.gw-openscience.org for [1239082257 .. 1239082267))
Reading data... [Done]
TimeSeries([-8.42599982e-19, -8.52439382e-19, -8.60740967e-19,
             ...,  1.38851953e-19,  1.37762006e-19,
             1.38095492e-19]
           unit: dimensionless,
           t0: 1239082257.0 s,
           dt: 0.000244140625 s,
           name: Strain,
           channel: None)
```

```python
%matplotlib inline
plot = ldata.plot()
```




```python
print(ldata.sample_rate)
```

```
4096.0 Hz
```

# GW data in time domain

```
from gwosc.datasets import event_gps
gps = event_gps('GW190412')
print(gps)
```

```
1239082262.2
```

`'G1'` - GEO600
`'H1'` - LIGO-Hanford
`'L1'` - LIGO-Livingston
`'V1'` - (Advanced) Virgo

```
segment = (int(gps)-5, int(gps)+5)
print(segment)
```

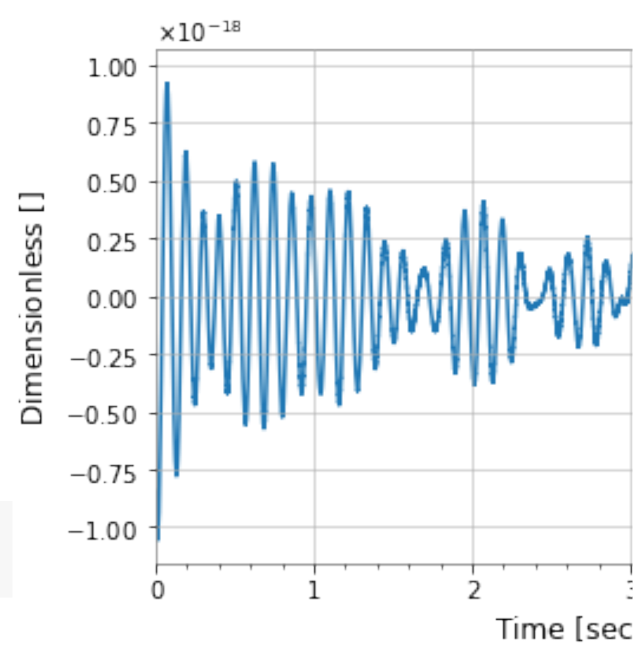
```
(1239082257, 1239082267)
```

```
from gwpy.timeseries import TimeSeries
ldata = TimeSeries.fetch_open_data('L1', *segment, verbose=True)
print(ldata)
```

```
Fetched 1 URLs from www.gw-openscience.org for [1239082257 ..
Reading data... [Done]
TimeSeries([-8.42599982e-19, -8.52439382e-19, -8.60740967e-19,
            ..., 1.38851953e-19, 1.37762006e-19,
             1.38095492e-19]
           unit: dimensionless,
           t0: 1239082257.0 s,
           dt: 0.000244140625 s,
           name: Strain,
           channel: None)
```

Read direct from downloaded file:
TimeSeries.read(filename, channel_name, start=t_start, end=t_end,format=format)

```
%matplotlib inline
plot = ldata.plot()
```




```
print(ldata.sample_rate)
```

```
4096.0 Hz
```

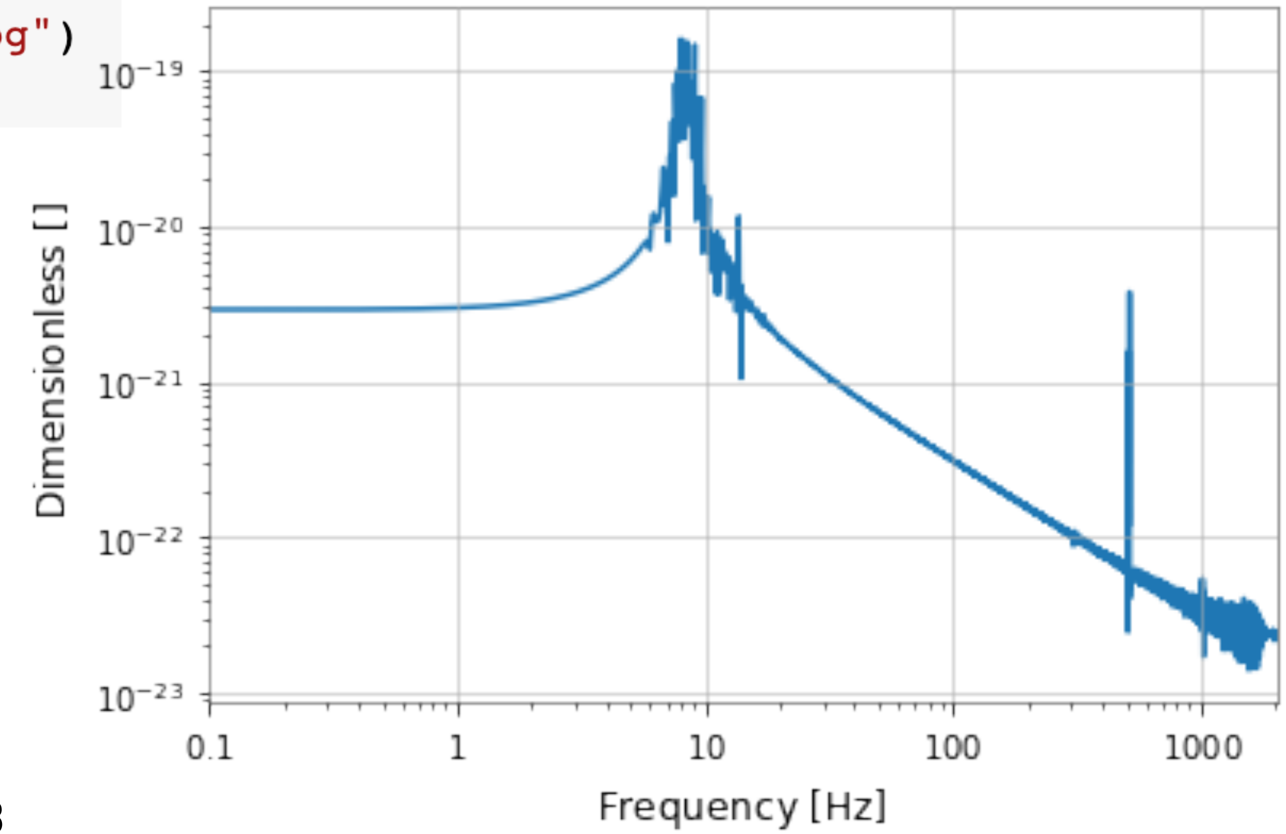
| Format | Read | Write | Auto-identify |
|---|---|---|---|
| csv | Yes | Yes | Yes |
| gwf | Yes | Yes | Yes |
| gwf.framecpp | Yes | Yes | No |
| gwf.framel | Yes | Yes | No |
| gwf.lalframe | Yes | Yes | No |
| hdf5 | Yes | Yes | Yes |
| hdf5.losc | Yes | No | No |
| txt | Yes | Yes | Yes |
| wav | Yes | No | No |

# GW data in frequency domain

```
1 fft = ldata.fft()
2 print(fft)
```

```
FrequencySeries([-1.45894353e-21+0.00000000e+00j,
                 -2.91834811e-21-4.52905623e-23j,
                 -2.91973217e-21-9.06203059e-23j, ...,
                 -2.38724887e-23+4.67871321e-26j,
                 -2.38346268e-23+1.80394122e-26j,
                 -2.38458080e-23+0.00000000e+00j]
                unit: dimensionless,
                f0: 0.0 Hz,
                df: 0.1 Hz,
                epoch: 1239082257.0,
                name: Strain,
                channel: None)
```

```
plot = fft.abs().plot(xscale="log", yscale="log")
plot.show(warn=False)
```

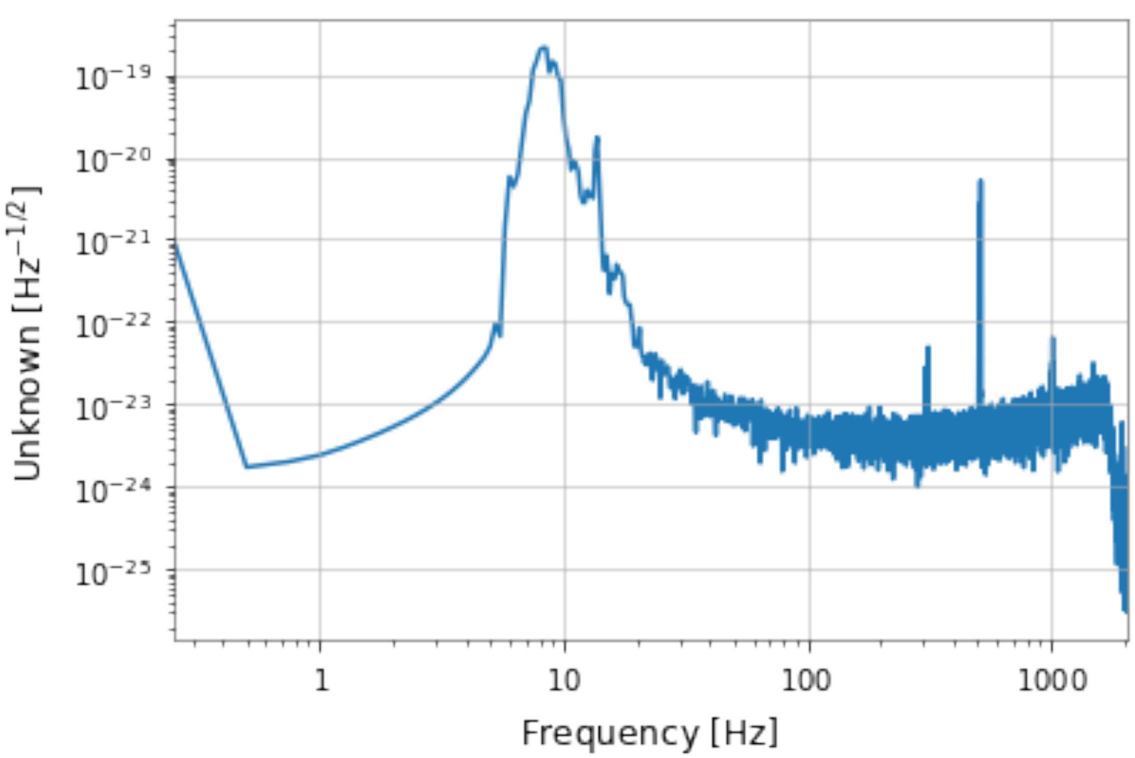


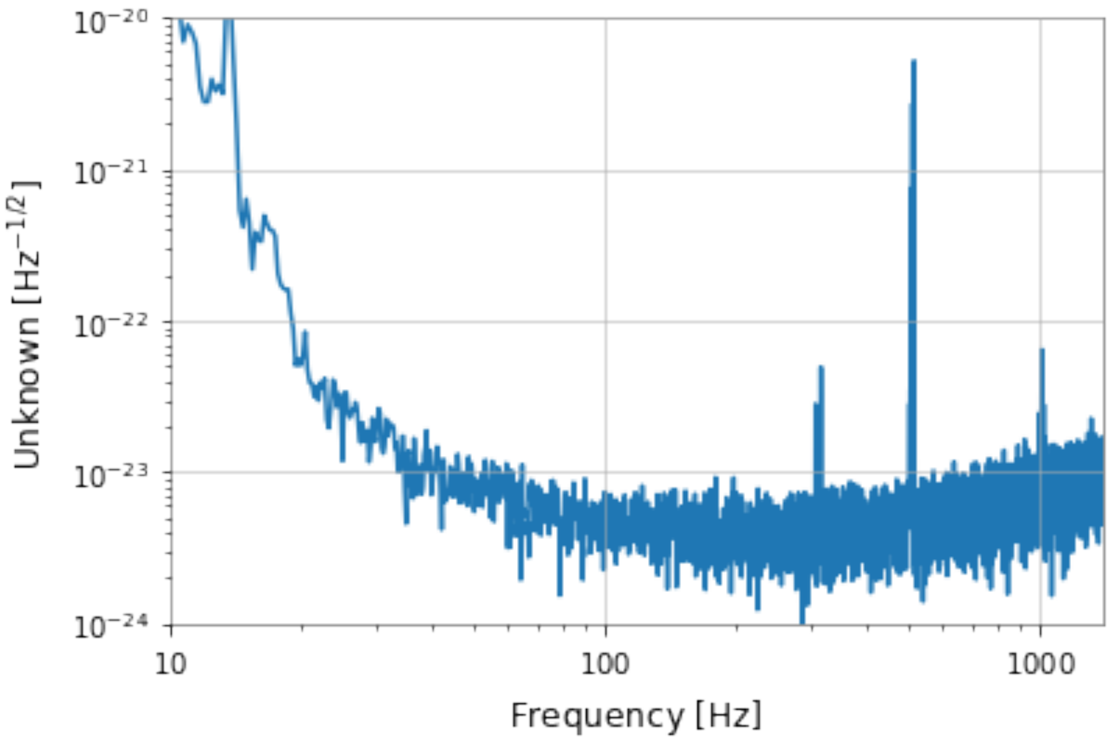
# GW data in frequency domain

```
from scipy.signal import get_window
window = get_window('hann', ldata.size)
lwin = ldata * window
```

```
fftamp = lwin.fft().abs()
plot = fftamp.plot(xscale="log", yscale="log")
plot.show(warn=False)
```

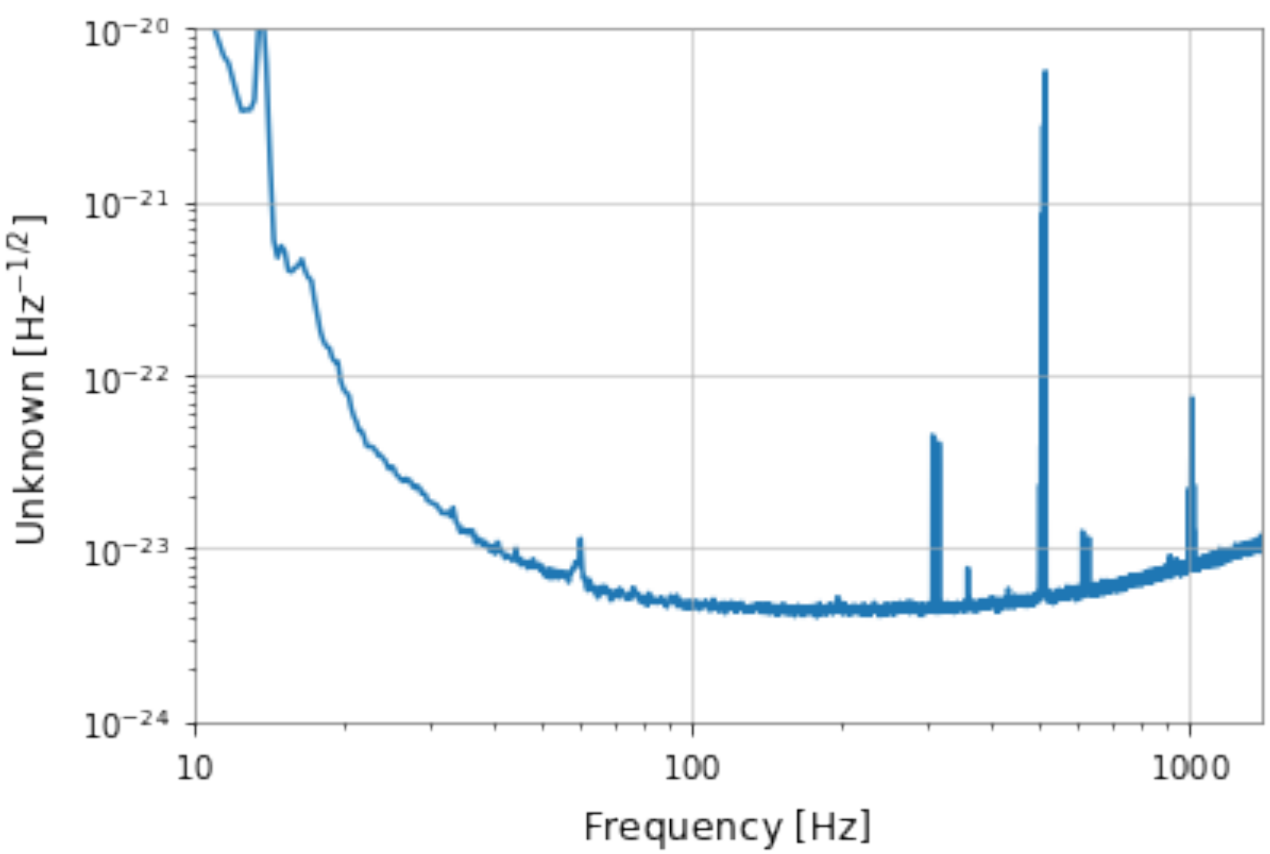
```
ax = plot.gca()
ax.set_xlim(10, 1400)
ax.set_ylim(1e-24, 1e-20)
plot
```

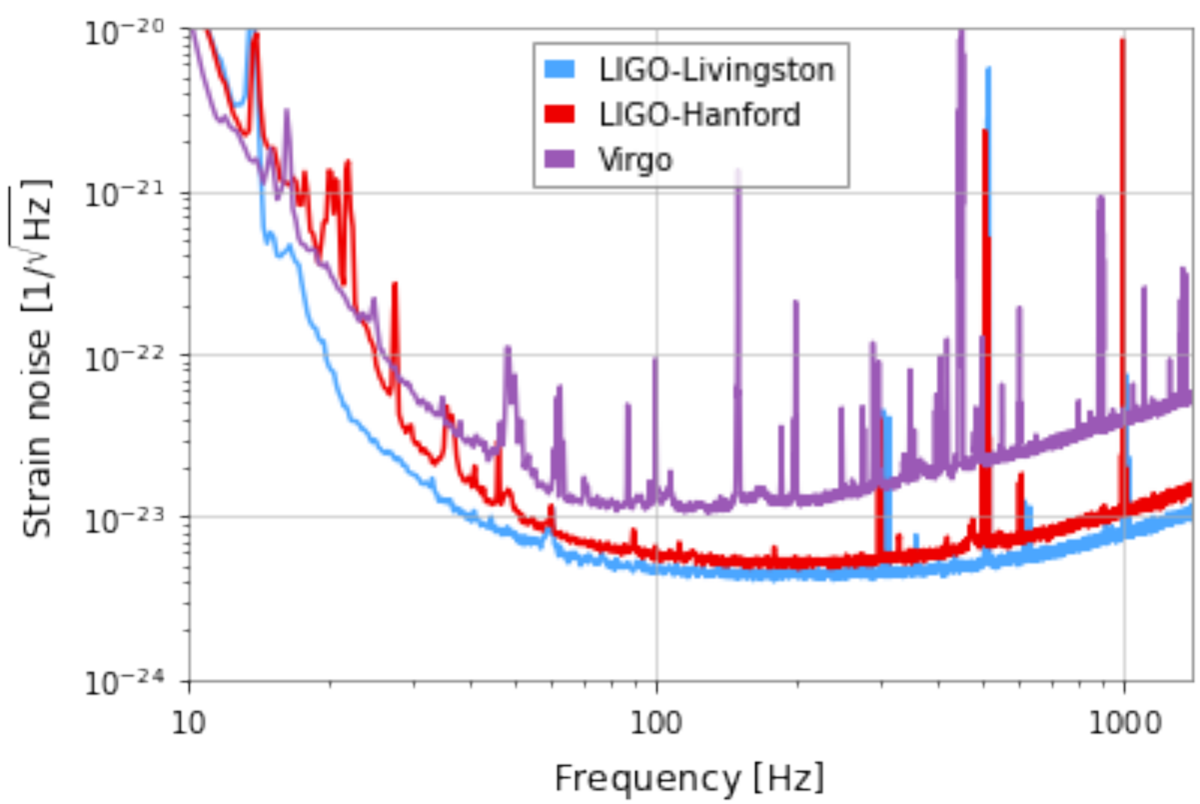


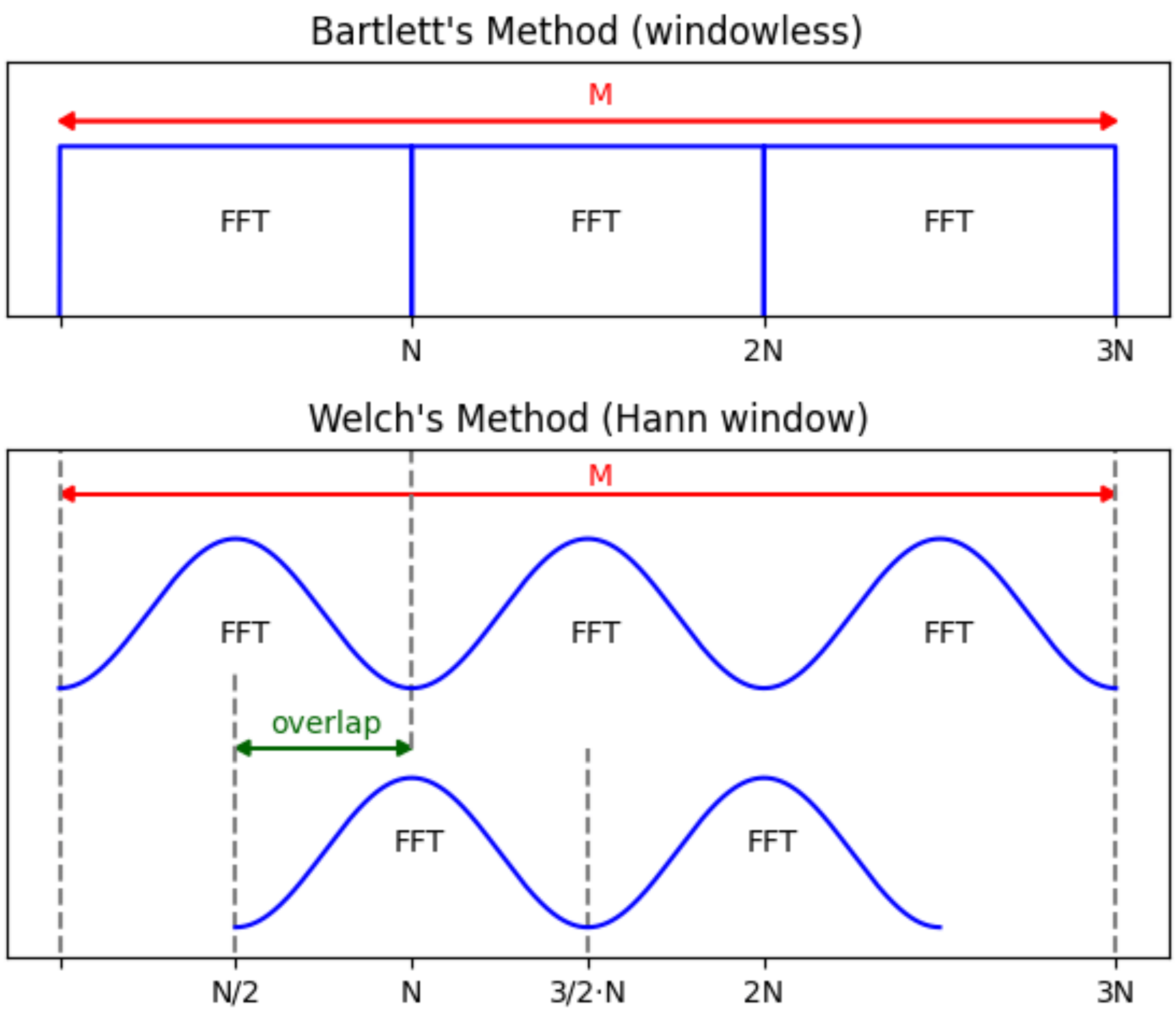
# GW data in frequency domain

```python
ldata2 = TimeSeries.fetch_open_data('L1', int(gps)-512, int(gps)+512, cache=True)
lasd2 = ldata2.asd(fftlength=4, method="median")
plot = lasd2.plot()
ax = plot.gca()
ax.set_xlim(10, 1400)
ax.set_ylim(1e-24, 1e-20)
plot.show(warn=False)
```



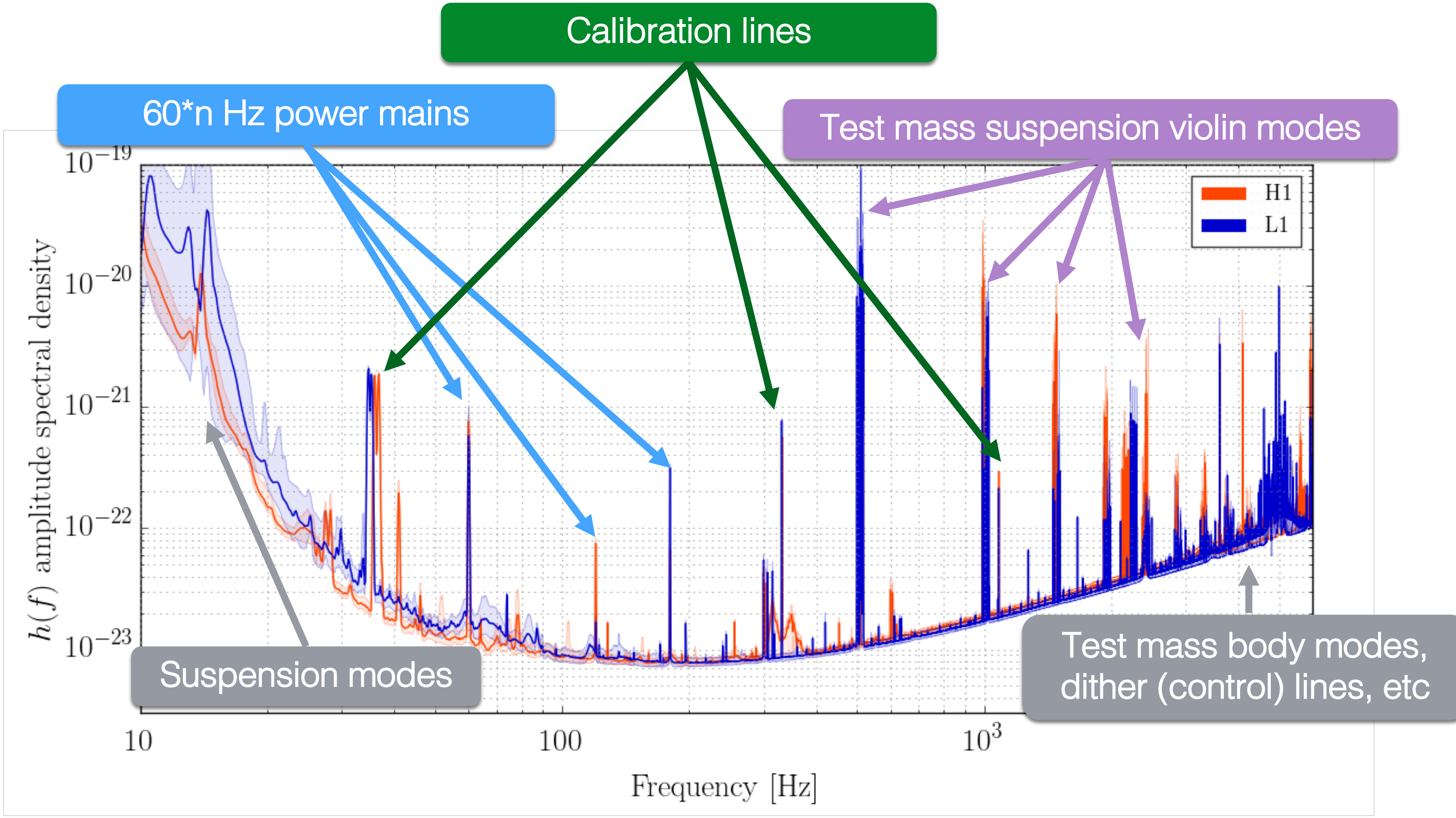



# Power Spectral Density - Welch method



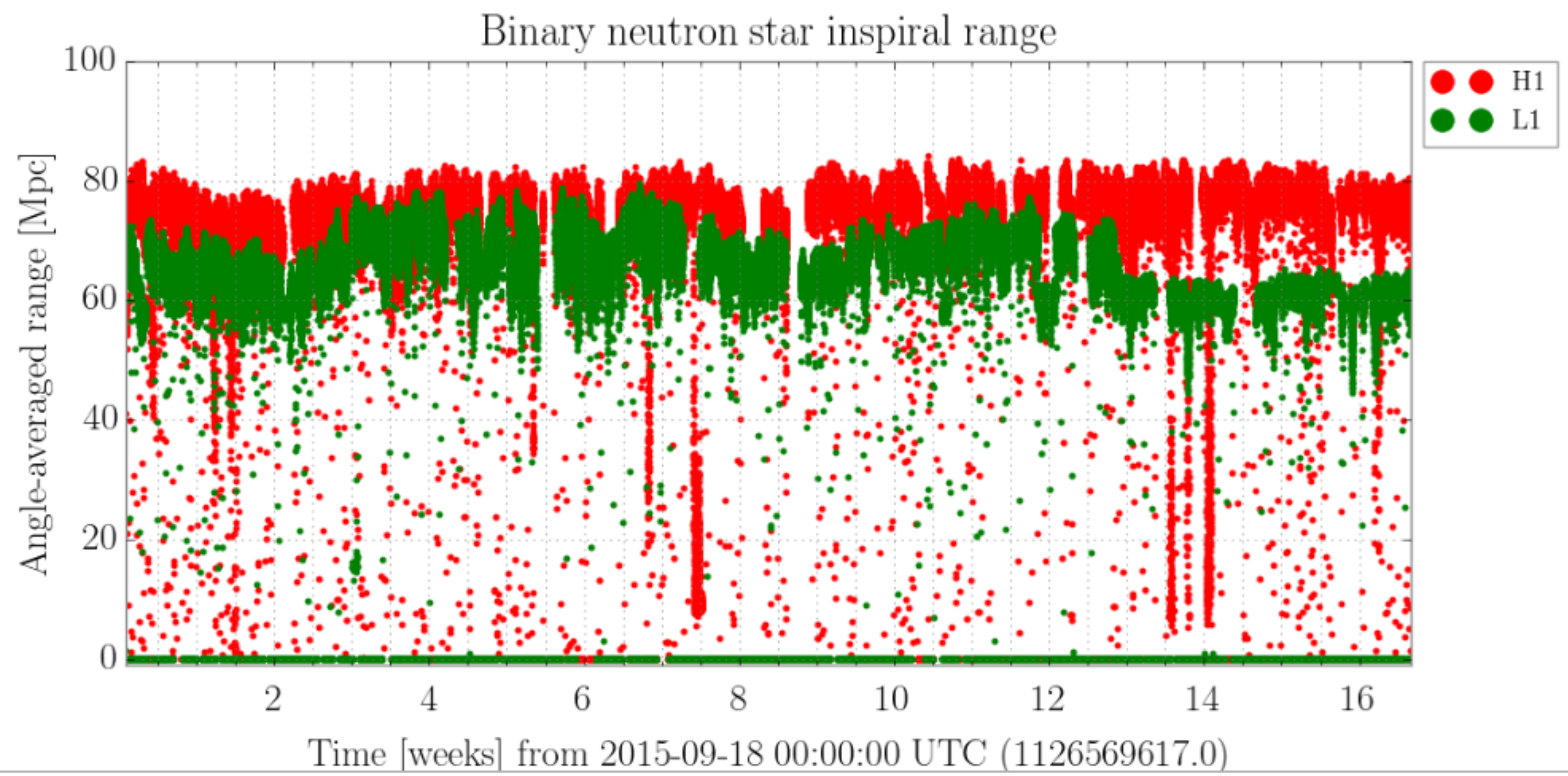

https://www.researchgate.net/figure/Welchs-and-Bartletts-methods-for-power-spectral-density-estimation-The-Bartletts_fig1_349283231

# Calibrated Strain noise spectral lines




Source: https://losc.ligo.org/events/GW150914/

# BNS Range




**BNS range** : The distance to which a LIGO detector can register a BNS signal with a single detector **signal-to-noise ratio (SNR) of 8**, averaged over source direction and orientation.
Each neutron star in the BNS system has a mass of **1.4 times the mass of the sun**, and negligible spin.

# BNS Range

Expected SNR

$$\varrho = \sqrt{4 \int_0^\infty \frac{|\tilde{h}(f)|^2}{S_n(f)} df}$$

$$= \left(\frac{1\ \mathrm{Mpc}}{D_{\mathrm{eff}}}\right) \sqrt{4 \mathcal{A}_{1\ \mathrm{Mpc}}^2(M, \mu) \int_0^\infty \frac{f^{-7/3}}{S_n(f)} df} \qquad \text{(D1)}$$

Horizon distance

$$D_{\mathrm{hor}} = \frac{1\ \mathrm{Mpc}}{\varrho} \sqrt{4 \mathcal{A}_{1\ \mathrm{Mpc}}^2(M, \mu) \int_0^\infty \frac{f^{-7/3}}{S_n(f)} df}, \qquad \text{(D2)}$$

snr=8, m1=m2=1.4 Msun, mu=0.7 Msun

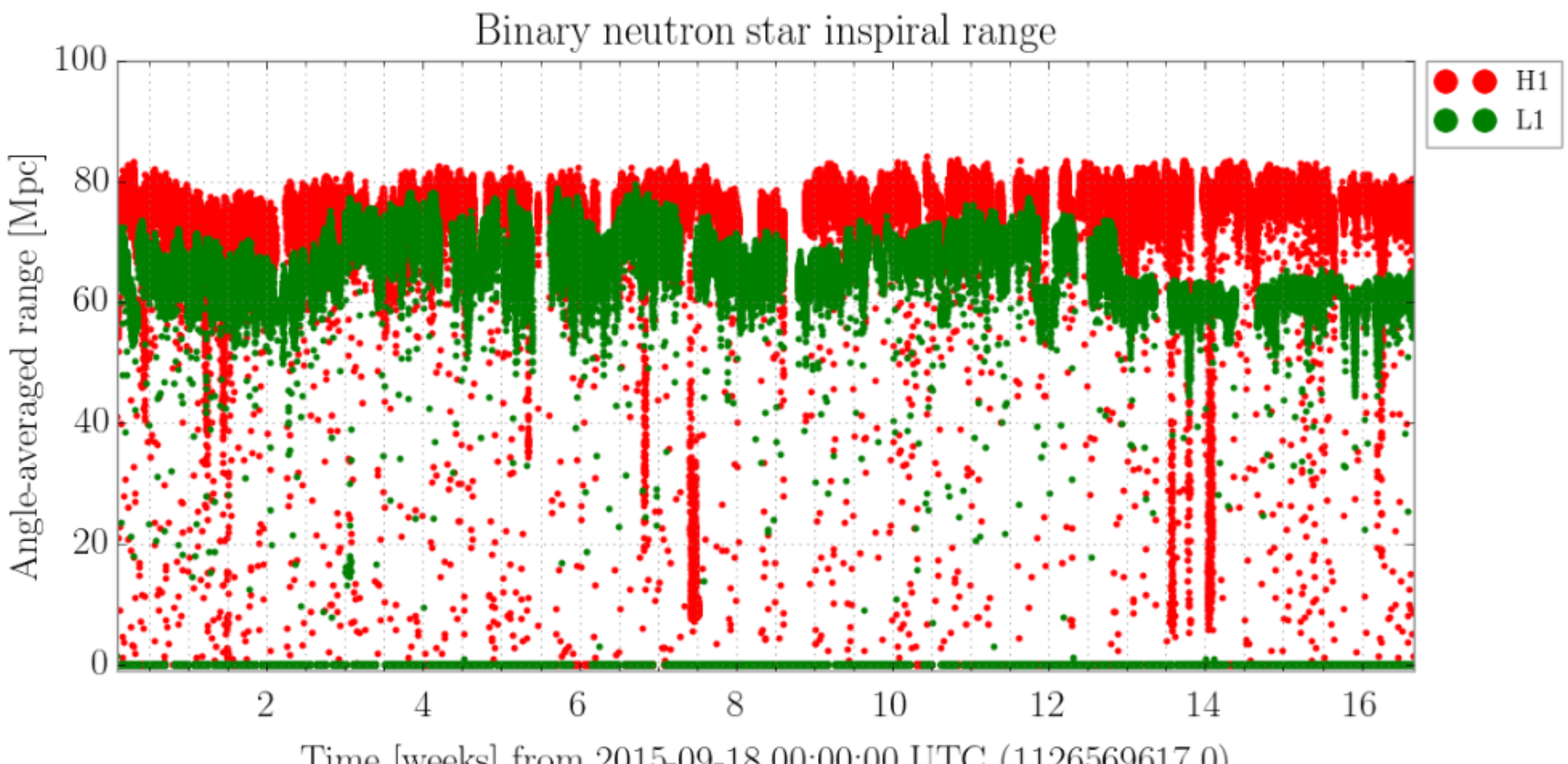
BNS range
(Sense-monitor range)

R = D_hor * F

1/F = 4/(3 * 1.84)^(1/3) ~ 2.12648

PHYSICAL REVIEW D 85, 122006 (2012)

# BNS Range




Step 1. Pick an event (GW150914,GW170104,GW170817,…..)
Step 2. Find the time segment of 1hr containing the event you pick
Step 3. Make BNS range plot for the time segment

**Hint: https://losc.ligo.org/s/events/GW170104/LOSC_Event_tutorial_GW170104.html#Binary-Neutron-Star-(BNS)-detection-range or search it in GWpy (https://gwpy.github.io/docs/latest/examples/index.html)**

# Example - BNS range by GWPy

First, we need to load some data. We can `fetch` the public data around the GW170817 BNS merger:

```
from gwpy.timeseries import TimeSeries
h1 = TimeSeries.fetch_open_data('H1', 1187006834, 1187010930, tag='C02')
l1 = TimeSeries.fetch_open_data('L1', 1187006834, 1187010930, tag='C02')
```

Then, we can measure the inspiral range directly:

```
from gwpy.astro import range_timeseries
h1range = range_timeseries(h1, 30, fftlength=4, fmin=10)
l1range = range_timeseries(l1, 30, fftlength=4, fmin=10)
```

We can now plot these trends to see the variation in LIGO sensitivity over an hour or so surrounding GW170817:

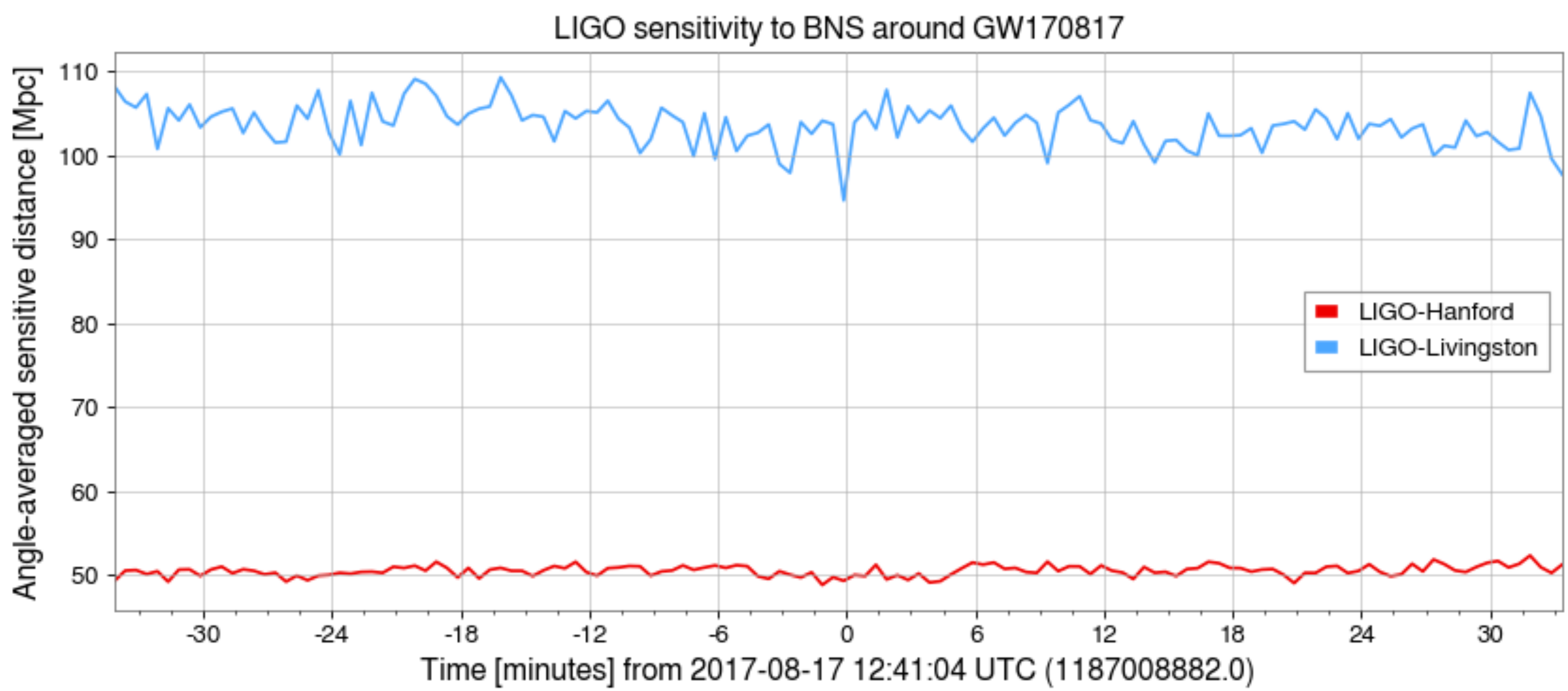
```
plot = h1range.plot(
    label='LIGO-Hanford', color='gwpy:ligo-hanford', figsize=(12, 5))
ax = plot.gca()
ax.plot(l1range, label='LIGO-Livingston', color='gwpy:ligo-livingston')
ax.set_ylabel('Angle-averaged sensitive distance [Mpc]')
ax.set_title('LIGO sensitivity to BNS around GW170817')
ax.set_epoch(1187008882)  # <- set 0 on plot to GW170817
ax.legend()
plot.show()
```

https://gwpy.github.io/docs/latest/examples/miscellaneous/range-timeseries.html

# Example - BNS range by GWPy

First, we need to load some data. We can **fetch** the public data around the GW170817 BNS merger:

```
from gwpy.timeseries import TimeSeries
h1 = TimeSeries.fetch_open_data('H1', 1187006834, 1187010930, tag='C02')
l1 = TimeSeries.fetch_open_data('L1', 1187006834, 1187010930, tag='C02')
```




```
ax.legend()
plot.show()
```

https://gwpy.github.io/docs/latest/examples/miscellaneous/range-timeseries.html

# Gravitational-wave event searches

There are two types of searches, online and offline

- Online searches are low-latency searches which aim to get quick results in order to get rapid alerts of events
- Offline searches use archived data using more computationally expensive techniques to get deeper searches into the data

What searches are there?

- Templated searches:
    - GstLAL - Online and Offline, lscsoft.docs.ligo.org/gstlal
    - **PyCBC** - Online and **Offline**, pycbc.org
    - MBTA - Online and Offline, T. Adams et al (2016)
    - SPIIR - Online only, Q. Chu (2017)
    - IAS - Offline only, Venumadhav et al. (2020)
- Non-templated search
    - cWB - Online and Offline gwburst.gitlab.io

PyCBC

gstLAL

cWB

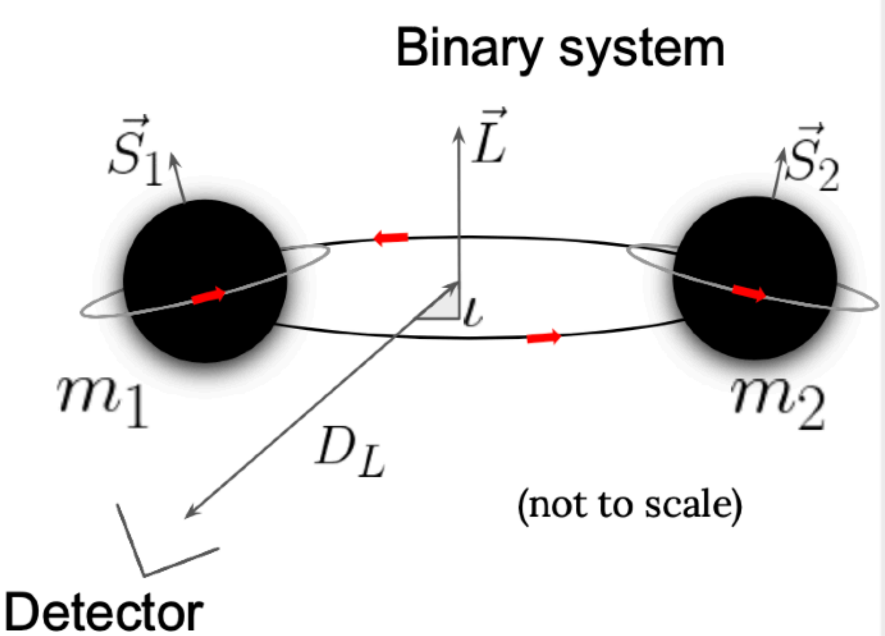
G. Davis in GW ODW #4, 2021

# Modelling colliding black holes

What will the signals from these systems look like in the data?

The signal from a binary system made up of black holes will be described by fifteen parameters

- Intrinsic parameters:
  - Component Masses: $m_1$ $m_2$
  - Component spins in each direction: $s_{1x}$ $s_{1y}$ $s_{1z}$ $s_{2x}$ $s_{2y}$ $s_{2z}$
- Extrinsic Parameters:
  - Location: Right Ascension and Declination
  - Inclination angle between line of sight and orbital plane, $\iota$
  - Polarisation angle,
  - Phase at coalescence
  - Luminosity distance, $D_L$
  - Time of coalescence




G. Davis in GW ODW #4, 2021

# Generating Waveforms

```
from pycbc.waveform import get_td_waveform
import pylab
```

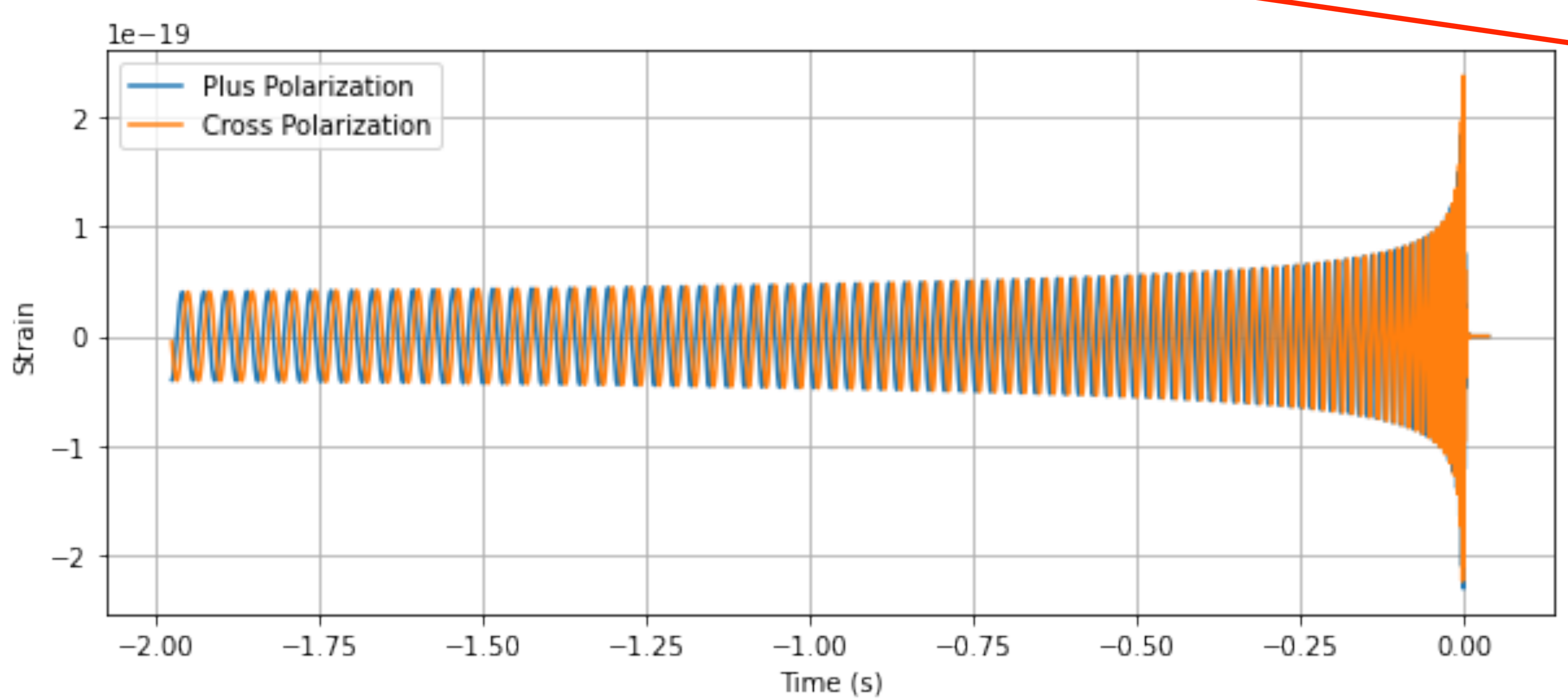
Spinning Effective One Body + Numerical Relativity

```
hp, hc = get_td_waveform(approximant="SEOBNRv4_opt",
                         mass1=10,
                         mass2=10,
                         delta_t=1.0/16384,
                         f_lower=30)
```

$M_\odot$ (solar mass)

Seconds

Hz



# Generating Waveforms

```
from pycbc.waveform import get_td_waveform
import pylab
```

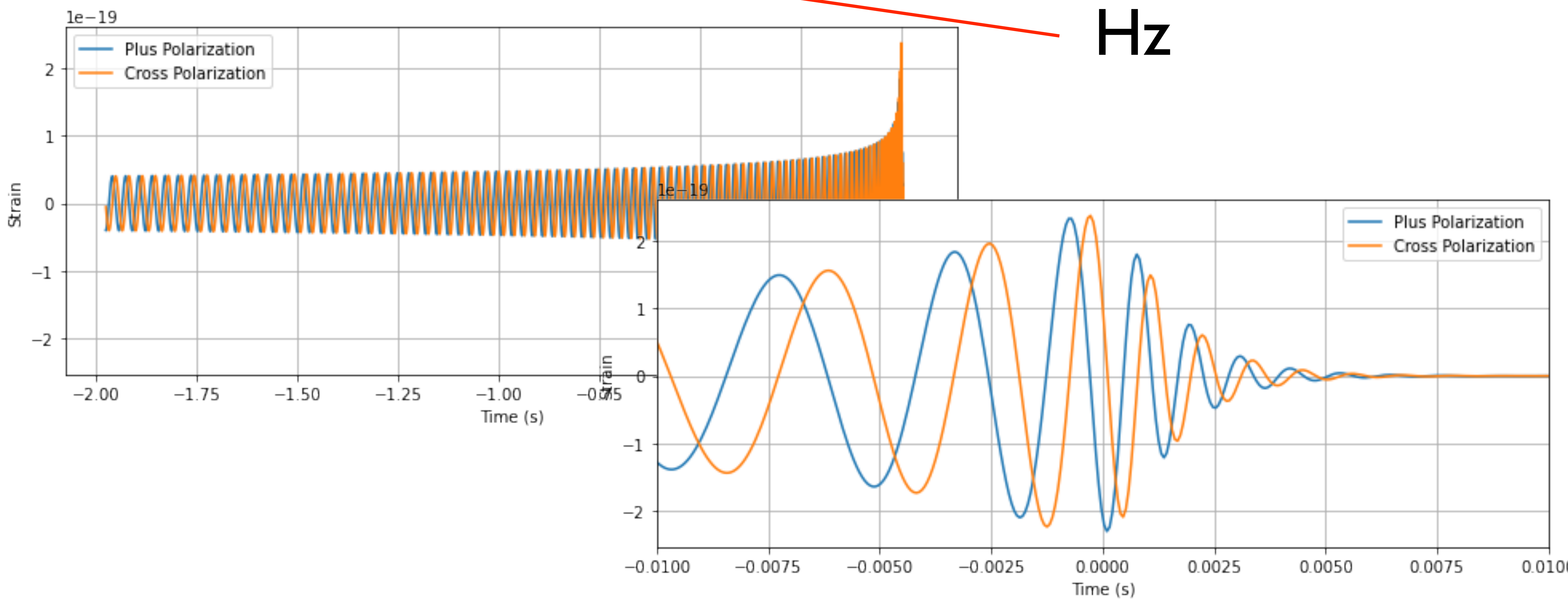
Spinning Effective One Body + Numerical Relativity

```
hp, hc = get_td_waveform(approximant="SEOBNRv4_opt",
                         mass1=10,
                         mass2=10,
                         delta_t=1.0/16384,
                         f_lower=30)
```

M⊙ (solar mass)

Seconds

Hz



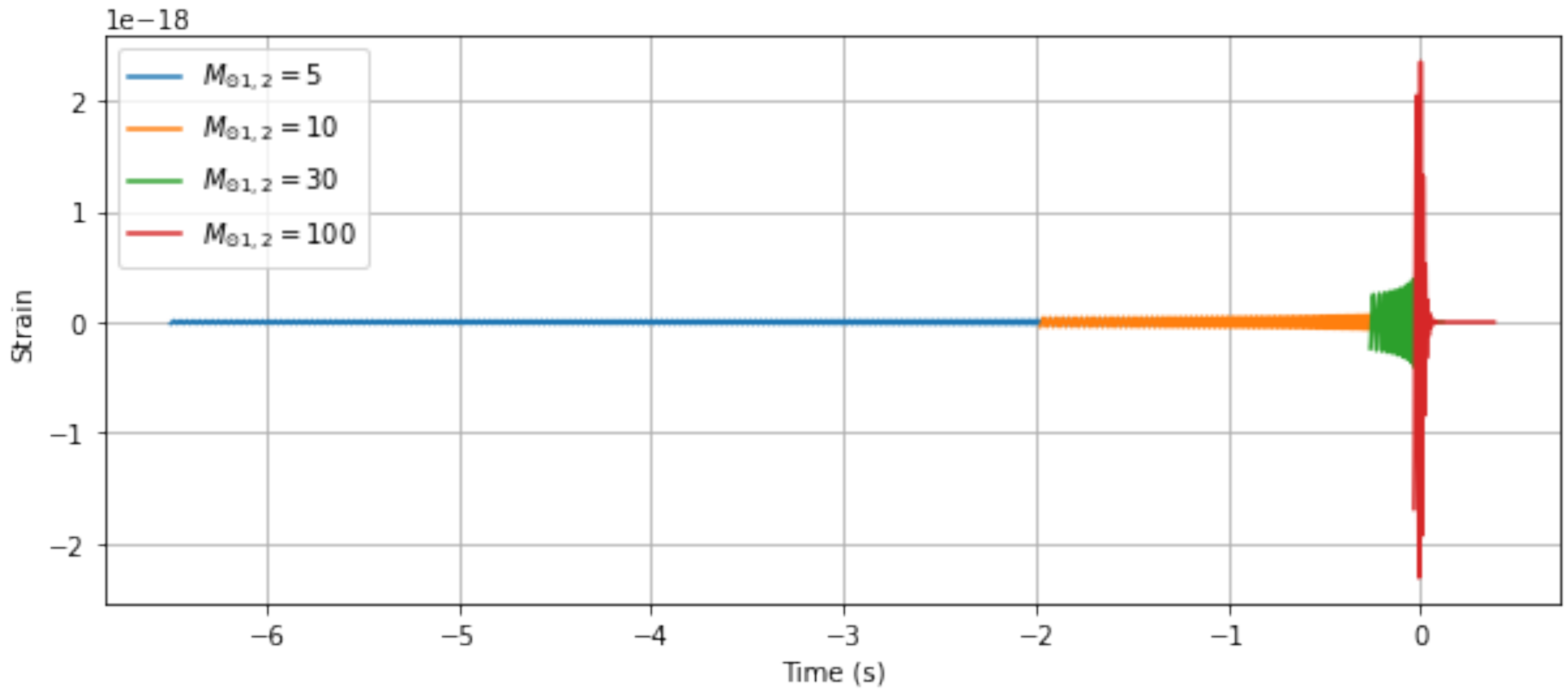
# Waveform Approximants

```
from pycbc.waveform import td_approximants, fd_approximants
print('Time domain waveform approximants: ',td_approximants())
print('Frequency domain waveform approximants: ', fd_approximants())
```

```
Time domain waveform approximants:  ['TaylorT1', 'TaylorT2', 'TaylorT3', 'SpinTaylorT1', 'SpinTaylorT4',
'SpinTaylorT5', 'PhenSpinTaylor', 'PhenSpinTaylorRD', 'EOBNRv2', 'EOBNRv2HM', 'TEOBResum_ROM', 'SEOBNRv1',
'SEOBNRv2', 'SEOBNRv2_opt', 'SEOBNRv3', 'SEOBNRv3_pert', 'SEOBNRv3_opt', 'SEOBNRv3_opt_rk4', 'SEOBNRv4',
'SEOBNRv4_opt', 'SEOBNRv4P', 'SEOBNRv4PHM', 'SEOBNRv2T', 'SEOBNRv4T', 'SEOBNRv4_ROM_NRTidalv2',
'SEOBNRv4_ROM_NRTidalv2_NSBH', 'HGimri', 'IMRPhenomA', 'IMRPhenomB', 'IMRPhenomC', 'IMRPhenomD',
'IMRPhenomD_NRTidalv2', 'IMRPhenomNSBH', 'IMRPhenomHM', 'IMRPhenomPv2', 'IMRPhenomPv2_NRTidal',
'IMRPhenomPv2_NRTidalv2', 'TaylorEt', 'TaylorT4', 'EccentricTD', 'SpinDominatedWf', 'NR_hdf5', 'NRSur7dq2',
'NRSur7dq4', 'SEOBNRv4HM', 'NRHybSur3dq8', 'IMRPhenomXAS', 'IMRPhenomXHM', 'IMRPhenomPv3', 'IMRPhenomPv3HM',
'IMRPhenomXP', 'IMRPhenomXPHM', 'TEOBResumS', 'IMRPhenomT', 'IMRPhenomTHM', 'TaylorF2', 'SEOBNRv1_ROM_EffectiveSpin',
'SEOBNRv1_ROM_DoubleSpin', 'SEOBNRv2_ROM_EffectiveSpin', 'SEOBNRv2_ROM_DoubleSpin', 'EOBNRv2_ROM', 'EOBNRv2HM_ROM',
'SEOBNRv2_ROM_DoubleSpin_HI', 'SEOBNRv4_ROM', 'SEOBNRv4HM_ROM', 'IMRPhenomD_NRTidal', 'SpinTaylorF2', 'TaylorF2NL',
'PreTaylorF2', ‘SpinTaylorF2_SWAPPER']

Frequency domain waveform approximants:  ['EccentricFD', 'TaylorF2', 'TaylorF2Ecc', 'TaylorF2NLTides',
'TaylorF2RedSpin', 'TaylorF2RedSpinTidal', 'SpinTaylorF2', 'EOBNRv2_ROM', 'EOBNRv2HM_ROM',
'SEOBNRv1_ROM_EffectiveSpin', 'SEOBNRv1_ROM_DoubleSpin', 'SEOBNRv2_ROM_EffectiveSpin', 'SEOBNRv2_ROM_DoubleSpin',
'SEOBNRv2_ROM_DoubleSpin_HI', 'Lackey_Tidal_2013_SEOBNRv2_ROM', 'SEOBNRv4_ROM', 'SEOBNRv4HM_ROM',
'SEOBNRv4_ROM_NRTidal', 'SEOBNRv4_ROM_NRTidalv2', 'SEOBNRv4_ROM_NRTidalv2_NSBH', 'SEOBNRv4T_surrogate', 'IMRPhenomA',
'IMRPhenomB', 'IMRPhenomC', 'IMRPhenomD', 'IMRPhenomD_NRTidal', 'IMRPhenomD_NRTidalv2', 'IMRPhenomNSBH',
'IMRPhenomHM', 'IMRPhenomP', 'IMRPhenomPv2', 'IMRPhenomPv2_NRTidal', 'IMRPhenomPv2_NRTidalv2', 'SpinTaylorT4Fourier',
'SpinTaylorT5Fourier', 'NRSur4d2s', 'IMRPhenomXAS', 'IMRPhenomXHM', 'IMRPhenomPv3', 'IMRPhenomPv3HM', 'IMRPhenomXP',
'IMRPhenomXPHM', 'SpinTaylorF2_SWAPPER', 'TaylorF2NL', 'PreTaylorF2', 'multiband', 'TaylorF2_INTERP', 'SpinTaylorT5',
'SEOBNRv1_ROM_EffectiveSpin_INTERP', 'SEOBNRv1_ROM_DoubleSpin_INTERP', 'SEOBNRv2_ROM_EffectiveSpin_INTERP',
'SEOBNRv2_ROM_DoubleSpin_INTERP', 'EOBNRv2_ROM_INTERP', 'EOBNRv2HM_ROM_INTERP', 'SEOBNRv2_ROM_DoubleSpin_HI_INTERP',
'SEOBNRv4_ROM_INTERP', 'SEOBNRv4HM_ROM_INTERP', 'SEOBNRv4', 'SEOBNRv4P', 'IMRPhenomC_INTERP', 'IMRPhenomD_INTERP',
'IMRPhenomPv2_INTERP', 'IMRPhenomD_NRTidal_INTERP', 'IMRPhenomPv2_NRTidal_INTERP', 'IMRPhenomHM_INTERP',
'IMRPhenomPv3HM_INTERP', 'IMRPhenomXHM_INTERP', 'IMRPhenomXPHM_INTERP', 'SpinTaylorF2_INTERP', 'TaylorF2NL_INTERP',
'PreTaylorF2_INTERP', 'SpinTaylorF2_SWAPPER_INTERP']
```

# WF w/ different masses

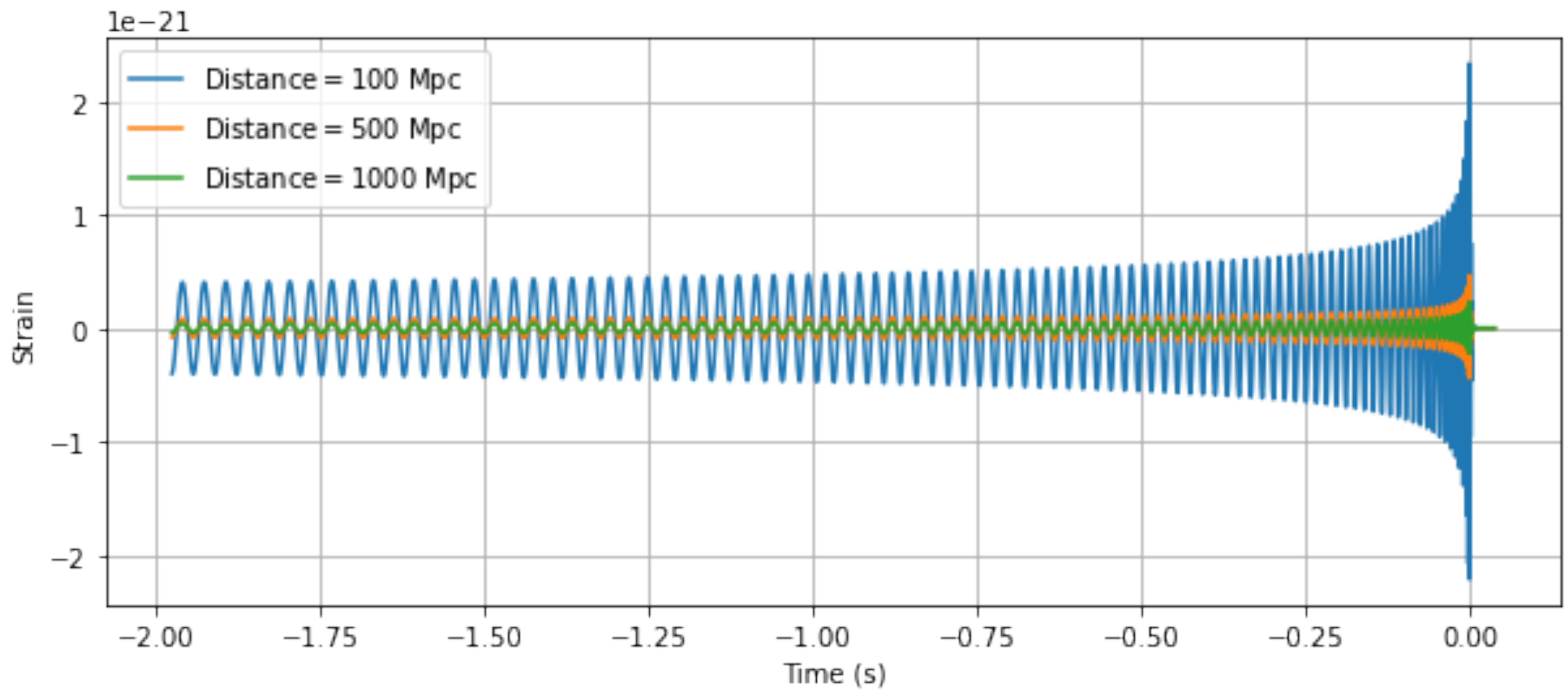
```
pylab.figure(figsize=pylab.figaspect(0.4))
for m in [5, 10, 30, 100]:
    hp, hc = get_td_waveform(approximant="SEOBNRv4_opt",
                             mass1=m,
                             mass2=m,
                             delta_t=1.0/4096,
                             f_lower=30)

    pylab.plot(hp.sample_times, hp, label='$M_{\odot 1,2}=%s$' % m)
```



# WF w/ different distances

```
pylab.figure(figsize=pylab.figaspect(0.4))
for d in [100, 500, 1000]:
    hp, hc = get_td_waveform(approximant="SEOBNRv4_opt",
                             mass1=10,
                             mass2=10,
                             delta_t=1.0/4096,
                             f_lower=30,
                             distance=d)

    pylab.plot(hp.sample_times, hp, label='Distance$=%s$ Mpc' % d)
```

Mpc



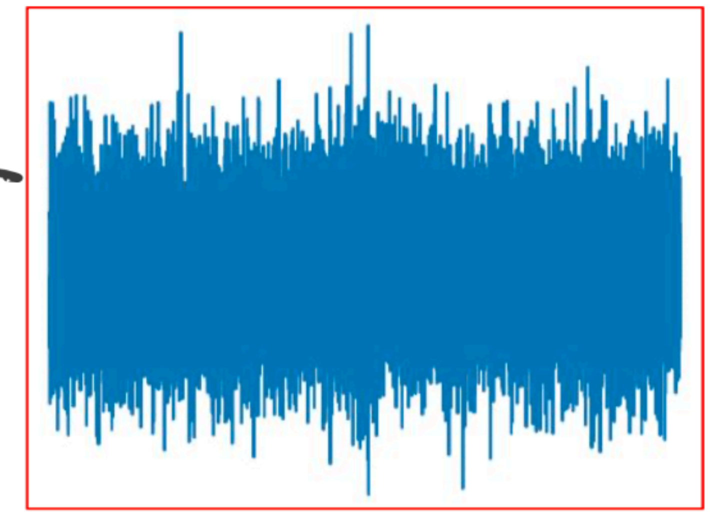
# Matched Filtering

Optimal for signals:

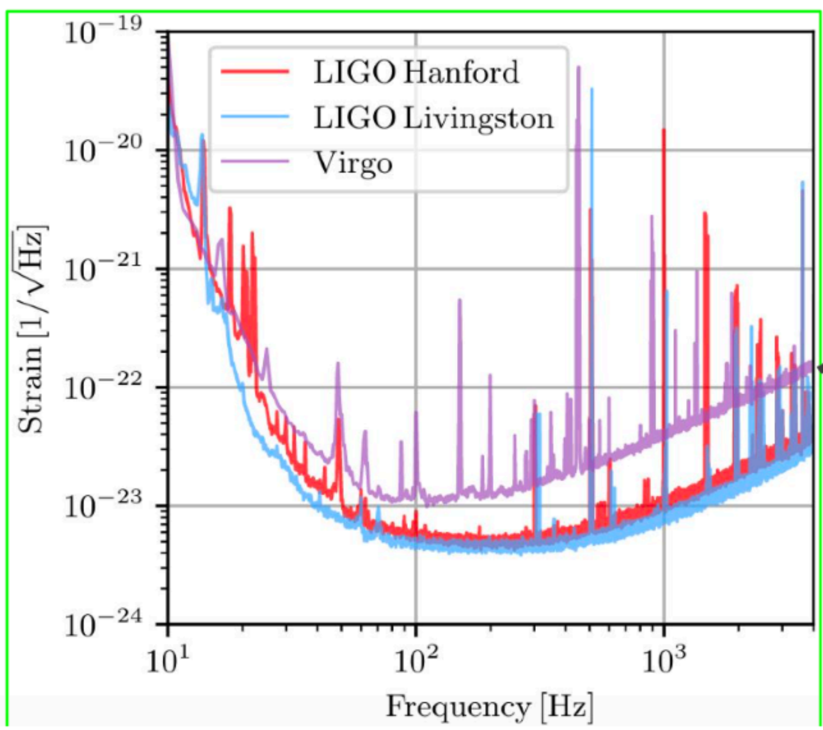
- in stationary Gaussian noise
- with known PSD

(Wainstein and Zubakov, 1962)

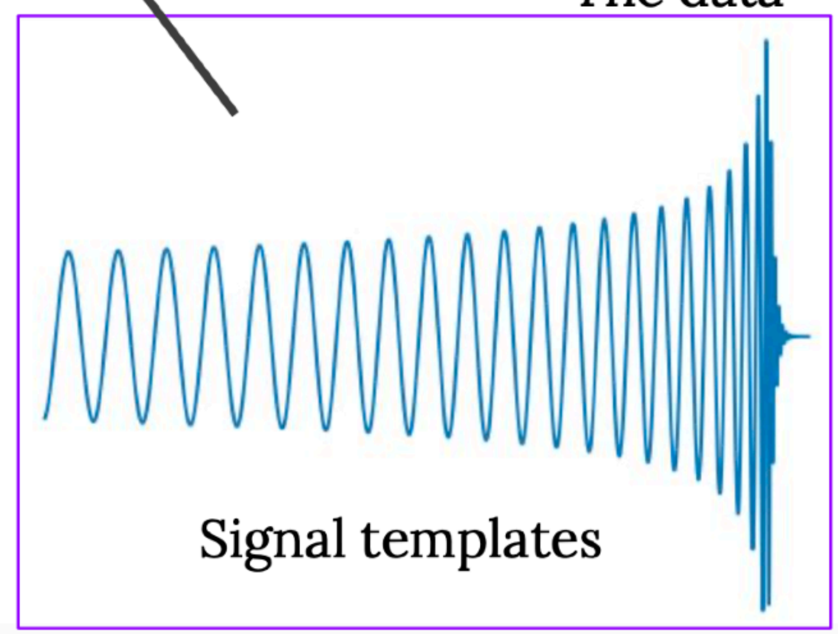
$$(s|h) = 4\Re \int_0^\infty \frac{\tilde{s}(f)\,\tilde{h}^\star(f)}{S_h(f)}\,\mathrm{d}f$$



The data




PSD




G. Davis in GW ODW #4, 2021

# Matched Filter

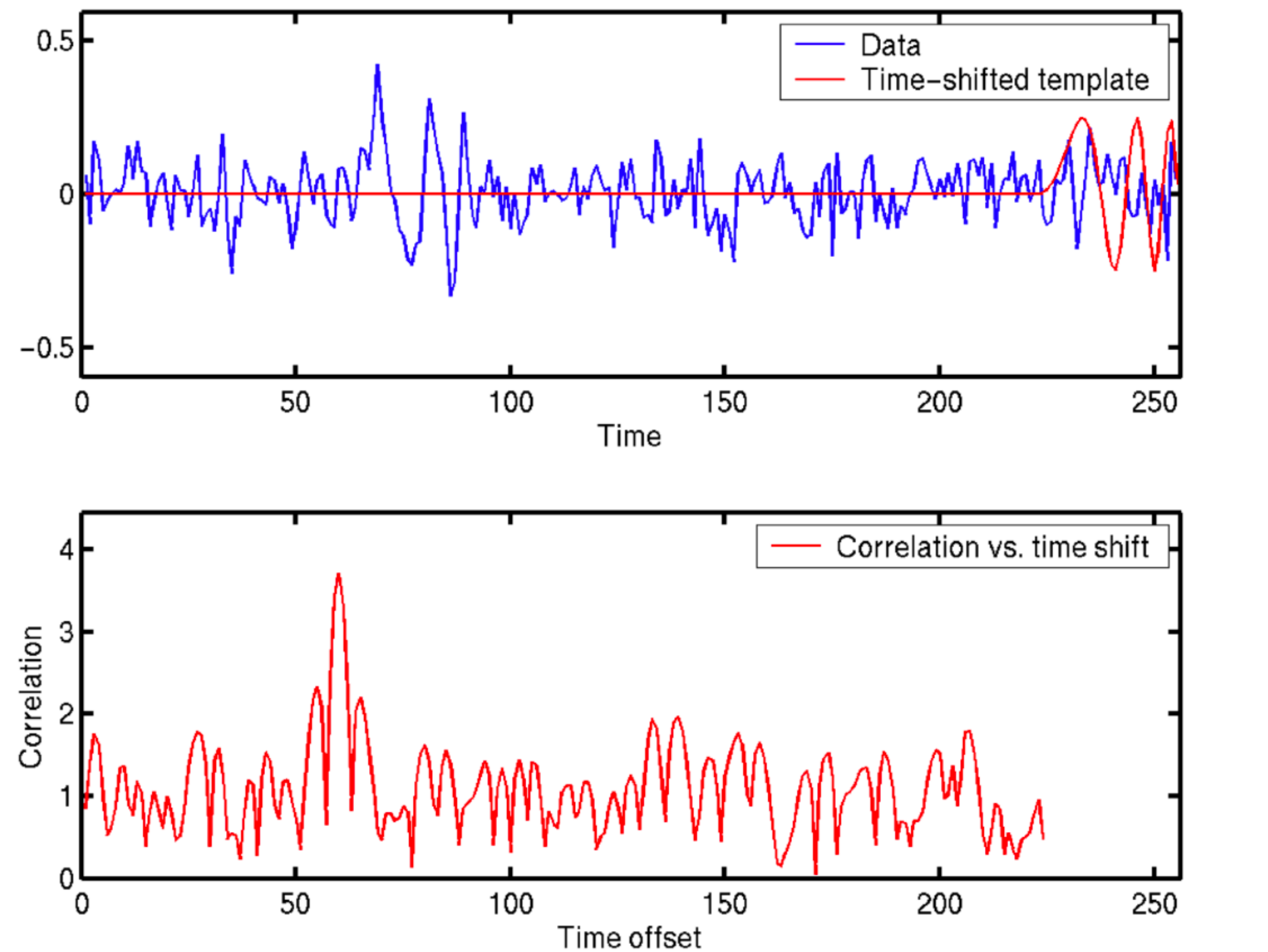
• Correlate data with expected signal (Here, plotting absolute value) G1400343

$$C(t) = \int_{-\infty}^{\infty} dt' \; s(t') \; h(t' - t)$$

Time offset

Data

Template with time offset



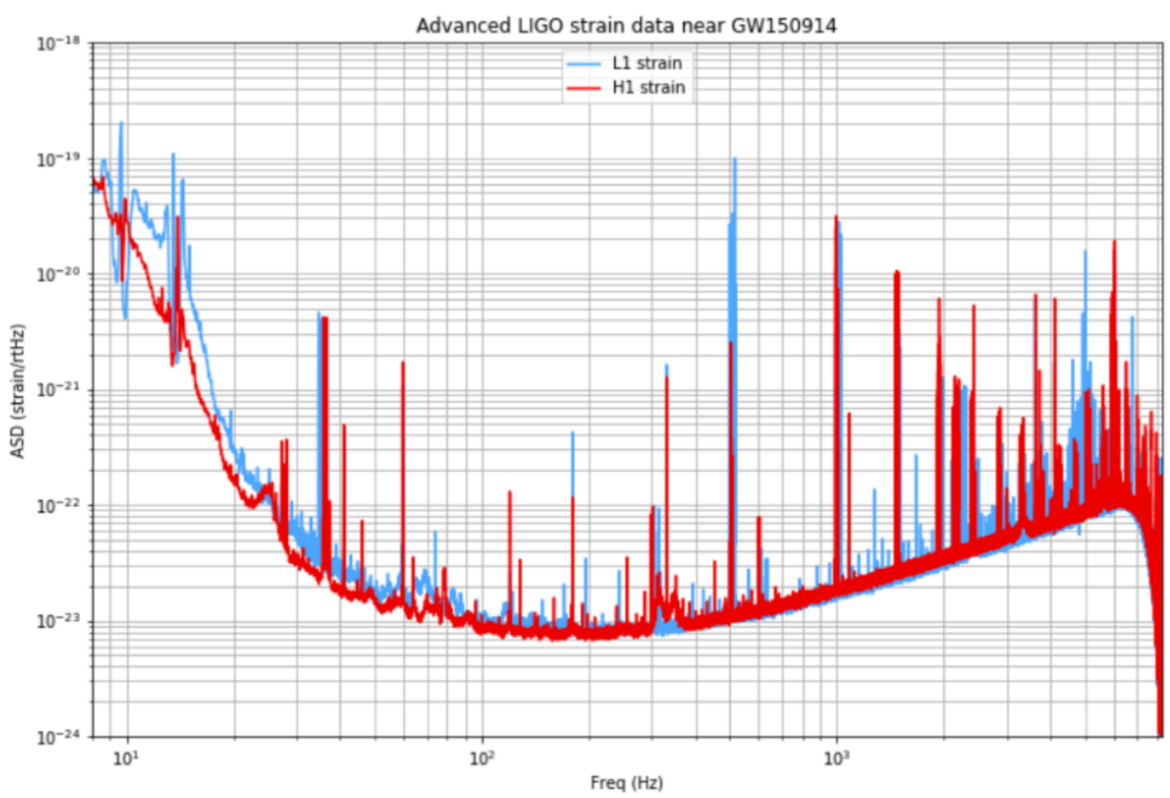

이형원교수님
지난 여름학교 강의중에서

# Spectral Lines - official info.

## O1 Instrumental Lines

The plot below shows the amplitude spectral density (ASD) of the strain noise in the H1 and L1 Advanced LIGO detectors, during a "typical" time in the O1 run. The plot shows frequency [Hz] on the X-axis, and the ASD value [ 1 / sqrt(Hz) ] on the y-axis. The first thing to note is that the data are not calibrated or valid below 10 Hz or above 5 kHz (and the data sampled at 4096 Hz are not valid above 2000 Hz).

The spectra reveal a large number of "lines" due to instrumental artifacts:




O1 : https://www.gw-openscience.org/o1speclines/
O2 : https://www.gw-openscience.org/o2speclines/
O3a : https://www.gw-openscience.org/O3/o3aspeclines/

# Noise backgrounds

## Complicated noise curves

Many lines in the data, not such an issue for transient searches, but can be an issue for continuous wave searches

To an okay approximation, the detector data is colored Gaussian noise - standard Gaussian noise just with certain frequencies louder than others

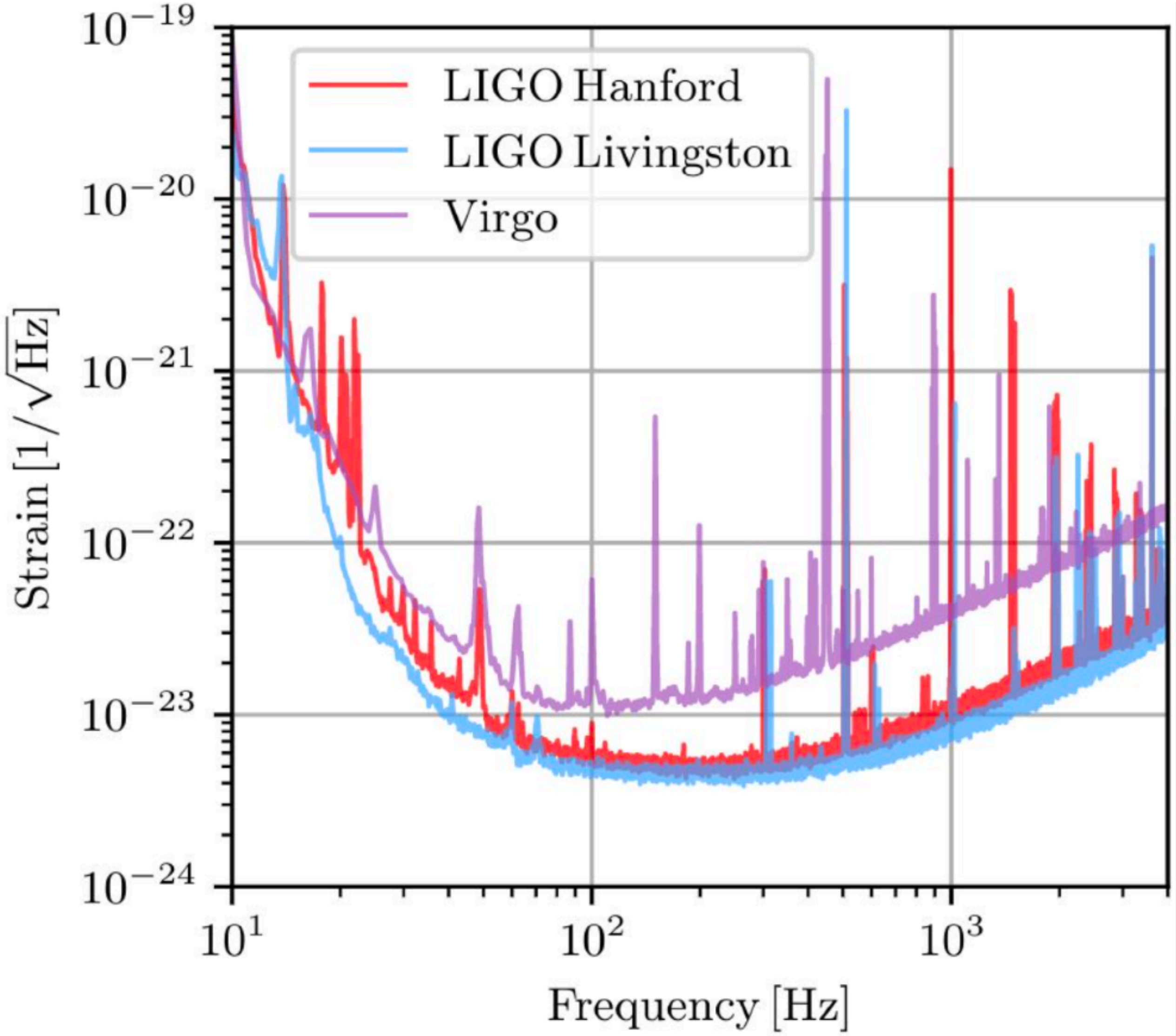
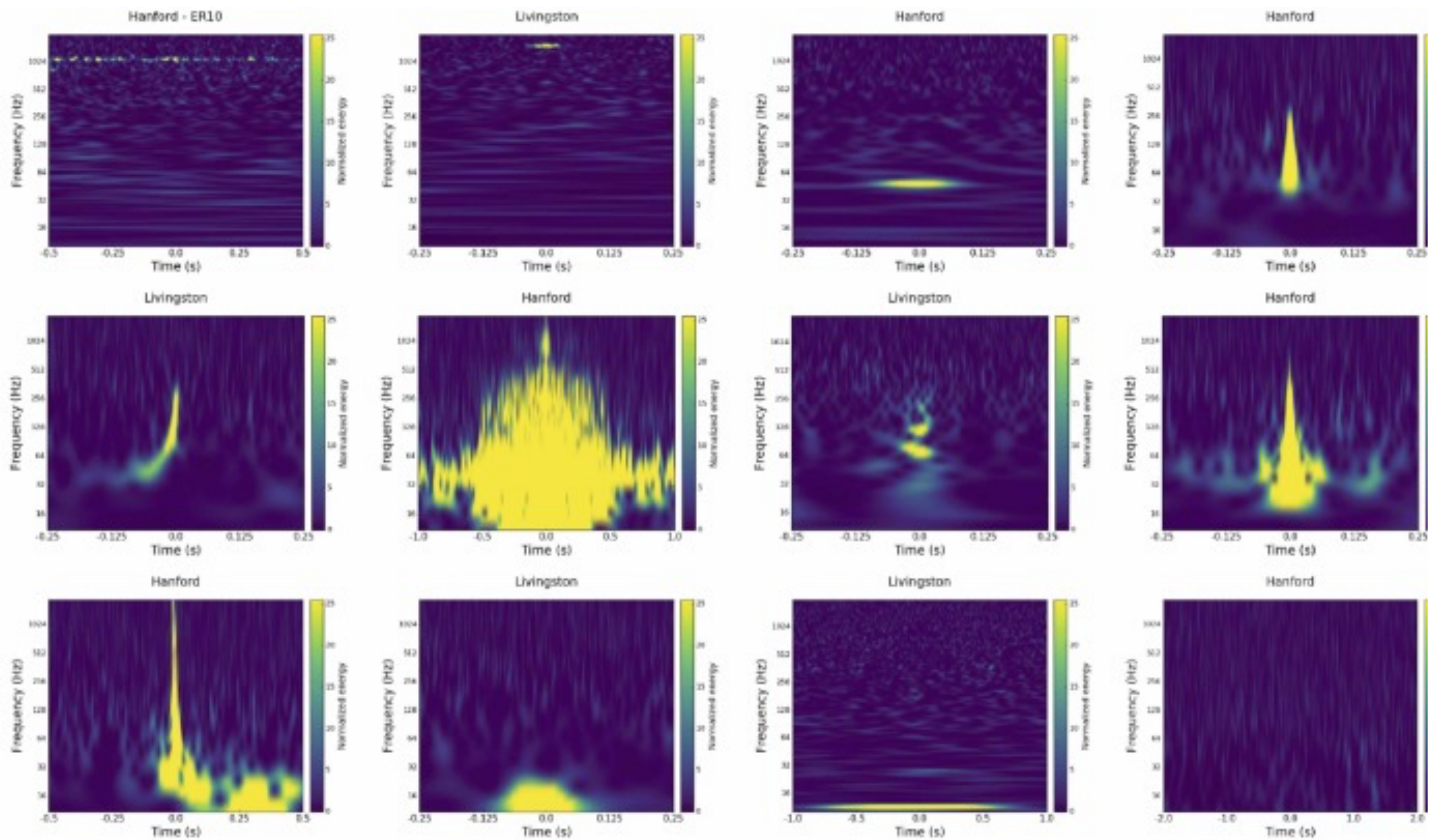



Image from Abbott et al (2020) GWTC-2 2010.14527

# Non-Gaussian Transient Noises




Bahaadini et al. (2018)

# Noise backgrounds

## Non-stationarity

The detector sensitivity is not constant, this can happen rapidly or slowly

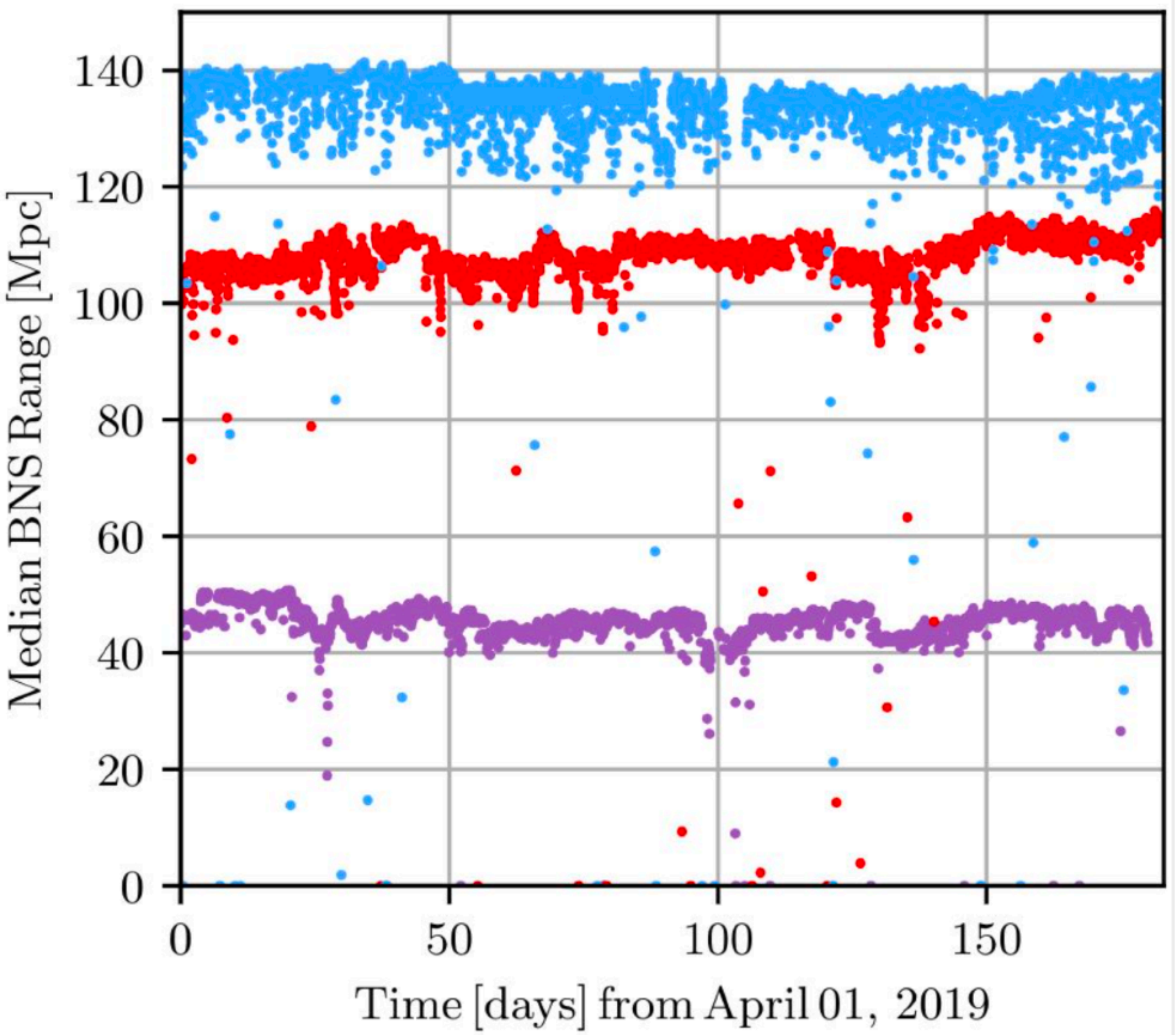
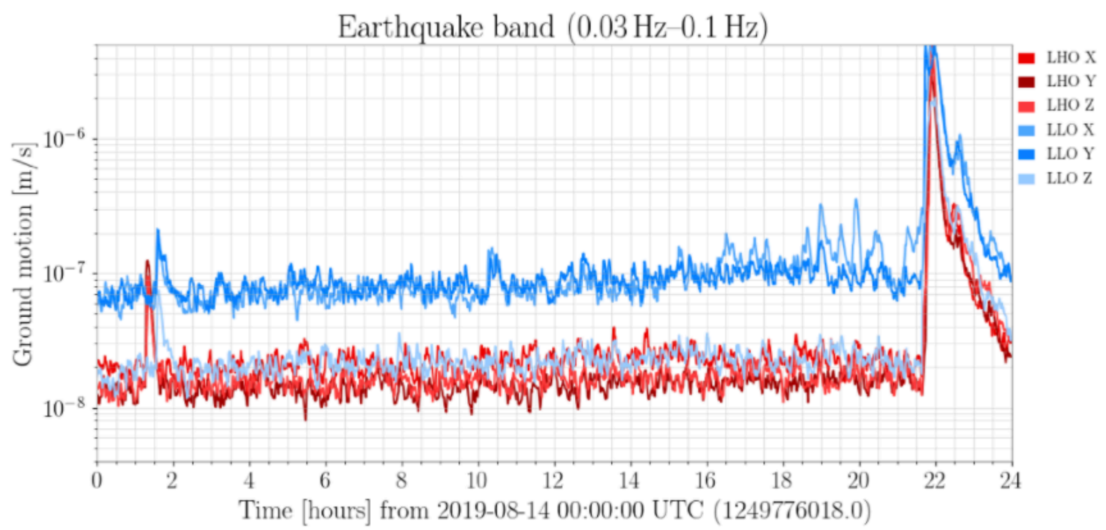
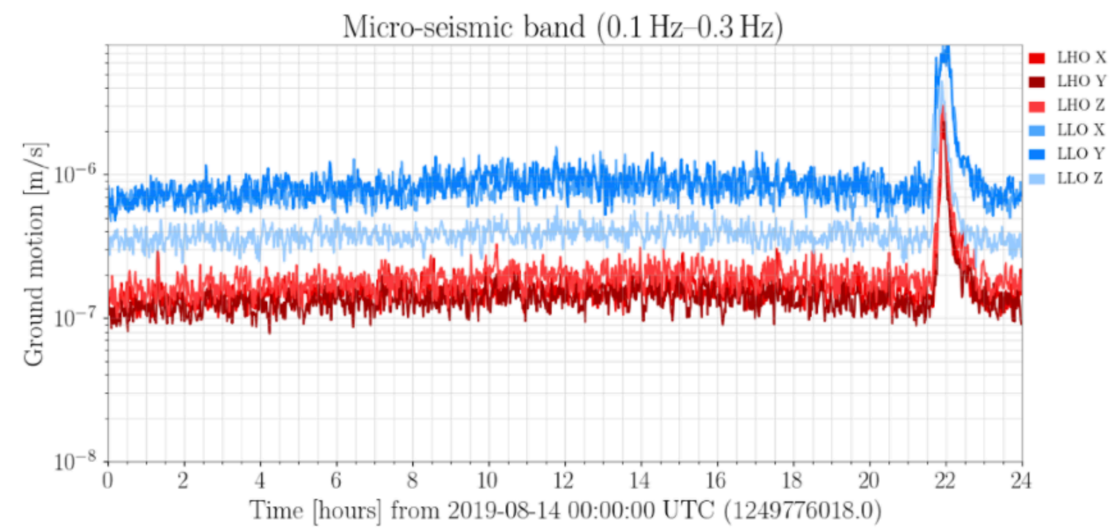


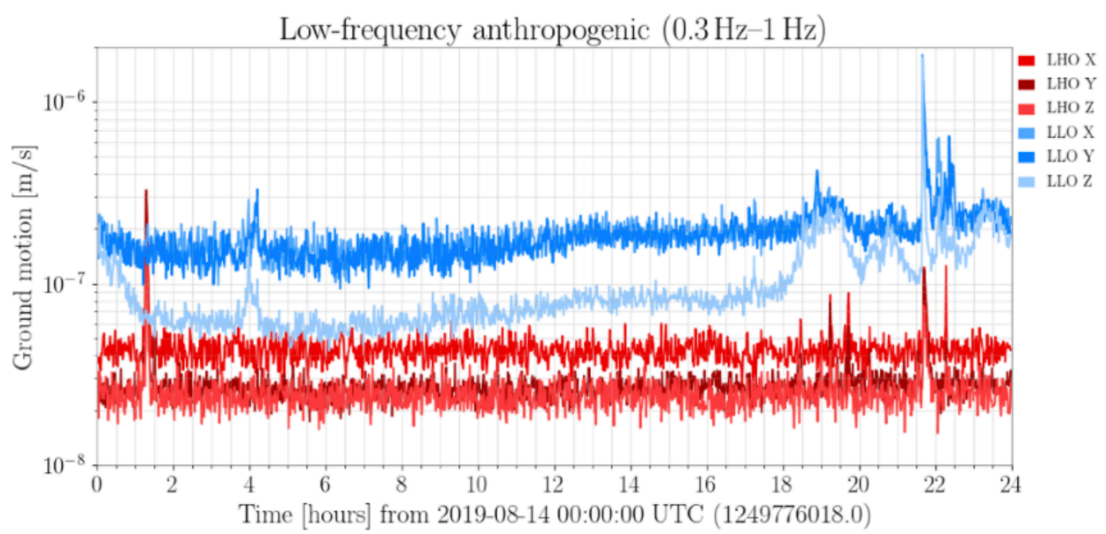

Image from Abbott et al (2020) GWTC-2 2010.14527

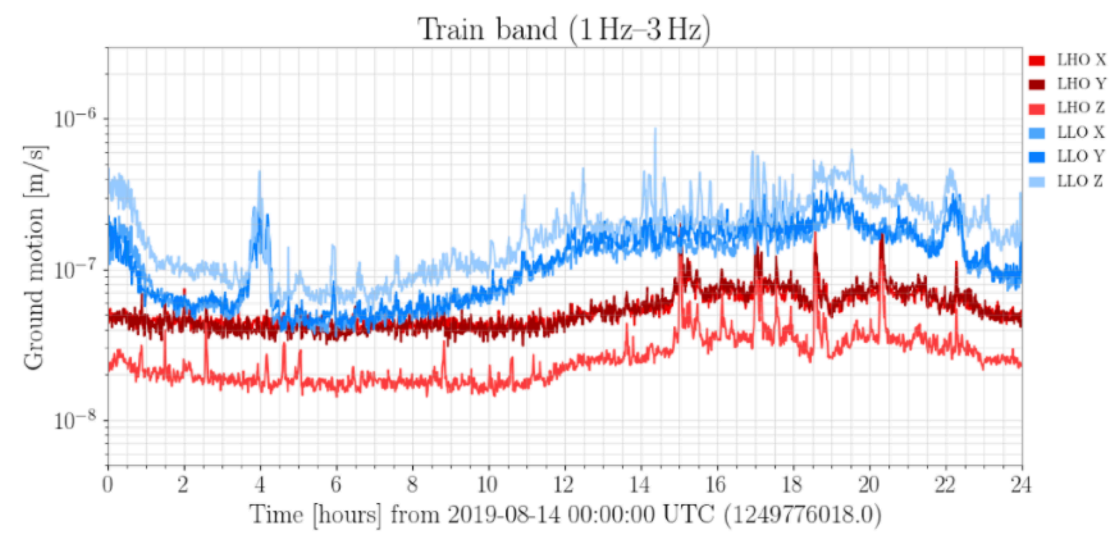
# Environment : Ground motion

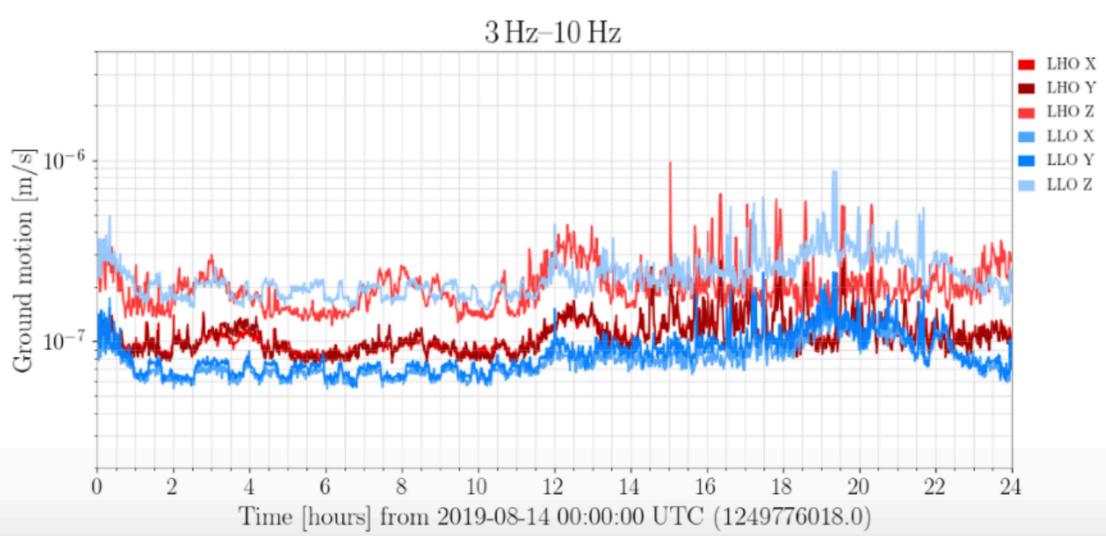
These plots display the ground motion at the LIGO Livingston and LIGO Hanford Observatories as measured by Streckeisen STS-2 seismometers at the corner station (where the X- and Y-arms meet). Each plot shows the root-mean-square ground motion in a different frequency band, which capture independent ground motion behavior.

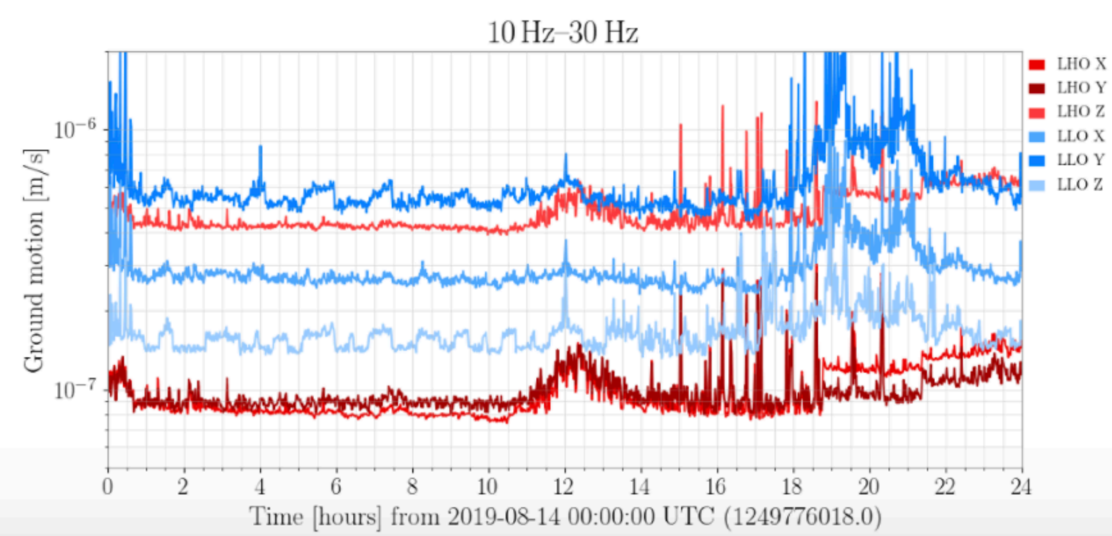


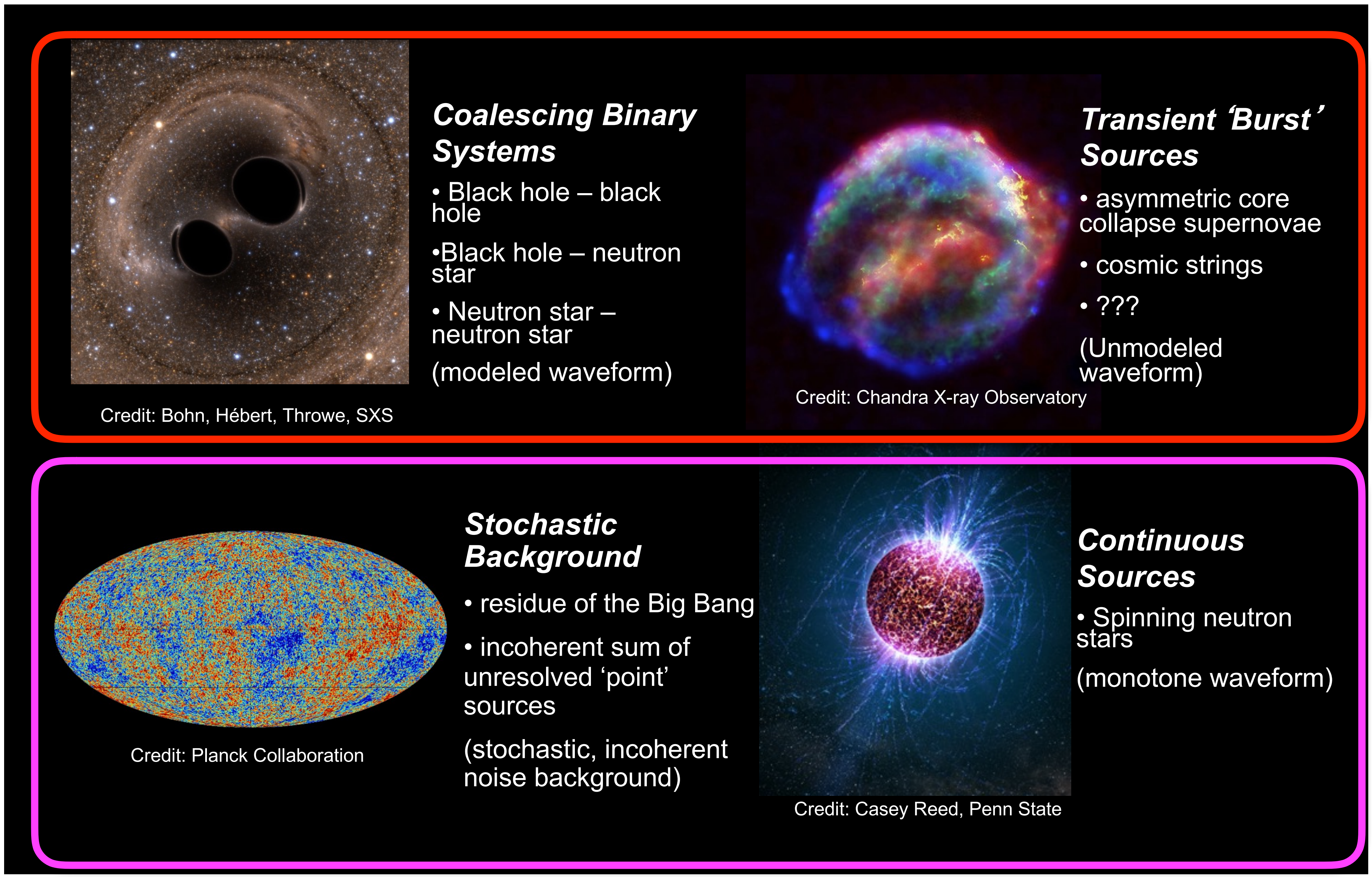



# Detection limitation by Noises

Non-Gaussian Transient Nosies : Glitches

Spectral lines : electrical or mechanical resonances




In D. Reitze's presentation in LIGO ODW #1, 2018

# Hypothesis Test

$H_0$ : null hypotheiss

$H_A$: Alternative hypothesis

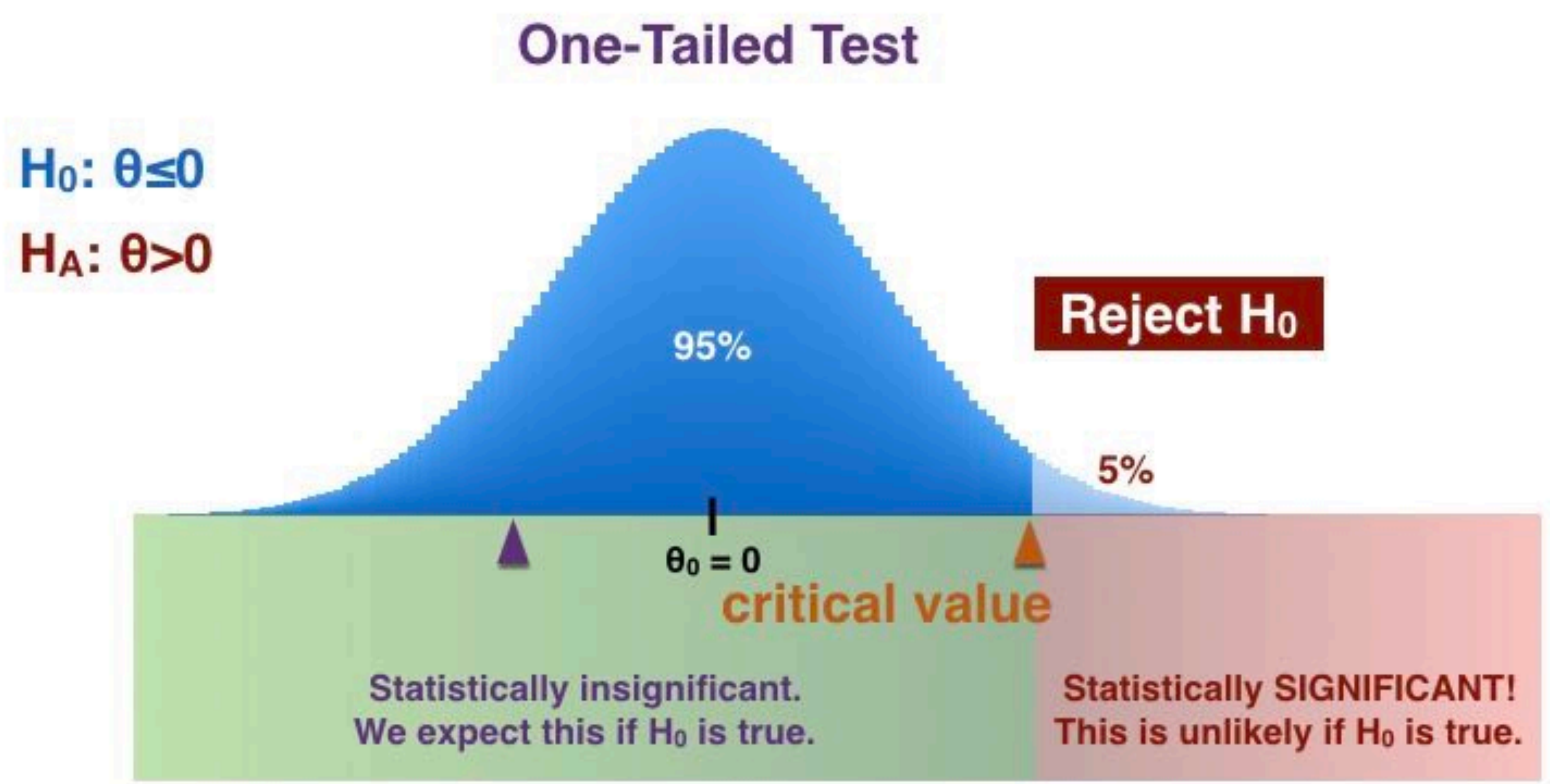



Image courtesy: https://prepnuggets.com/glossary/one-tailed-hypothesis-test/

# Hypothesis Test

$H_0$ : null hypotheiss

$H_1$: Alternative hypothesis

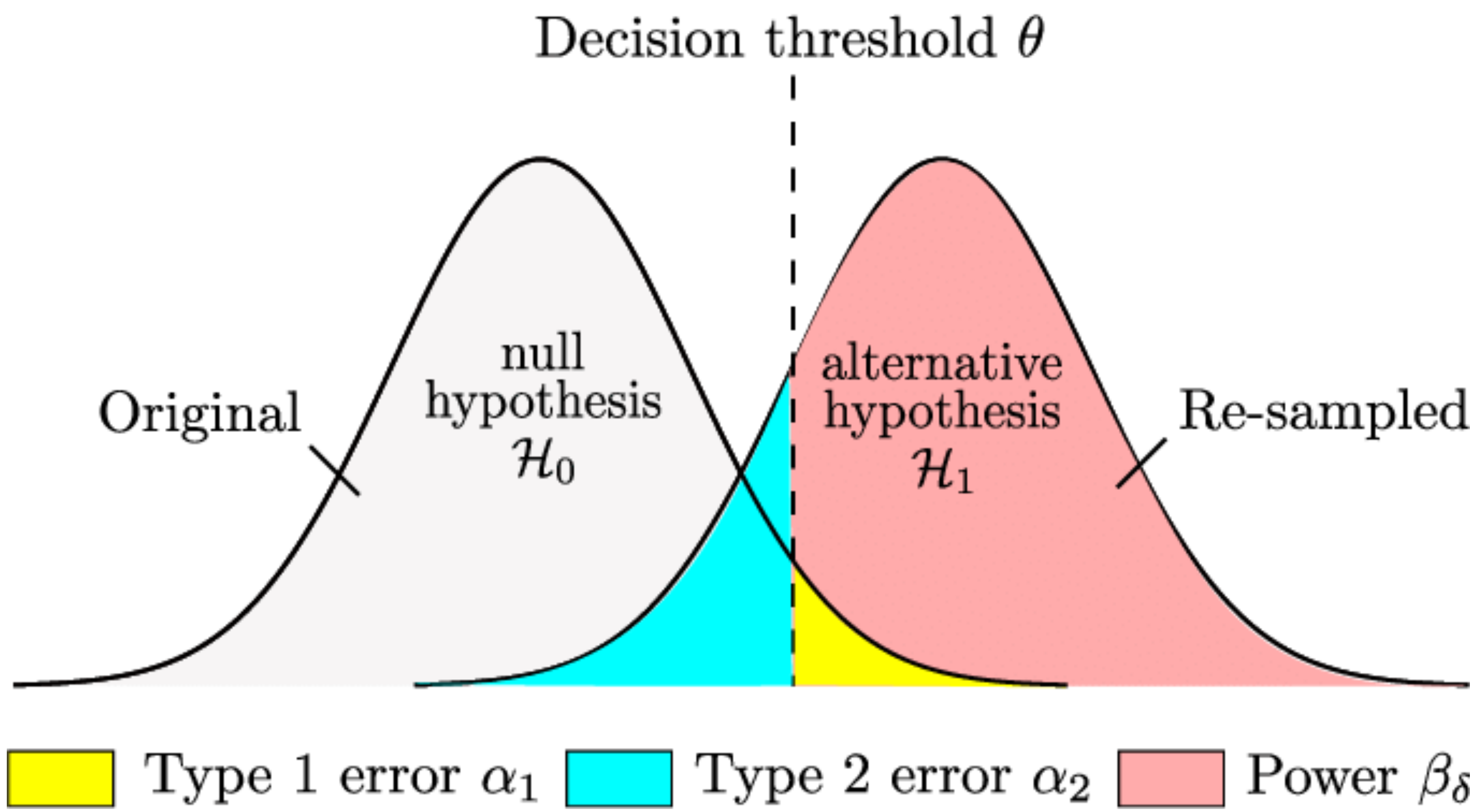
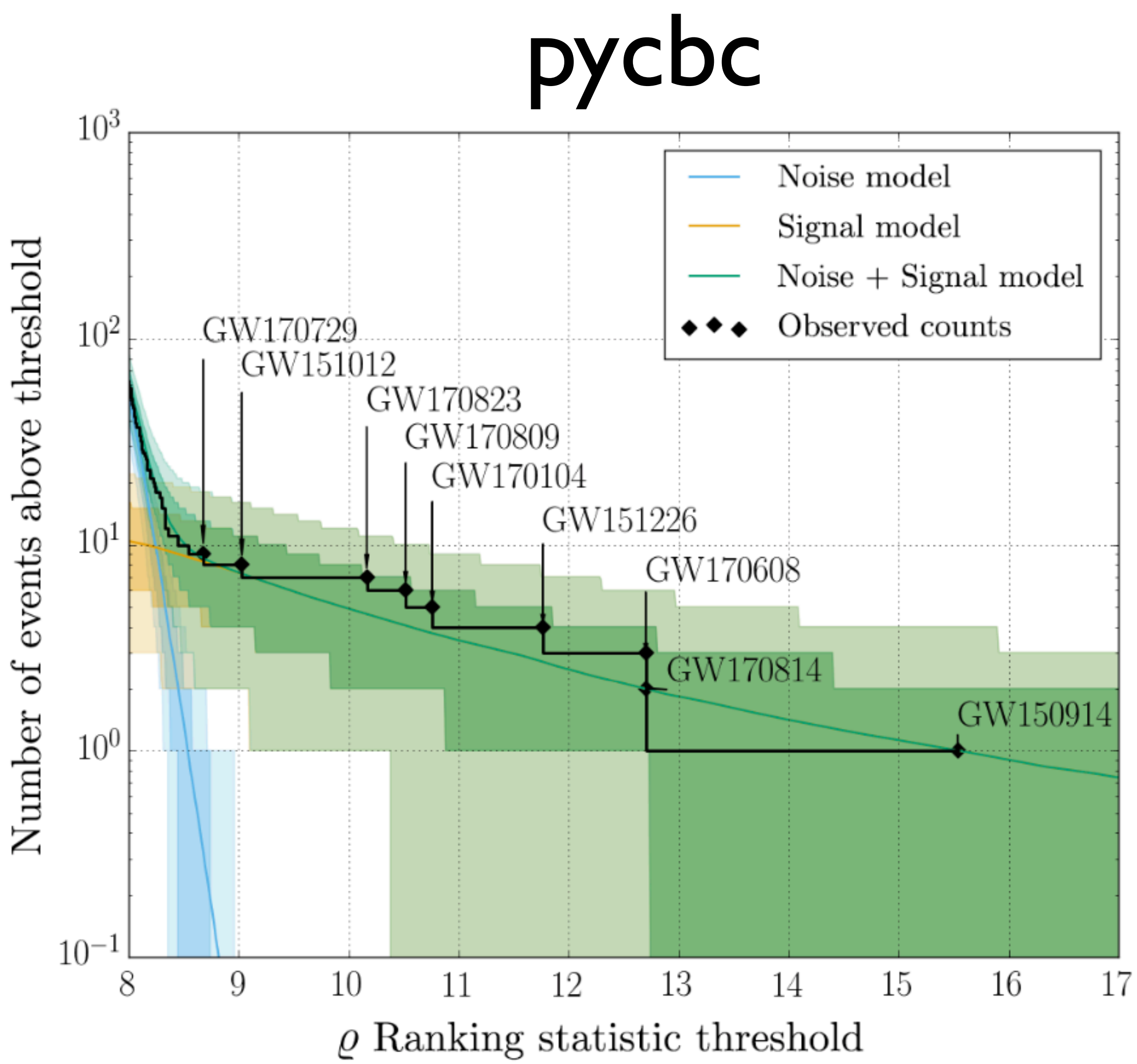
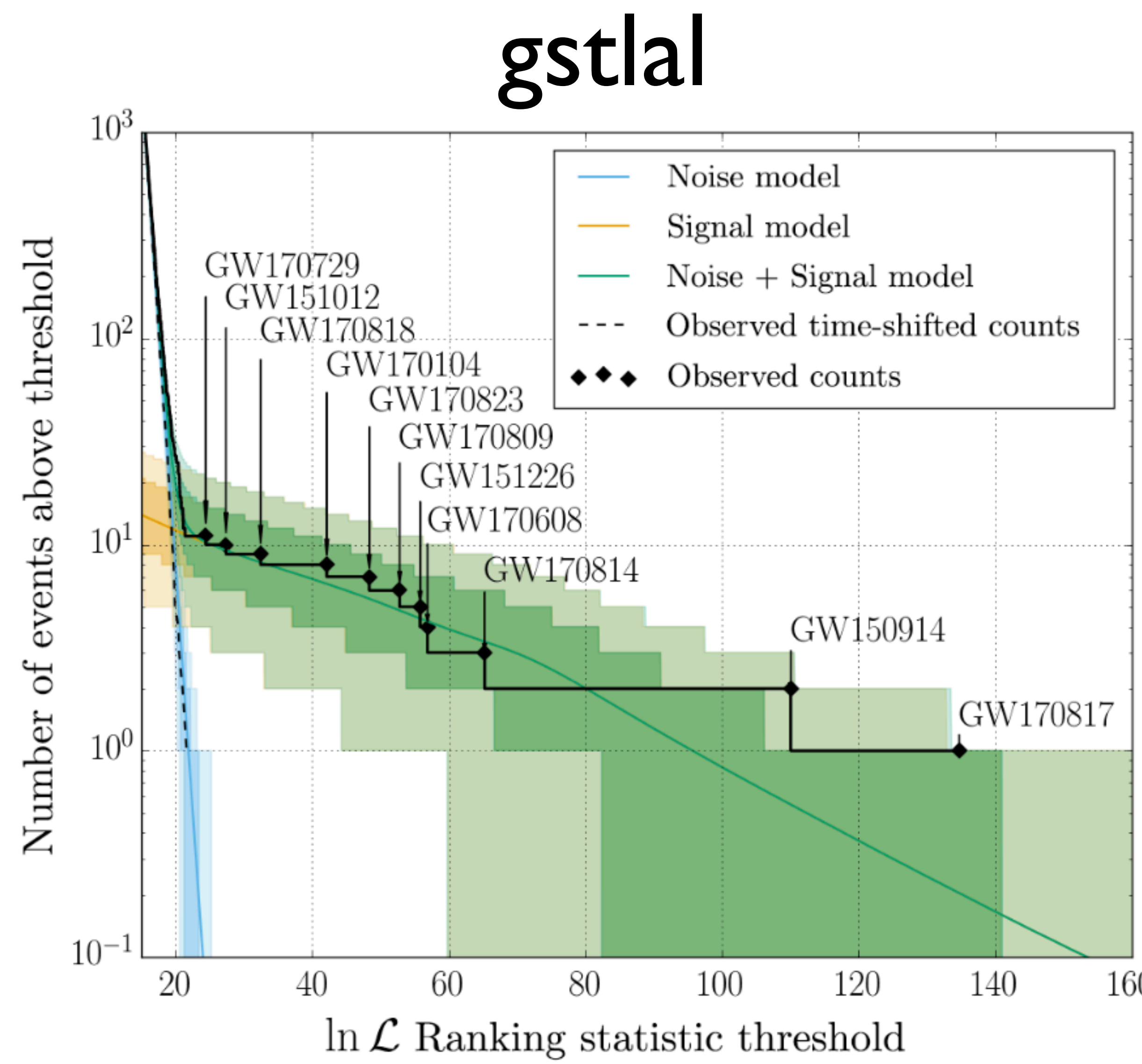



Image courtesy: Le Nhan

# Signal Significance

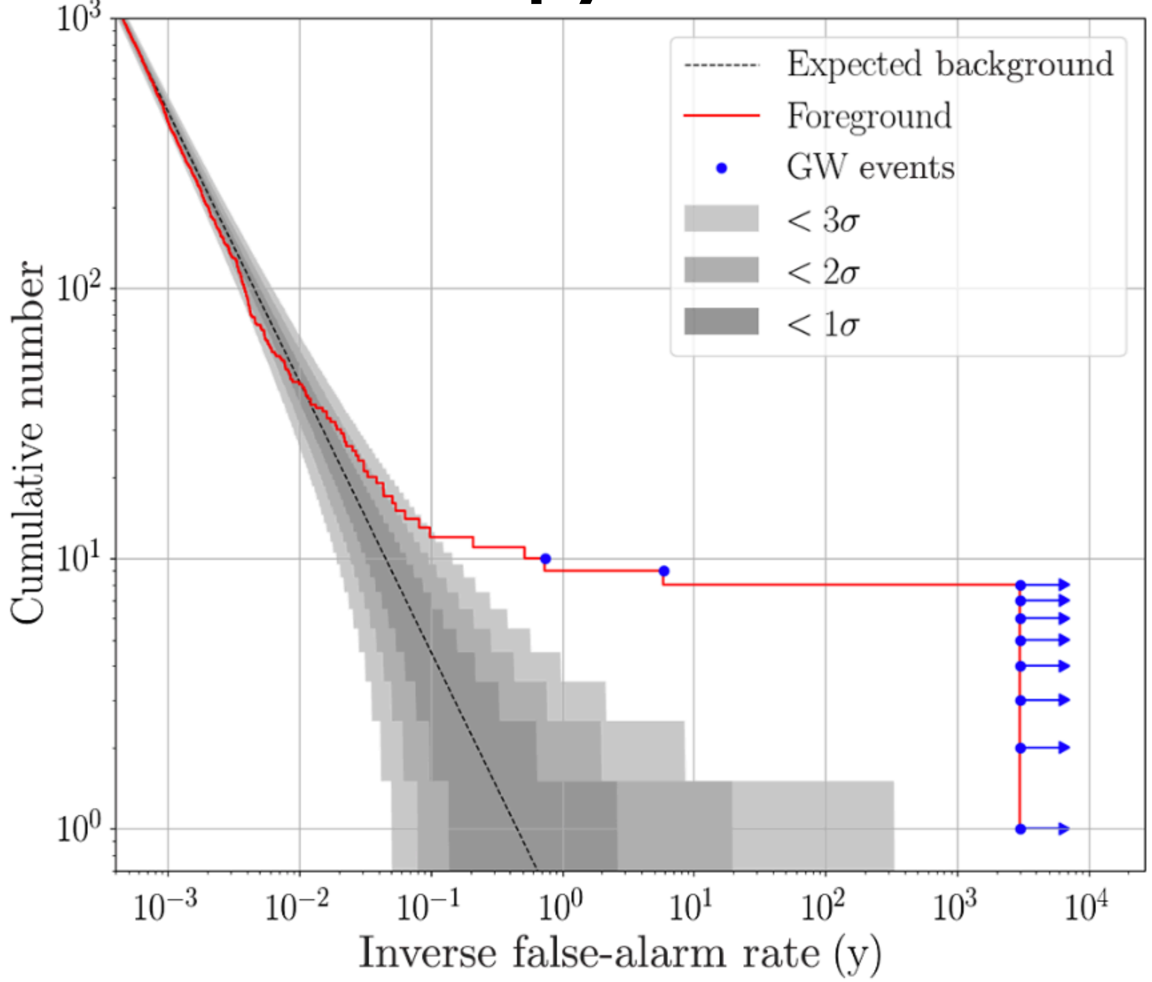


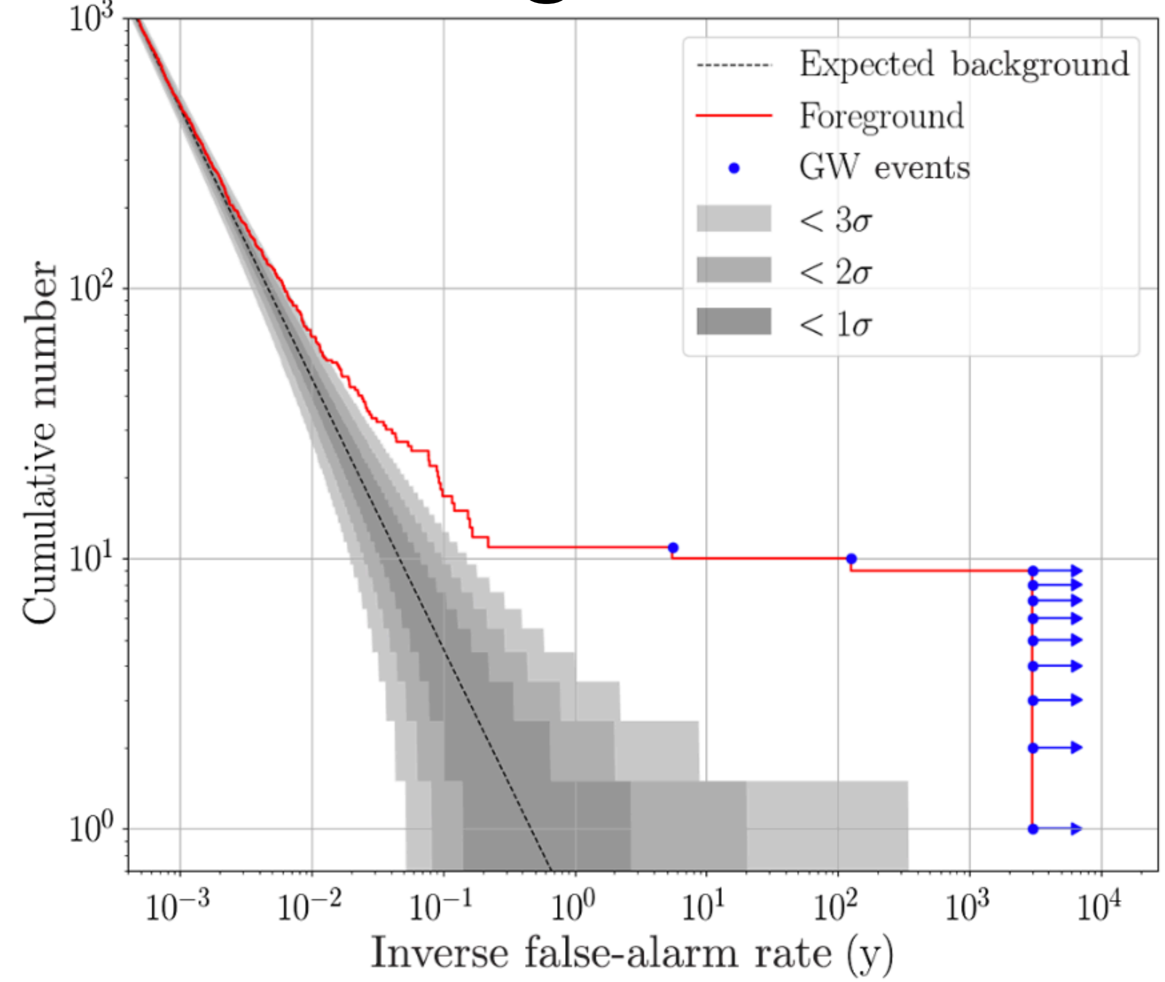

GWTC-1, PHYS. REV. X 9, 031040 (2019)

# Signal Significance

pycbc




gstlal



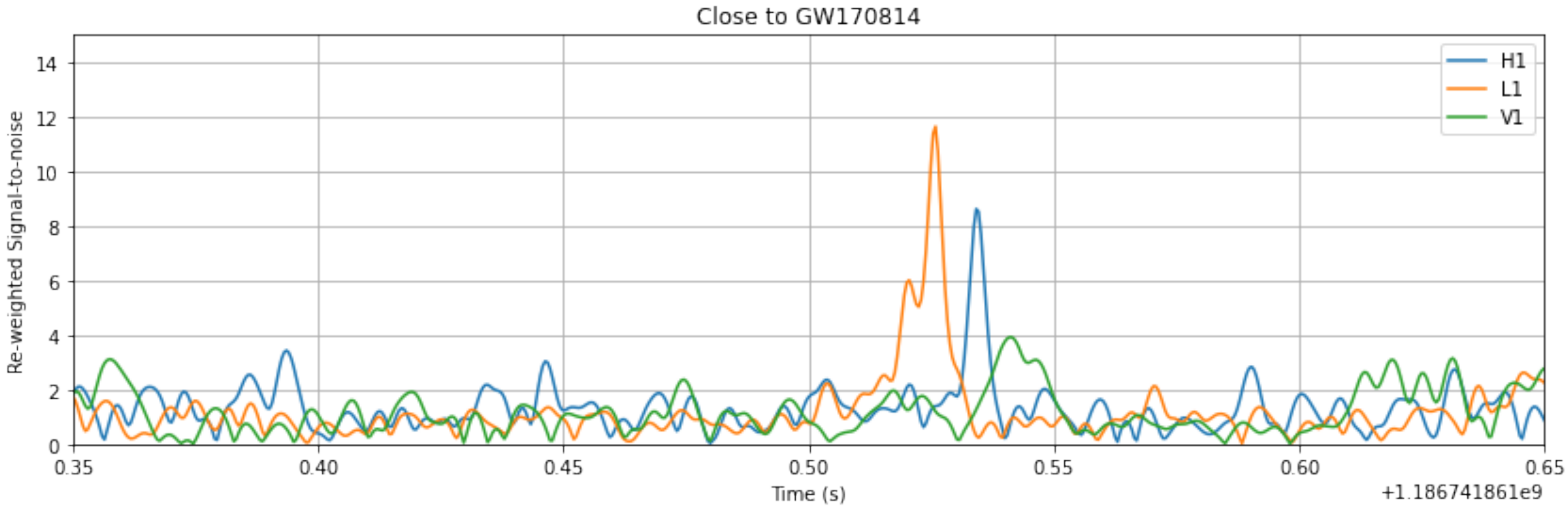

$$p = 1 - e^{-T/\mathrm{IFAR}}$$

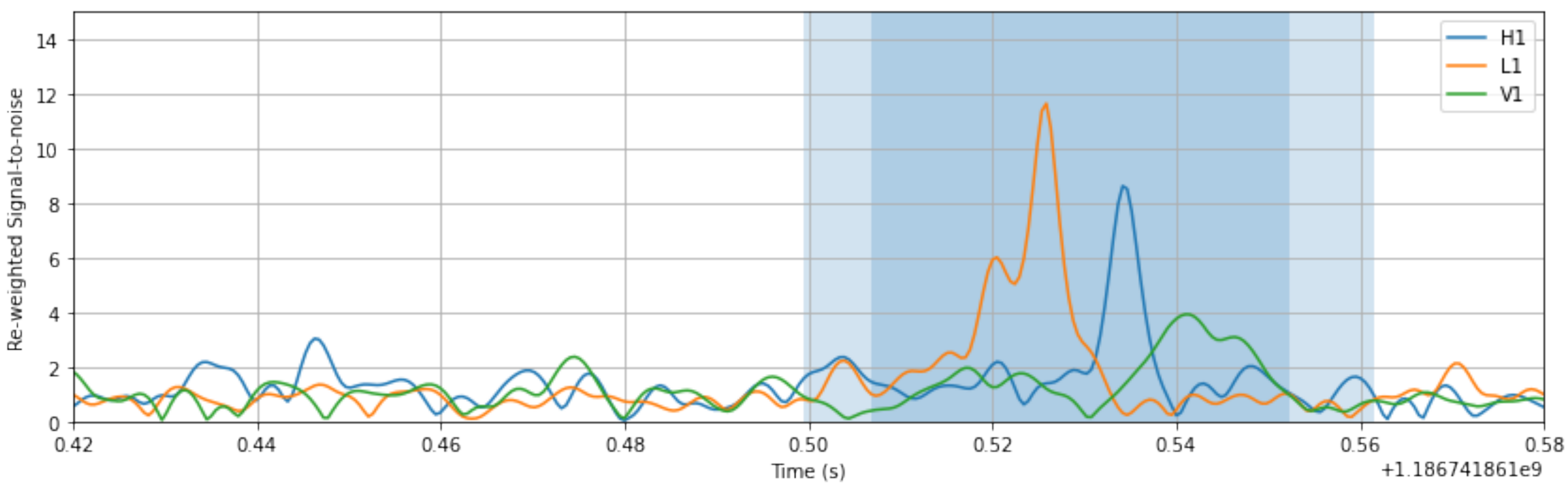
$$\mathrm{FAR} = \frac{NP(\log \mathcal{L}^* \geq \log \mathcal{L} | \mathrm{noise})}{T},$$

$$p = 1 - e^{-NP(\log \mathcal{L}^* \geq \log \mathcal{L} | \mathrm{noise})}.$$

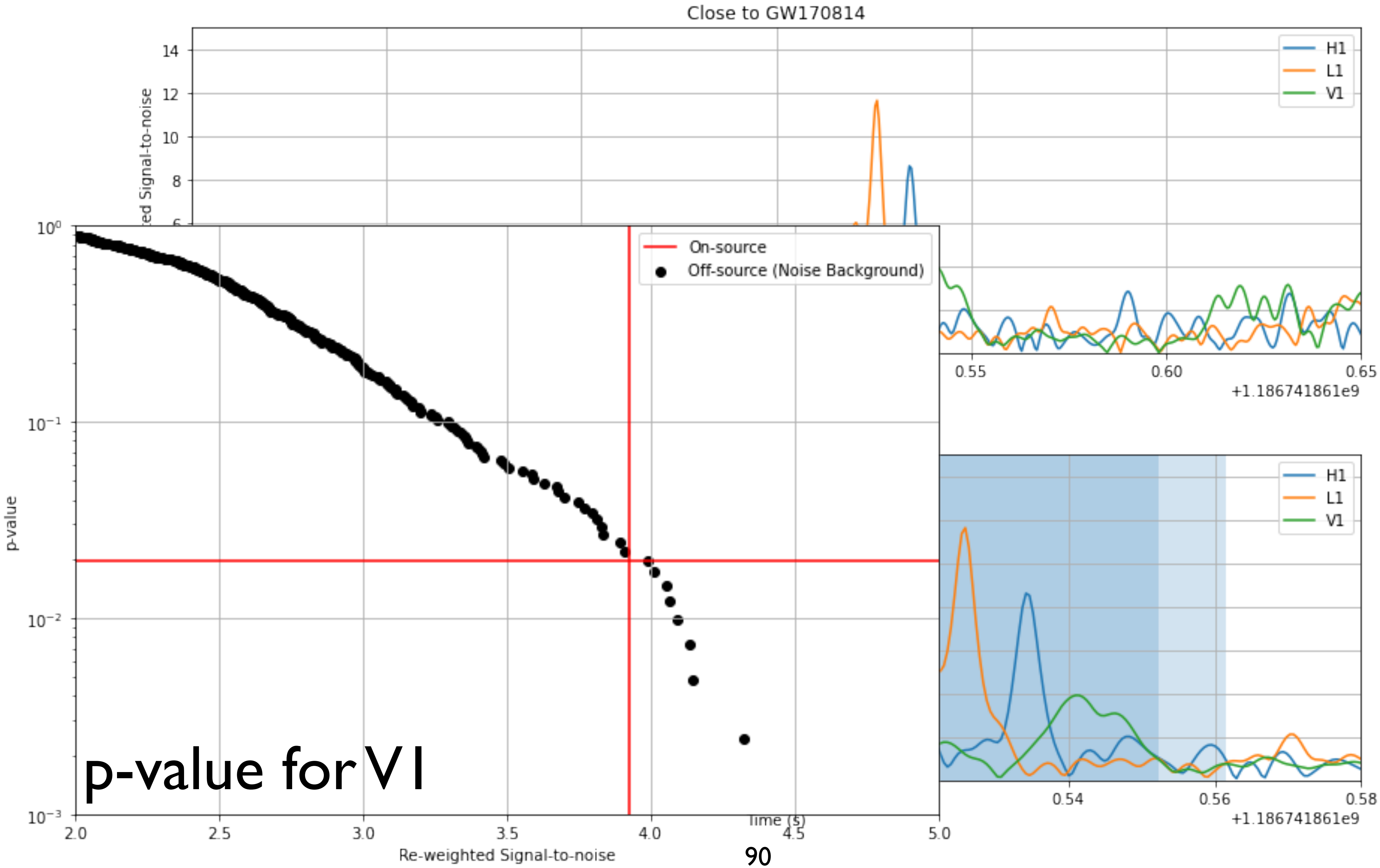
GWTC-1, PHYS. REV. X 9, 031040 (2019)

# Signal Consistency and Significance



# Signal Consistency and Significance



# GW Open Data Workshop

## GW Open Data Workshop

May 23 - 25, 2022

**This workshop is now available as an online course**

Start Course

### Overview

LIGO, Virgo, and KAGRA have now completed three observing runs (O1, O2, and O3), with all observational quality strain data available to the public. These observations include over 90 detections of compact object mergers. With more detector upgrades in progress and future observing runs planned, it is a very exciting time in the field!

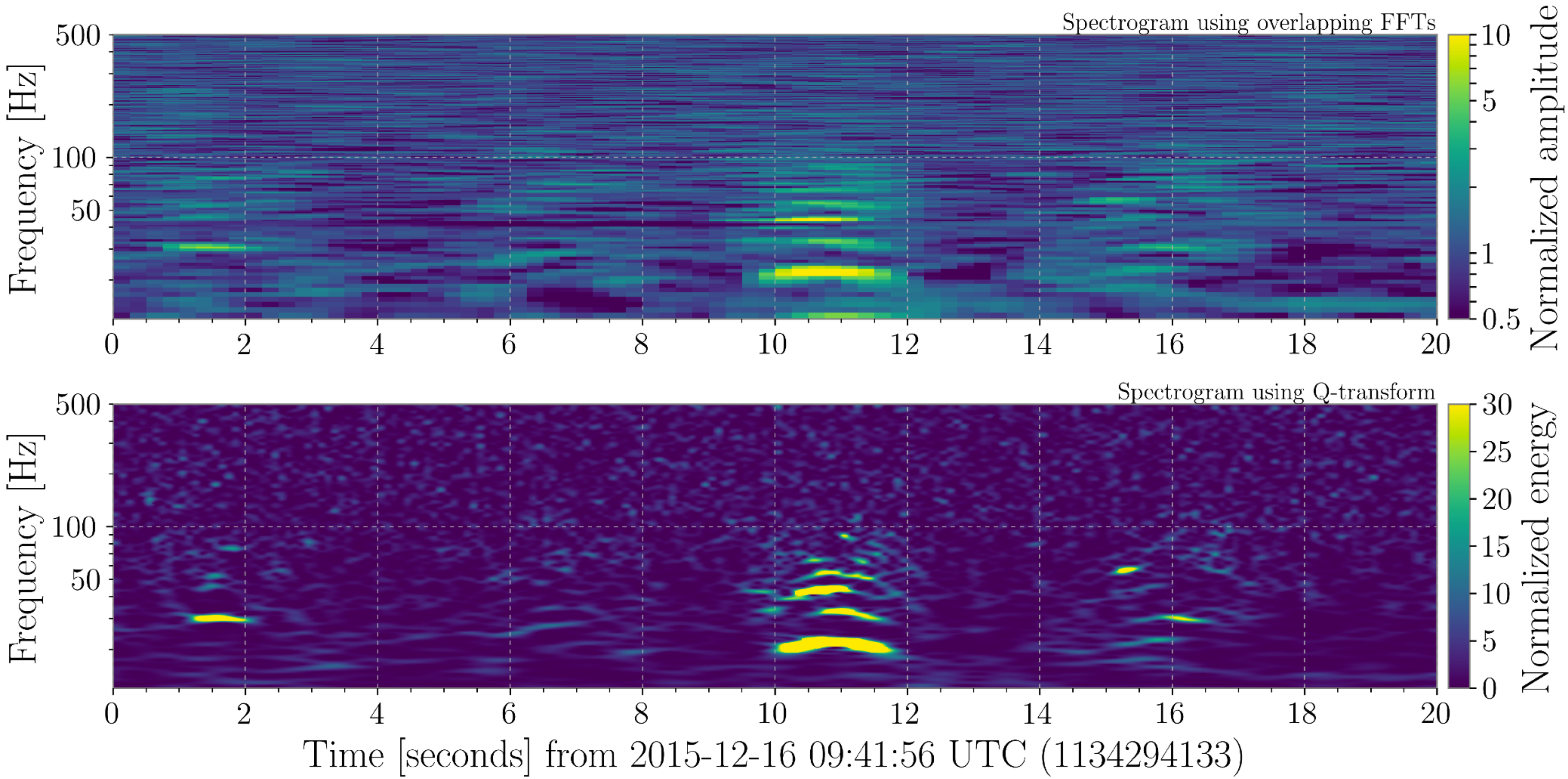
This Open Data Workshop is the 5th in a series of workshops that began in 2018. Participants will receive a crash-course in gravitational-wave data analysis. The workshop includes lectures by data analysis experts, hands on experience with software tutorials, and a data challenge designed to test your new skill in GW data analysis.

View Poster

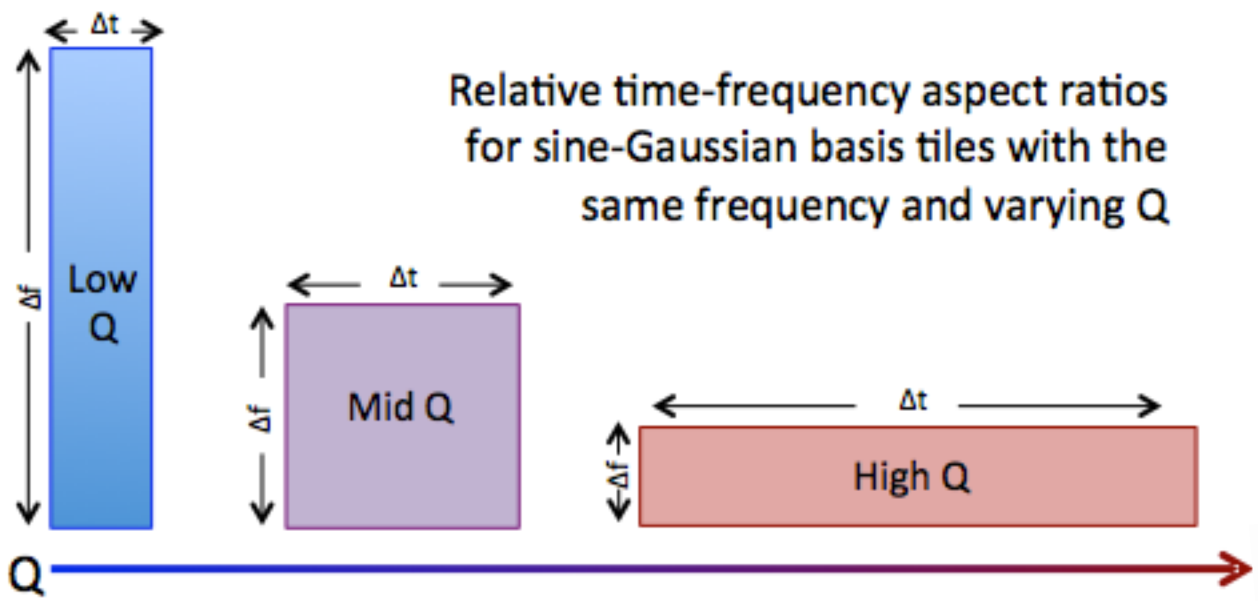
GW ODW #5,
https://www.gw-openscience.org/odw/odw2022/

Git Hub : https://github.com/gw-odw/odw-2022

# Features in GW data

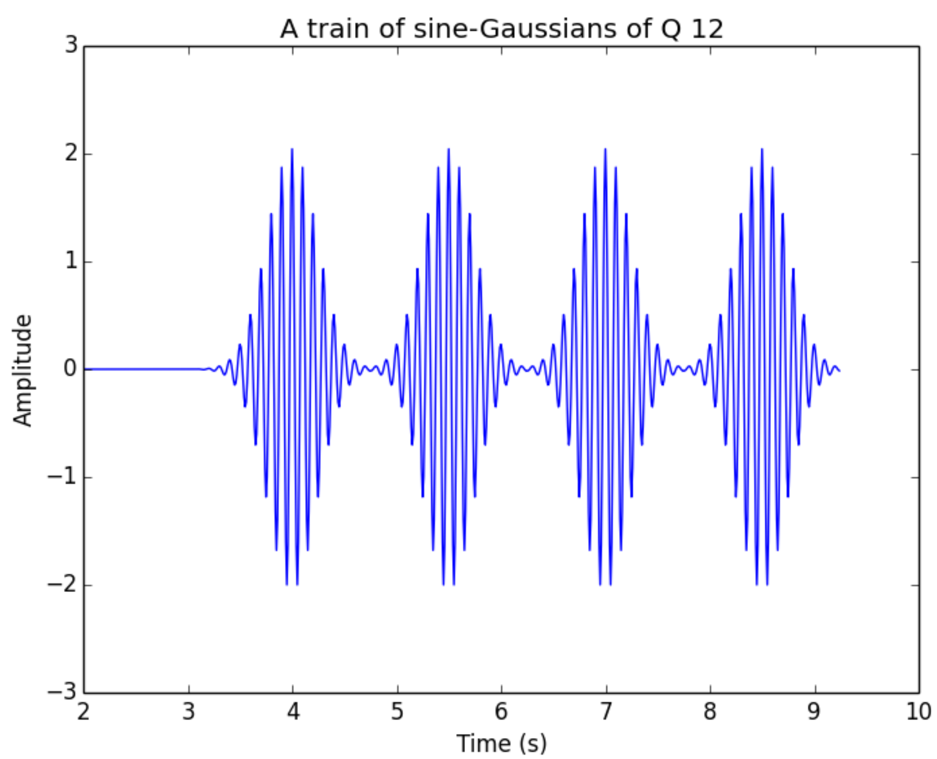


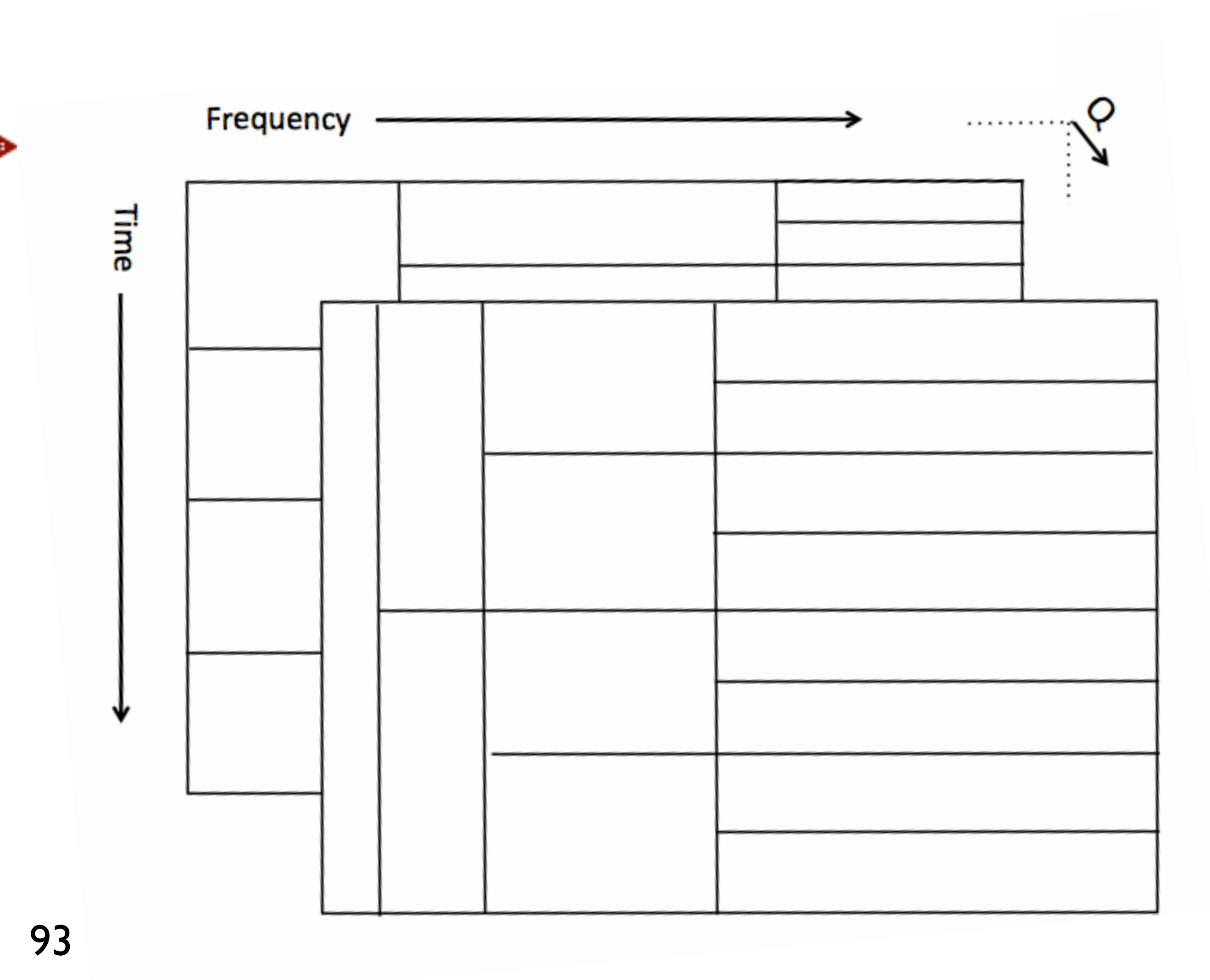
# Q-transform




$$\Delta t \Delta f = 1/4\pi$$

$$Q = f_0/\Delta f \ = \ 4\pi f_0 \Delta t.$$



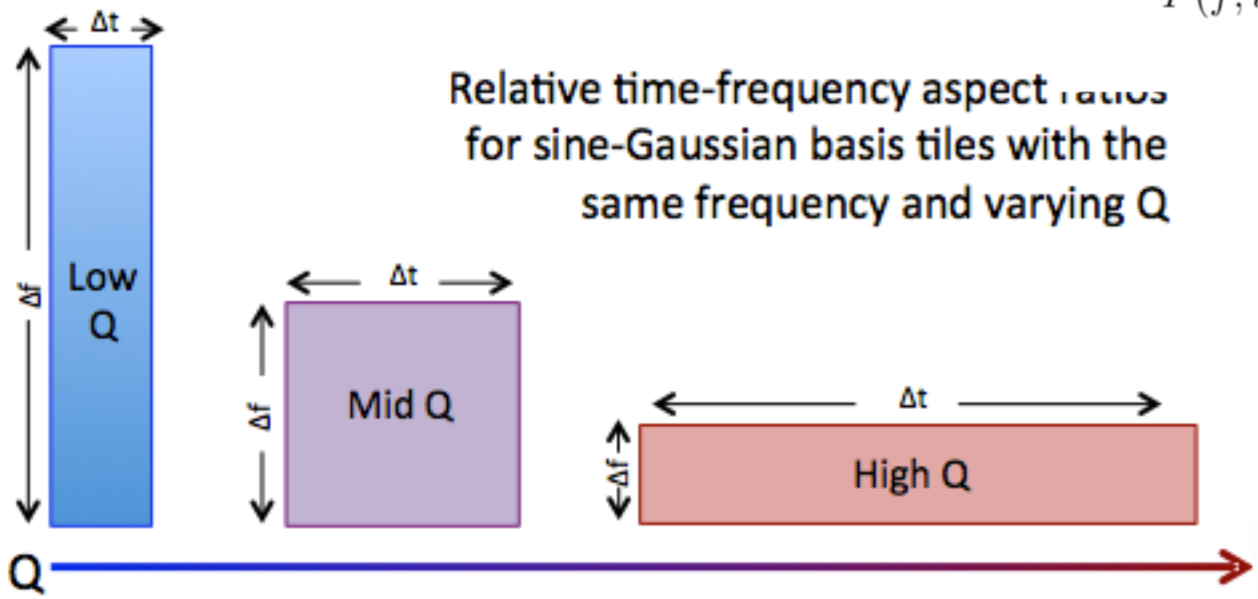



# Q-transform

$$Y(t, t_0, f_0, Q) = \left(\frac{8\pi f_0^2}{Q^2}\right)^{\frac{1}{4}} \exp\left[\frac{-4\pi^2 f_0^2}{Q^2}(t - t_0)^2\right] \exp[-i2\pi f_0(t - t_0)] \quad (3.11)$$

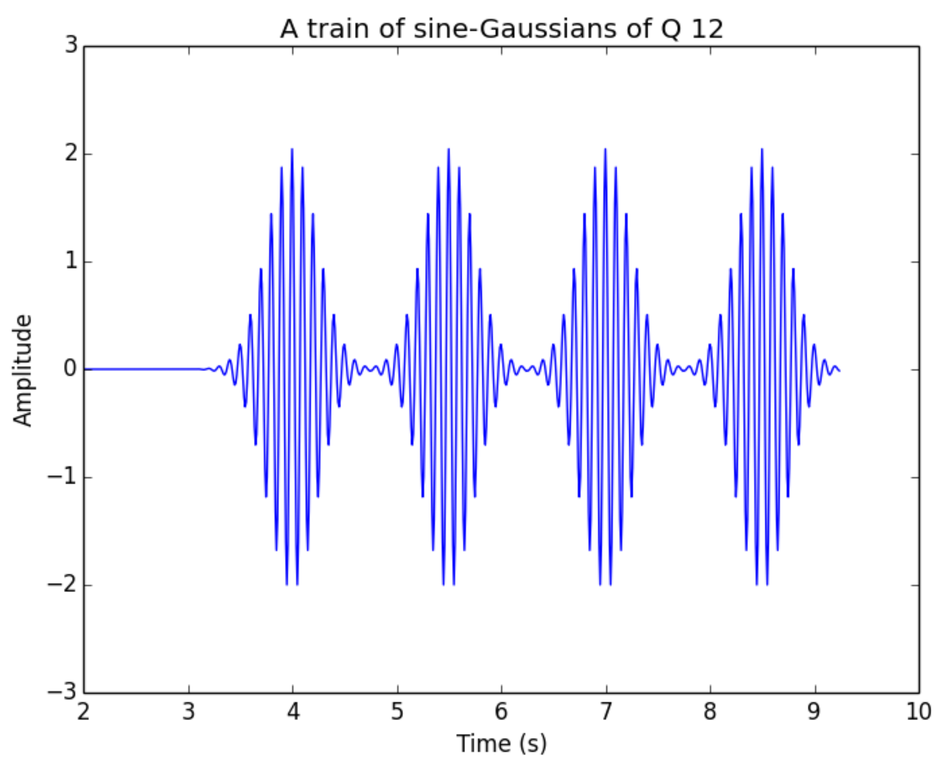
or equivalently in the frequency domain:

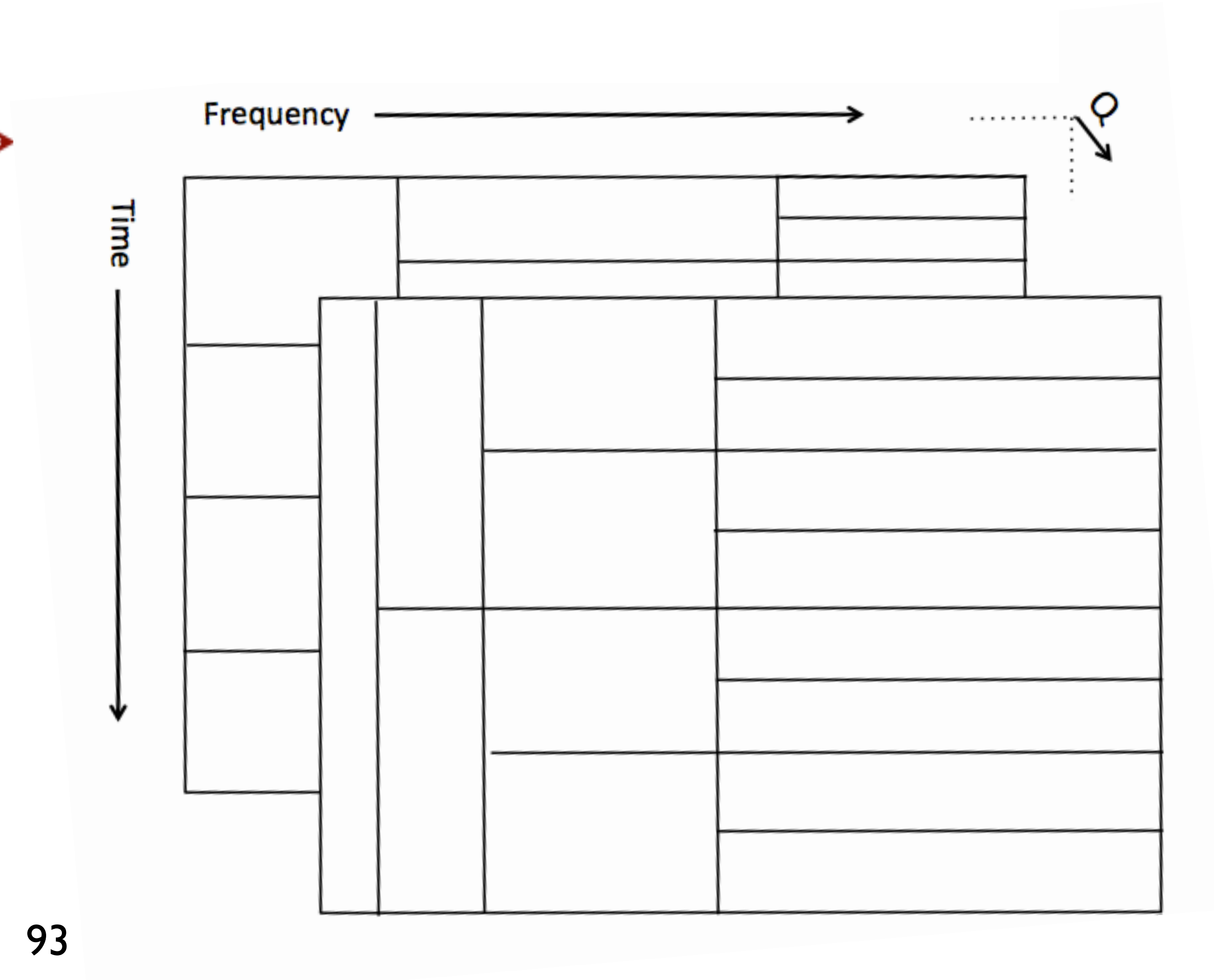
$$\widetilde{Y}(f, t_0, f_0, Q) = \left(\frac{Q^2}{2\pi f_0^2}\right)^{\frac{1}{4}} \exp\left[\frac{-Q^2}{4 f_0^2}(f - f_0)^2\right] \exp[-i2\pi t_0(f - f_0)]. \quad (3.12)$$




$$\Delta t \Delta f = 1/4\pi$$

$$Q = f_0/\Delta f = 4\pi f_0 \Delta t.$$

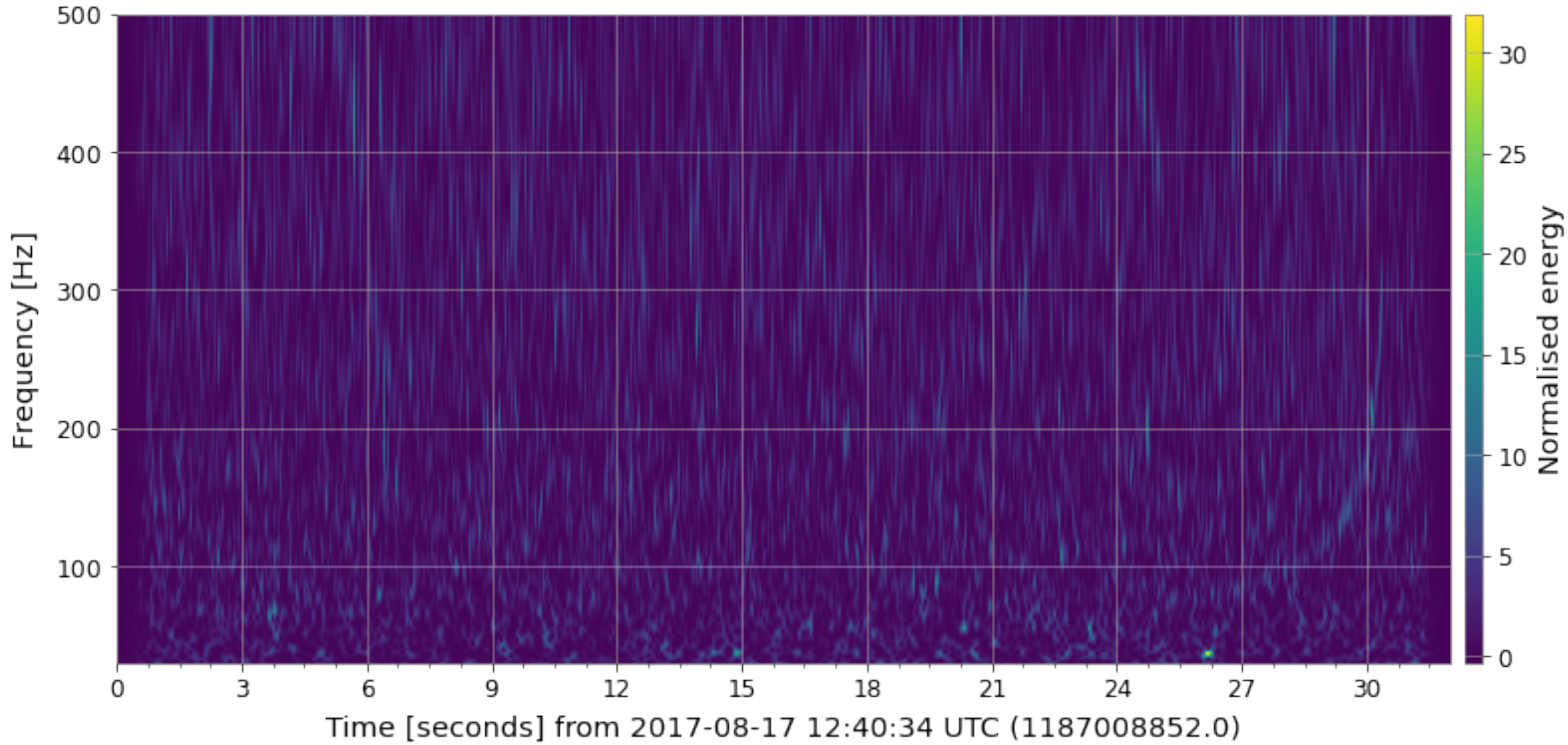


# GW170817: Q-transform in gwpy

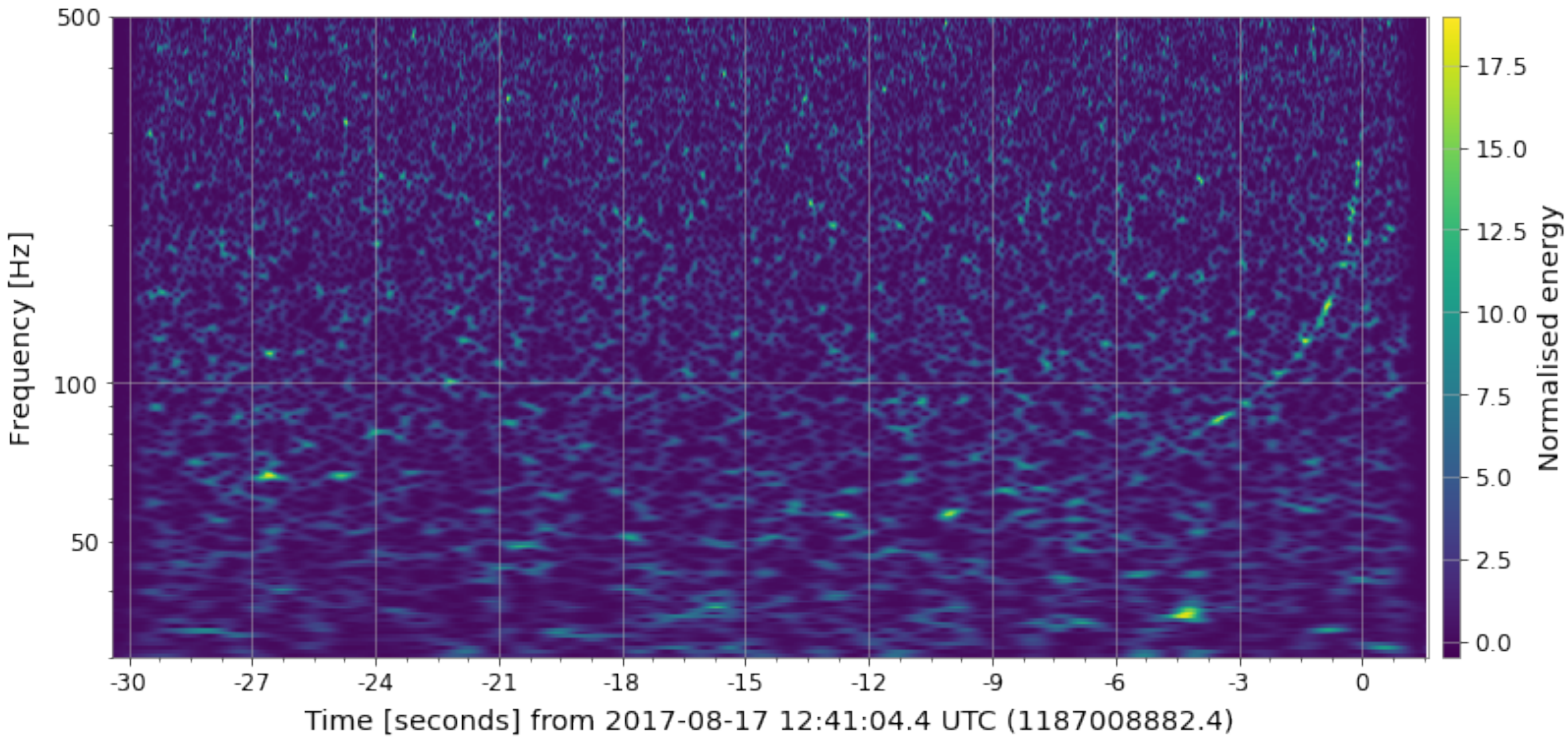
```
segment = (int(gps) - 30, int(gps) + 2)
hdata = TimeSeries.fetch_open_data('H1', *segment, verbose=True, cache=True)
```

```
hq = hdata.q_transform(frange=(30, 500))
plot = hq.plot()
plot.colorbar(label="Normalised energy")
```



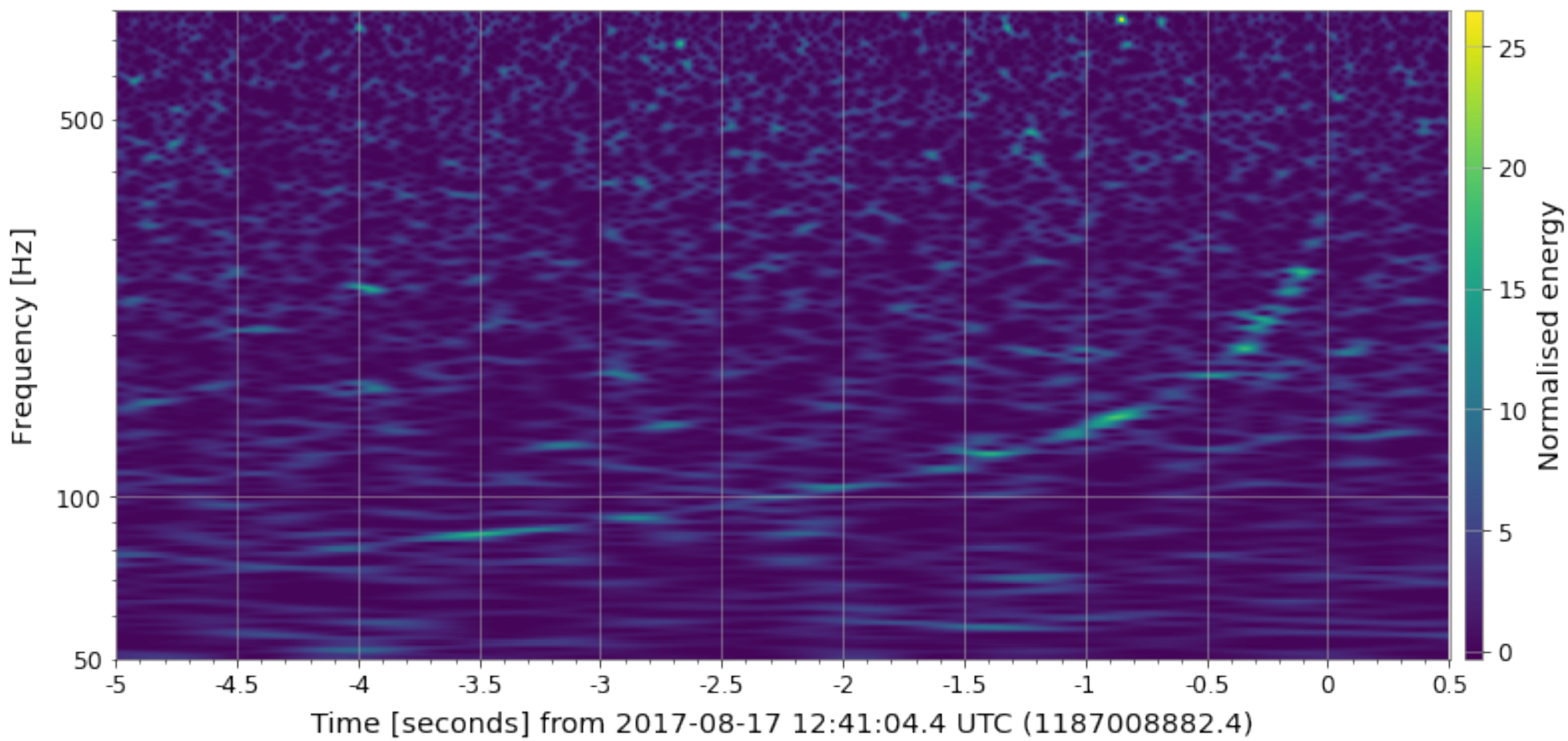
# GW170817: Q-transform in gwpy

```python
hq = hdata.q_transform(frange=(30, 500), qrange=(100, 110))
plot = hq.plot()
ax = plot.gca()
ax.set_epoch(gps)
ax.set_yscale('log')
ax.colorbar(label="Normalised energy")
```



# GW170817: Q-transform in gwpy

```python
#-- Use OUTSEG for small time range
hq2 = hdata.q_transform(frange=(50, 800), qrange=(90, 110), outseg=(gps-5,gps+0.5))
plot = hq2.plot()
ax = plot.gca()
ax.set_epoch(gps)
ax.set_yscale('log')
ax.colorbar(label="Normalised energy")
```



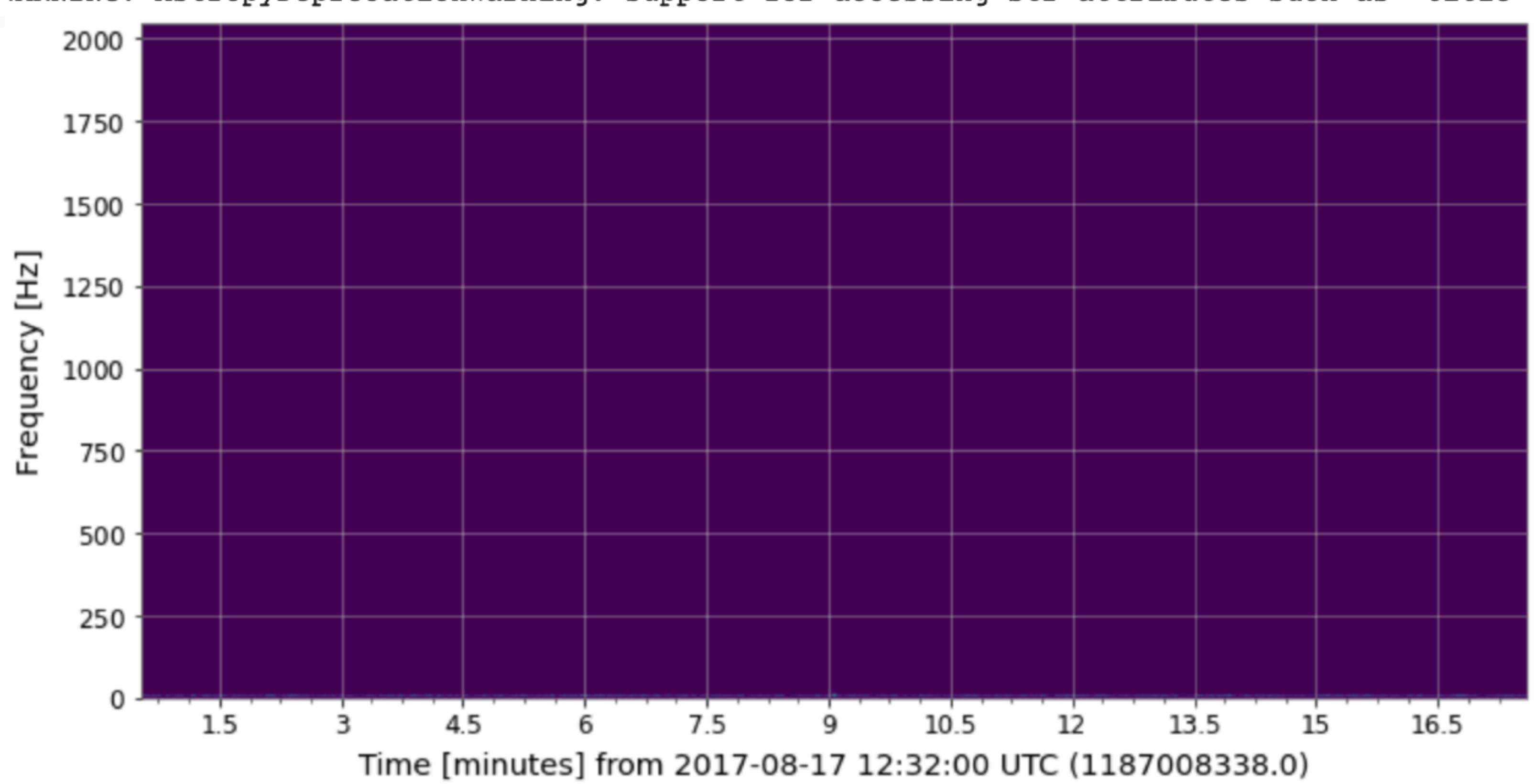
# GW170817: Spectrograms in gwpy

```python
from gwosc.datasets import event_gps
from gwpy.timeseries import TimeSeries

gps = event_gps('GW170817')
print("GW170817 GPS:", gps)

ldata = TimeSeries.fetch_open_data('L1', int(gps)-512, int(gps)+512, cache=True)
print("GW170817 data")
print(ldata)
```

```
GW170817 GPS: 1187008882.4
GW170817 data
TimeSeries([2.06056010e-20, 1.59181918e-20, 2.18438811e-20, ...,
            1.25504332e-19, 1.23976846e-19, 1.22231459e-19]
           unit: dimensionless,
           t0: 1187008370.0 s,
           dt: 0.000244140625 s,
           name: Strain,
           channel: None)
```

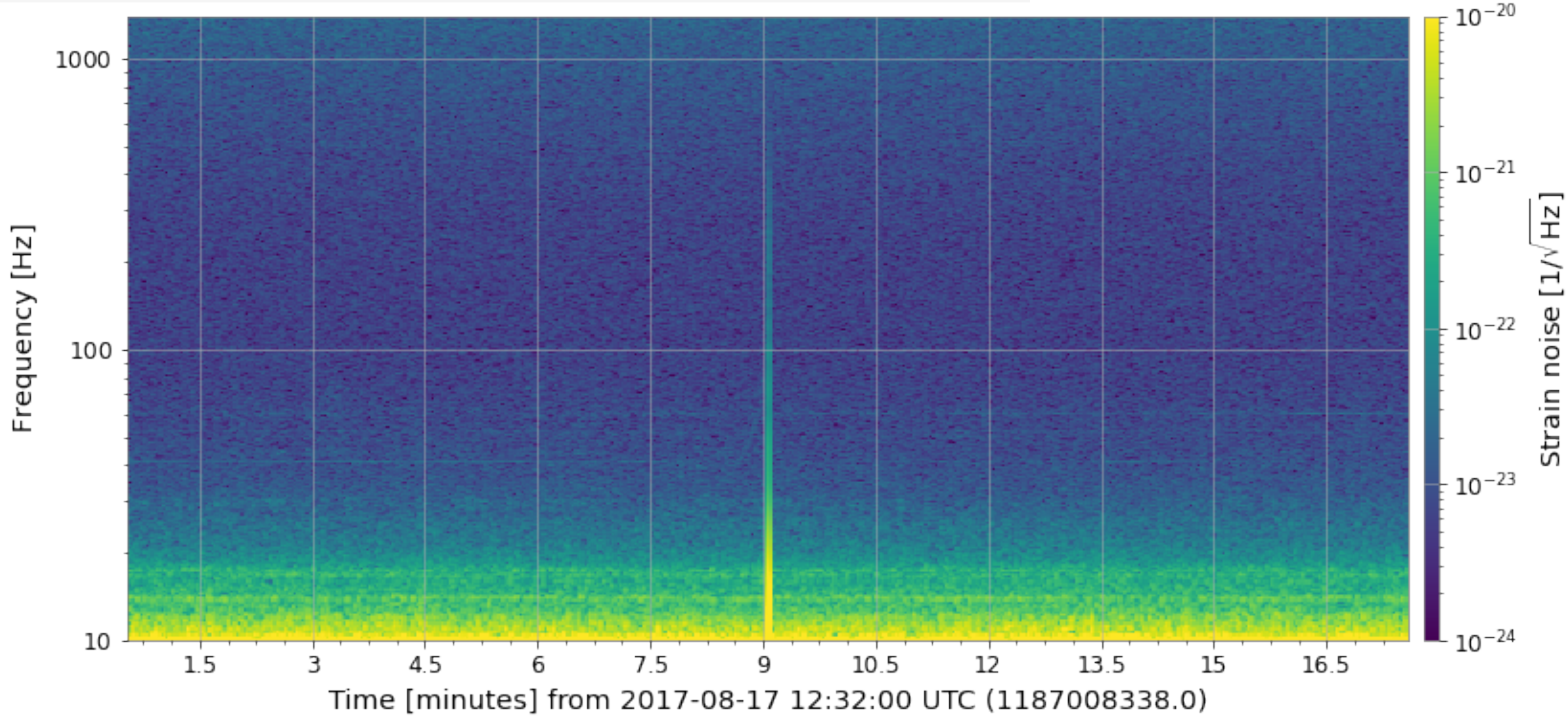
```python
specgram = ldata.spectrogram2(fftlength=4, overlap=2, window='hann') ** (1/2.)
plot = specgram.plot()
```

```
WARNING: AstropyDeprecationWarning: support for accessing str attributes such as 'title' from Phys
```



# GW170817: Spectrograms in gwpy

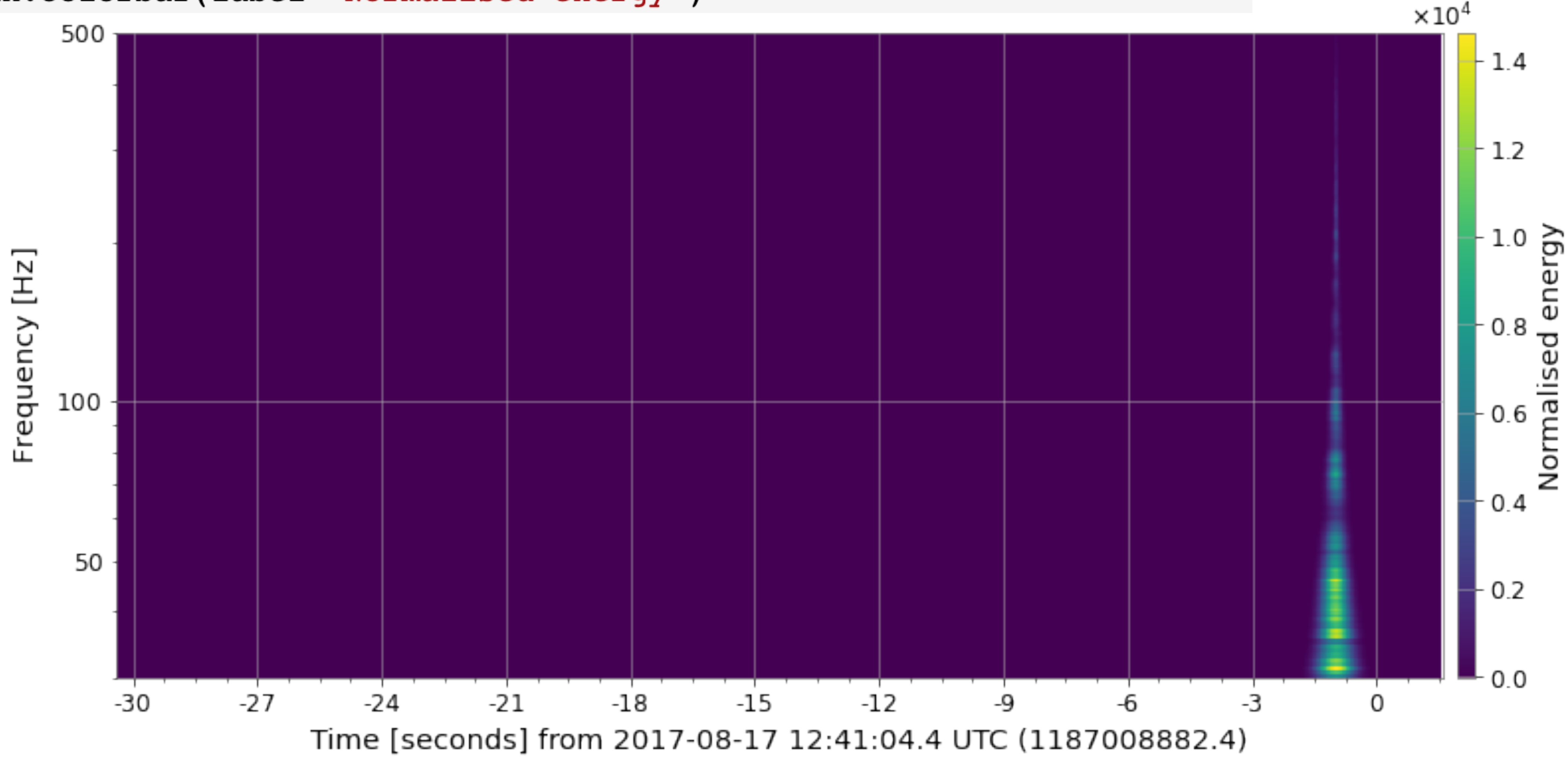
```python
ax = plot.gca()
ax.set_yscale('log')
ax.set_ylim(10, 1400)
ax.colorbar(
    clim=(1e-24, 1e-20),
    norm="log",
    label=r"Strain noise [$1/\sqrt{\mathrm{Hz}}$]",
)
plot  # refresh
```



# GW170817: Q-transform in gwpy

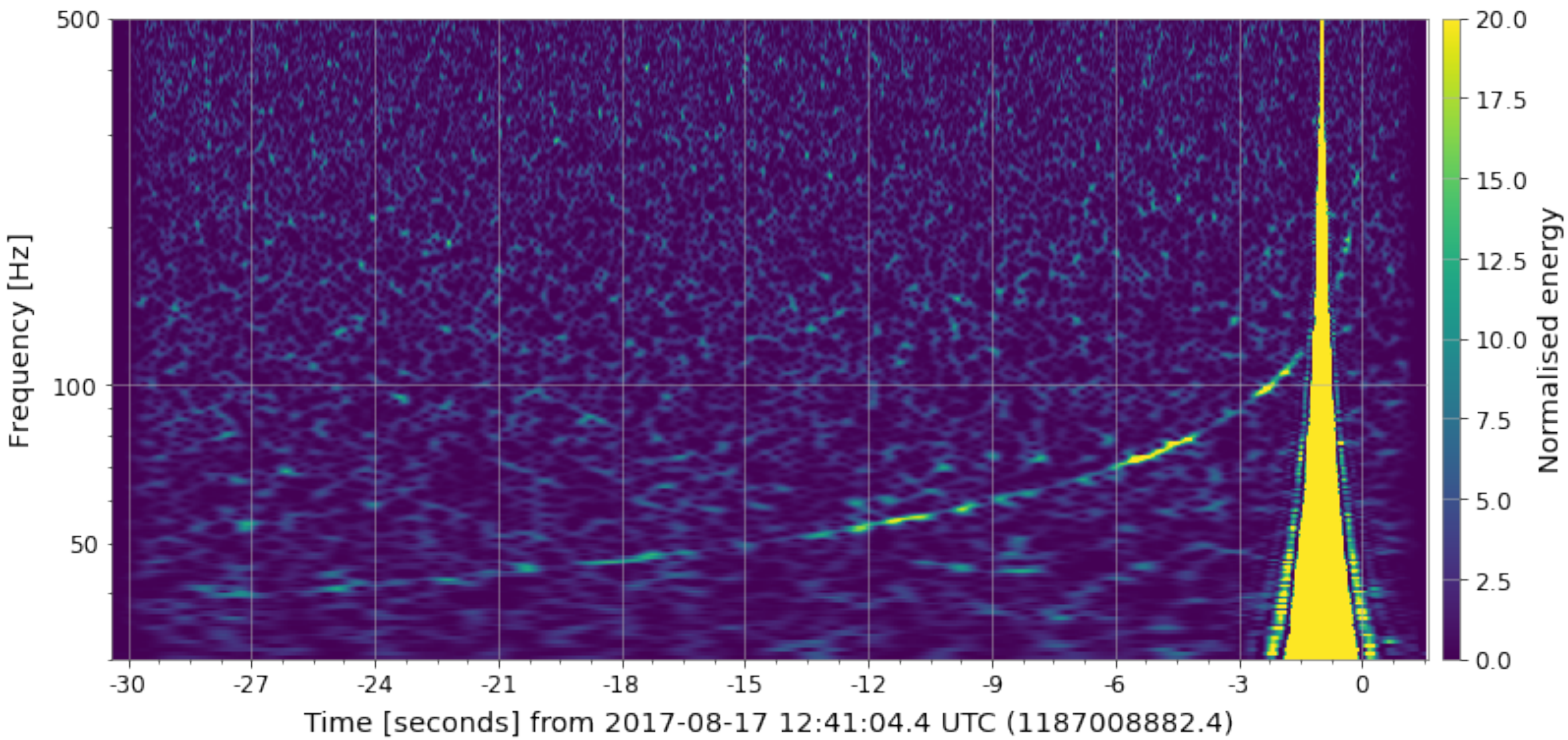
```
ldata = TimeSeries.fetch_open_data('L1', *segment, verbose=True)
```

```
lq = ldata.q_transform(frange=(30, 500), qrange=(100, 110))
plot = lq.plot()
ax = plot.gca()
ax.set_epoch(gps)
ax.set_yscale('log')
ax.colorbar(label="Normalised energy")
```



# GW170817: Q-transform in gwpy

```
plot.colorbars[0].mappable.set_clim(0,20)
plot.refresh()
plot
```



# Estimation of Q-value

- $Q$ The number of cycles in a given frequency bin
- $\Delta f$ The width of the frequency bin
- $\varepsilon$ The fractional width of the frequency bin, e.g. $\varepsilon = \Delta f / f$
- $\tau$ The time to change frequency bins

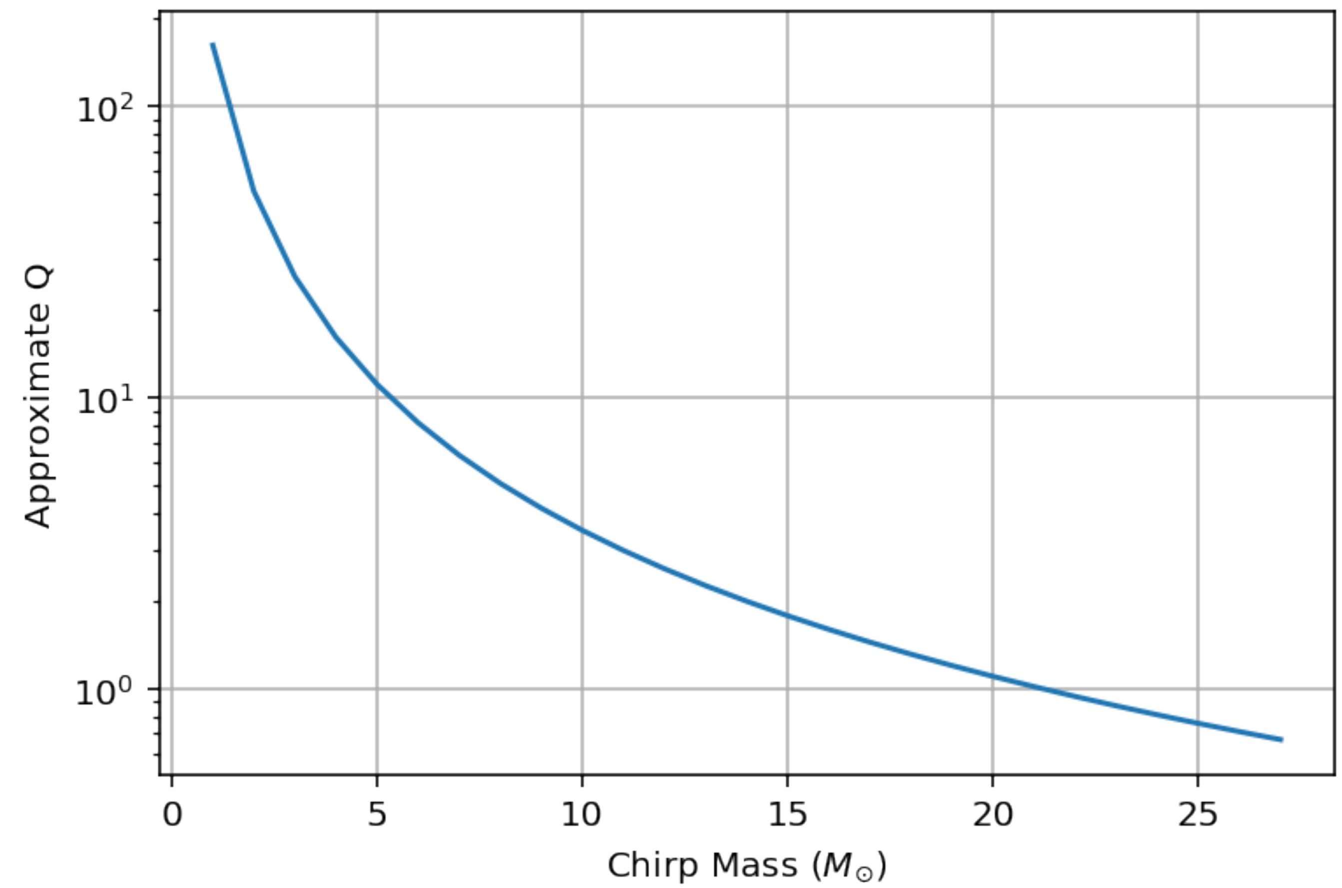
$$\Delta f = \dot{f}\tau$$

$$\tau = \Delta f / \dot{f} = \varepsilon f / \dot{f}$$

$$Q = \tau f$$

$$Q = \frac{\varepsilon f^2}{\dot{f}}$$

$$\mathcal{M} = \frac{c^3}{G}\left(\left(\frac{5}{96}\right)^3 \pi^{-8} (f_{\mathrm{GW}})^{-11} \left(\dot{f}_{\mathrm{GW}}\right)^3\right)^{1/5}$$




$$Q = \varepsilon \times (\alpha/M)^{5/3} \times f^{-5/3}$$

https://github.com/jkanner/gw-intro/blob/main/extra/Estimate%20Q-value.ipynb

# Noise Subtraction

After data collection we remove several independently measured terrestrial contributions to the detector noise:

- LIGO - remove calibration lines and 60Hz AC power mains harmonics. We also remove some additional noise due to non stationary couplings

- Virgo - remove broadband noise, including frequency noise from the laser, noise introduced when controlling the displacement of the beam splitter and amplitude noise of the 56 MHz modulation frequency.

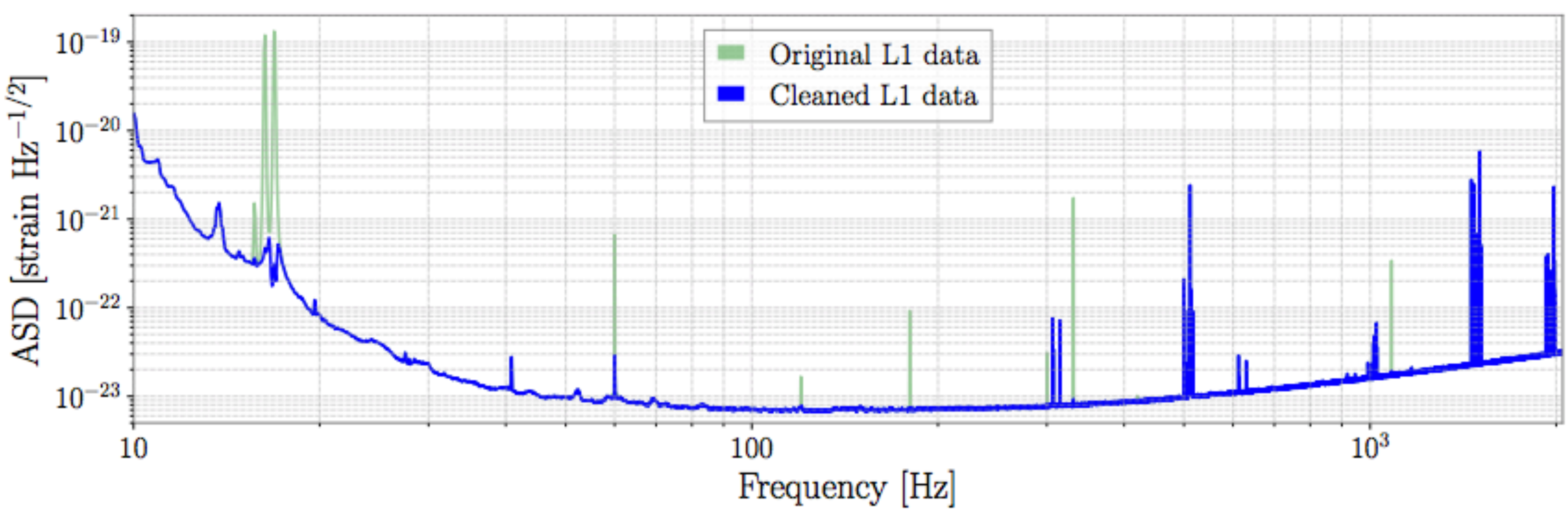
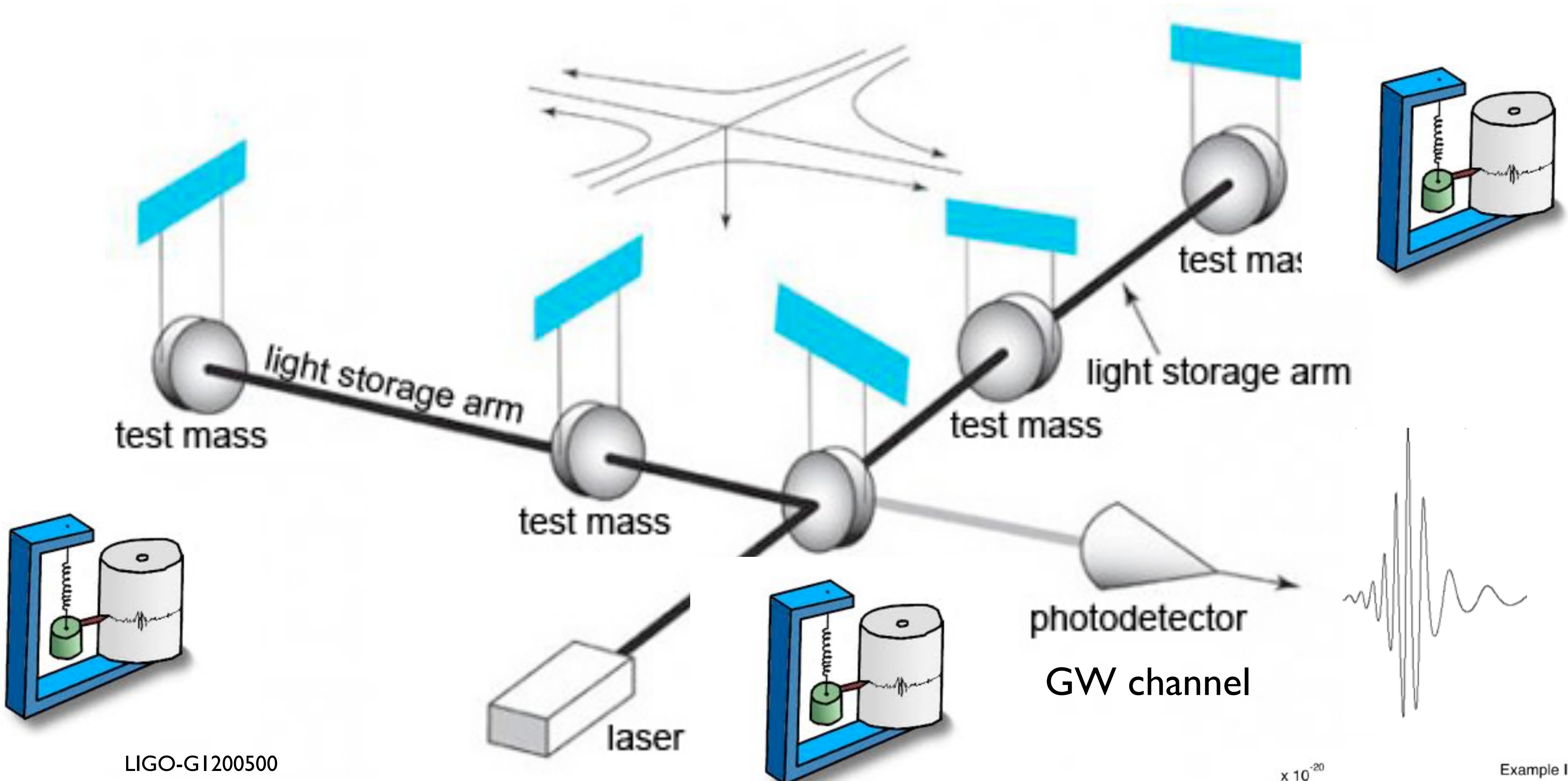



Image: D. Davis et al., *Improving the sensitivity of Advanced LIGO using noise subtraction*, CQG **36** 055011 (2019)



Laura Nuttall in GW ODW #4, 2021

# Auxiliary Channels

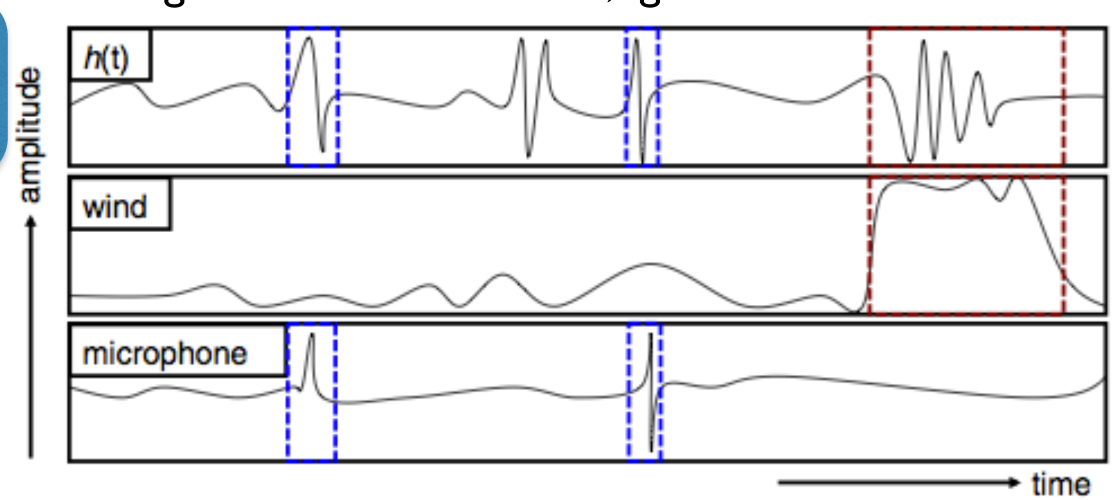


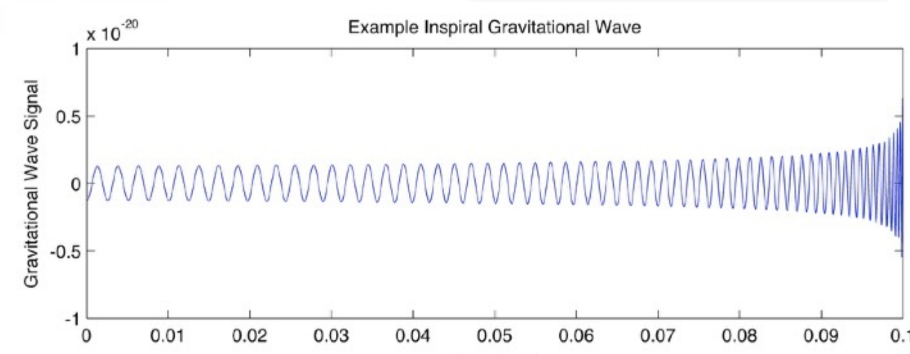

LIGO-G1200500

Signal Search

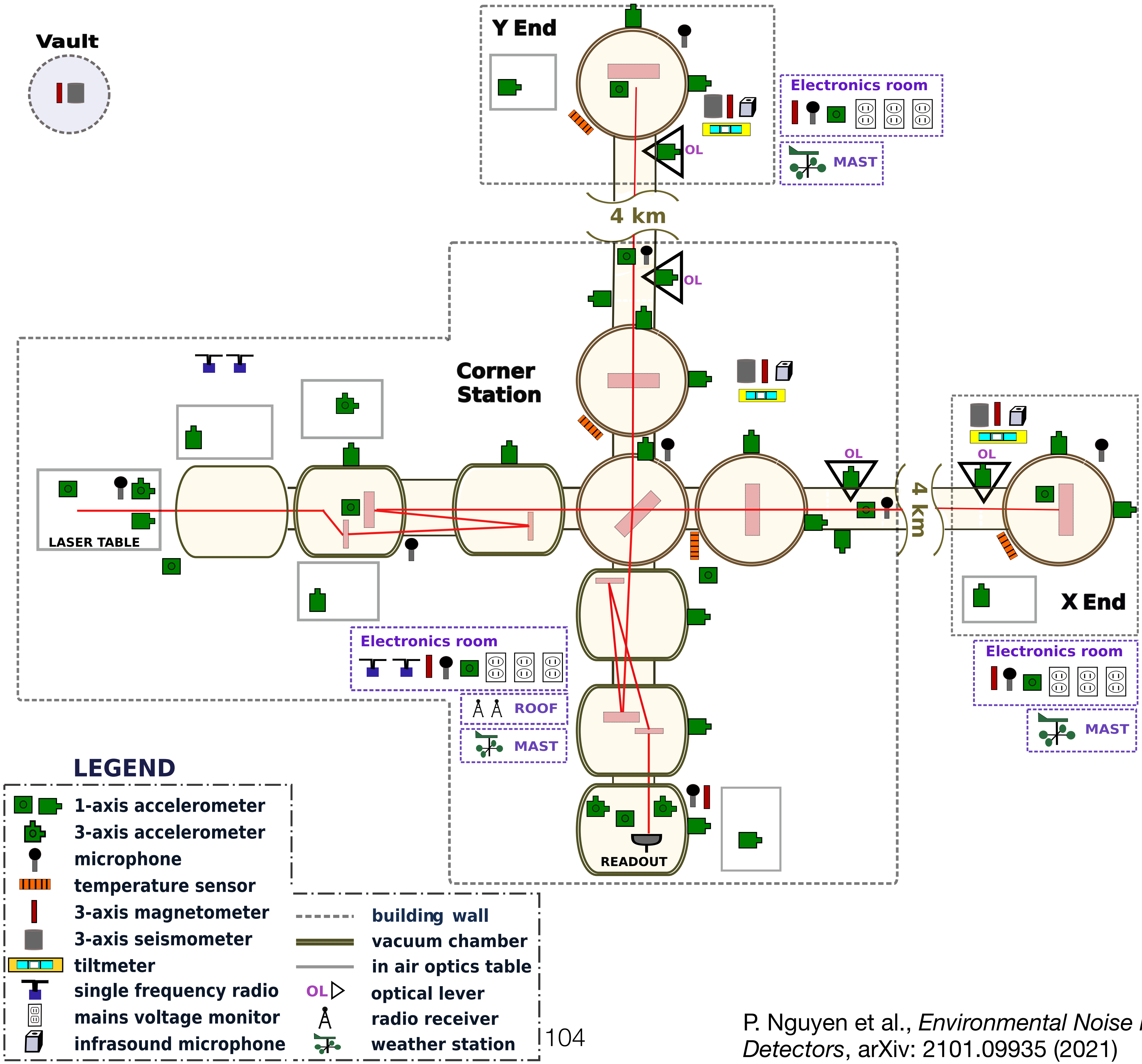
Detector Characterization

non-gaussian noise artifacts, "glitch"



# Physical Environment Channels

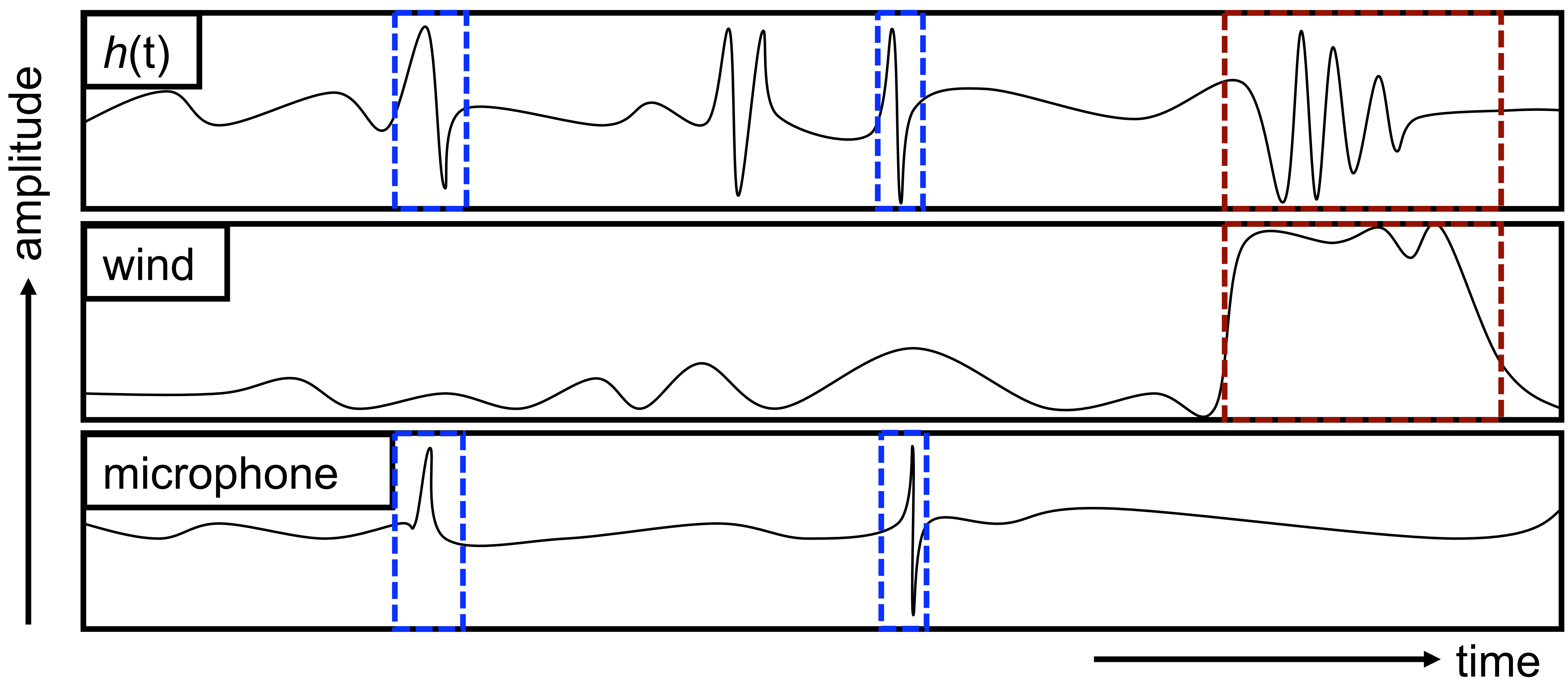




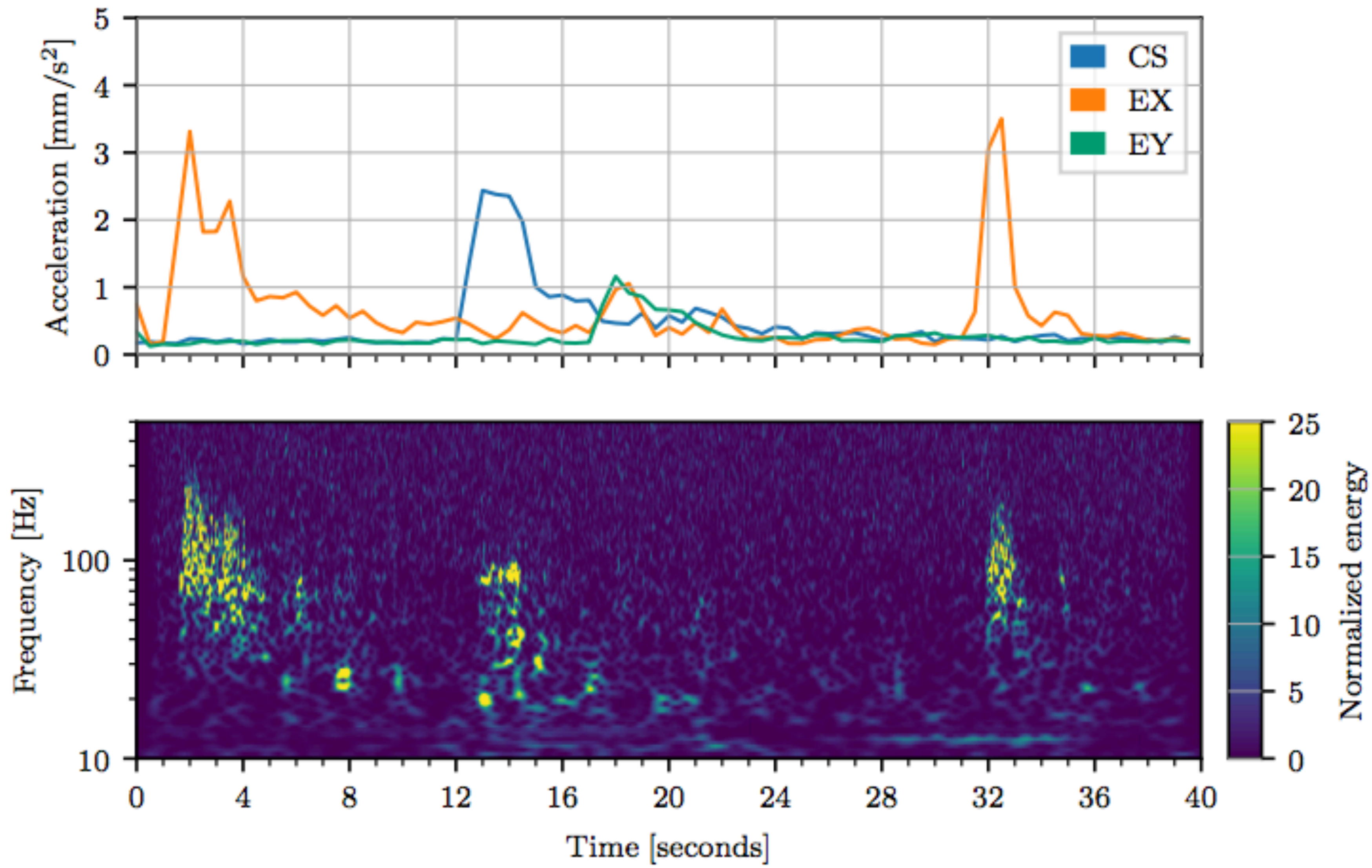

P. Nguyen et al., *Environmental Noise in Advanced LIGO Detectors*, arXiv: 2101.09935 (2021)

# Correlations with Auxiliary channels

- We record over 200,000 channels per detector that monitor environment and detector behaviour
- We can use them to help track down and trace instrumental causes of glitches that pollute the searches.



# Thunderstorms

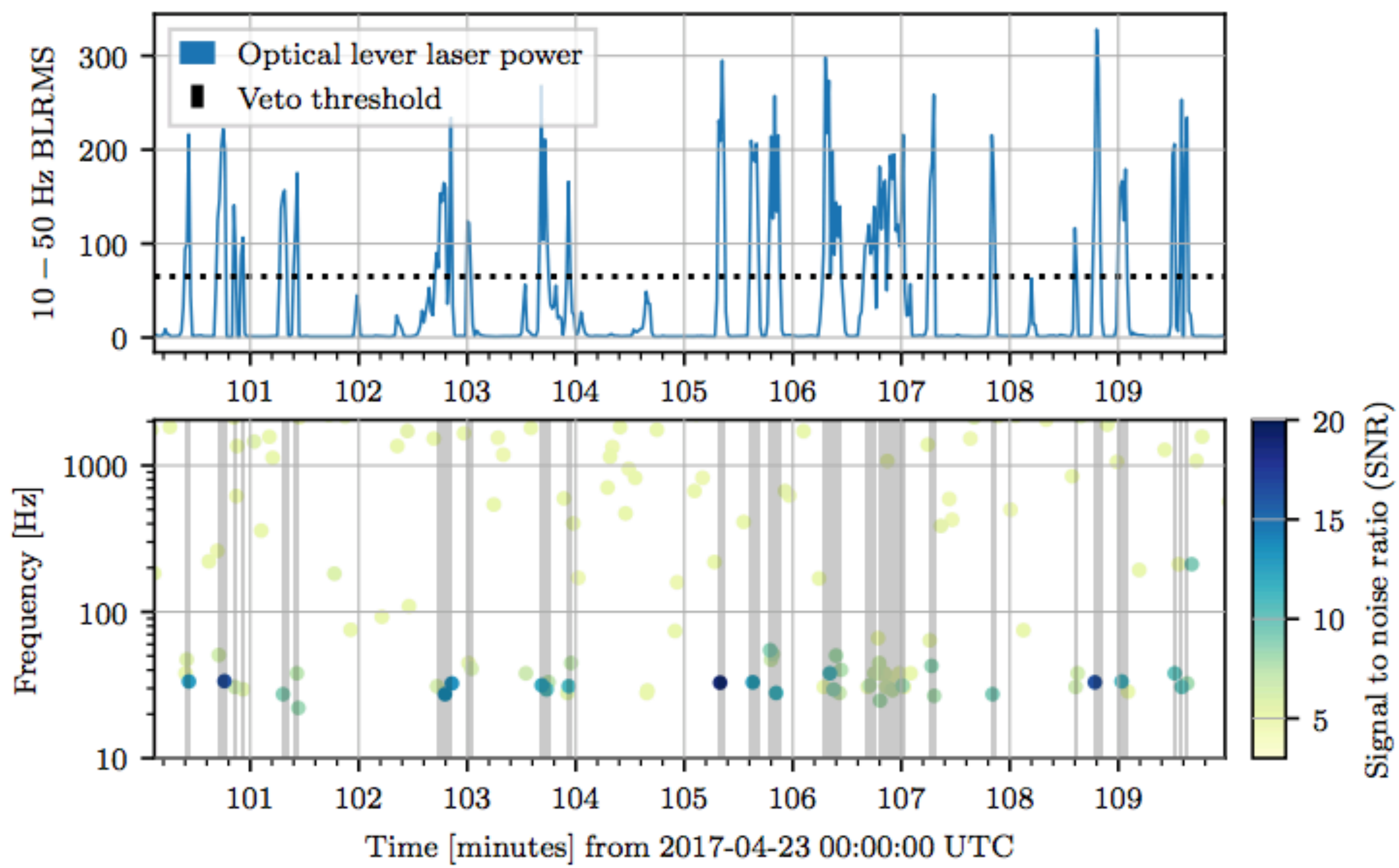



- Top: Data between 10-100 Hz from accelerometers located in the corner station (CS), End X station (EX) and End Y station (EY)

- Bottom: Spectrogram of the GW strain channel at the same time. Excess noise in the frequency range of 20 Hz to 200 Hz coincides with the thunderclaps, with intensity depending on the thunder's location.

D. Davis et al., *LIGO Detector Characterization in the Second and Third Observing Runs*, arXiv: 2101.11673 (2021)

# Example of a data quality veto in O2



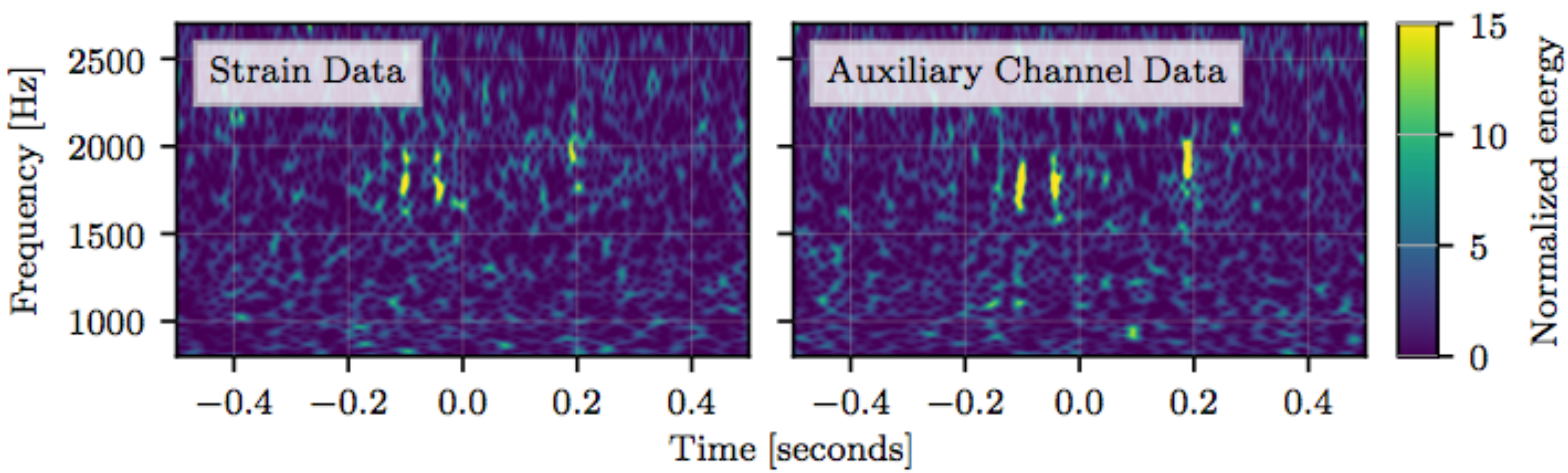

D. Davis et al., *LIGO Detector Characterization in the Second and Third Observing Runs*, arXiv: 2101.11673 (2021)

# S191110af

- Potential Burst Source in LIGO and Virgo

- Looking at the LIGO-Hanford data, there was a clear correlation between an auxiliary channel (i.e. not sensitive to GWs) and the gravitational-wave strain channel

  - Similar morphology between the two channels
  - Origin of this event is instrumental (from the output mode cleaner) rather than astrophysical




D. Davis et al., *LIGO Detector Characterization in the Second and Third Observing Runs*, arXiv: 2101.11673 (2021)

# Data quality of individual events

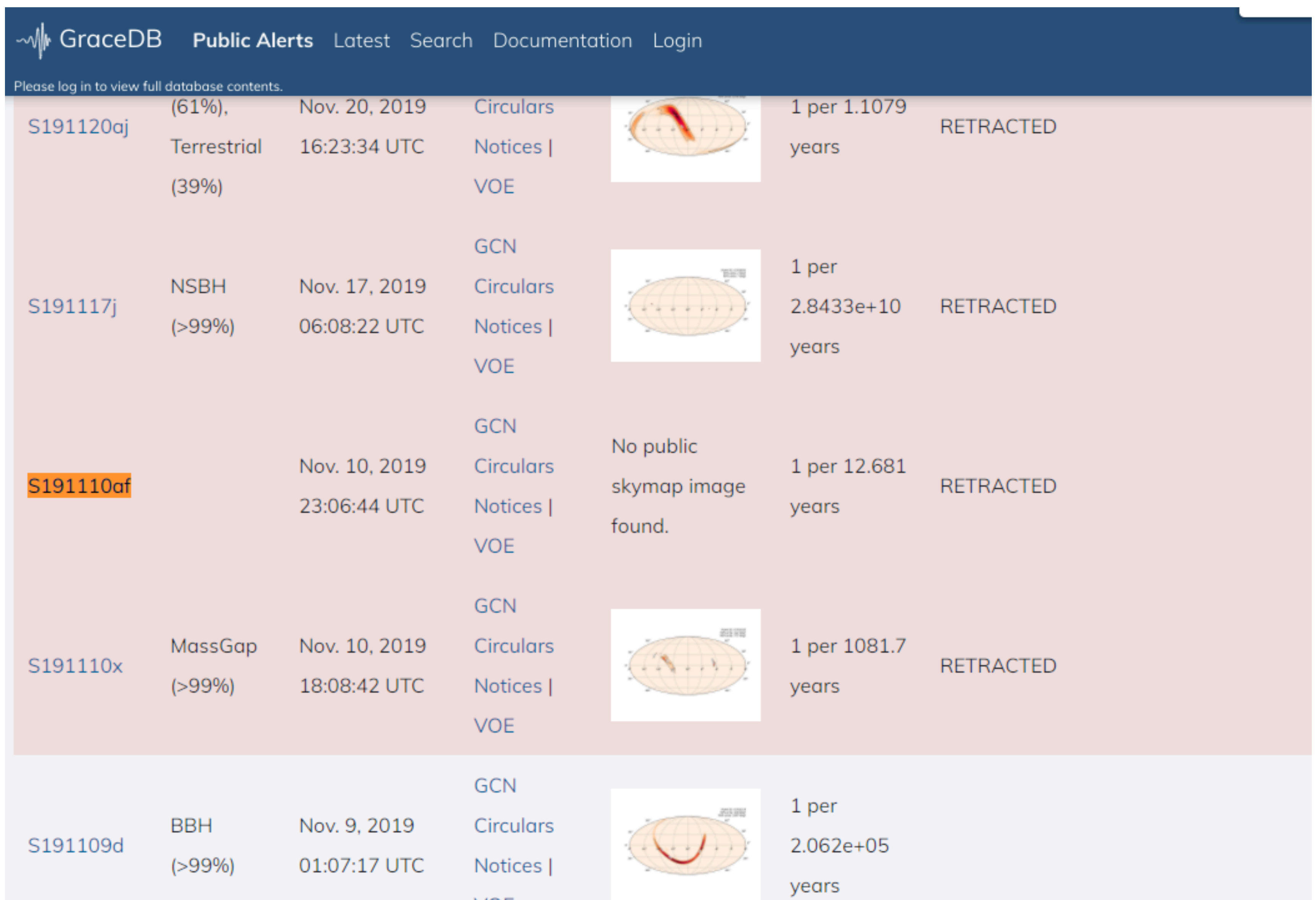
Evaluation of the data quality around an event is important to:

- identify a clear instrumental origin and issue a retraction

# Data quality of individual events

Evaluation of the data quality around an event is important to:

- identify a clear instrumental origin and issue a retraction


GraceDB Public Alerts Latest Search Documentation Login

Please log in to view full database contents.

| | | | | | | |
|---|---|---|---|---|---|---|
| S191120aj | (61%), Terrestrial (39%) | Nov. 20, 2019 16:23:34 UTC | Circulars Notices \| VOE | | 1 per 1.1079 years | RETRACTED |
| S191117j | NSBH (>99%) | Nov. 17, 2019 06:08:22 UTC | GCN Circulars Notices \| VOE | | 1 per 2.8433e+10 years | RETRACTED |
| S191110af | | Nov. 10, 2019 23:06:44 UTC | GCN Circulars Notices \| VOE | No public skymap image found. | 1 per 12.681 years | RETRACTED |
| S191110x | MassGap (>99%) | Nov. 10, 2019 18:08:42 UTC | GCN Circulars Notices \| VOE | | 1 per 1081.7 years | RETRACTED |
| S191109d | BBH (>99%) | Nov. 9, 2019 01:07:17 UTC | GCN Circulars Notices \| VOE | | 1 per 2.062e+05 years | |

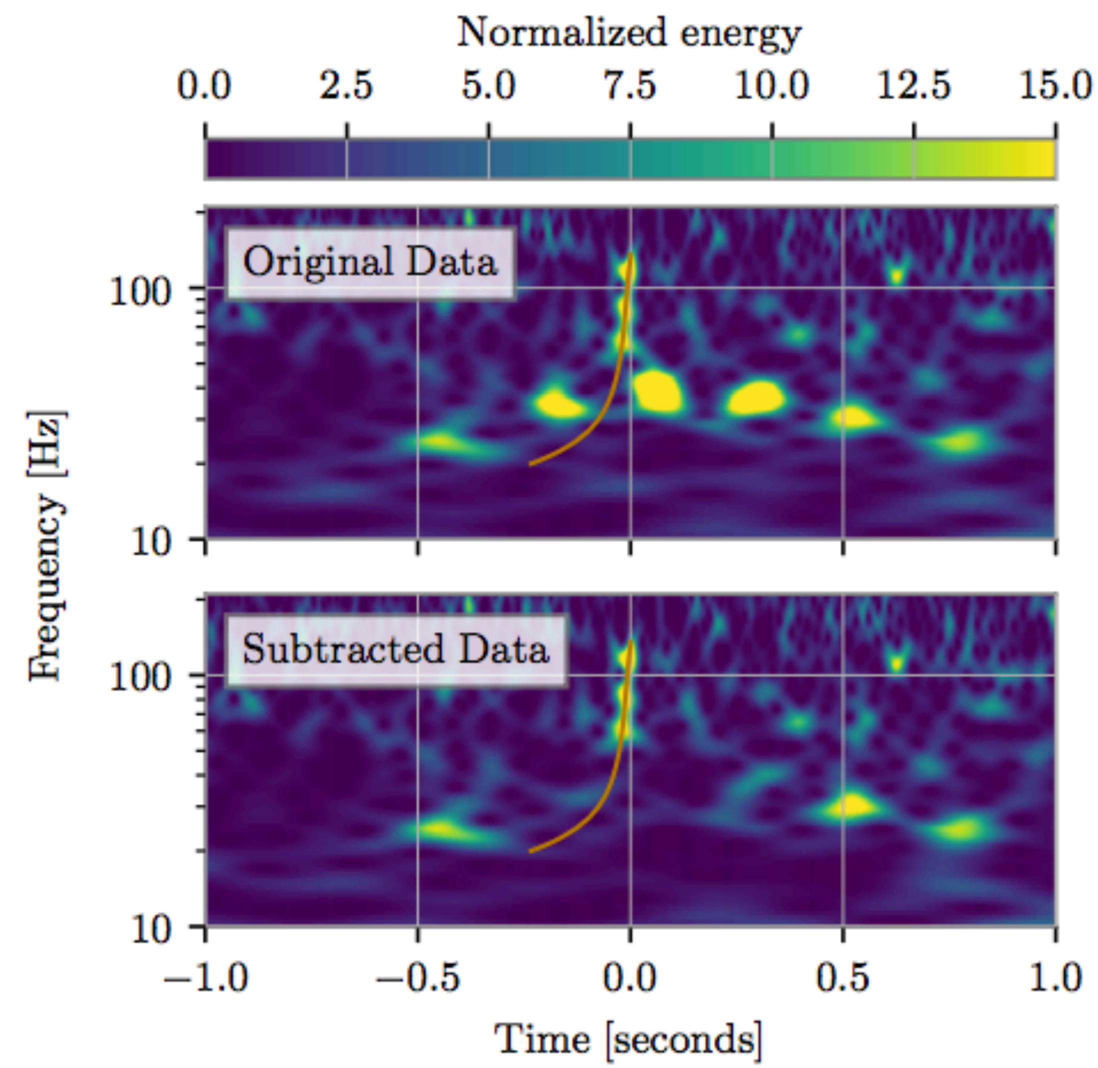
Laura Nuttall in GW ODW #4, 2021

https://gracedb.ligo.org/superevents/public/O3/

# Data quality of individual events

Evaluation of the data quality around an event is important to:

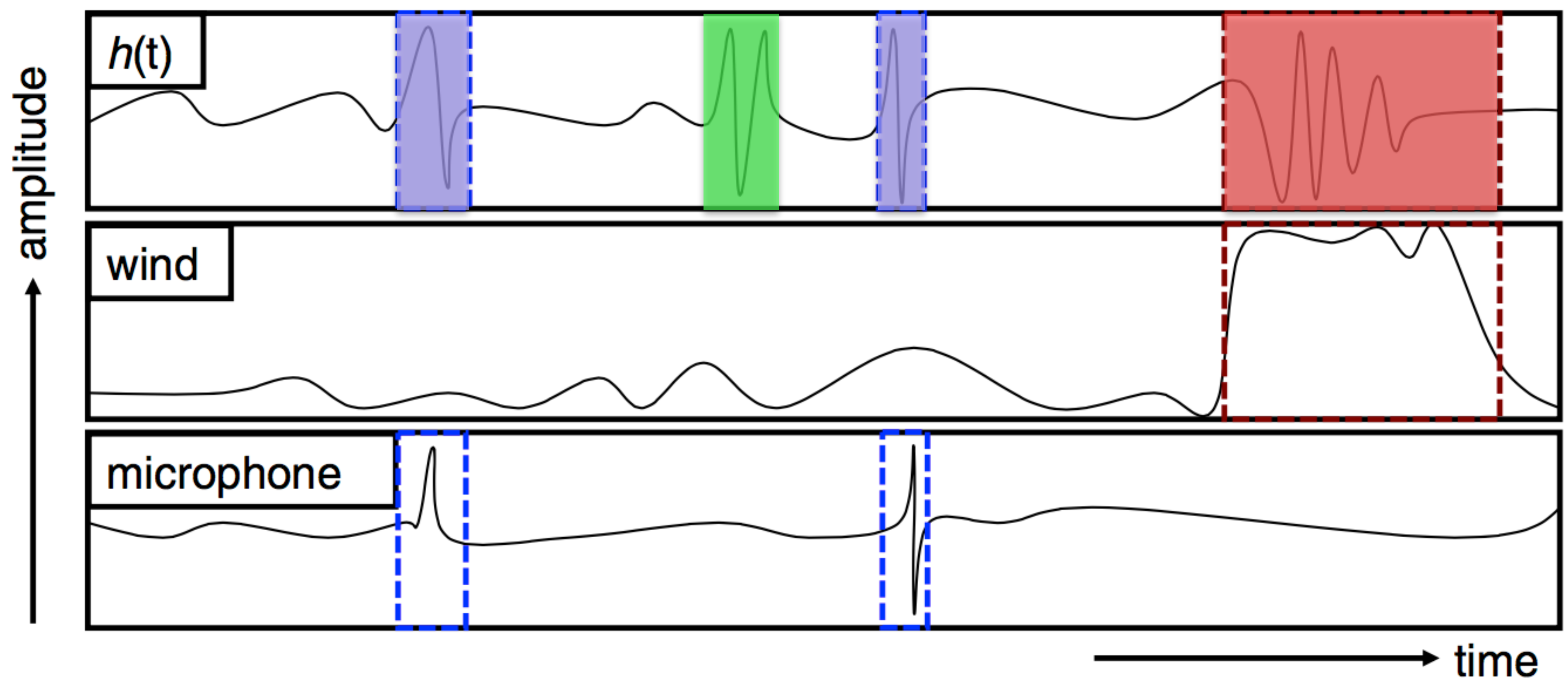
- identify a clear instrumental origin and issue a retraction
- rule out an instrumental origin (i.e. all GW events that have been published)
- identify if any instrumental noise needs to be mitigated before an analysis to determine the GW parameters is completed
  - i.e. glitch subtraction around candidate events.




Scattered light was present around the event GW190701_203306. Despite the overlap, the excess power from the glitch is successfully modelled and subtracted

R. Abbott et al., *GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo During the First Half of the Third Observing Run*, arXiv: 2010.14527 (2020)

# Veto

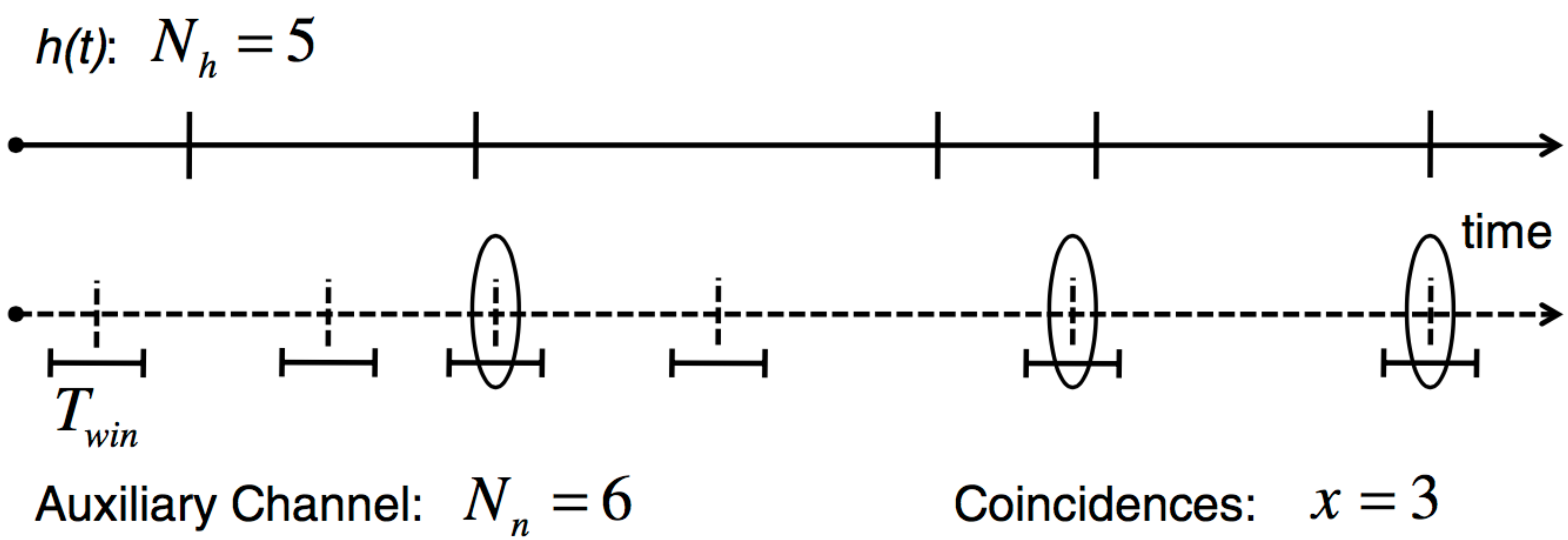



GW channel

Auxiliary channels

**Data Quality Flags**: exclude periods of data for known noises
**Data Quality Triggers**: short duration vetoes generated by algorithms that identify significant statistical correlation between a transient in h(t) and transient noise in auxiliary channels

# Counting Experiment




**Poisson Statistics**

# Poisson statistics

- Poisson distribution expresses probability of a number of independent events occurring in a given time period
- apply to coincidence
- Definitions
  - $N_{de}$ = **number** of triggers in DARM ERR channel
  - $N_n$ = number of triggers in auxilliary channel n
  - $T_{win}$ = full time window centered on auxilliary channel trigger
  - $T_{tot}$ = total live-time analyzed
- From these calculate mean number of expected coincidences

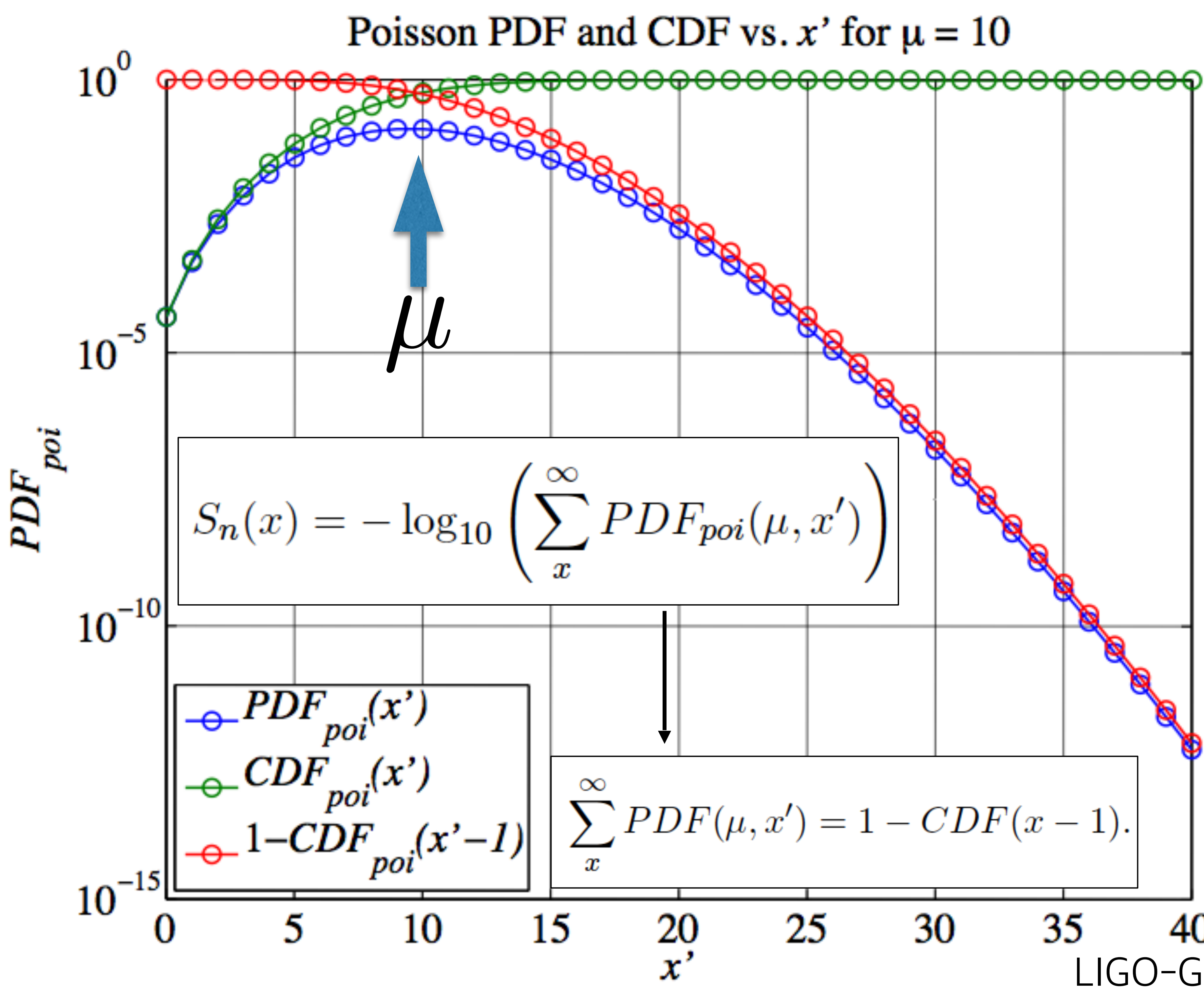
Probability Density Function

$$PDF_{poi}(\mu, x') = \frac{\mu^{x'} e^{-\mu}}{x'!},$$

Mean number of coincidence

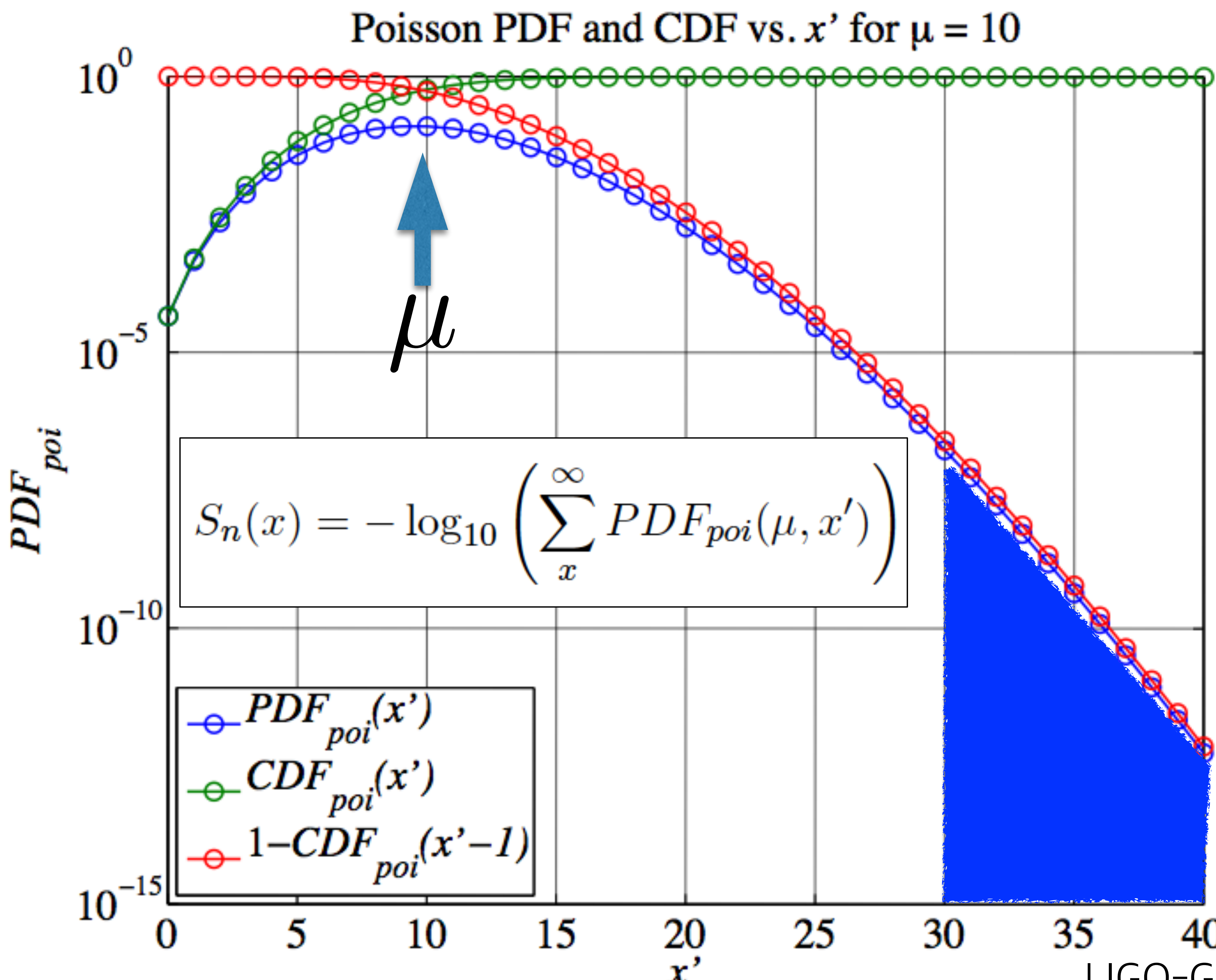
$$\mu = \frac{N_{de} N_n T_{win}}{T_{tot}}$$

# Statistical Significance




LIGO-G0900878

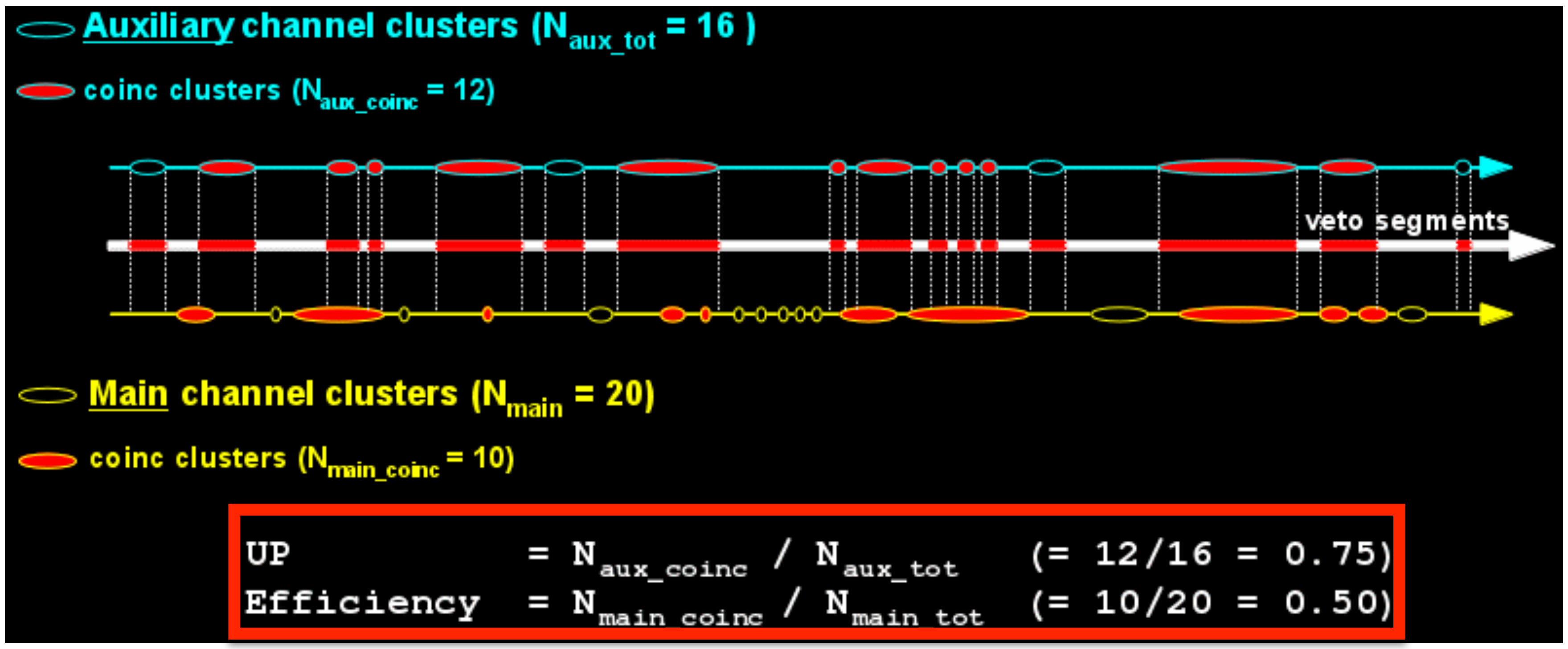
# Statistical Significance




LIGO-G0900878

# Veto Algorithms (1)

1. Use-Percentage Veto (UPV)




Journal of Physics: Conference Series 243, 012005 (2010); 14th GWDAW

2. Hierarchical Veto (Hveto)

$$S = -\log_{10}\Sigma_{k=n}^{\infty}[\frac{\mu^k e^{-\mu}}{k!}])$$

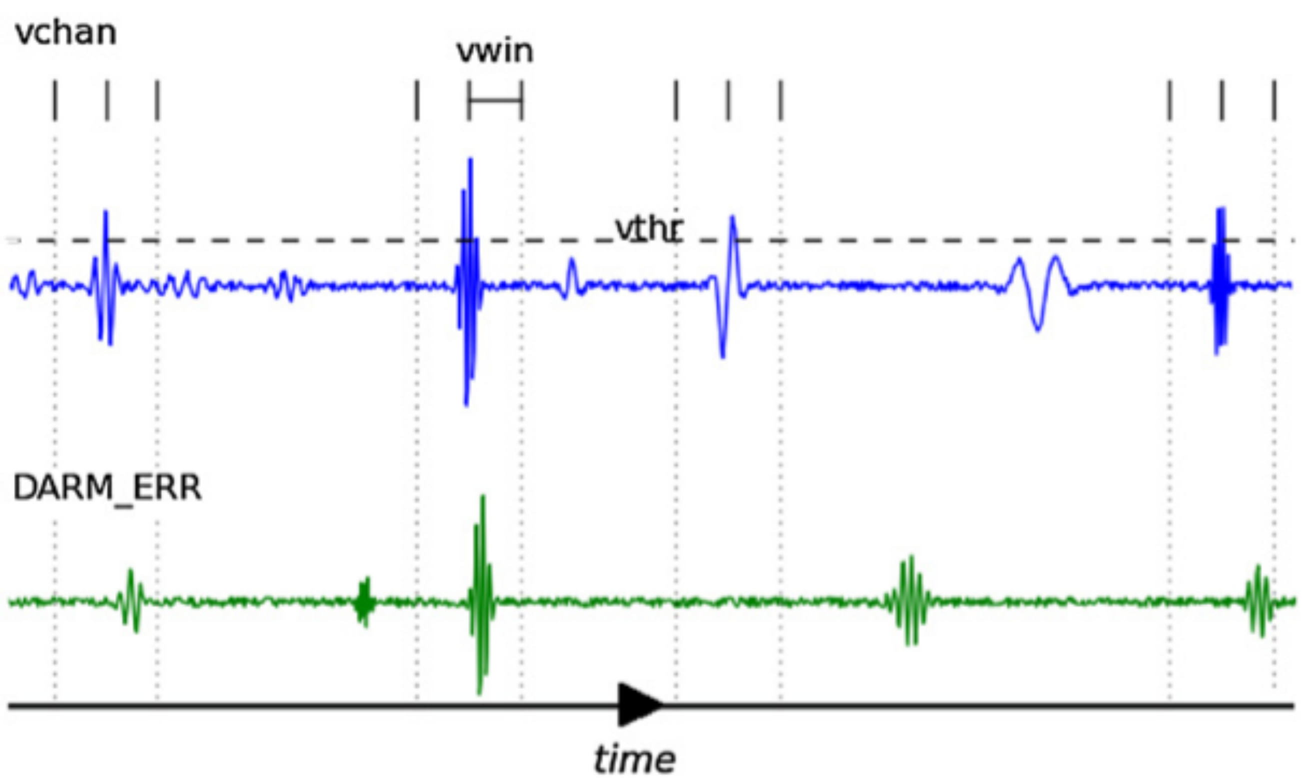
$$\mu = \frac{N_{main\_tot} N_{aux\_tot} T_{win}}{T_{tot}}$$

n : the number of coincidences
T_win : full width of coincidence time window
T_tot : a given total analysis time

Class. Quantum. Gav. 28, 235005 (2011)

# Veto Algorithms (2)

Ordered-Veto List (OVL) - used in iDQ




veto efficiency

$$e = n_c / N_{GW}$$

fractional dead time

$$f = t/T$$

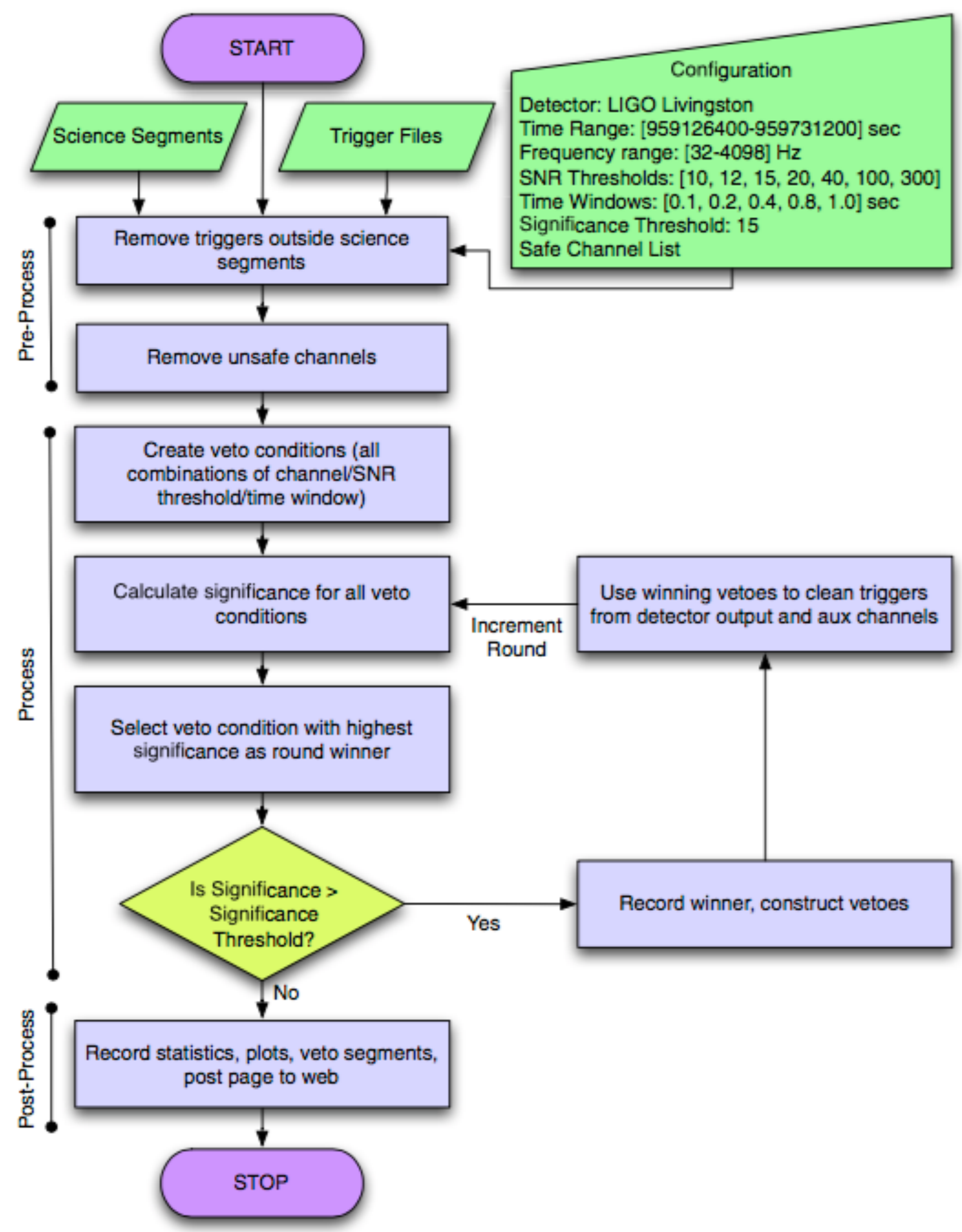
$$e/f = \frac{n_c}{t(N_{GW}/T)} \sim \frac{n_c}{t\lambda_{GW}}$$

$$t\lambda_{GW} \sim < n_c >$$

$$p = \sum_{k=n_c}^{\infty} \frac{\langle n_c \rangle^k}{k!} e^{-\langle n_c \rangle} = \sum_{k=n_c}^{\infty} \frac{(n_c(f/\varepsilon))^k}{k!} e^{-n_c(f/\varepsilon)}$$

Class. Quantum. Gav. 30, 155010 (2013)

# hveto algorithm

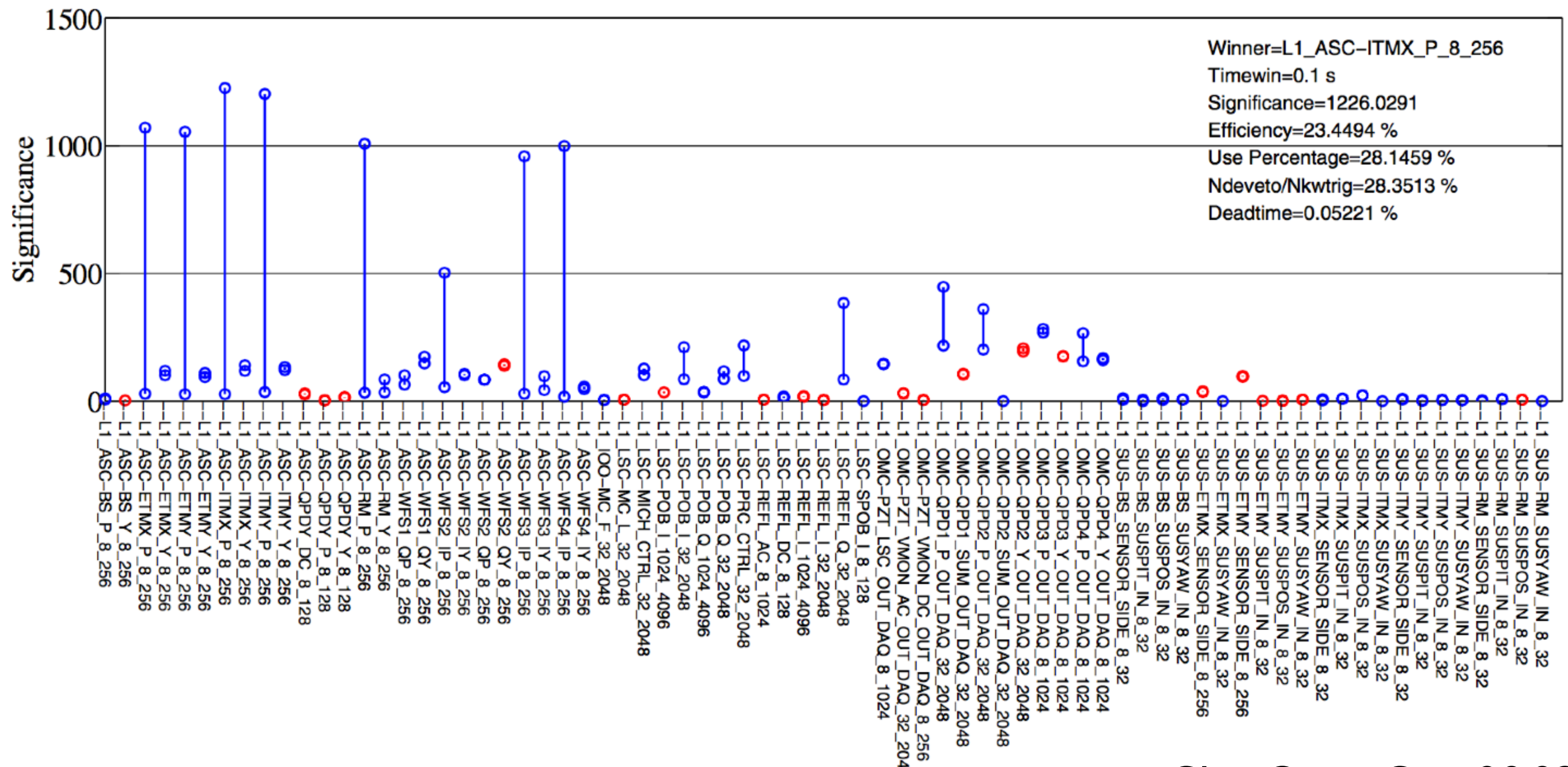


# hVeto

$$S = -\log_{10}\Sigma_{k=n}^{\infty}[\frac{\mu^k e^{-\mu}}{k!}])$$

$$\mu = \frac{N_{main\_tot}N_{aux\_tot}T_{win}}{T_{tot}}$$

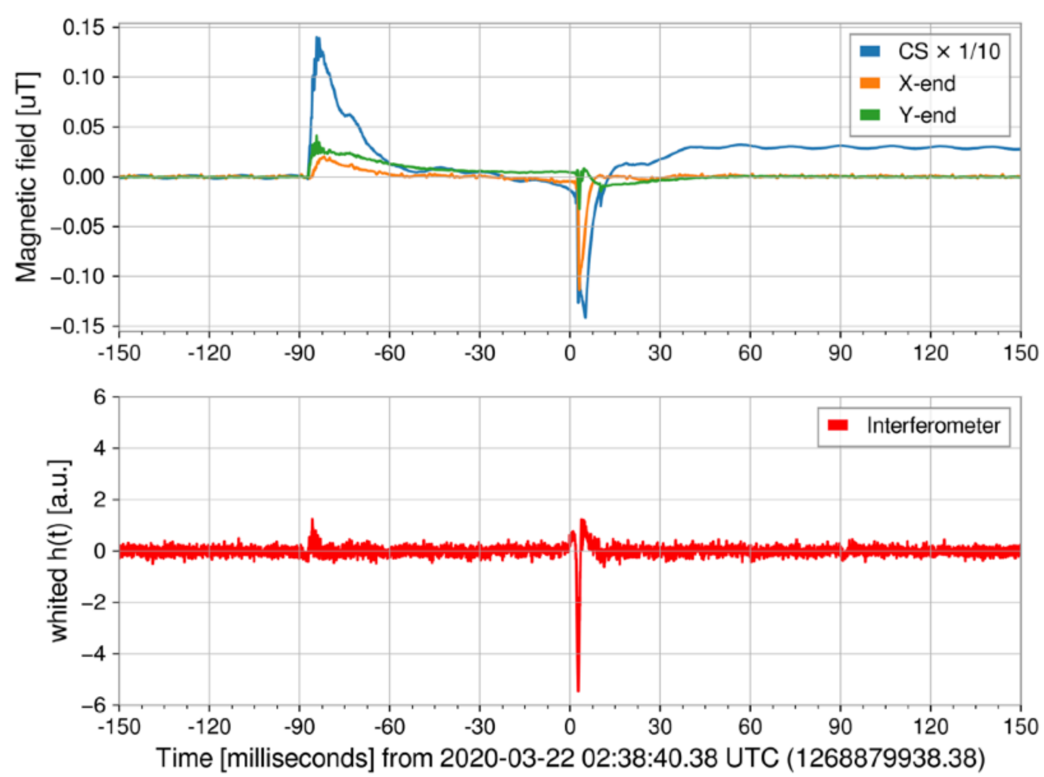
n : the number of coincidences
T_win : full width of coincidence time window
T_tot : a given total analysis time



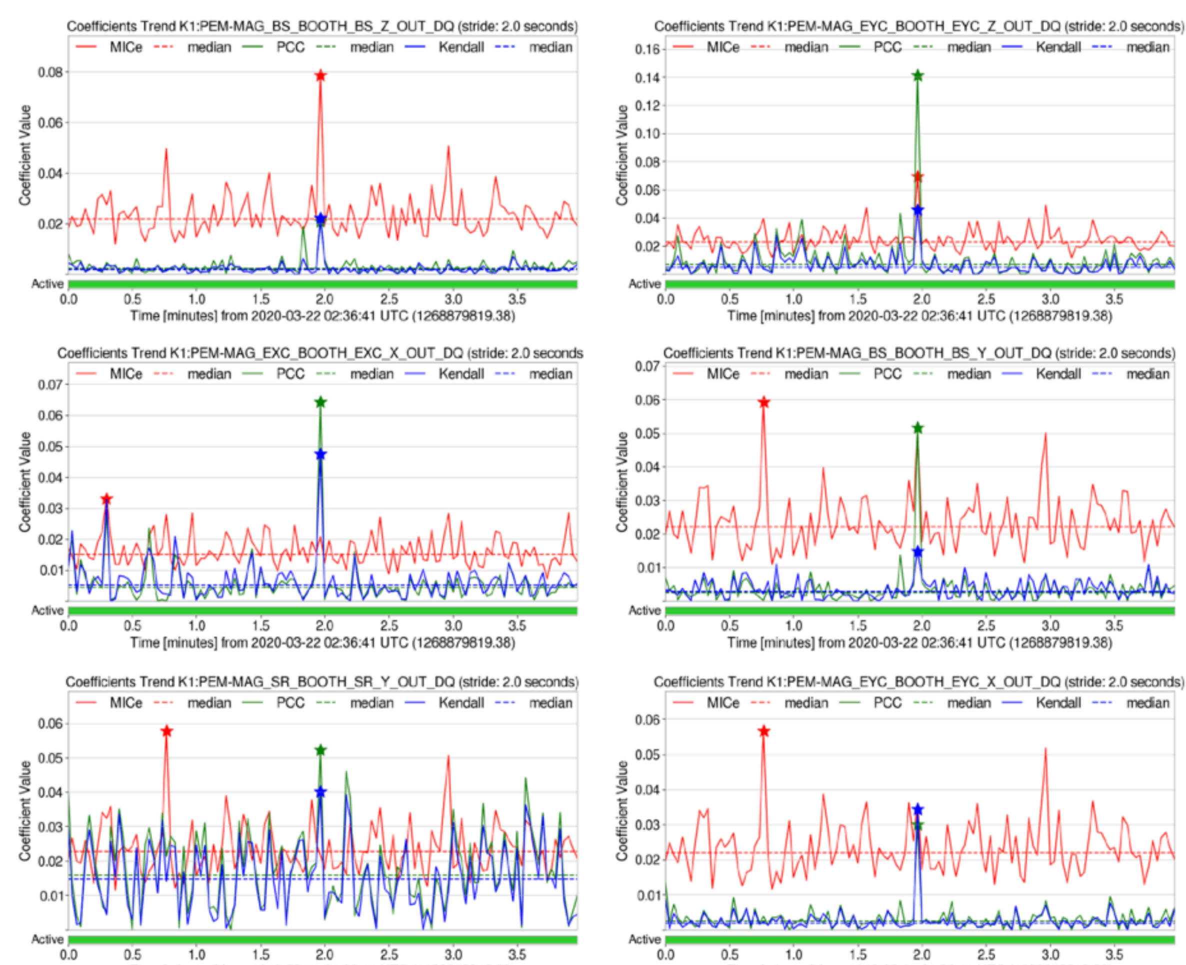

ClassQuantGrav.28.235005(2011)

# CAGMon

1. Developers : J. J. Oh (오정근), P. J. Jung (정필종)
2. Wiki: https://kgwg.nims.re.kr/wiki/DetChar/CAGMon
3. Code : https://github.com/pjjung/cagmon

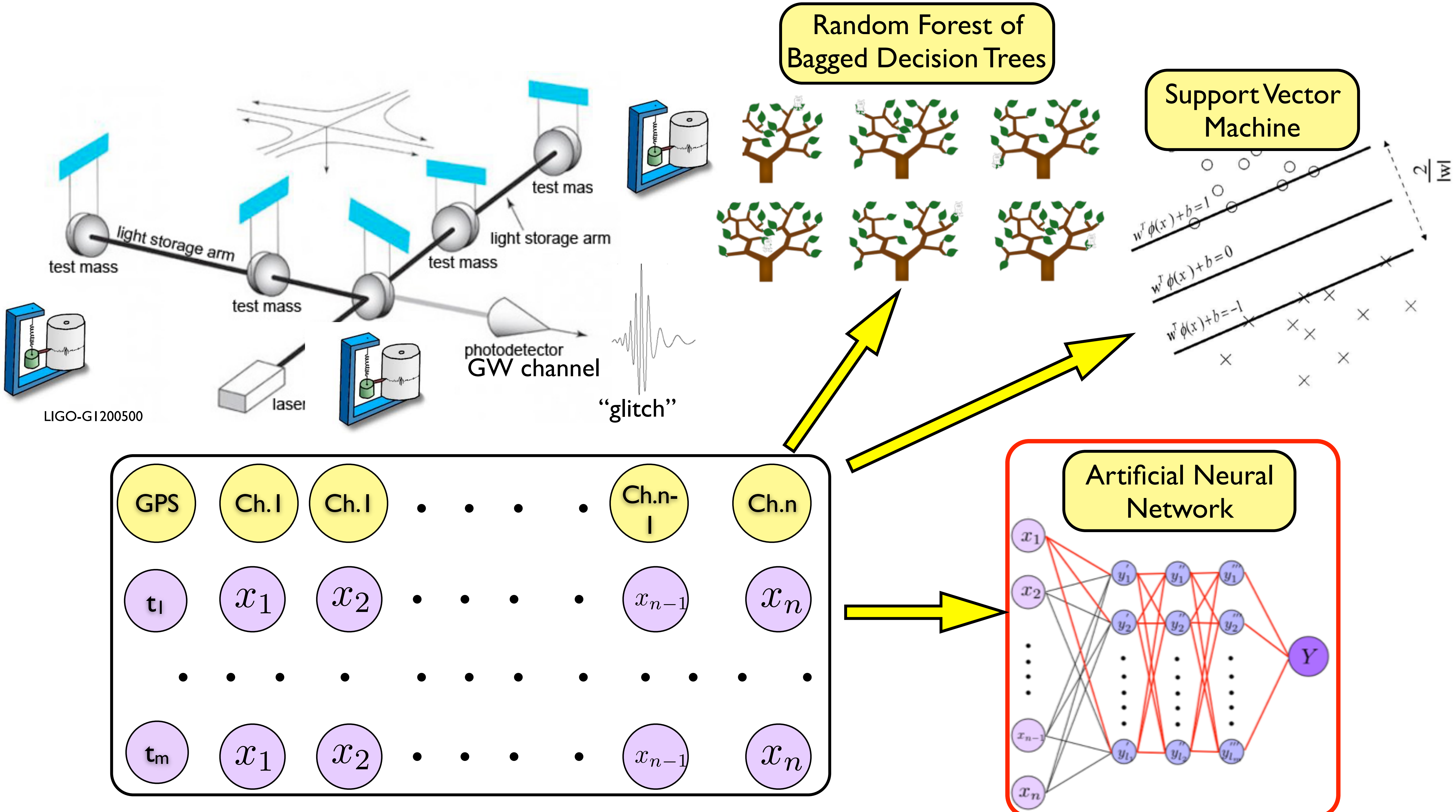



T. Washimi *et al* 2021 *JINST* **16** P07033



P.J. Jung et al. , arxiv:2204.00370, accepted by PRD

# Glitch Classification by ML



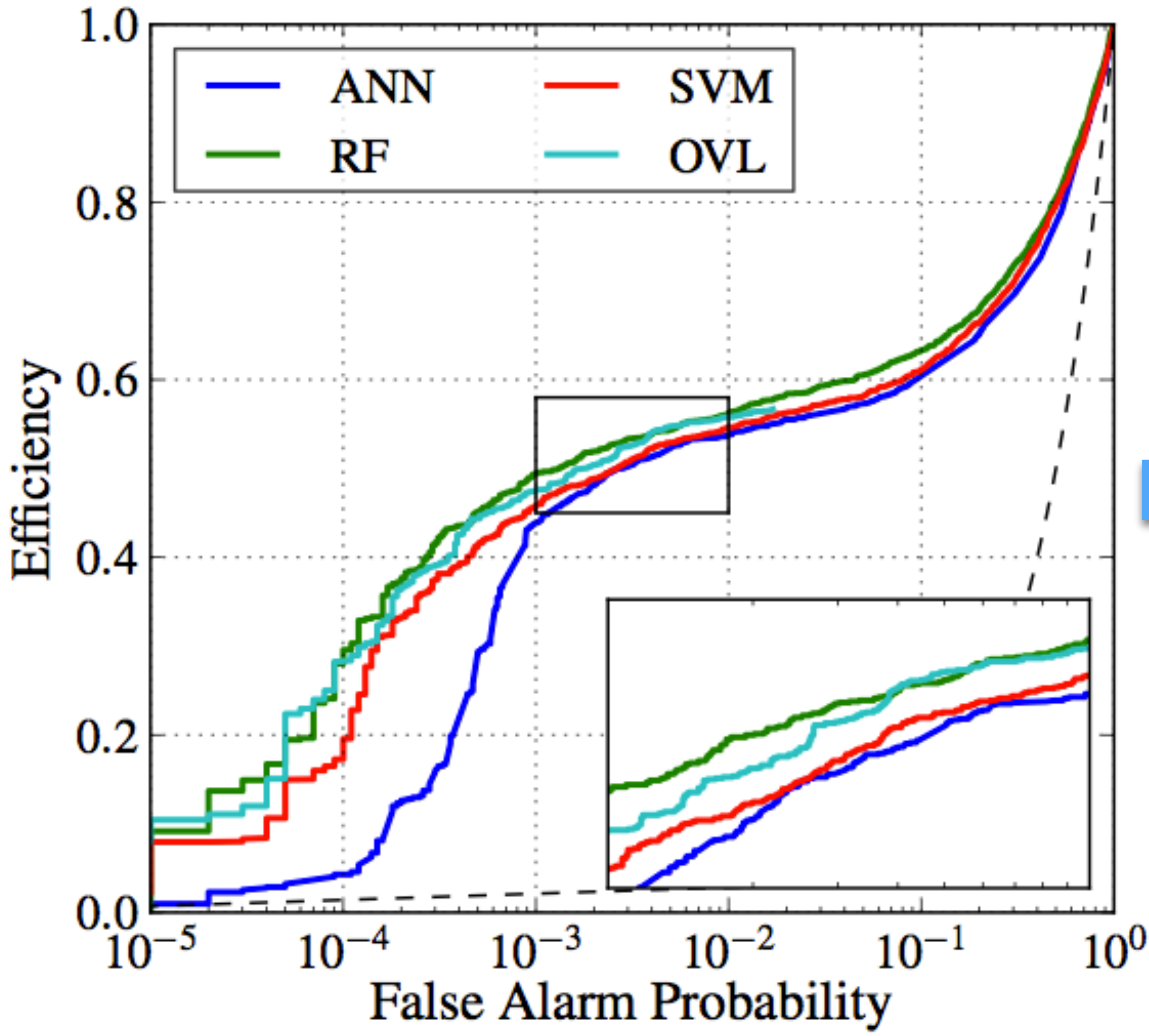

Vectorized information

# MLA application to DQ

1. Ordered Veto List (OVL) + 3 Machine Learning Algorithms

   - application to hundreds of channels among 200,000 auxiliary channels

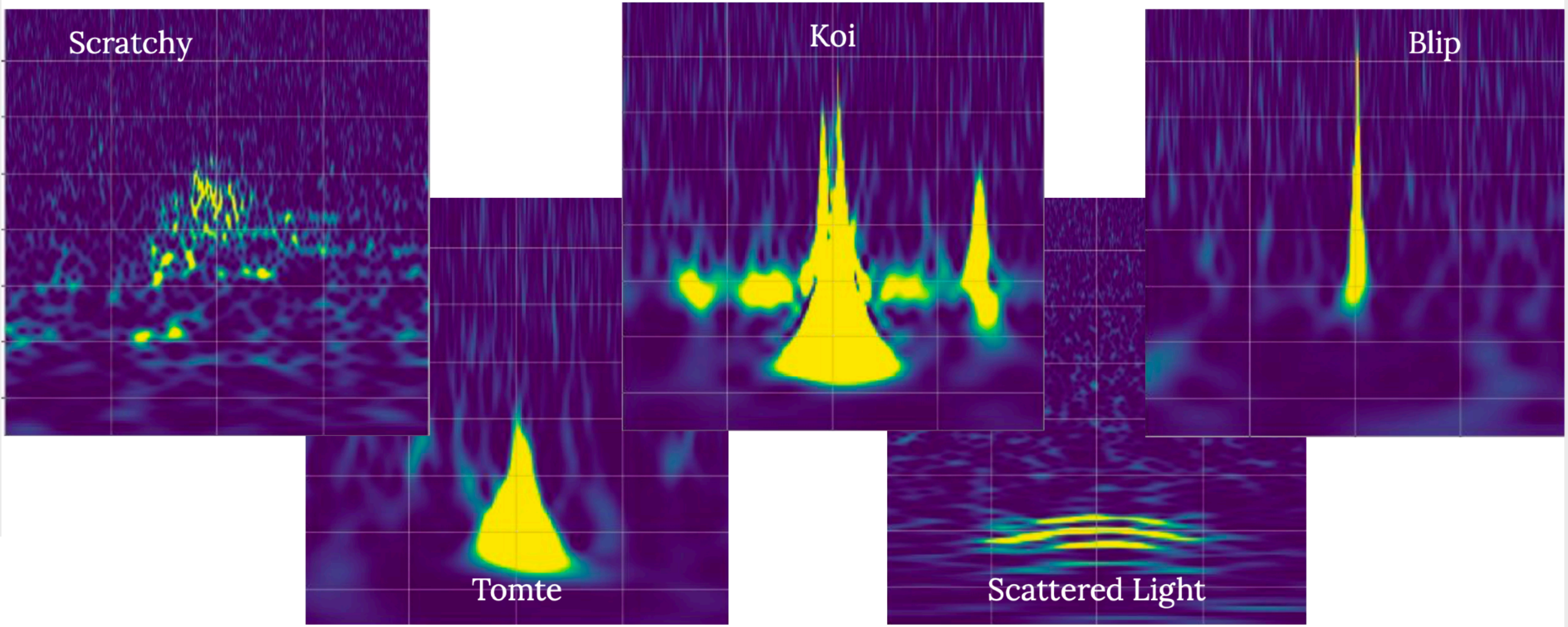
Phys. Rev. D 88, 062003 (2013)




Low Latency DQ pipeline (iDQ) for GraceDB

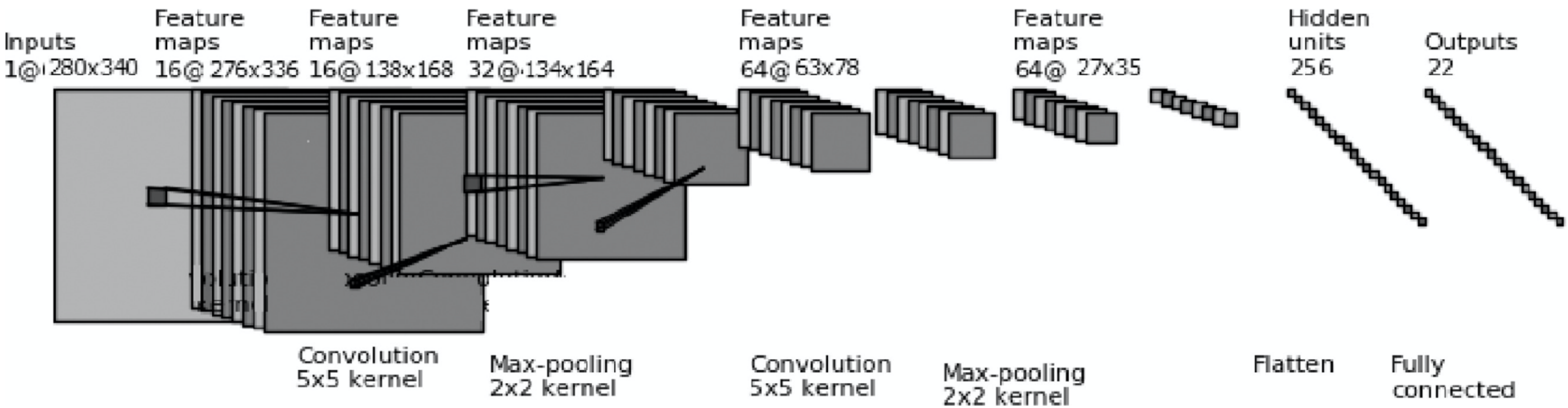
# Gravity Spy

## Non-Gaussian Glitches



# Gravity Spy

downsample: 140*170 —> merged view images (0.5, 1.0, 2.0, 4.0s)




+

Support Vector Machine (SVM)
Ensemble Learning

samples: 8583
train: 6008
validation: 1288
Test: 1287

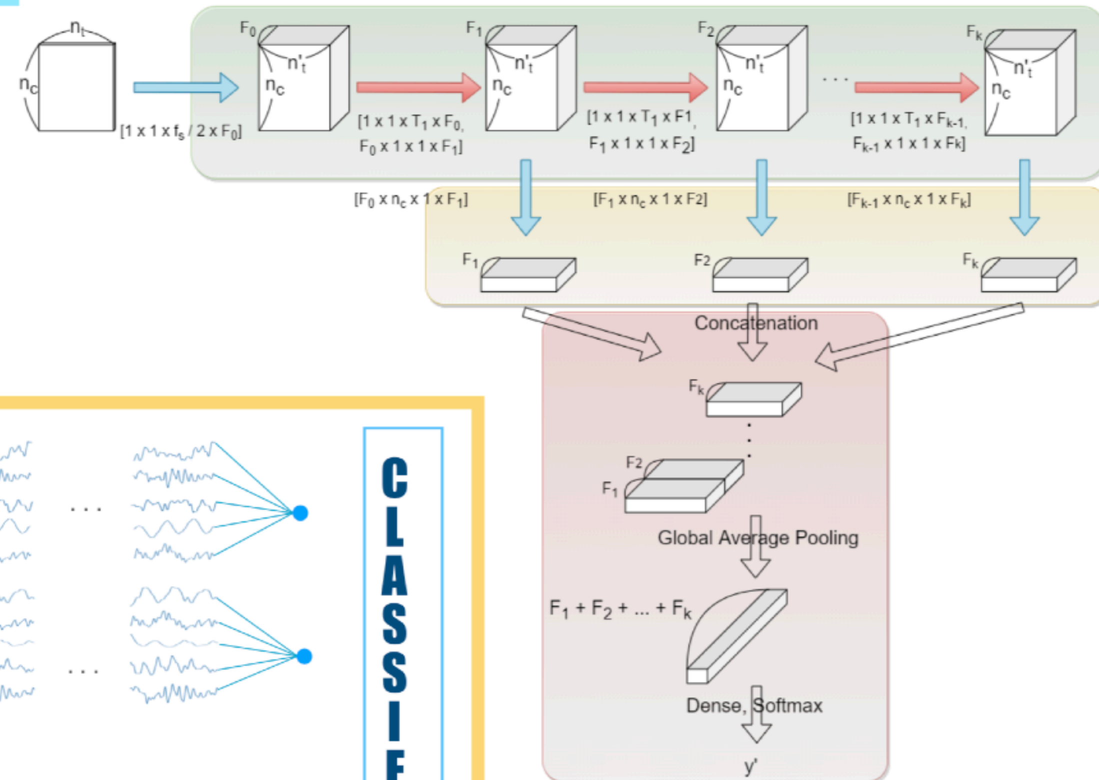
Bahaadini et al. (2018)

**DL Models:** vanilla and MSNN model are implemented using `PyTorch`

# Our work

## Multi-Scale Neural Net (MSNN) [6]

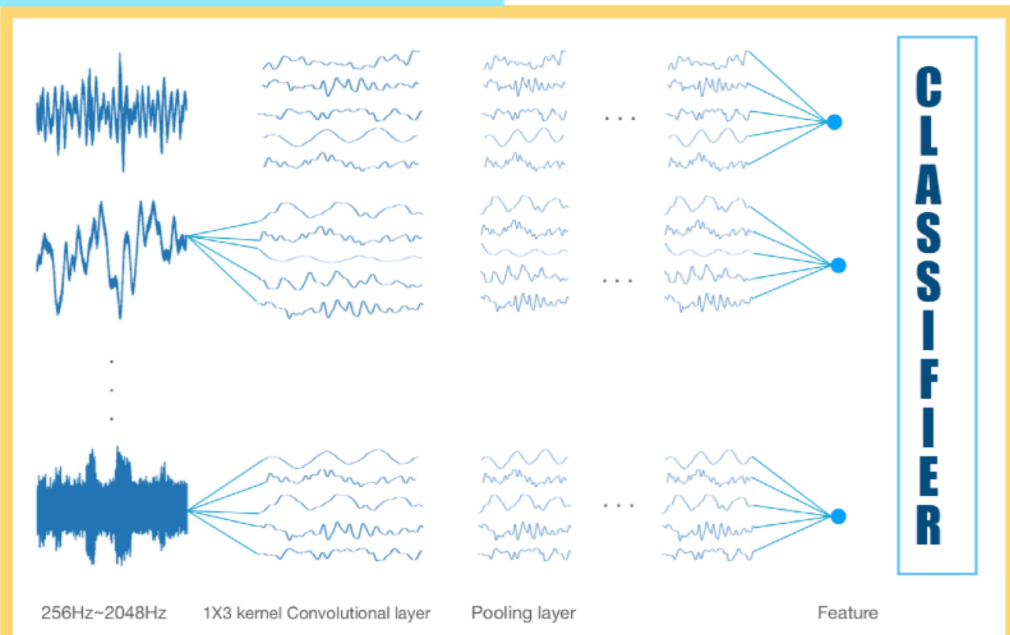
- Seperable Convolution
- 2d Convolution
- Spectral-Temporal Feature Representation Block
- Spatial Feature Representation Block
- Classification Block



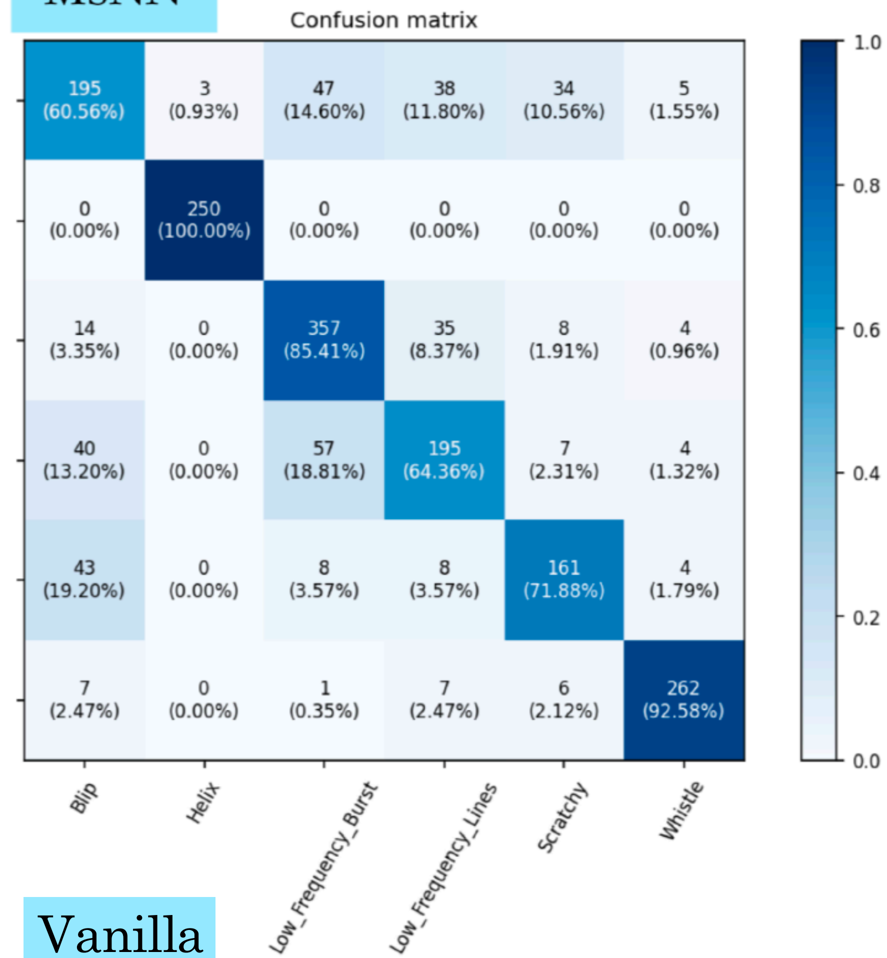

**MSNN Layer Structure**

Input: 1 x 290 x 8192
1st 2Dconv: 1 x 1025 x 50
2nd SepConv: 1 x 3 x 10
3nd SepConv: 1 x 3 x 10

## Vanilla model (multi-input CNN)




## MSNN



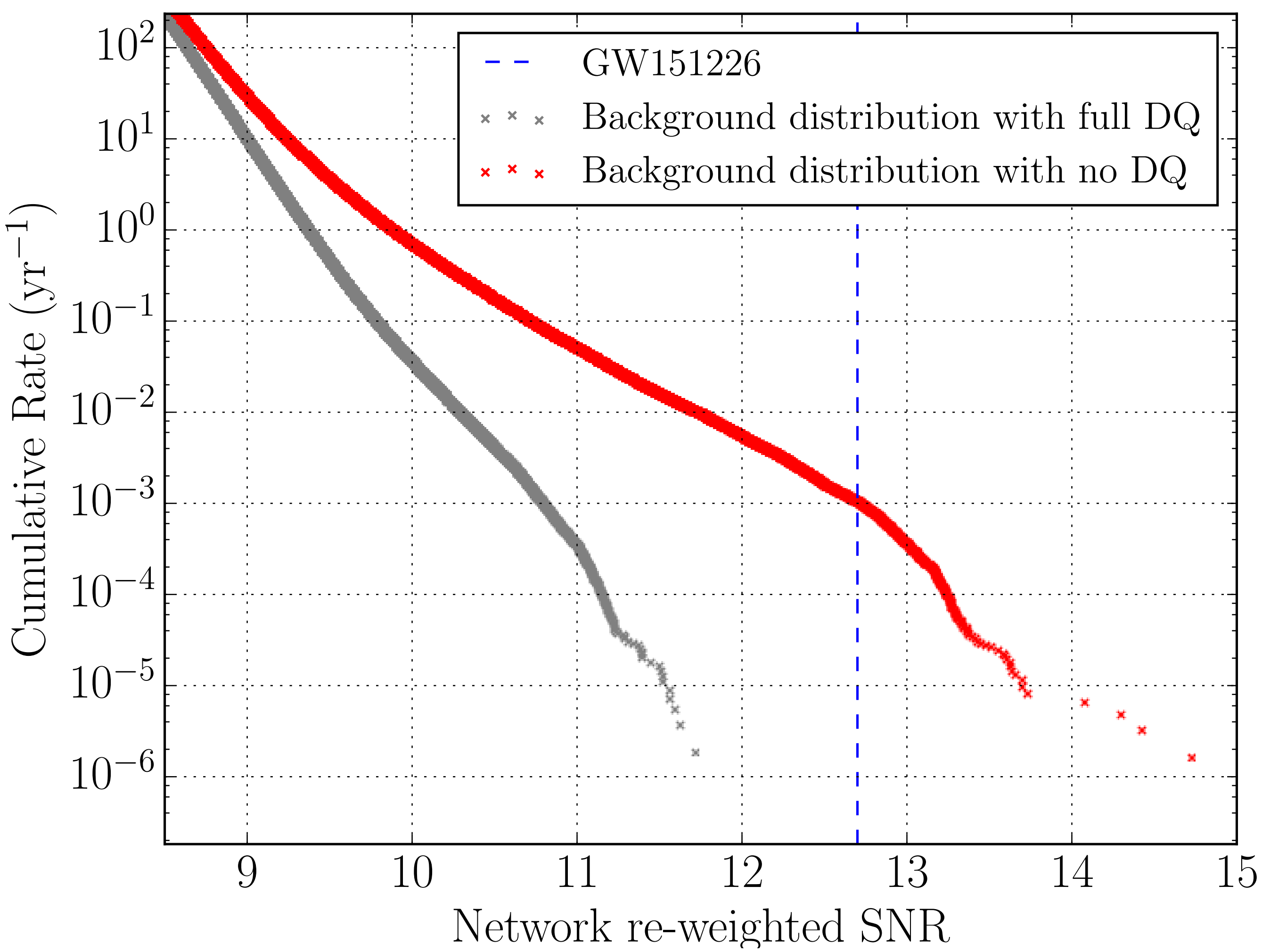

## Vanilla


## Results and future work

- 10-fold cross validation accuracy
  - Vanilla model: 70.0%
  - MSNN model: 78.89%
- **Future work**: Layer-wise relevance propagation will be implemented in the MSNN model to identify the channels responsible for glitches

W. Kim et al., in preparation

# Data Quality Impact on GW searches




LIGO-Virgo collaboration (2017) - arXiv 1710.02185

The false alarm rate of GW151226 **improves by a factor of 567**, from 1 in 320 years to 1 in 183000 years, **with interferometer data quality information!**

# Data Quality Information

**DATA (Data Available)**: Failing this level indicates that LIGO/Virgo data are not publicly available because the instruments or data calibration were not operating in an acceptable condition.

**CAT1 (Category 1)**: Failing a data quality check at this category indicates **a critical issue with a key detector component not operating in its nominal configuration**.

- These times are identical for each data analysis group.
- *Times that fail CAT1 flags are not available.*

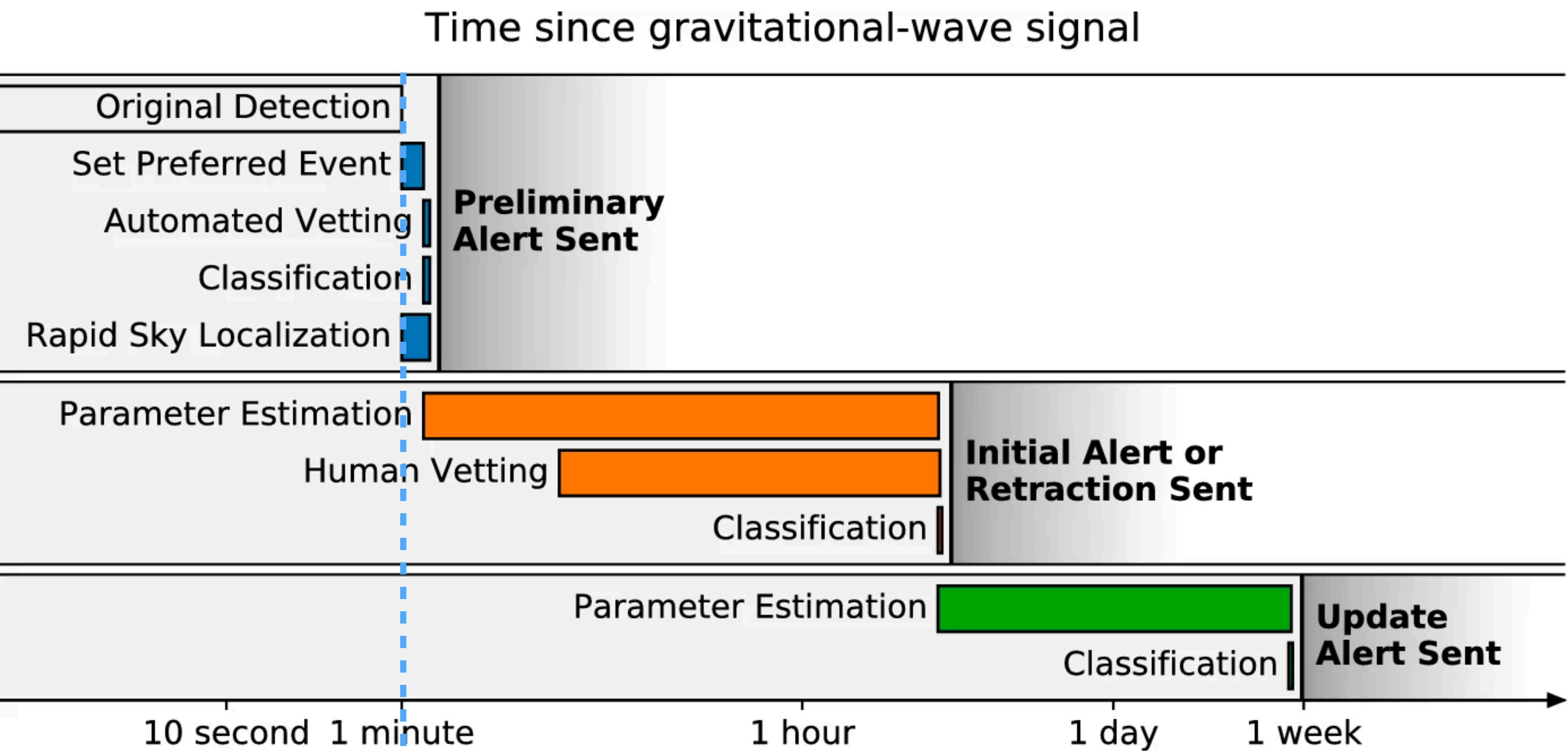
**CAT2 (Category 2)**: Failing a data quality check at this category indicates times when there is a **known, understood physical coupling to the gravitational wave channel**. For example, high seismic activity.

**CAT3 (Category 3)**:
- Burst: Failing a data quality check at this category indicates times when there is **statistical coupling to the gravitational wave channel** which is not fully understood.
- CBC: Category not used

Data quality levels are defined in a cumulative way: a time which fails a given category automatically fails all higher categories.

Data quality categories are defined independently for different analysis groups: if something fails at CAT2_BURST, it could pass CAT2_CBC.

Living Reviews in Relativity (2020) 23:3
https://doi.org/10.1007/s41114-020-00026-9




Low-latency DQ

GraceDB : https://gracedb.ligo.org/

# Bayesian Inference

Bayes' theorem

The likelihood could be the function of errors

**Posterior** $$p(\theta|d) = \frac{p(d|\theta)}{p(d)} \cdot p(\theta)$$

$$= \frac{p(d|\theta)}{\int p(d|\theta)p(\theta)d\theta} \cdot p(\theta)$$

Prior choices can influence results

The evidence is unimportant for parameter estimation (but not model selection !)

$$p(\theta|d) \sim p(d|\theta)p(\theta)$$

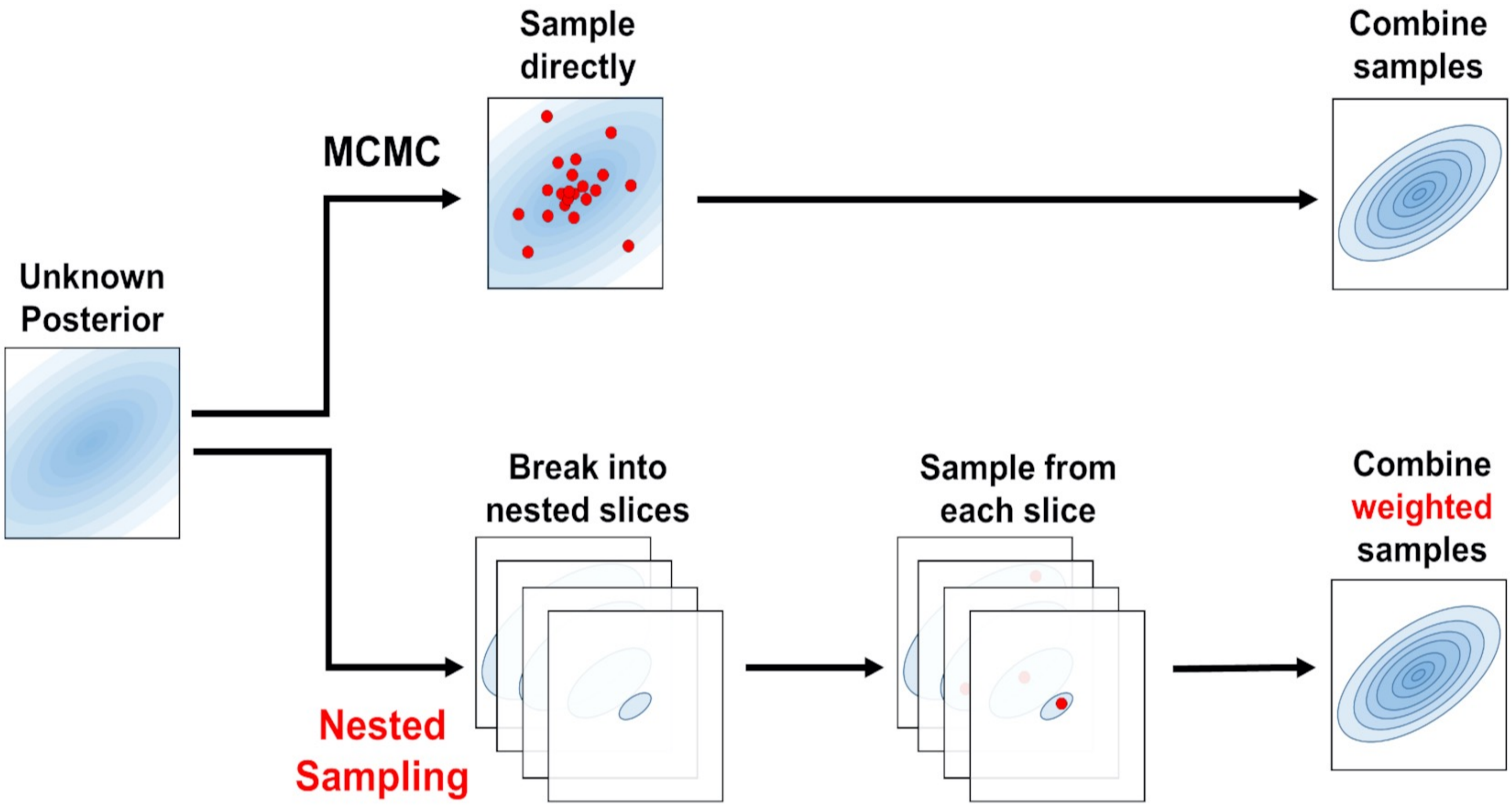
Prior, p(θ) : the distribution of the parameter(s) before any data is observed

Likelihood, p(d|θ) : the distribution of the observed data conditional on its parameters

Posterior, p(θ|d): the distribution of the parameter(s) after taking into account the observed data

Model evidence, p(d):the distribution of the observed data marginalized over the parameter(s)

**Figure 1.** A schematic representation of the different approaches MCMC methods and nested sampling methods take to ...




*Mon Not R Astron Soc*, Volume 493, Issue 3, April 2020, Pages 3132–3158, https://doi.org/10.1093/mnras/staa278

# Metroplis algorithm

Let $f(x)$ be a function that is proportional to the desired probability density function $P(x)$ (a.k.a. a target distribution)[a].

1. Initialization: Choose an arbitrary point $x_t$ to be the first observation in the sample and choose an arbitrary probability density $g(x \mid y)$ (sometimes written $Q(x \mid y)$) that suggests a candidate for the next sample value $x$, given the previous sample value $y$. In this section, $g$ is assumed to be symmetric; in other words, it must satisfy $g(x \mid y) = g(y \mid x)$. A usual choice is to let $g(x \mid y)$ be a Gaussian distribution centered at $y$, so that points closer to $y$ are more likely to be visited next, making the sequence of samples into a random walk[b]. The function $g$ is referred to as the *proposal density* or *jumping distribution*.
2. For each iteration *t*:
   - *Generate* a candidate $x'$ for the next sample by picking from the distribution $g(x' \mid x_t)$.
   - *Calculate* the *acceptance ratio* $\alpha = f(x')/f(x_t)$, which will be used to decide whether to accept or reject the candidate[c]. Because *f* is proportional to the density of *P*, we have that $\alpha = f(x')/f(x_t) = P(x')/P(x_t)$.
   - *Accept or reject*:
     - Generate a uniform random number $u \in [0, 1]$.
     - If $u \le \alpha$, then *accept* the candidate by setting $x_{t+1} = x'$,
     - If $u > \alpha$, then *reject* the candidate and set $x_{t+1} = x_t$ instead.

Wikipedia/Metropolis–Hastings_algorithm

# Metropolis-Hastings alogorithm

1. Initialise
   1. Pick an initial state $x_0$.
   2. Set $t = 0$.
2. Iterate
   1. *Generate* a random candidate state $x'$ according to $g(x' \mid x_t)$.
   2. *Calculate* the acceptance probability $A(x', x_t) = \min\left(1, \frac{P(x')}{P(x_t)} \frac{g(x_t \mid x')}{g(x' \mid x_t)}\right)$.
   3. *Accept or reject*:
      1. generate a uniform random number $u \in [0, 1]$;
      2. if $u \le A(x', x_t)$, then *accept* the new state and set $x_{t+1} = x'$;
      3. if $u > A(x', x_t)$, then *reject* the new state, and copy the old state forward $x_{t+1} = x_t$.
   4. *Increment*: set $t = t + 1$.

Wikipedia/Metropolis–Hastings_algorithm

# Example - MCMC

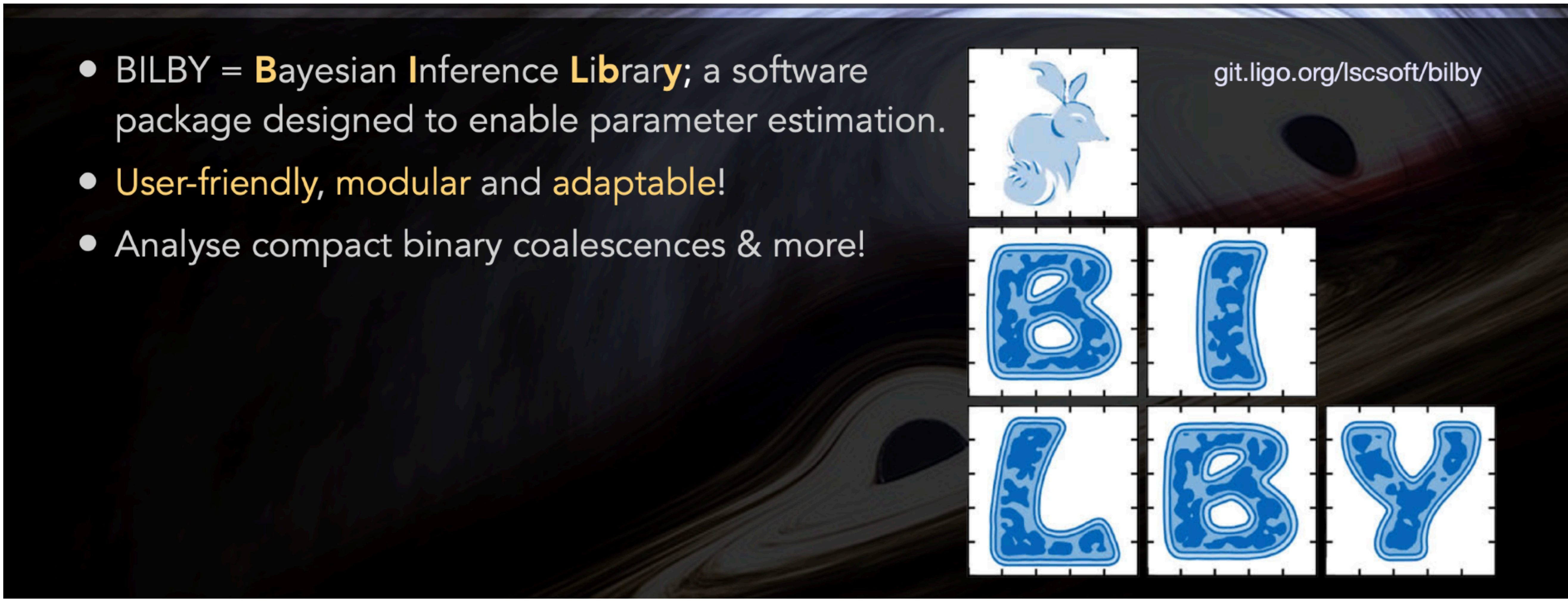
https://github.com/gw-odw/odw-2018/blob/master/parameter_estimation/IntroToMCMC.ipynb

# Nested Sampling

```
Start with N points θ_1,...,θ_N sampled from prior.
for i = 1 to j do        % The number of iterations j is chosen by guesswork.
    L_i := min(current likelihood values of the points);
    X_i := exp(-i/N);
    w_i := X_{i-1} - X_i
    Z := Z + L_i · w_i;
    Save the point with least likelihood as a sample point with weight w_i.
    Update the point with least likelihood with some Markov chain Monte Carlo steps
according to the prior, accepting only steps that
    keep the likelihood above L_i.
end
return Z;
```

wikipedia/Nested_sampling_algorithm

# Bilby




In S.Galaudage' slides, GW ODW #4, 2021

# Gravitational-wave likelihood

The residual between the data and best-match waveform template should also follow a unit Gaussian about the square root of the PSD when there is a signal.

$$\mathcal{L}\left(d_i|\theta\right) = \frac{1}{2\pi P_i}\exp\left(-2\Delta f\frac{\left|d_i - h_i(\theta)\right|^2}{P_i}\right)$$

PSD

frequency resolution

$$\mathcal{L}\left(d|\theta\right) = \prod_i^N \mathcal{L}\left(d_i|\theta\right)$$

In S.Galaudage' slides, GW ODW #4, 2021

# Model Selection

- Calculating the evidence for the signal & noise allows you to calculate a Bayes factor.

$$\mathrm{BF}_N^S = \frac{\mathcal{Z}_S}{\mathcal{Z}_N}$$

- You can do the same thing on a population level with different models. Having a $BF \gtrsim 3000$ or $\ln BF \gtrsim 8$ is considered significant.
- Allows you to do model selection and determine which models best fit your data.

In S.Galaudage' slides, GW ODW #4, 2021

# PE w/ Bilby - data

```
import bilby
from bilby.core.prior import Uniform
from bilby.gw.conversion import
convert_to_lal_binary_black_hole_parameters,
generate_all_bbh_parameters

from gwpy.timeseries import TimeSeries
```

* Download data

```
# Definite times in relation to the trigger time (time_of_event), duration and post_trigger_duratio
post_trigger_duration = 2
duration = 4
analysis_start = time_of_event + post_trigger_duration - duration

# Use gwpy to fetch the open data
H1_analysis_data = TimeSeries.fetch_open_data(
    "H1", analysis_start, analysis_start + duration, sample_rate=4096, cache=True)

L1_analysis_data = TimeSeries.fetch_open_data(
    "L1", analysis_start, analysis_start + duration, sample_rate=4096, cache=True)
```

# PE w/ Bilby - data

* Set up empty interferometers

```
H1 = bilby.gw.detector.get_empty_interferometer("H1")
L1 = bilby.gw.detector.get_empty_interferometer("L1")
```

```
H1.set_strain_data_from_gwpy_timeseries(H1_analysis_data)
L1.set_strain_data_from_gwpy_timeseries(L1_analysis_data)
```

```
* Download the PSD data

psd_duration = duration * 32
psd_start_time = analysis_start - psd_duration

H1_psd_data = TimeSeries.fetch_open_data(
    "H1", psd_start_time, psd_start_time + psd_duration, sample_rate=4096, cache=True)

L1_psd_data = TimeSeries.fetch_open_data(
    "L1", psd_start_time, psd_start_time + psd_duration, sample_rate=4096, cache=True)
```

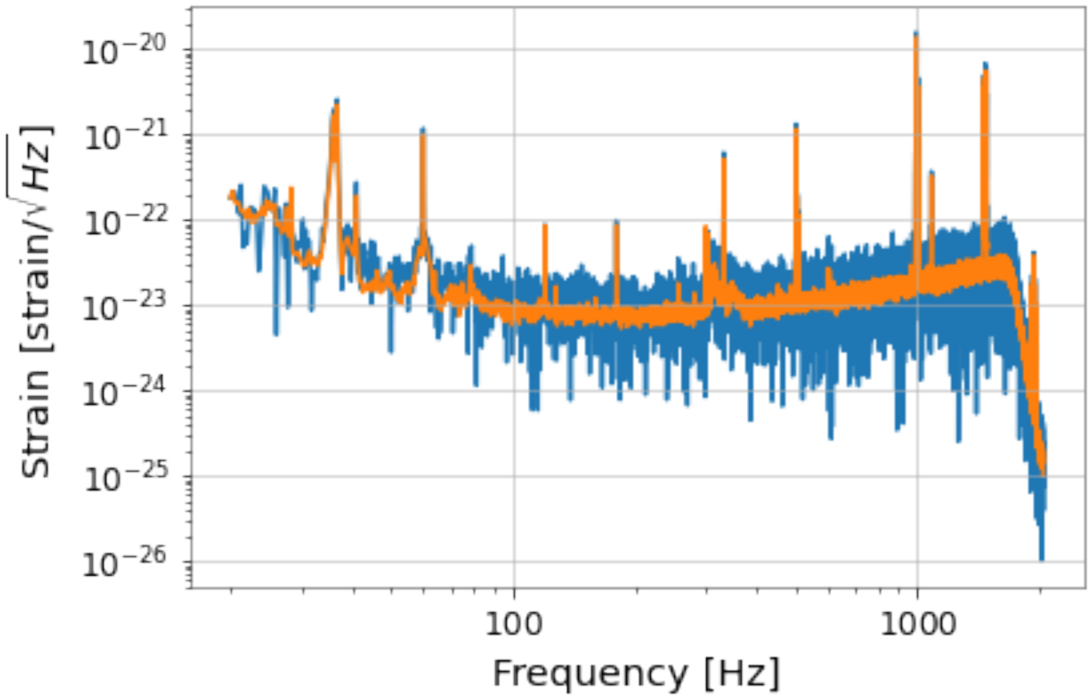
```
psd_alpha = 2 * H1.strain_data.roll_off / duration
H1_psd = H1_psd_data.psd(fftlength=duration, overlap=0, window=("tukey", psd_alpha), method="median")
L1_psd = L1_psd_data.psd(fftlength=duration, overlap=0, window=("tukey", psd_alpha), method="median")
```

# PE w/ Bilby - data

* Initialise the PSD

```
H1.power_spectral_density = bilby.gw.detector.PowerSpectralDensity(
    frequency_array=H1_psd.frequencies.value, psd_array=H1_psd.value)
L1.power_spectral_density = bilby.gw.detector.PowerSpectralDensity(
    frequency_array=H1_psd.frequencies.value, psd_array=L1_psd.value)
```

```
fig, ax = plt.subplots()
idxs = H1.strain_data.frequency_mask  # This is a boolean
mask of the frequencies which we'll use in the analysis
ax.loglog(H1.strain_data.frequency_array[idxs],

np.abs(H1.strain_data.frequency_domain_strain[idxs]))
ax.loglog(H1.power_spectral_density.frequency_array[idxs],
          H1.power_spectral_density.asd_array[idxs])
ax.set_xlabel("Frequency [Hz]")
ax.set_ylabel("Strain [strain/$\sqrt{Hz}$]")
plt.show()
```



# PE w/ Bilby - priors

```
prior = bilby.core.prior.PriorDict()
prior['chirp_mass'] = Uniform(name='chirp_mass', minimum=30.0,maximum=32.5)
prior['mass_ratio'] = Uniform(name='mass_ratio', minimum=0.5, maximum=1)
prior['phase'] = Uniform(name="phase", minimum=0, maximum=2*np.pi)
prior['geocent_time'] = Uniform(name="geocent_time", minimum=time_of_event-0.1,
maximum=time_of_event+0.1)
prior['a_1'] =  0.0
prior['a_2'] =  0.0
prior['tilt_1'] =  0.0
prior['tilt_2'] =  0.0
prior['phi_12'] =  0.0
prior['phi_jl'] =  0.0
prior['dec'] =  -1.2232
prior['ra'] =  2.19432
prior['theta_jn'] =  1.89694
prior['psi'] =  0.532268
prior['luminosity_distance'] = 412.066
```

Bayes' theorem

The likelihood could be the function of errors

**Posterior**

$$p(\theta|d) = \frac{p(d|\theta)}{p(d)} \cdot p(\theta)$$

$$= \frac{p(d|\theta)}{\int p(d|\theta)p(\theta)d\theta} \cdot p(\theta)$$

Prior choices can influence results

$$p(\theta|d) \sim p(d|\theta)p(\theta)$$

The evidence is unimportant for parameter estimation (but not model selection !)

# PE w/ Bilby - Likelihood

```
# First, put our "data" created above into a list of intererometers (the
order is arbitrary)
interferometers = [H1, L1]

# Next create a dictionary of arguments which we pass into the
LALSimulation waveform - we specify the waveform approximant here
waveform_arguments = dict(
    waveform_approximant='IMRPhenomPv2', reference_frequency=100.,
catch_waveform_errors=True)

# Next, create a waveform_generator object. This wraps up some of the jobs
of converting between parameters etc
waveform_generator = bilby.gw.WaveformGenerator(
    frequency_domain_source_model=bilby.gw.source.lal_binary_black_hole,
    waveform_arguments=waveform_arguments,
    parameter_conversion=convert_to_lal_binary_black_hole_parameters)

# Finally, create our likelihood, passing in what is needed to get going
likelihood = bilby.gw.likelihood.GravitationalWaveTransient(
    interferometers, waveform_generator, priors=prior,
    time_marginalization=True, phase_marginalization=True,
distance_marginalization=False)
```

# PE w/ Bilby - run

```
result_short = bilby.run_sampler(
    likelihood, prior, sampler='dynesty', outdir='short', label="GW150914",
    conversion_function=bilby.gw.conversion.generate_all_bbh_parameters,
    sample="unif", nlive=500, dlogz=3)
    # Arguments are used to make things fast - not recommended for general use
```

dynesty: https://arxiv.org/abs/1904.02180
https://dynesty.readthedocs.io/en/latest/dynamic.html

Samplers:
Nested Sampling: dynesty, nestle, cpnest
MCMC : bilby_mcmc, emcee, ptemcee, pymc3

# PE w/ Bilby - run

```
04:51 bilby INFO    : Running for label 'GW150914', output will be saved to 'short'
04:51 bilby INFO    : Using lal version 7.1.2
04:51 bilby INFO    : Using lal git version Branch: None;Tag: lalsuite-v6.82;Id: cf792129c2473f42ce6c6ee21d8234254cefd337;;Builder: Unknown User <>;Repository status: UNCLEAN: Modified
working tree
04:51 bilby INFO    : Using lalsimulation version 2.5.1
04:51 bilby INFO    : Using lalsimulation git version Branch: None;Tag: lalsuite-v6.82;Id: cf792129c2473f42ce6c6ee21d8234254cefd337;;Builder: Unknown User <>;Repository status: UNCLEAN:
Modified working tree
04:51 bilby INFO    : Search parameters:
04:51 bilby INFO    :   chirp_mass = Uniform(minimum=30.0, maximum=32.5, name='chirp_mass', latex_label='$\\mathcal{M}$', unit=None, boundary=None)
04:51 bilby INFO    :   mass_ratio = Uniform(minimum=0.5, maximum=1, name='mass_ratio', latex_label='$q$', unit=None, boundary=None)
04:51 bilby INFO    :   time_jitter = Uniform(minimum=-0.000244140625, maximum=0.000244140625, name=None, latex_label=None, unit=None, boundary='periodic')
04:51 bilby INFO    :   phase = 0.0
04:51 bilby INFO    :   geocent_time = 1126259460.3999023
04:51 bilby INFO    :   a_1 = 0.0
04:51 bilby INFO    :   a_2 = 0.0
04:51 bilby INFO    :   tilt_1 = 0.0
04:51 bilby INFO    :   tilt_2 = 0.0
04:51 bilby INFO    :   phi_12 = 0.0
04:51 bilby INFO    :   phi_jl = 0.0
04:51 bilby INFO    :   dec = -1.2232
04:51 bilby INFO    :   ra = 2.19432
04:51 bilby INFO    :   theta_jn = 1.89694
04:51 bilby INFO    :   psi = 0.532268
04:51 bilby INFO    :   luminosity_distance = 412.066
04:51 bilby INFO    : Generating frequency domain strain from given time domain strain.
04:51 bilby INFO    : Applying a tukey window with alpha=0.1, roll off=0.2
04:51 bilby INFO    : Single likelihood evaluation took 1.482e-02 s
0it [00:00, ?it/s]04:51 bilby INFO    : Using sampler Dynesty with kwargs {'bound': 'multi', 'sample': 'unif', 'verbose': True, 'periodic': None, 'reflective': None,
'check_point_delta_t': 600, 'nlive': 500, 'first_update': None, 'walks': 100, 'npdim': None, 'rstate': None, 'queue_size': 1, 'pool': None, 'use_pool': None, 'live_points': None,
'logl_args': None, 'logl_kwargs': None, 'ptform_args': None, 'ptform_kwargs': None, 'enlarge': 1.5, 'bootstrap': None, 'vol_dec': 0.5, 'vol_check': 8.0, 'facc': 0.2, 'slices': 5,
'update_interval': 300, 'print_func': <bound method Dynesty._print_func of <bilby.core.sampler.dynesty.Dynesty object at 0x7fe0fd1bbed0>>, 'dlogz': 3, 'maxiter': None, 'maxcall': None,
'logl_max': inf, 'add_live': True, 'print_progress': True, 'save_bounds': False, 'n_effective': None, 'maxmcmc': 5000, 'nact': 5}
04:51 bilby INFO    : Checkpoint every check_point_delta_t = 600s
04:51 bilby INFO    : Using dynesty version 1.1
04:51 bilby INFO    : Resume file short/GW150914_resume.pickle does not exist.
04:51 bilby INFO    : Generating initial points from the prior
980it [00:51,  6.17it/s, bound:0 nc: 15 ncall:3.3e+03 eff:30.0% logz-ratio=266.09+/-0.08 dlogz:3.012>3]04:52 bilby INFO    : Written checkpoint file short/GW150914_resume.pickle
04:52 bilby INFO    : Writing 190 current samples to short/GW150914_samples.dat
982it [00:53, 18.23it/s, bound:0 nc:  1 ncall:3.3e+03 eff:45.3% logz-ratio=268.04+/-0.13 dlogz:0.005>3]04:52 bilby INFO    : Sampling time: 0:00:41.658335
04:52 bilby INFO    : Reconstructing marginalised parameters.

100%|██████████| 1482/1482 [00:46<00:00, 31.90it/s]04:53 bilby INFO    : Generating sky frame parameters.
100%|██████████| 1482/1482 [00:00<00:00, 2545.58it/s]
04:53 bilby INFO    : Computing SNRs for every sample.
100%|██████████| 1482/1482 [00:21<00:00, 70.41it/s]
04:54 bilby INFO    : Summary of results:
nsamples: 1482
ln_noise_evidence: -8534.562
ln_evidence: -8266.517 +/-  0.128
ln_bayes_factor: 268.045 +/-  0.128
```

# PE w/ Bilby - results

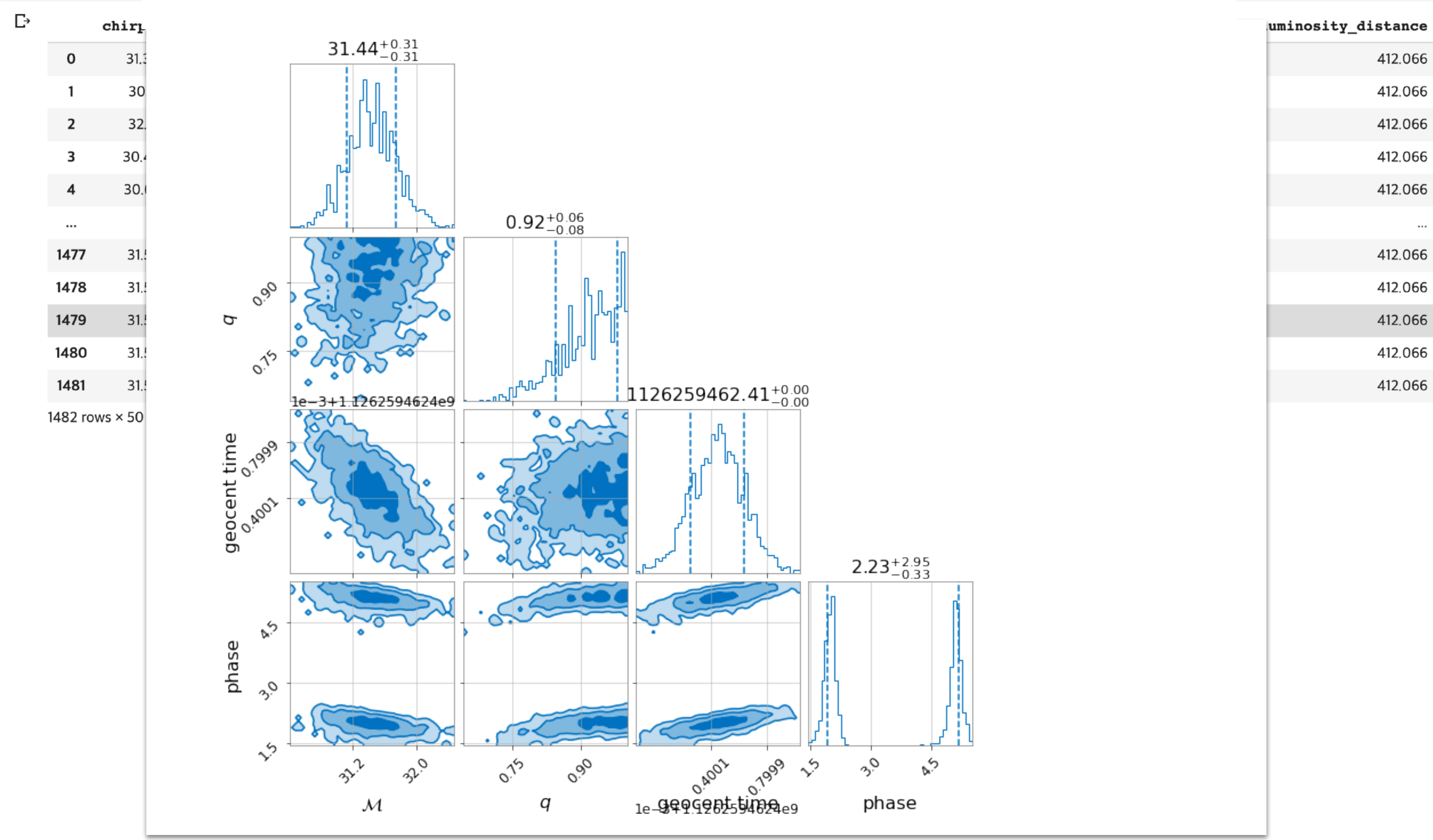
```
1 result_short.posterior
```

| | chirp_mass | mass_ratio | time_jitter | phase | geocent_time | a_1 | a_2 | tilt_1 | tilt_2 | phi_12 | phi_jl | dec | ra | theta_jn | psi | luminosity_distance |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 31.304732 | 0.642621 | 0.000201 | 4.248570 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **1** | 30.431251 | 0.865419 | -0.000241 | 5.339548 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **2** | 32.351273 | 0.823104 | 0.000208 | 1.453658 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **3** | 30.489016 | 0.749595 | 0.000115 | 1.913821 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **4** | 30.647587 | 0.683362 | -0.000116 | 4.763118 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **1477** | 31.527348 | 0.986097 | -0.000080 | 5.085407 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **1478** | 31.527348 | 0.986097 | -0.000080 | 2.036470 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **1479** | 31.527348 | 0.986097 | -0.000080 | 1.935416 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **1480** | 31.527348 | 0.986097 | -0.000080 | 2.048065 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |
| **1481** | 31.527348 | 0.986097 | -0.000080 | 2.041310 | 1.126259e+09 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | 0.0 | -1.2232 | 2.19432 | 1.89694 | 0.532268 | 412.066 |

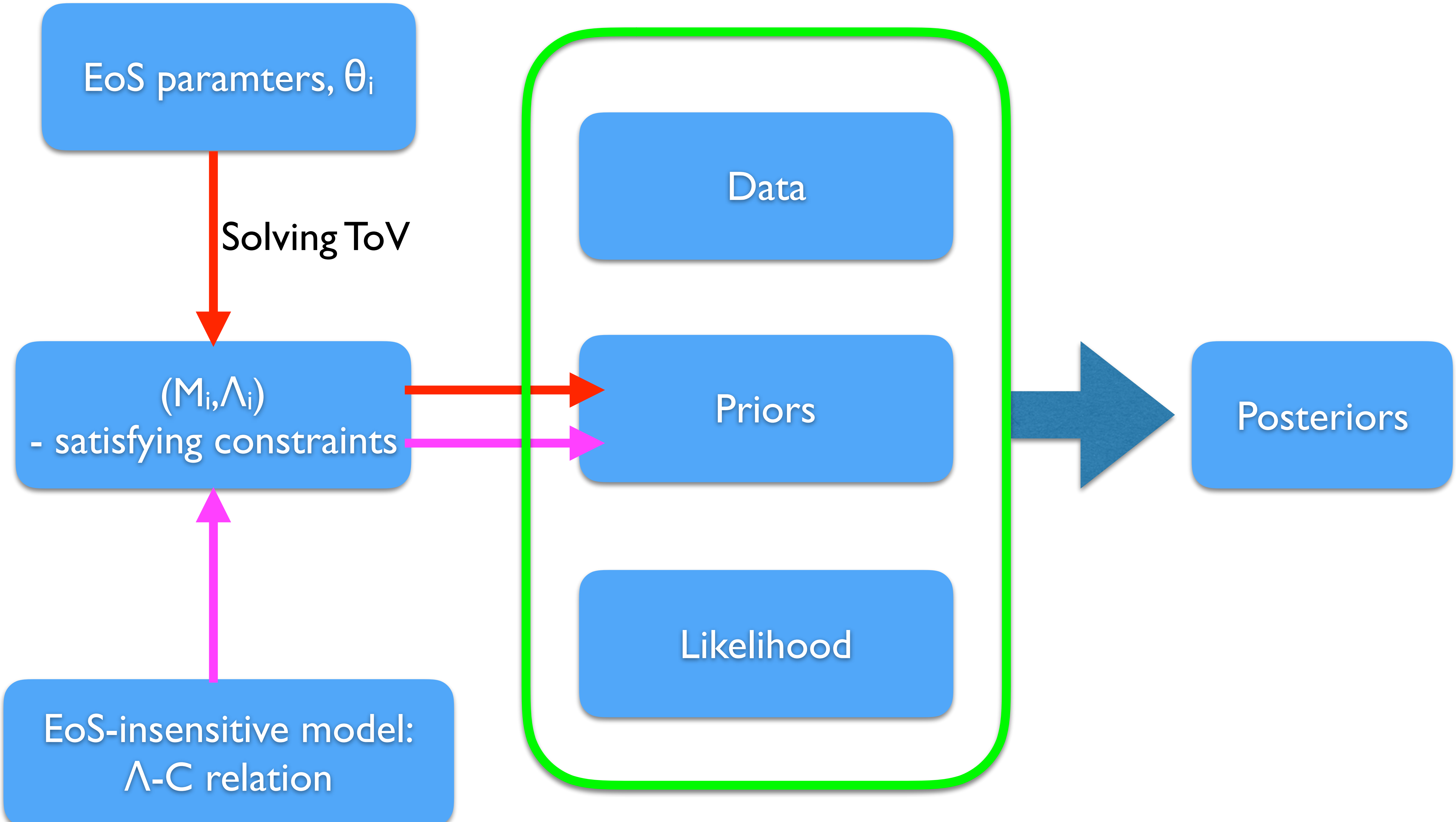
1482 rows × 50 columns

# PE w/ Bilby - results

```
result_short.plot_corner(parameters=["chirp_mass", "mass_ratio", "geocent_time", "phase"], prior=True)
```



# Bilby - schematic procedure w/ NS EoS



# PE w/ Bilby - BNS

```
priors = bilby.gw.prior.BNSPriorDict()
for key in ['psi', 'geocent_time', 'ra', 'dec', 'chi_1', 'chi_2',
            'theta_jn', 'luminosity_distance', 'phase']:
    priors[key] = injection_parameters[key]
priors.pop('mass_1')
priors.pop('mass_2')
priors.pop('lambda_1')
priors.pop('lambda_2')
priors.pop('mass_ratio')
priors['chirp_mass'] = bilby.core.prior.Gaussian(1.215, 0.1, name='chirp_mass',
unit='$M_{\\odot}$')
priors['symmetric_mass_ratio'] = bilby.core.prior.Uniform(0.1, 0.25,
name='symmetric_mass_ratio')
priors['eos_spectral_gamma_0'] = bilby.core.prior.Uniform(0.2, 2.0, name='gamma0',
latex_label='$\\gamma_0')
priors['eos_spectral_gamma_1'] = bilby.core.prior.Uniform(-1.6, 1.7, name='gamma1',
latex_label='$\\gamma_1')
priors['eos_spectral_gamma_2'] = bilby.core.prior.Uniform(-0.6, 0.6, name='gamma2',
latex_label='$\\gamma_2')
priors['eos_spectral_gamma_3'] = bilby.core.prior.Uniform(-0.02, 0.02,
name='gamma3', latex_label=‘$\\gamma_3’)

priors['eos_check'] = bilby.gw.prior.EOSCheck()
```

# PE w/ Bilby - BNS

```
# Initialise the likelihood by passing in the interferometer data (IFOs)
# and the waveform generator
likelihood = bilby.gw.GravitationalWaveTransient(
    interferometers=interferometers, waveform_generator=waveform_generator,
    time_marginalization=False, phase_marginalization=False,
    distance_marginalization=False, priors=priors)

# Run sampler.  In this case we're going to use the `dynesty` sampler
result = bilby.run_sampler(
    likelihood=likelihood, priors=priors, sampler='dynesty', npoints=1000,
    injection_parameters=injection_parameters, outdir=outdir, label=label,
    conversion_function=bilby.gw.conversion.generate_all_bns_parameters,
    resume=True)
```

# Example - Bilby

```python
#!/usr/bin/env python3
"""
An example of how to use bilby to perform parameter estimation for
non-gravitational wave data consisting of a Gaussian with a mean and variance
"""
import bilby
import numpy as np

# A few simple setup steps
label = 'gaussian_example'
outdir = 'outdir'

# Here is minimum requirement for a Likelihood class to run with bilby. In this
# case, we setup a GaussianLikelihood, which needs to have a log_likelihood
# method. Note, in this case we will NOT make use of the `bilby`
# waveform_generator to make the signal.

# Making simulated data: in this case, we consider just a Gaussian

data = np.random.normal(3, 4, 100)


class SimpleGaussianLikelihood(bilby.Likelihood):
    def __init__(self, data):
        """
        A very simple Gaussian likelihood

        Parameters
        ----------
        data: array_like
            The data to analyse
        """
        super().__init__(parameters={'mu': None, 'sigma': None})
        self.data = data
        self.N = len(data)
```

```python
    def log_likelihood(self):
        mu = self.parameters['mu']
        sigma = self.parameters['sigma']
        res = self.data - mu
        return -0.5 * (np.sum((res / sigma)**2) +
                       self.N * np.log(2 * np.pi * sigma**2))


likelihood = SimpleGaussianLikelihood(data)
priors = dict(mu=bilby.core.prior.Uniform(0, 5, 'mu'),
              sigma=bilby.core.prior.Uniform(0, 10, 'sigma'))

# And run sampler
result = bilby.run_sampler(
    likelihood=likelihood, priors=priors, sampler='dnest4', npoints=50000,
    walks=100, outdir=outdir, label=label)
result.plot_corner()
```

- Data : $N(3,4)$



이형원교수님
지난 여름학교 강의중에서

https://git.ligo.org/lscsoft/bilby/blob/master/examples/core_examples/gaussian_example.py

# Posterior samples from PE

```python
import h5py
import pandas as pd
import corner
```

```python
label = 'GW150914'

# if you do not have wget installed, simply download manually
# https://dcc.ligo.org/LIGO-P1800370/public/GW150914_GWTC-1.hdf5
# from your browser
! wget https://dcc.ligo.org/LIGO-P1800370/public/{label}_GWTC-1.hdf5

posterior_file = './'+label+'_GWTC-1.hdf5'
posterior = h5py.File(posterior_file, 'r')
```

```python
print('This file contains four datasets: ',posterior.keys())
```

```
This file contains four datasets:  <KeysViewHDF5
['IMRPhenomPv2_posterior', 'Overall_posterior', 'SEOBNRv3_posterior',
‘prior']>
```

# Posterior samples from PE

```
print(posterior['Overall_posterior'].dtype.names)
('costheta_jn', 'luminosity_distance_Mpc', 'right_ascension', 'declination',
'm1_detector_frame_Msun', 'm2_detector_frame_Msun', 'spin1', 'spin2', 'costilt1',
'costilt2')
```

- `luminosity_distance_Mpc`: luminosity distance [Mpc]
- `m1_detector_frame_Msun`: primary (larger) black hole mass (detector frame) [solar mass]
- `m2_detector_frame_Msun`: secondary (smaller) black hole mass (detector frame) [solar mass]
- `right_ascension`, `declination`: right ascension and declination of the source [rad].
- `costheta_jn`: cosine of the angle between line of sight and total angular momentum vector of system.
- `spin1`, `costilt1`: primary (larger) black hole spin magnitude (dimensionless) and cosine of the zenith angle between the spin and the orbital angular momentum vector of system.
- `spin2`, `costilt2`: secondary (smaller) black hole spin magnitude (dimensionless) and cosine of the zenith angle between the spin and the orbital angular momentum vector of system.

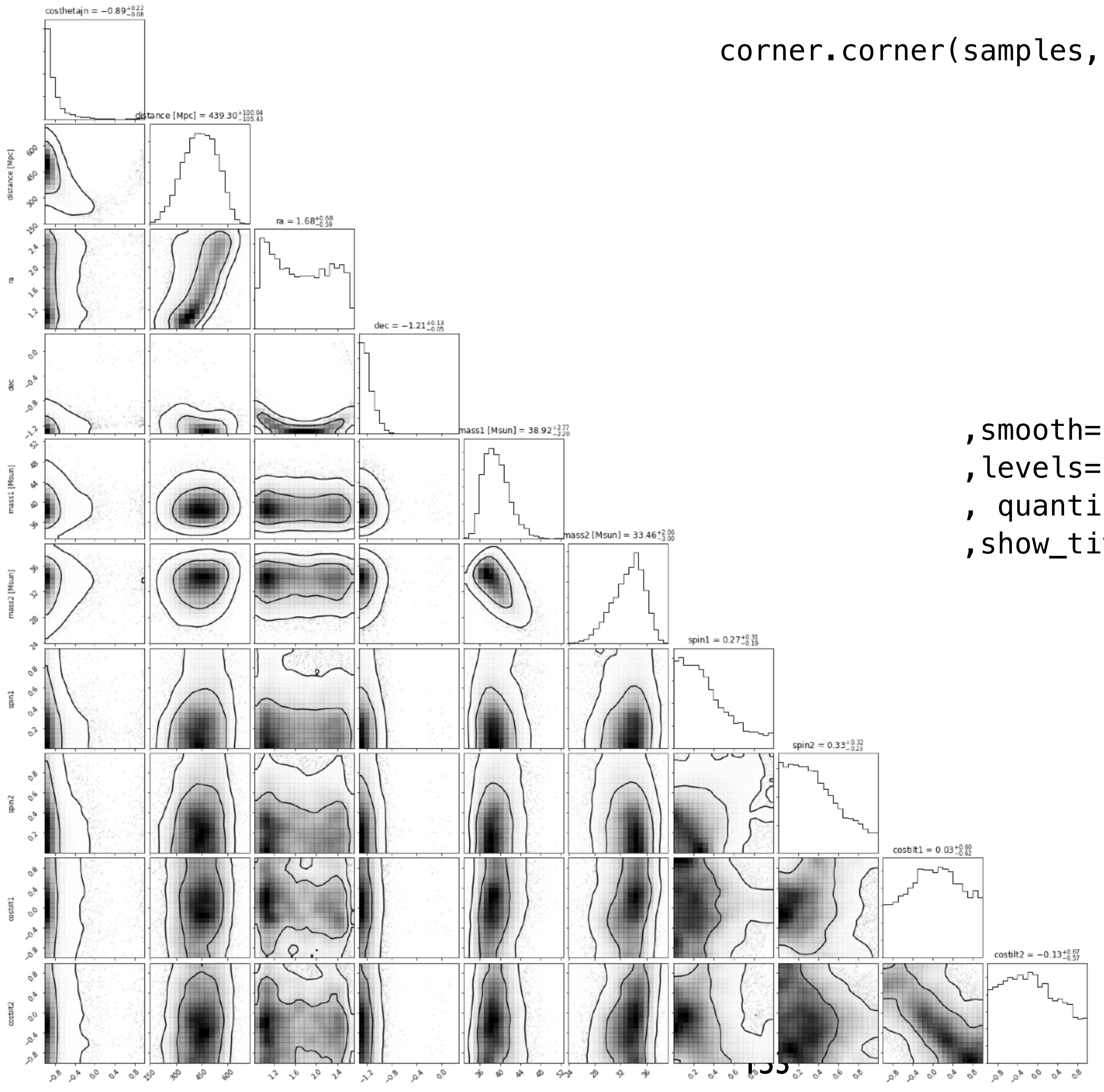
# Posterior samples from PE

```
samples=pd.DataFrame.from_records(np.array(posterior['Overall_posterior']))
```

| | costheta_jn | luminosity_distance_Mpc | right_ascension | declination | m1_detector_frame_Msun | m2_detector_frame_Msun | spin1 | spin2 | costilt1 | costilt2 |
|---|---|---|---|---|---|---|---|---|---|---|
| **0** | -0.976633 | 517.176717 | 1.456176 | -1.257815 | 39.037380 | 37.044563 | 0.417147 | 0.867740 | -0.280624 | 0.403853 |
| **1** | -0.700404 | 401.626864 | 2.658802 | -0.874661 | 34.620096 | 34.184416 | 0.125709 | 0.260679 | -0.757349 | -0.312285 |
| **2** | -0.840752 | 369.579071 | 1.106548 | -1.136396 | 37.894343 | 33.970520 | 0.581047 | 0.926893 | 0.649781 | -0.510843 |
| **3** | -0.583657 | 386.935268 | 2.077180 | -1.246351 | 36.412973 | 35.684463 | 0.235808 | 0.094391 | 0.116578 | -0.720505 |
| **4** | -0.928271 | 345.104345 | 0.993604 | -1.069243 | 39.477251 | 31.645008 | 0.511521 | 0.868009 | -0.438237 | 0.269333 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **8345** | -0.691637 | 306.985025 | 1.485646 | -1.269228 | 37.561962 | 33.355792 | 0.484003 | 0.627191 | 0.194507 | -0.408345 |
| **8346** | -0.834615 | 462.649414 | 2.065362 | -1.265618 | 37.824298 | 36.674075 | 0.589654 | 0.650758 | -0.737792 | 0.875384 |
| **8347** | -0.911463 | 448.930876 | 1.536913 | -1.257956 | 38.063291 | 35.757913 | 0.708407 | 0.714805 | 0.852085 | -0.797475 |
| **8348** | -0.856914 | 561.020036 | 2.367289 | -1.211824 | 44.884396 | 31.592433 | 0.389284 | 0.521304 | -0.251461 | 0.830526 |
| **8349** | -0.919556 | 519.641782 | 1.916675 | -1.250801 | 37.275183 | 35.445032 | 0.391824 | 0.516908 | -0.705305 | 0.600727 |

8350 rows × 10 columns

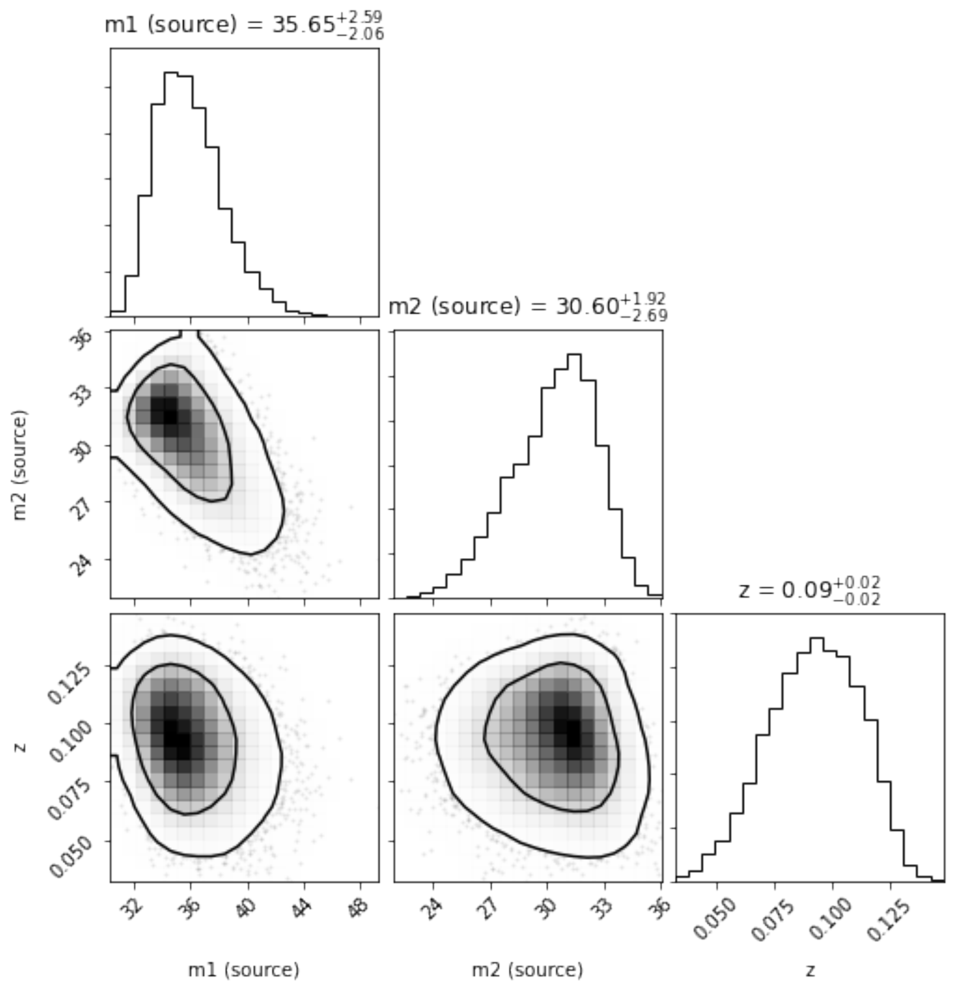
# Posterior samples from PE




```
corner.corner(samples,labels=['costhetajn',
                                 'distance [Mpc]',
                                 'ra',
                                 'dec',
                                 'mass1 [Msun]',
                                 'mass2 [Msun]',
                                 'spin1',
                                 'spin2',
                                 'costilt1',
                                 'costilt2']
              ,smooth=1.0
              ,levels=(0.68,0.95)
              , quantile=(0.16,0.84)
              ,show_titles=True);
```

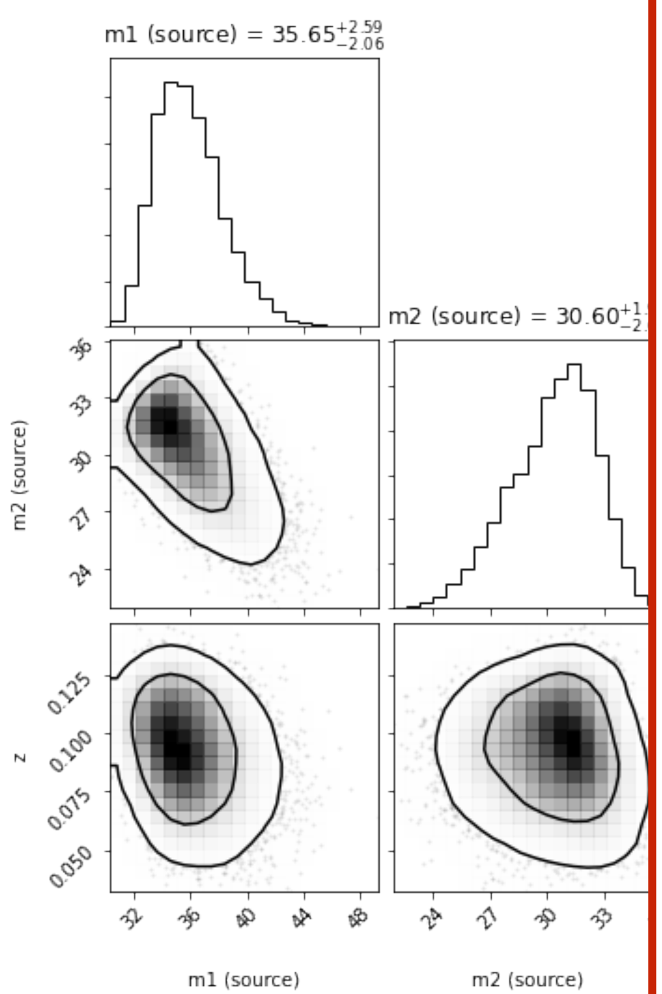
# Computing new quantities from posterior samples

```
import astropy.units as u
from astropy.cosmology import Planck15, z_at_value

z = np.array([z_at_value(Planck15.luminosity_distance, dist * u.Mpc) for dist in
samples['luminosity_distance_Mpc']])

samples['m1_source_frame_Msun']=samples['m1_detector_frame_Msun']/(1.0+z)
samples['m2_source_frame_Msun']=samples['m2_detector_frame_Msun']/(1.0+z)
samples['redshift']=z
```



# Computing new quantities from posterior samples

```
import astropy.units as u
from astropy.cosmology import Planck15, z_at_value

z = np.array([z_at_value(Planck15.luminosity_distance, dist * u.Mpc) for dist in
samples['luminosity_distance_Mpc']])

samples['m1_source_frame_Msun']=samples['m1_detector_frame_Msun']/(1.0+z)
samples['m2_source_fram
samples['redshift']=z
```




```
import bilby
# calculate the detector frame chirp mass
mchirp = ((samples['m1_detector_frame_Msun'] *
samples['m2_detector_frame_Msun'])**(3./5))/\
         (samples['m1_detector_frame_Msun'] +
samples['m2_detector_frame_Msun'])**(1./5)
# initialize a SampleSummary object to describe the chirp mass posterior
samples
chirp_mass_samples_summary =
bilby.core.utils.SamplesSummary(samples=mchirp, average='median')
print('The median chirp mass = {}
Msun'.format(chirp_mass_samples_summary.median))
print('The 90% confidence interval for the chirp mass is {} - {}
Msun'.format(chirp_mass_samples_summary.lower_absolute_credible_interval,
chirp_mass_samples_summary.upper_absolute_credible_interval))
```

```
The median chirp mass = 31.23055308109465 Msun
The 90% confidence interval for the chirp mass is
29.655877108464615 - 32.97324559242388 Msun
```

지난 겨울학교때…

# 데이터 세트

1. BBH1 : time segment = 1240642018 - 1240643042
   - BBH1-H1.gwf, BBH1-L1.gwf
2. BBH2 : time segment = 1240641118 - 1240642142
   - BBH2-H1.gwf, BBH2-L1.gwf
3. BNS1 : time segment = 1262492106 - 1262493130
   - BNS1-H1.gwf, BNS1-L1.gwf
4. BNS2 : time segment = 1262492018 - 1262493042
   - BNS2-H1.gwf, BNS1-L1.gwf

Channel name:
"H1:HWINJ_INJECTED" for H1
"L1:HWINJ_INJECTED" for L1

# How to read data

## Pycbc

```
Read local file:
pycbc.frame.read_frame(file, channel_name)
```

## Bilby

```
ifo_list = bilby.gw.detector.InterferometerList([])
ifo = bilby.gw.detector.get_empty_interferometer('H1')
ifo.strain_data.set_from_frame_file(frame_file=h1gwf,
                                    channel=h1channel,
                                    sampling_frequency=4096,
                                    start_time=gps_start_time,
                                    duration=duration)
ifo_list.append(ifo)
```

# 블랙홀 쌍성병합 문제

문제 **1: Signal Search [15점]**

주어진 시계열 데이터 별로 SNR > 15인 중력파 신호들을 찾고, event time, mass parameters (M1, M2), luminosity distance를 찾는 프로그램을 작성하시오.

- 소스파일: prob1.py
- 출력파일: output1.txt

– 첫번째 행에는 BBH1에 대한 4개의 숫자 (event time, M1, M2, luminosity distance 순서로)
– 두번째 행에는 BBH2에 대한 4개의 숫자 (event time, M1, M2, luminosity distance 순서로)

문제 **2: Parameter Estimation [15점]**

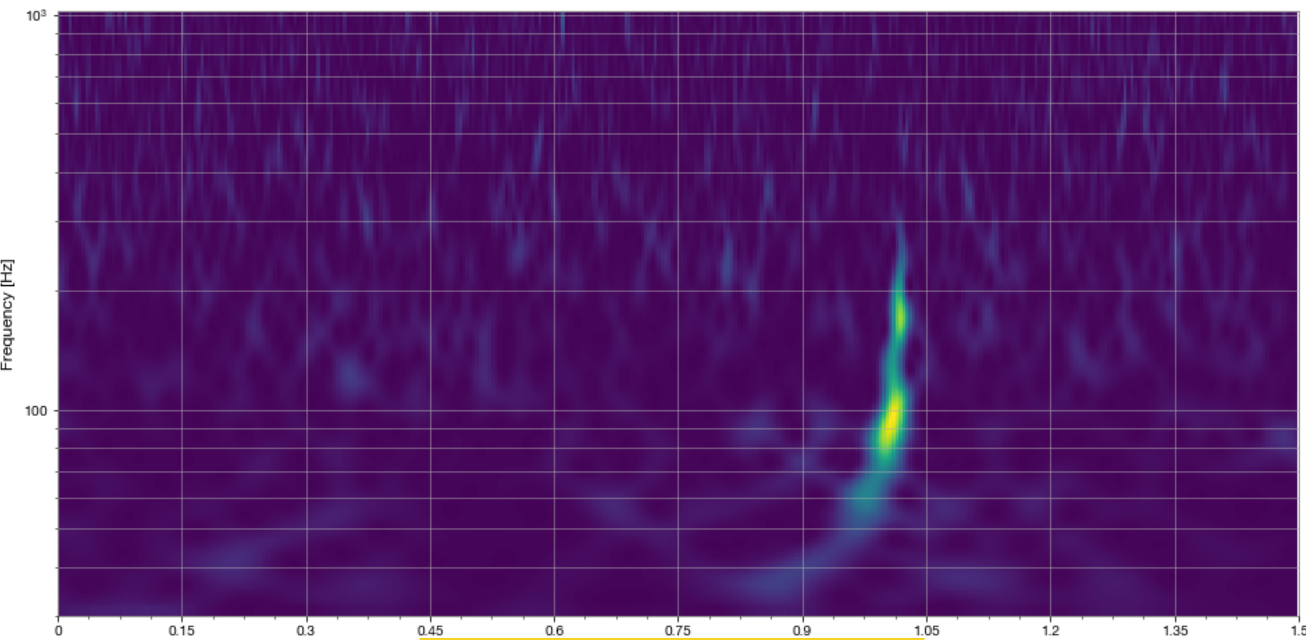
BBH1에 대해 문제 1에서 찾은 event time 주변에서 모수추정을 하시오. M1, M2, luminosity distance 의 posterior samples 을 이용하여 50%와 90% credible region을 찾는 프로그램을 작성하시오.

- 소스파일: prob2.py
- 출력파일: output2.txt

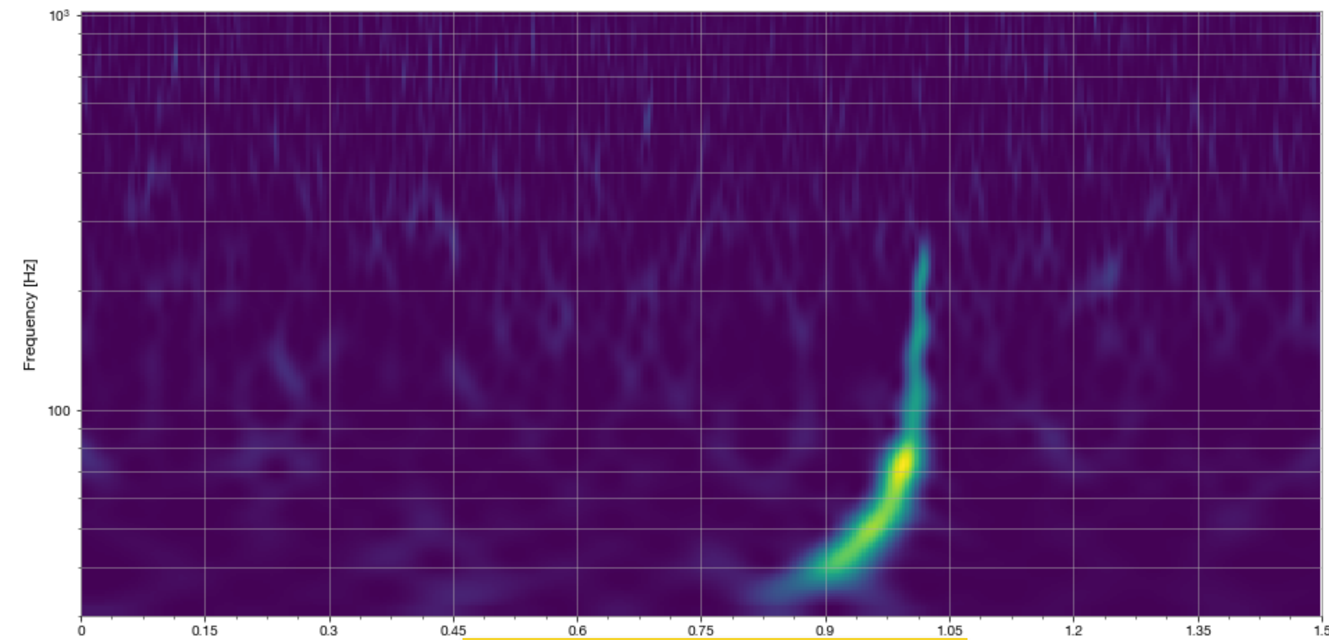
– 첫번째 행에는 50% credible region에 대한 6개의 숫자 (M1, M2, luminosity distance)
– 두번째 행에는 90% credible region에 대한 6개의 숫자 (M1, M2, luminosity distance)

Hint: M_i=[10,100] Msun, Luminosity distance=[400,2000] Mpc

# BBH1

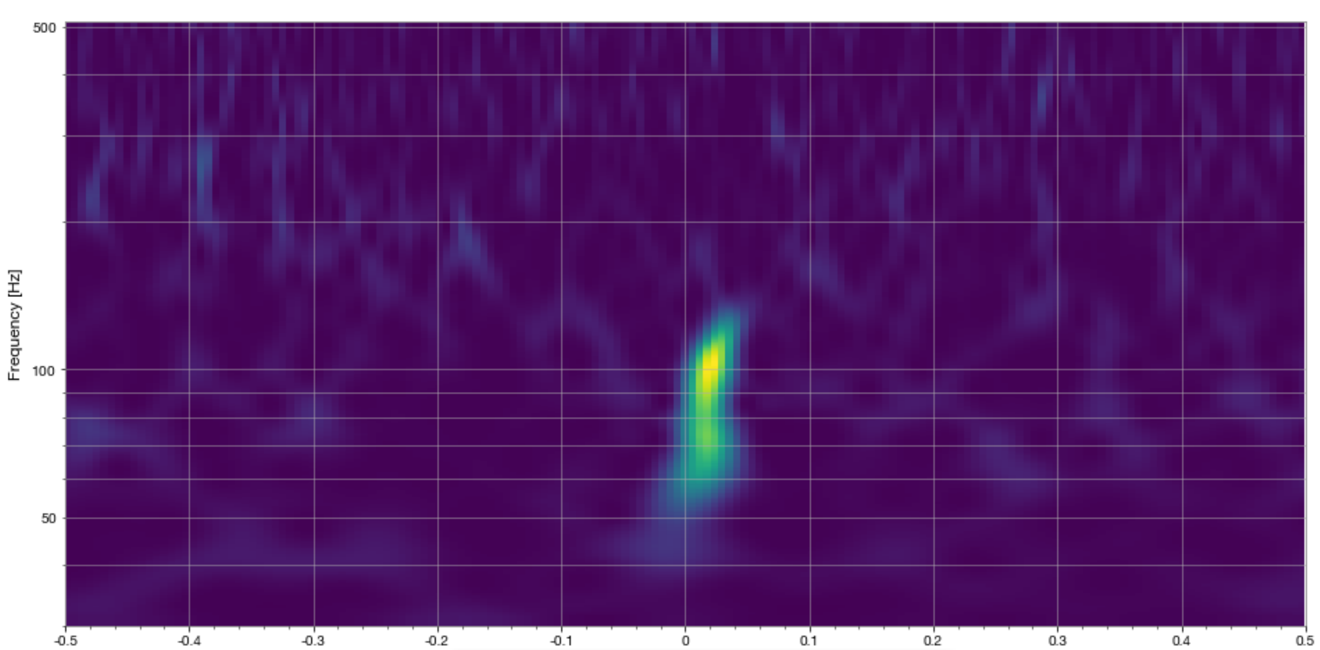



H1

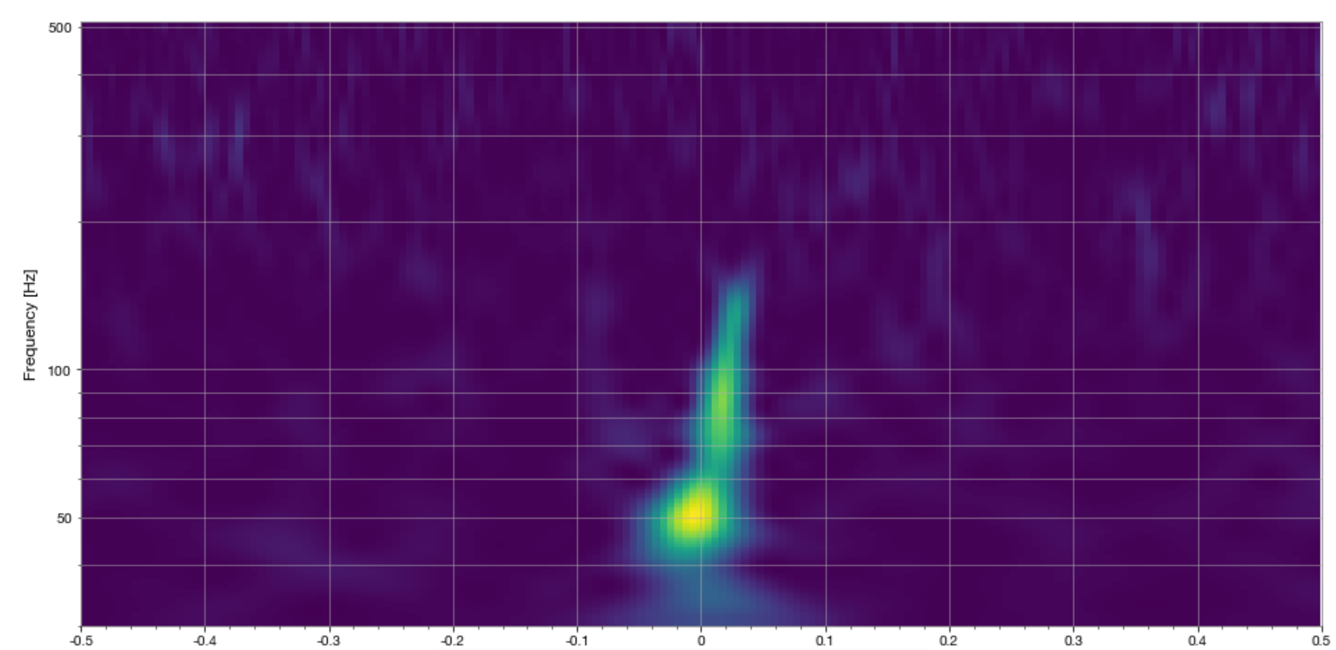



L1

# BBH2



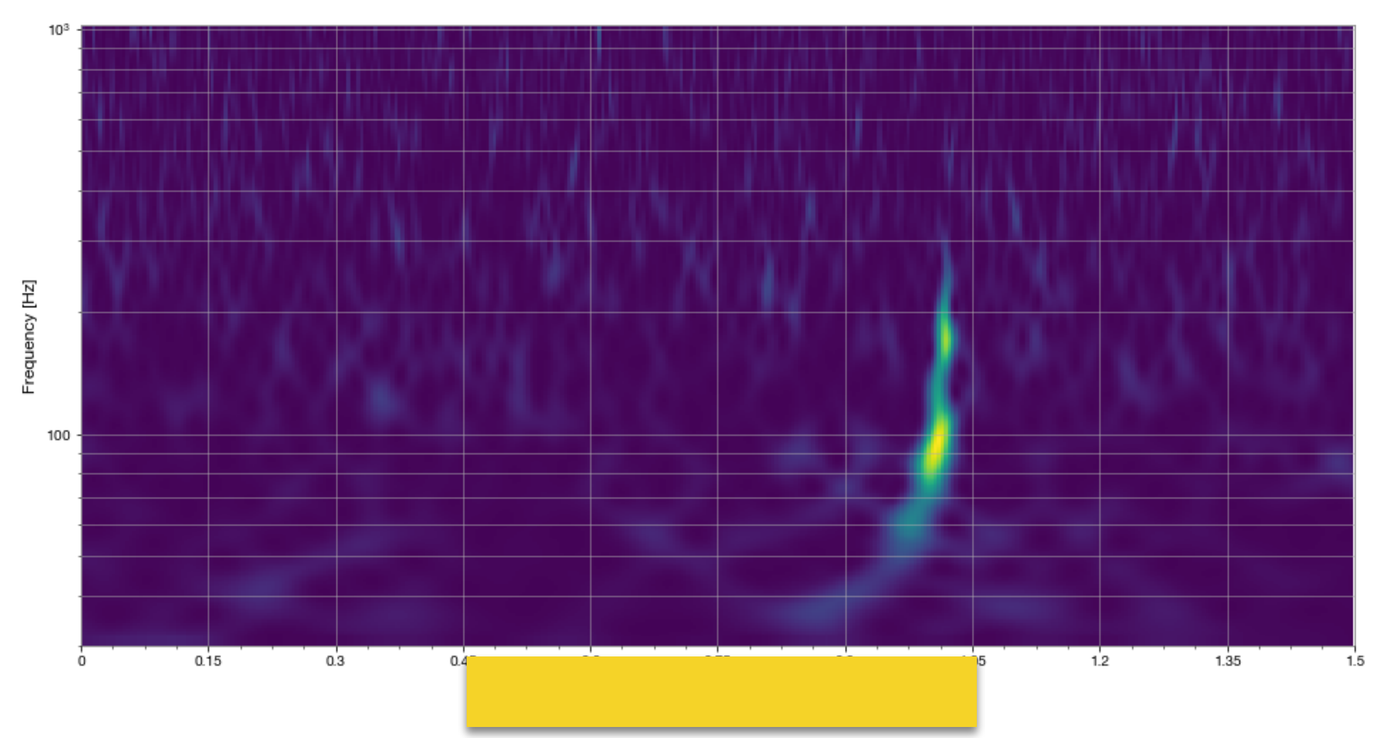

H1




L1

# Spectrogram (Q-transform)

```
from gwpy.timeseries import TimeSeries
data = TimeSeries.read(‘BBH1-H1.gwf','H1:HWINJ_INJECTED',
                    start=GPS_START_TIME,end=GPS_END_TIME)
qspecgram = data.q_transform(qrange=(10,90),frange=(30, 1024),
                    outseg=(start_time,end_time))
plot=qspecgram.plot(figsize=[16,8])
ax=plot.gca()
ax.set_xscale('seconds')
ax.set_yscale('log')
ax.set_ylim(30,1024)
ax.set_xlim(event_time-1.,event_time+0.5)
ax.set_ylabel('Frequency [Hz]')
ax.grid(True,axis='y', which=‘both')
ax.colorbar(cmap=‘viridis',
      label='Normalized energy')
plot.show()
plot.savefig(‘BBH1-H1.png')
```



# 중성자별 쌍성병합 문제 (1)

문제 **3: Signal Search [15점]**
주어진 시계열 데이터 별로 SNR > 15인 중력파 신호들을 찾고, event time, mass parameters ($M_1$, $M_2$), tidal deformabilities($\Lambda_1$, $\Lambda_2$), luminosity distance를 찾는 프로그램을 작성하시오.

- 소스파일: prob3.py
- 출력파일: output3.txt

– 첫번째 행에는 BNS1에 대한 6개의 숫자 (event time, $M_1$ , $M_2$ , $\Lambda_1$ , $\Lambda_2$ , luminosity distance)

– 두번째 행에는 BNS2에 대한 6개의 숫자 (event time, $M_1$ , $M_2$ , $\Lambda_1$ , $\Lambda_2$ , luminosity distance)

문제 **4: Parameter Estimation [15점]**

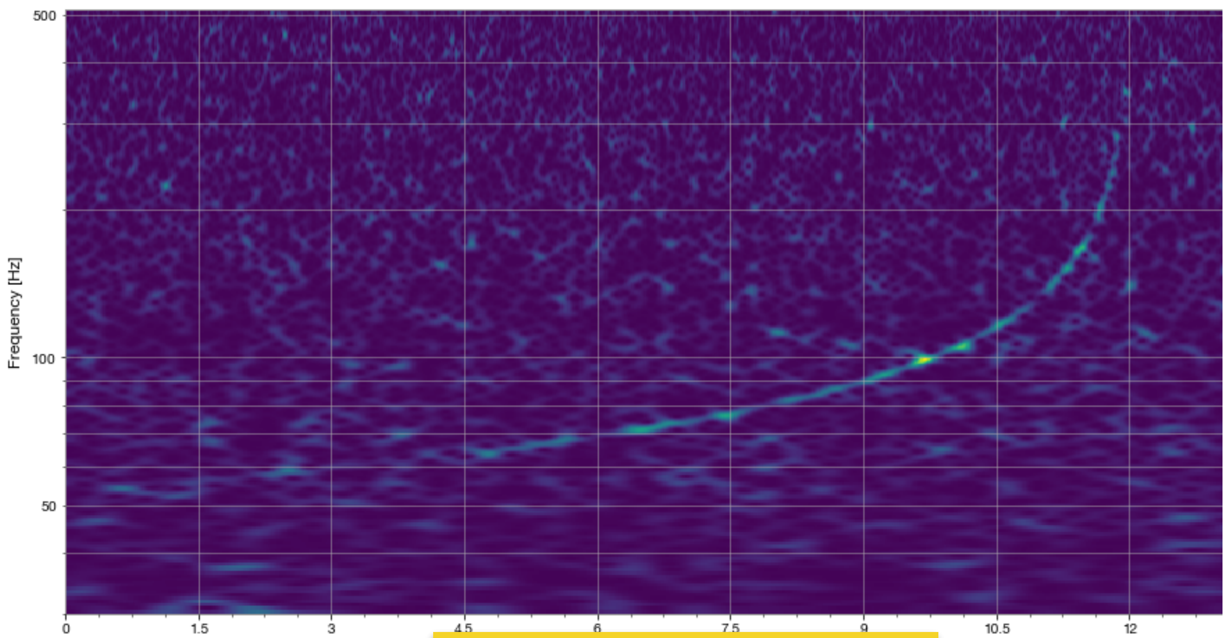
BNS1에 대해 문제 3에서 찾은 event time 주변에서 모수추정을 하시오. $M_1$, $M_2$, $\Lambda_1$, $\Lambda_2$, luminosity distance 의 posterior samples 을 이용하여 50%와 90% credible region을 찾는 프로그램을 작성하 시오.

- 소스파일: prob4.py
- 출력파일: output4.txt

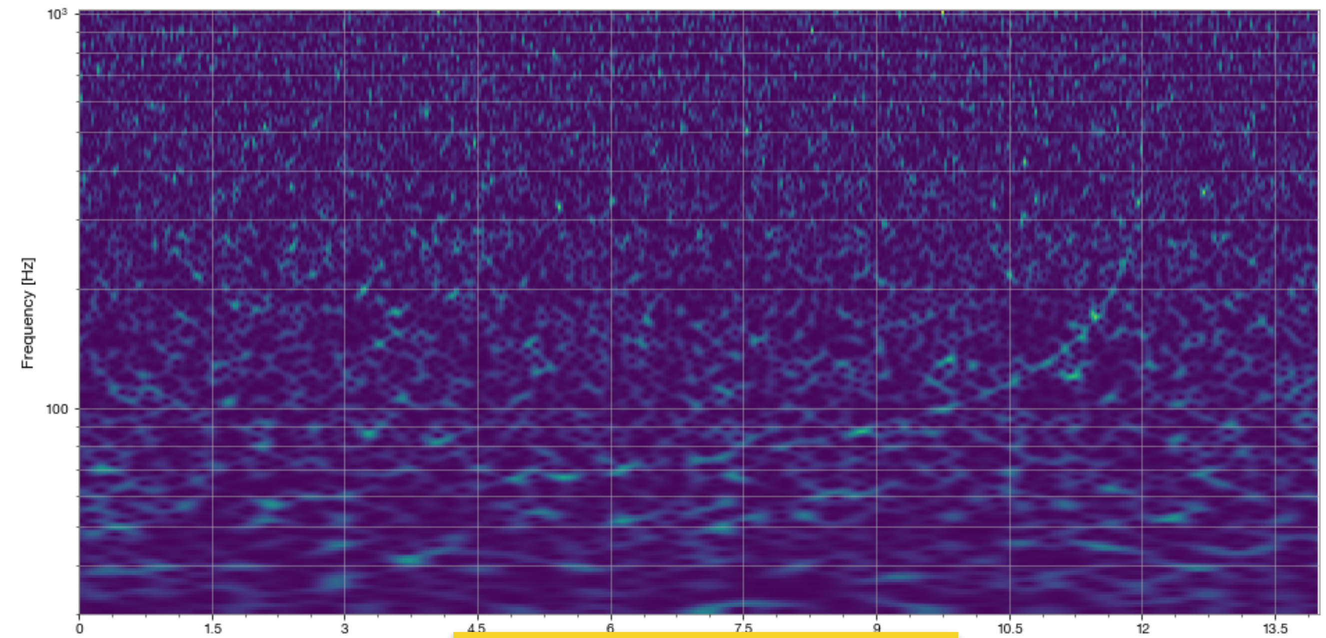
– 첫번째 행에는 50% credible region에 대한 10개의 숫자 ($M_1$, $M_2$, $\Lambda_1$, $\Lambda_2$, luminosity distance)
– 두번째 행에는 90% credible region에 대한 10개의 숫자 ($M_1$, $M_2$, $\Lambda_1$, $\Lambda_2$, luminosity distance)

## Hint: M_i=[1,5] Msun, Λ_i=[10,800], $D_L$=[30,80] Mpc

# BNS1

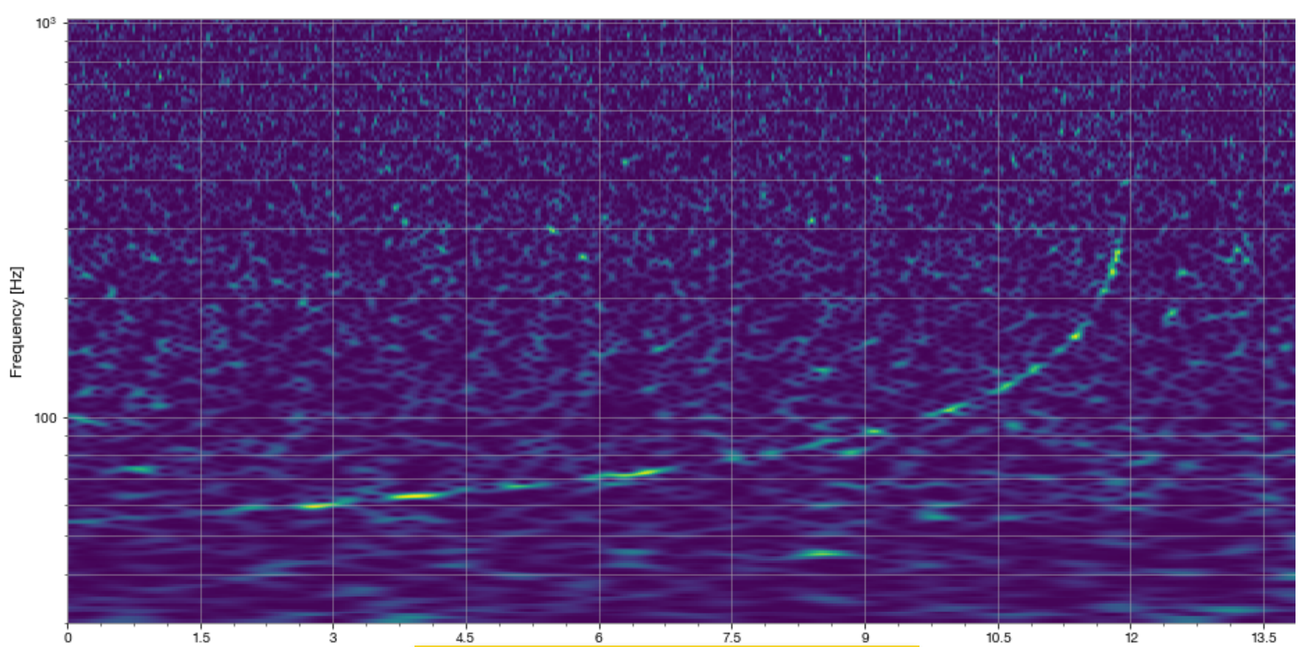



H1

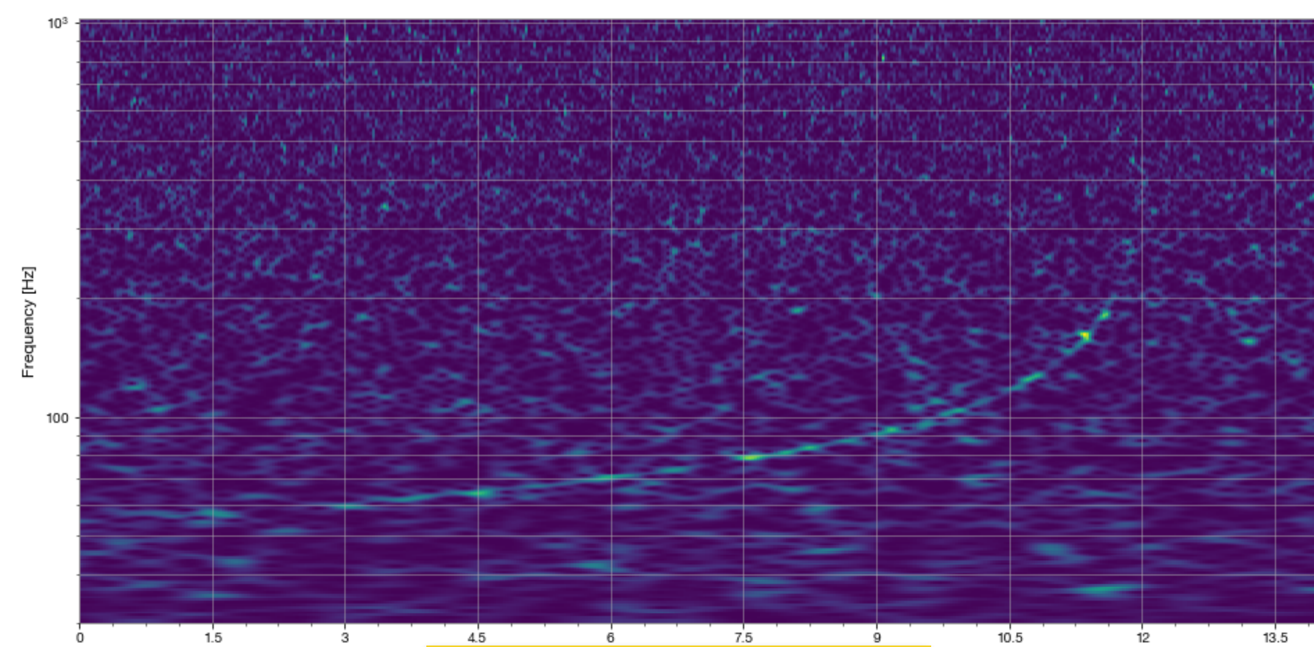



L1

# BNS2




H1




L1

# Hint 1: Signal search

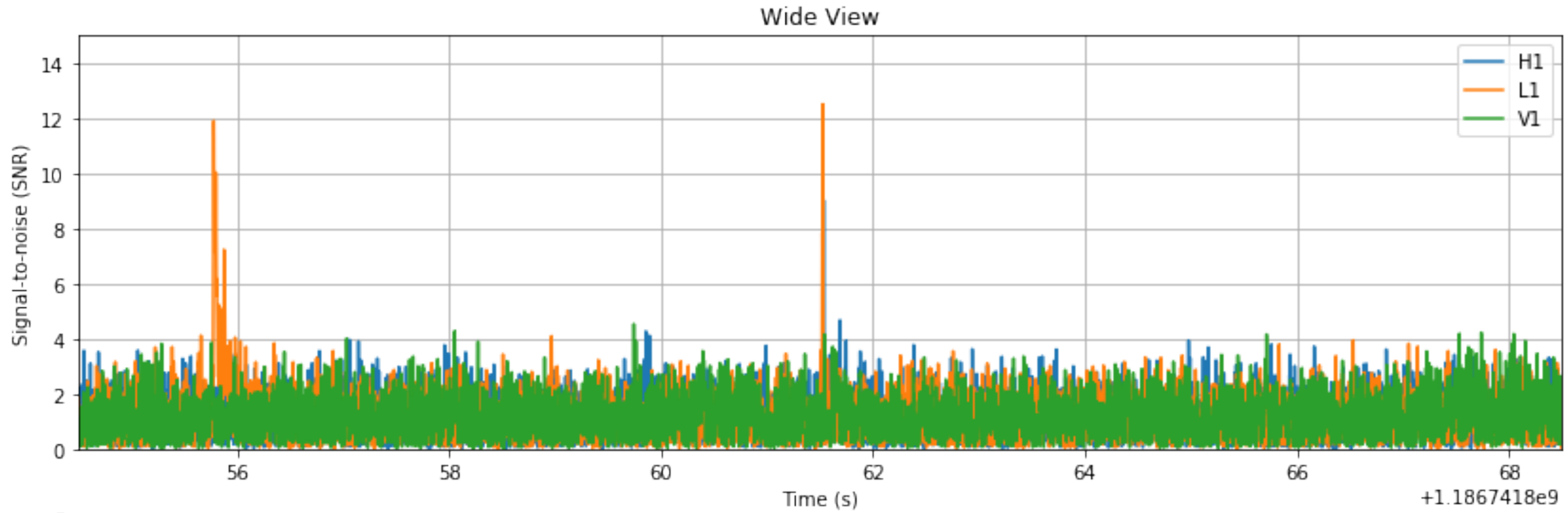
```
from pycbc.frame import read_frame
from pycbc.filter import resample_to_delta_t, highpass, matched_filter
from pycbc.psd import interpolate, inverse_spectrum_truncation
from pycbc.waveform import get_td_waveform
from pycbc.vetoes import power_chisq
from pycbc.events.ranking import newsnr
```

```
snr = matched_filter(template, conditioned, psd=psd,
                     low_frequency_cutoff=30)
snr = snr.crop(4+4,4)

nbins=26
chisq = power_chisq(hp, conditioned, nbins, psd, low_frequency_cutoff=30.0)
chisq = chisq.crop(4+4,4)
dof = nbins * 2 - 2
chisq /= dof
nsnr = newsnr(abs(snr), chisq)

peak = nsnr.argmax()
snrp = nsnr[peak]
time = snr.sample_times[peak]

#peak = abs(snr).numpy().argmax()
#snrp = snr[peak]
#time = snr.sample_times[peak]
```

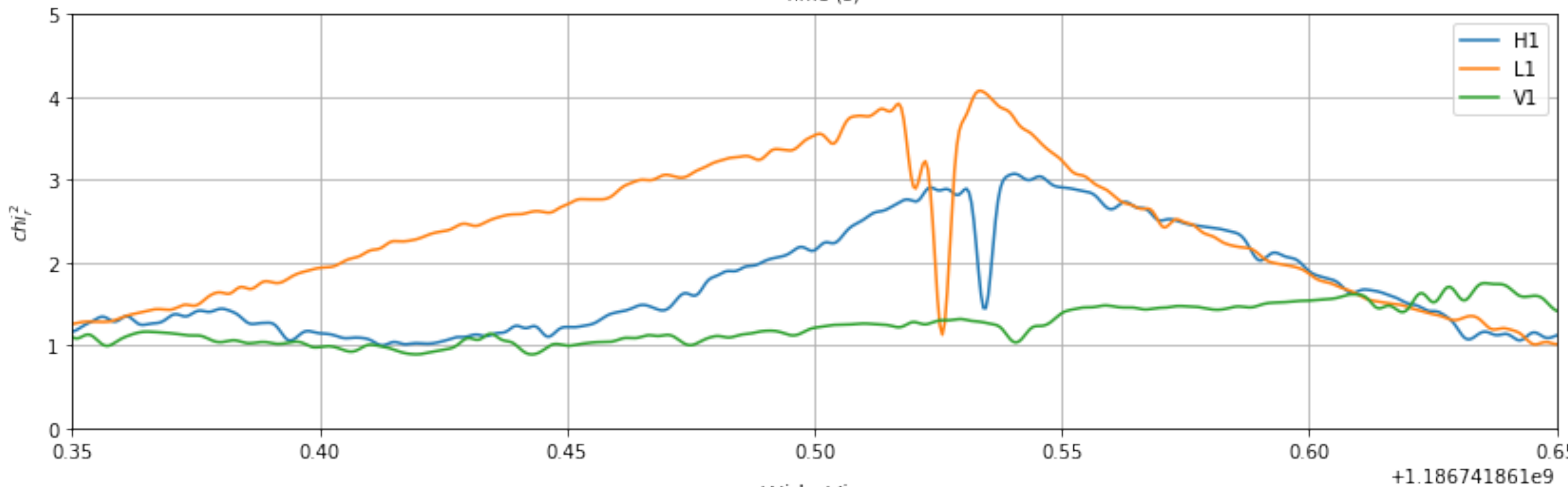
$$\chi^2 = \Sigma_{i=0}^{p}(\rho_i - \rho/p)^2$$

**https://github.com/gw-odw/odw-2021/blob/master/Tutorials/Day_2/Tuto_2.3_Signal_consistency_and_significance.ipynb**

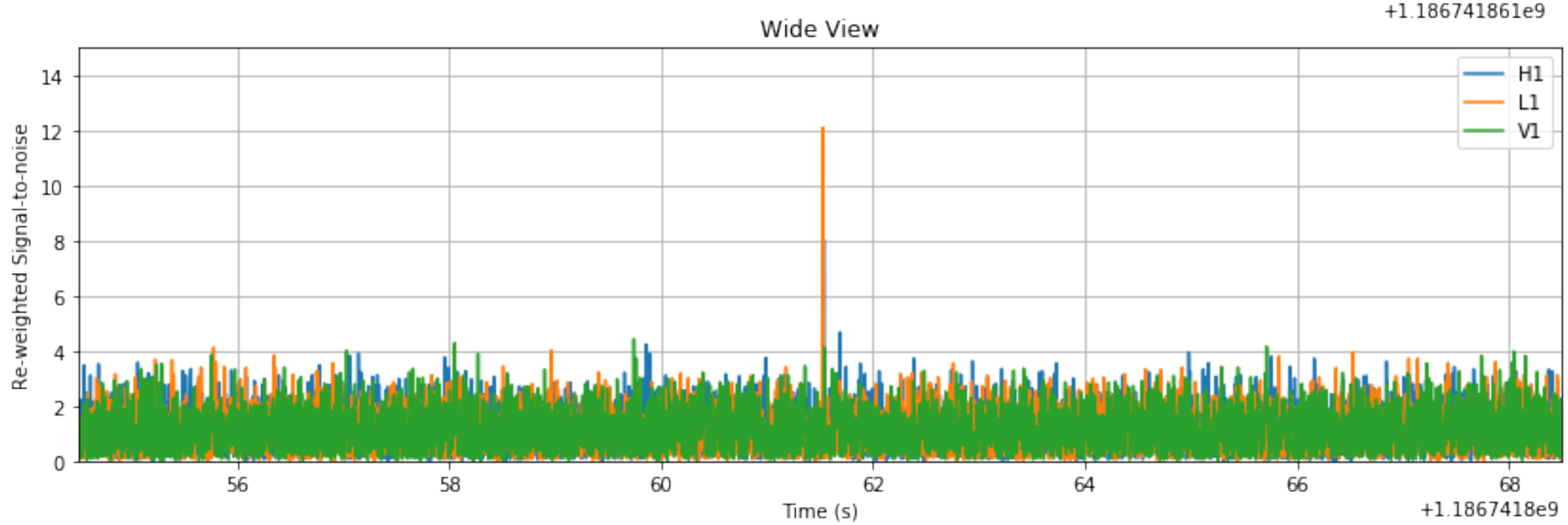
**SNR**




**$Chi^2_r$**




**New SNR**



# Hint 2: Signal search

- 문제1: prob1.py, output1.txt, 문제3: prob3.py, output3.txt
  - gps event time, m1, m2, ($\Lambda_1$,$\Lambda_2$), luminosity distance
  - gps event time 을 정확히 찾는 것이 점수가 높음.
  - SNR 또는 new SNR의 peak을 찾을때 m1,m2 쌍을 잘 찾는것이 좋음.
  - Matched filtering에서 Luminosity distance를 주지 않으면 1Mpc으로 가정하고 계산됨. SNR 값에 크게 영향주지 않음.
  - BNS1, BNS2 신호들은 m1=m2 임.
  - 특정 신호의 L1 또는 H1 데이터는 SNR peak time이 event time이 아님.

# Hint 3: Parameter Estimation

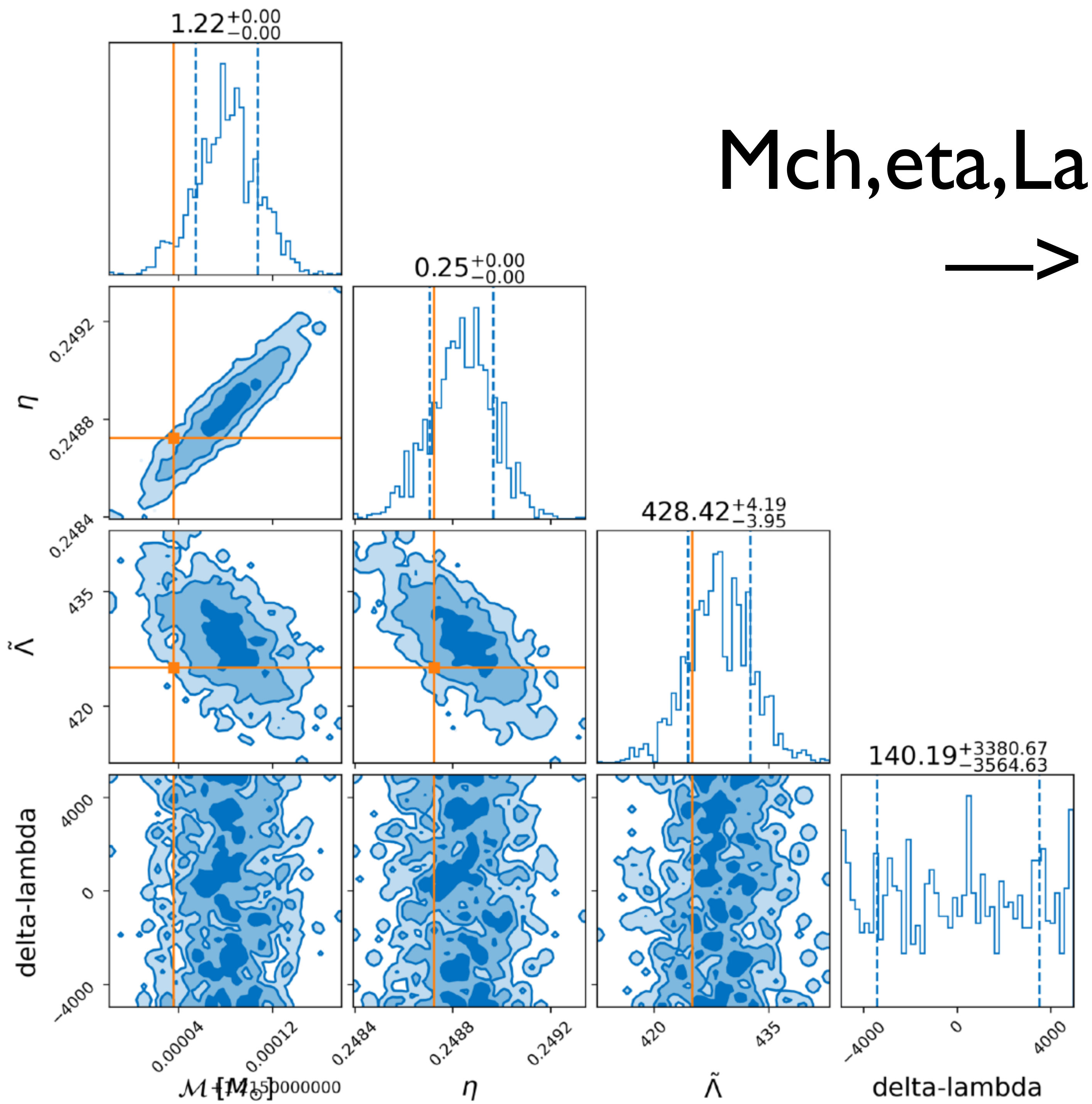
```
import bilby

filename = {'H1':'../BNS1-H1.gwf', 'L1':'../BNS1-L1.gwf'}
channel = {'H1':'H1:HWINJ_INJECTED','L1':'L1:HWINJ_INJECTED'}

ifo_list = bilby.gw.detector.InterferometerList([])

for detector in ['H1','L1']:
    ifo = bilby.gw.detector.get_empty_interferometer(detector)
    ifo.strain_data.set_from_frame_file(frame_file=filename[detector],
                                        channel=channel[detector],
                                        sampling_frequency=4096,
                                        start_time=start_time,
                                        duration=duration)
    ifo_list.append(ifo)


priors = bilby.gw.prior.BNSPriorDict()

priors.pop('mass_ratio')
priors.pop('mass_1')
priors.pop('mass_2')
priors.pop('lambda_1')
priors.pop('lambda_2')

priors['geocent_time'] = event_time

priors['chirp_mass'] = bilby.core.prior.Uniform(1.15, 1.25, name='chirp_mass', unit='$M_{\\odot}$')
priors['symmetric_mass_ratio'] = bilby.core.prior.Uniform(0.1, 0.25, name='symmetric_mass_ratio')
priors['lambda_tilde'] = bilby.core.prior.Uniform(100, 800, name='lambda_tilde')
priors['delta_lambda'] = bilby.core.prior.Uniform(-700, 700, name='delta_lambda')
```

# Hint 3: Parameter Estimation

```
waveform_arguments = dict(waveform_approximant='IMRPhenomPv2_NRTidal',minimum_frequency=30.)
waveform_generator = bilby.gw.WaveformGenerator(
    duration=duration, sampling_frequency=sampling_frequency,
    frequency_domain_source_model=bilby.gw.source.lal_binary_neutron_star,
    parameter_conversion=bilby.gw.conversion.convert_to_lal_binary_neutron_star_parameters,
    waveform_arguments=waveform_arguments)

for interferometer in ifo_list:
    interferometer.minimum_frequency = 30

likelihood=bilby.gw.likelihood.GravitationalWaveTransient(
    ifo_list,waveform_generator,
    time_marginalization=False, phase_marginalization=False,
    distance_marginalization=True,
    priors=priors)

result=bilby.run_sampler(likelihood, priors, sampler='dynesty',
                         outdir=outdir, label=label, nlive=200, nact=3,npool=4,
                         n_check_point=100,check_point_plot=True,use_ratio=True,
                         conversion_function=bilby.gw.conversion.generate_all_bbh_parameters)


result.plot_corner(parameters=['mass_1', ‘mass_2','lambda_1','lambda_2','luminosity_distance'],
                   filename='{}/corner.png'.format(outdir))
```

**!! 프로그램 런타임이 길 수 있음. 10분이상 또는 더 많이...**
**채점결과 같은 점수일 경우에 런타임이 빠른 경우 추가점 발생 할 수 있음.**

# Example - posteriors for BNS w/ Bilby




Mch,eta,Lambda,delta Lambda
—> M1,M2,L1,L2

# 중성자별 쌍성병합 문제 (2)

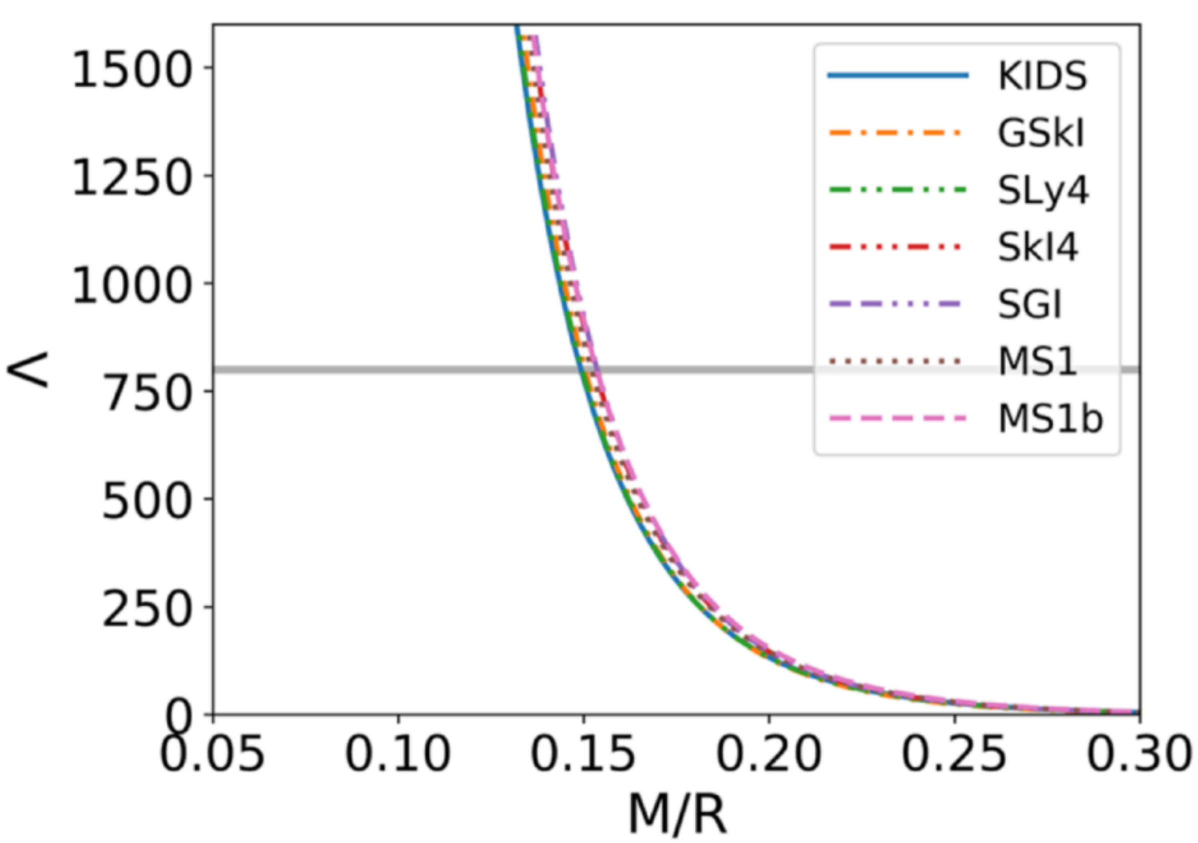
문제 **5: Radius estimation in BNS [10점]**

문제 4에서 찾은 중성자별 쌍성 신호에서 산출한 (M1, Λ1), (M2, Λ2) posterior samples 들로 부터 Λ–C 관계식을 이용하여 중성자별 반경을 산출하여, (M1, R1), (M2, R2)의 50%, 90% credible region을 찾 는 프로그램을 작성하시오.

- 소스파일: prob5.py
- 출력파일: output5.txt

– 첫번째 행에는 50% credible region에 대한 8개의 숫자 (M1, R1, M2, R2 순서로)

– 두번째 행에는 90% credible region에 대한 8개의 숫자 (M1, R1, M2, R2 순서로)




Insensitive to EoS

K. Yagi and N. Yunes, Phys. Rep. 681 (2017) 1

$$C = a_0 + a_1(\ln\Lambda) + a_2(\ln\Lambda)^2$$

a_0=0.360, a_1= - 0.0355, a_2= 0.000705

C=GM/Rc^2

# 중성자별 쌍성병합 문제 (3)

문제 **6: Relation of Radius and Tidal deformability in BNS [15점]**

최근 중성자별 상태방정식 연구들에서는 중성자별 질량 1.4M⊙일때, 조력변형성(Λ)과 반경(R)의 관계가 power law ($\Lambda \approx R^{\alpha}$)를 따름을 보여주고 있다. 문제5에서 구한 결과를 이용하여 α를 산출하고, 50%, 90% credible region을 찾는 프로그램을 작성하시오.(Bilby를 이용하여 직접 likelihood 함수를 구성하 여 결과를 산출하시오.)

- 소스파일: prob6.py
- 출력파일: output6.txt

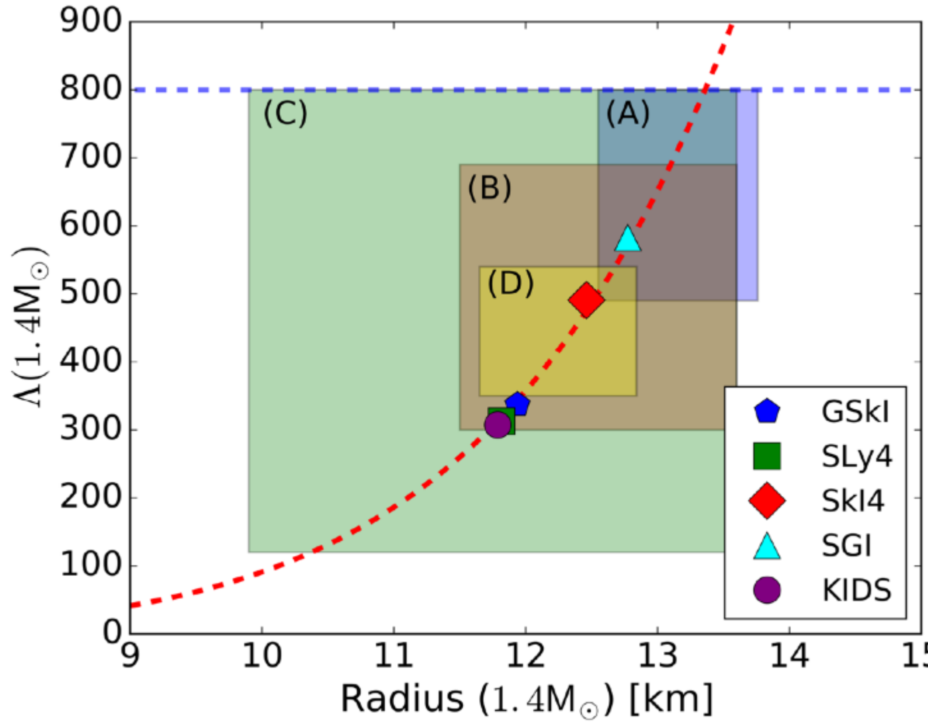
– 첫번째 행에는 50% credible region에 대한 2개의 숫자

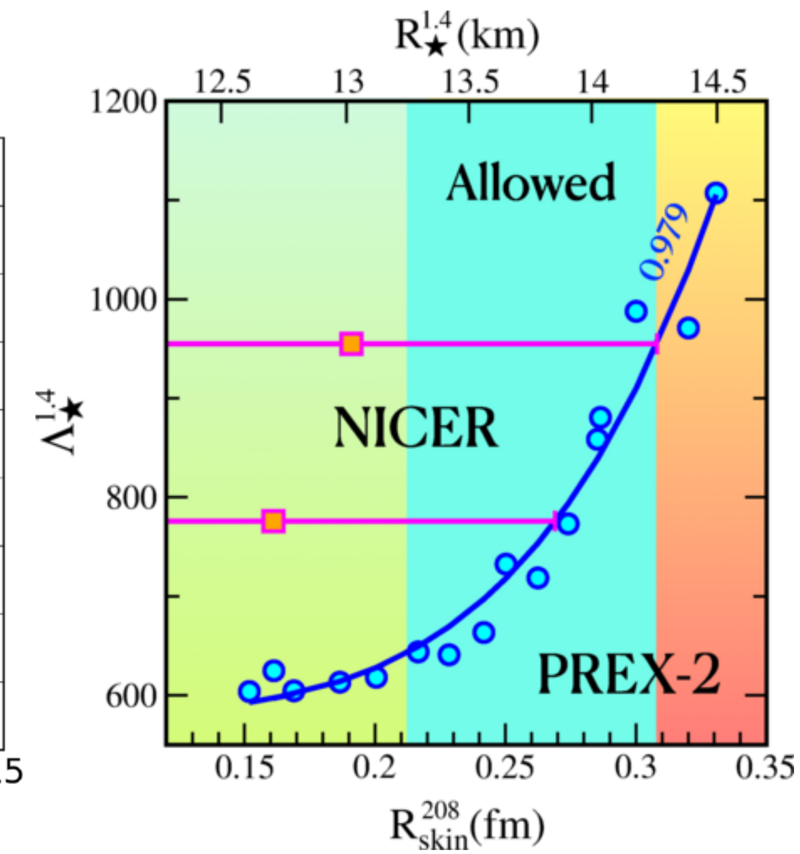
– 두번째 행에는 90% credible region에 대한 2개의 숫자

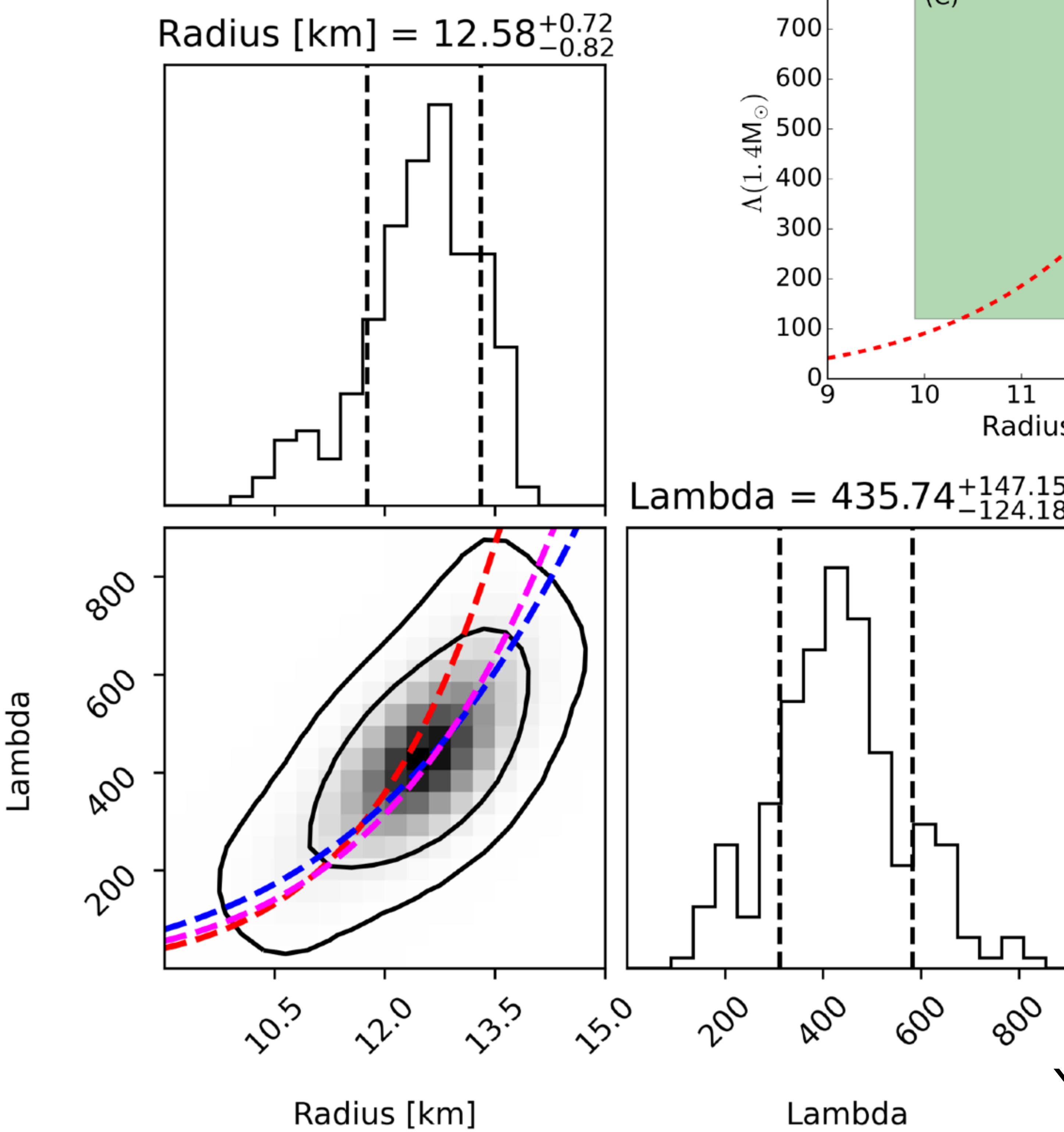
# Lambda-Radius relation

Implication of PREX 2 - PhysRevLett.126.172503 (2021)

Kim et al., PhysRevC.98.065805 (2018)




Red line: $\Lambda(1.4M_\odot) = 2.88 * 10^{-6}\ (R/km)^{7.5}$

[C] PhysRevLett.120.172703.pdf

Blue line: $\Lambda(1.4M_\odot) = 1.35 * 10^{-3}\ (R/km)^{5.0}$

Implication of PREX 2 - PhysRevLett.126.172503 (2021)

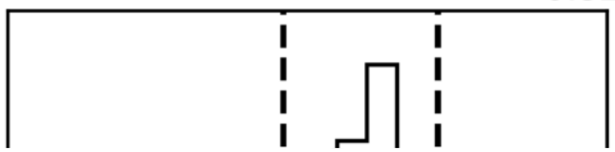
=> $\Lambda \sim R^{4.8}$

Magenta line: $\Lambda(1.4M_\odot) = 1.05 * 10^{-4}\ (R/km)^{6.0}$

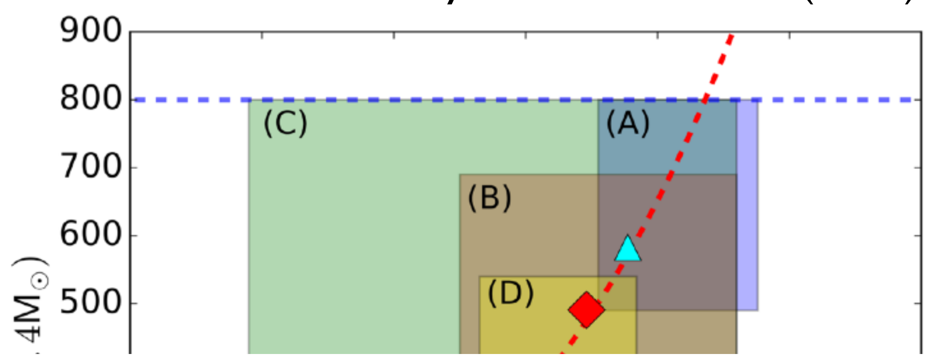
Y.-M. Kim, in progress

# Lambda-Radius relation

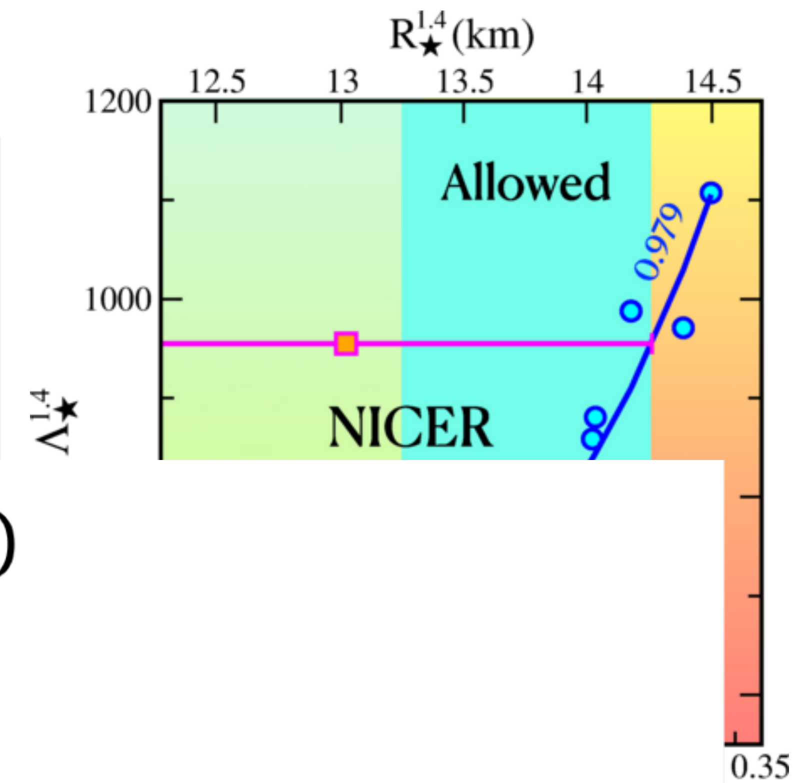
Radius [km] = $12.58^{+0.72}_{-0.82}$



Kim et al., PhysRevC.98.065805 (2018)




Implication of PREX 2 - PhysRevLett.126.172503 (2021)




```python
#!/usr/bin/env python3
"""
An example of how to use bilby to perform parameter estimation for
non-gravitational wave data consisting of a Gaussian with a mean and variance
"""
import bilby
import numpy as np

# A few simple setup steps
label = 'gaussian_example'
outdir = 'outdir'

# Here is minimum requirement for a Likelihood class to run with bilby. In this
# case, we setup a GaussianLikelihood, which needs to have a log_likelihood
# method. Note, in this case we will NOT make use of the `bilby`
# waveform_generator to make the signal.

# Making simulated data: in this case, we consider just a Gaussian

data = np.random.normal(3, 4, 100)


class SimpleGaussianLikelihood(bilby.Likelihood):
    def __init__(self, data):
        """
        A very simple Gaussian likelihood

        Parameters
        ----------
        data: array_like
            The data to analyse
        """
        super().__init__(parameters={'mu': None, 'sigma': None})
        self.data = data
        self.N = len(data)
```

```python
    def log_likelihood(self):
        mu = self.parameters['mu']
        sigma = self.parameters['sigma']
        res = self.data - mu
        return -0.5 * (np.sum((res / sigma)**2) +
                       self.N * np.log(2 * np.pi * sigma**2))


likelihood = SimpleGaussianLikelihood(data)
priors = dict(mu=bilby.core.prior.Uniform(0, 5, 'mu'),
              sigma=bilby.core.prior.Uniform(0, 10, 'sigma'))

# And run sampler
result = bilby.run_sampler(
    likelihood=likelihood, priors=priors, sampler='dnest4', npoints=50000,
    walks=100, outdir=outdir, label=label)
result.plot_corner()
```

- Data : $N(3,4)$

이 코드 응용 필요

Lambda

6.0



Radius [km] Lambda

Y.-M. Kim, in progress

# 중성자별 쌍성병합 문제 (4)

문제 **7: Challenge [15점]**

폴리트로프 상태방정식 ($P = K\varrho^{\Gamma}$)을 이용하여 중성자별의 질량, 반경, 조력변형성을 이론적으 로 산출해 볼수 있다. 폴리트로프 상태방정식을 이용하여, 위에서 구한 (M,Λ) 또는 (M,R) 로부터 likelihood function을 직접 구성하여, Γ posterior samples 을 구하고, 50% credible region을 구하는 프로그램을 작성하시오. (Bilby를 이용하여 직접 likelihood 함수를 구성하여 결과를 산출하시오. LALSIMULATION, Bilby에서 제공하는 TOV solver 사용 가능.)

- 소스파일: prob7.py
- 출력파일: output7.txt

– 첫번째 행에는 50% credible region에 대한 2개의 숫자

– 두번째 행에는 90% credible region에 대한 2개의 숫자

Hint: lalsimulation, bilby에선 상태방정식 테이블을 geometric unit(G=c=1)을 사용한다. 직접상태방정식을 만들때는 밀도, 압력을 geometric unit에서 m 단위로 바꿔 사용해야 함. lalsimulation은 MKS 단위 사용.

# Polytropic equation of state

**Reference for Piece-wise Polytropic EoSs : Read et al. PRD 79, 124032 (2009)**

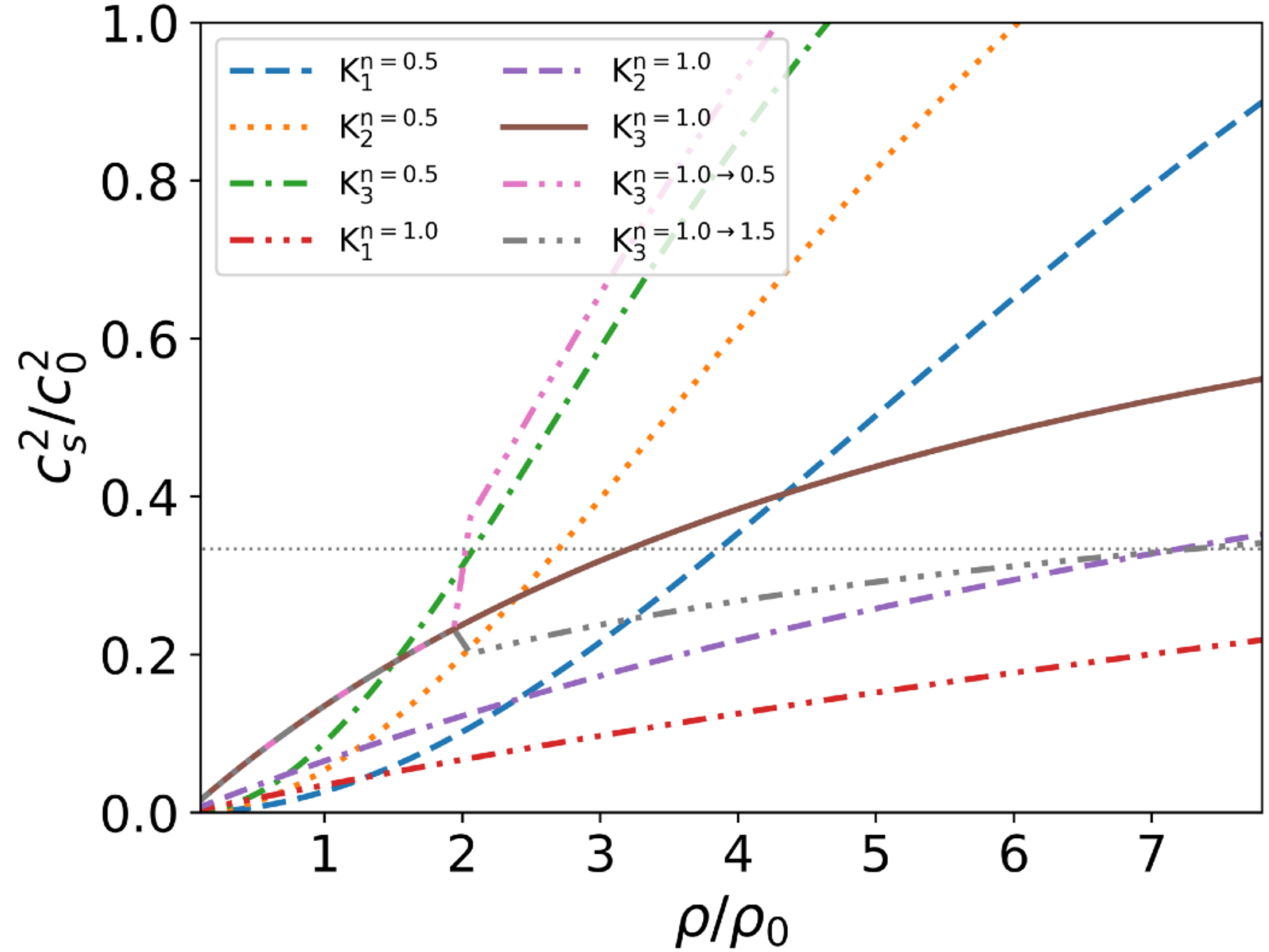
$$p = K\rho^{\Gamma} = K\rho^{1+1/n}$$

$$\begin{aligned}\epsilon =&(1+a)\rho c_0^2 + \frac{p}{\Gamma - 1} \\ =&(1+a)\rho c_0^2 + \frac{K}{\Gamma - 1}\rho^{\Gamma},\end{aligned}$$

(a=0)

$$\frac{c_s^2}{c_0^2} = \frac{\mathrm{d}p}{\mathrm{d}\epsilon} = \Gamma\frac{p}{\epsilon + p}$$

$K_1<K_2<K_3$




M. Kim et al., JKPS, 78, 932-941(2021)
https://link.springer.com/article/10.1007/s40042-021-00084-4

# TOV solver in lalsimulation

```
import lalsimulation as lalsim
```

## EoS generation

Polytrope EoS 내장함수 사용

```
eos = lalsim.SimNeutronStarEOSPolytrope(Gamma,
                                        reference_pressure_SI,
                                         reference_density_SI)
```

Polytrope EoS table 직접 계산해서 eosfile 생성 (첫번째 컬럼 pressure, 두번째 컬럼 density)

```
eos = lalsim.SimNeutronStarEOSFromFile(eosfile)
```

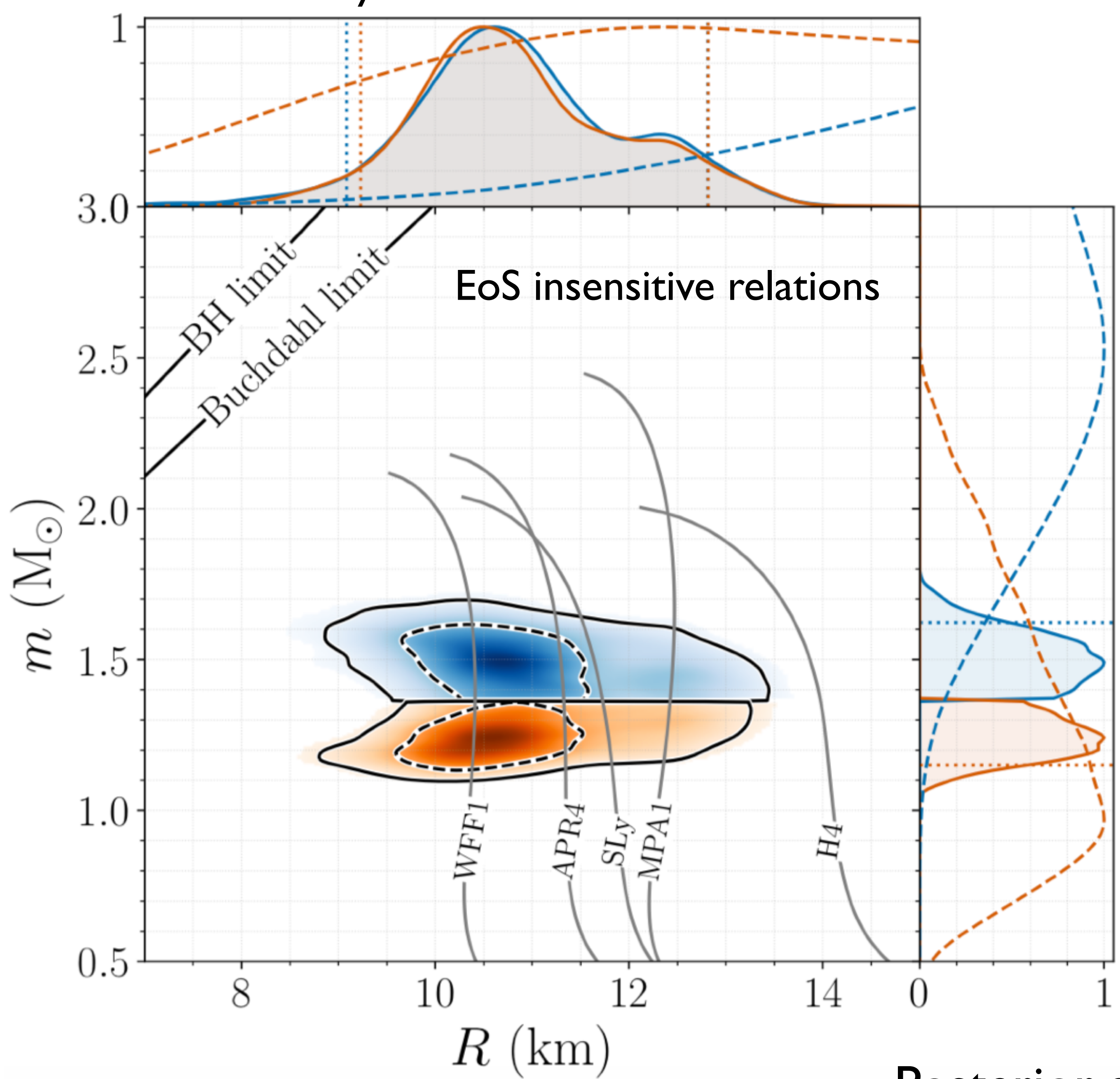
## Solving TOV

```
eosfam = lalsim.CreateSimNeutronStarFamily(eos)

mass = 1.4 * lal.MSUN_SI
radius = lalsim.SimNeutronStarRadius(mass,eosfam)
k2 = lalsim.SimNeutronStarLoveNumberK2(mass,eosfam)
```

참고 자료: https://lscsoft.docs.ligo.org/lalsuite/lalsimulation/group___l_a_l_sim_neutron_star___h.html

# M-R Samples from GW170817

Abbott et al. (LSC and Virgo),
PhysRevLett.121.161101




Posterior samples:
https://dcc.ligo.org/LIGO-P1800115/public

# Sample code

```
import bilby
import numpy as np
import lal
import lalsimulation as lalsim
```

```
# A few simple setup steps
label = "pp_eos_example"
outdir = "outdir"
```

```
class SimpleGaussianLikelihood(bilby.Likelihood):
    def __init__(self, data):
        """
        A very simple Gaussian likelihood

        Parameters
        ----------
        data: array_like
            The data to analyse
        """
        super().__init__(parameters={"logp1": None, "gamma1": None,"gamma2": None,"gamma3": None})
        self.data = data
        self.mass = data[0] #* lal.MSUN_SI # kg
        self.radius = data[1] #* 1000 # m
        self.N = len(self.data[0])
        #self.mr_kernel = stats.gaussian_kde(self.data)
        #self.N = 2000
```

# Sample code (continued)

```
def log_likelihood(self):
    # In PRD 79, 124032 (2009)
    # the unit : cgs
    # in lalsim, we need SI unit
    # logp1  has cgs unit (dyne/cm^2) in table 3 of PRD 79, 124032
    # therefore, logp1_si == values of log(p1) in table 3 minus 1 (1 Pa. = 10 dyne/cm^2)
    # logp1_si == log(p1) - 1

    logp1 = self.parameters["logp1"] - 1
    gamma1 = self.parameters["gamma1"]
    gamma2 = self.parameters["gamma2"]
    gamma3 = self.parameters["gamma3"]

    #mr_kernel = stats.gaussian_kde(self.data)
    #masses, radii = self.mr_kernel.resample(size=self.N)
    masses = self.mass
    radii = self.radius

    eos = lalsim.SimNeutronStarEOS4ParameterPiecewisePolytrope(logp1, gamma1, gamma2, gamma3)
    eosfam=lalsim.CreateSimNeutronStarFamily(eos)
    eos_fail = False
```

# Sample code (continued)

```python
def log_likelihood(self):
    # In PRD 79, 124032 (2009)
    # the unit : cgs
    # in lalsim, we need SI unit
    # logp1  has cgs unit (dyne/cm^2) in table 3 of PRD 79, 124032
    # therefore, logp1_si == values of log(p1) in table 3 minus 1 (1 Pa. = 10 dyne/cm^2)
    # logp1_si == log(p1) - 1

    logp1 = self.parameters["logp1"] - 1
    gamma1 = self.parameters["gamma1"]
    gamma2 = self.parameters["gamma2"]
    gamma3 = self.parameters

    #mr_kernel = stats.gauss
    #masses, radii = self.mr
    masses = self.mass
    radii = self.radius

    eos = lalsim.SimNeutronS
    eosfam=lalsim.CreateSimN
    eos_fail = False
```

```python
if lalsim.SimNeutronStarFamMinimumMass(eosfam)/lal.MSUN_SI > min(masses):
    eos_fail = True
if lalsim.SimNeutronStarMaximumMass(eosfam)/lal.MSUN_SI < max(masses):
    eos_fail = True
if lalsim.SimNeutronStarMaximumMass(eosfam)/lal.MSUN_SI < 2.0:
    eos_fail = True
if lalsim.SimNeutronStarMaximumMass(eosfam)/lal.MSUN_SI > 3.0:
    eos_fail = True

if eos_fail:
    return np.nan
else:
    radius_expected = []
    for m in masses:
        radius_expected.append(lalsim.SimNeutronStarRadius(m*lal.MSUN_SI,eosfam))
    radius_expected = np.array(radius_expected)/1000
    res = radii - radius_expected
    sigma = np.std(radii)
    return -0.5 * (
        np.sum((res / sigma) ** 2) + self.N * np.log(2 * np.pi * sigma**2)
    )
```

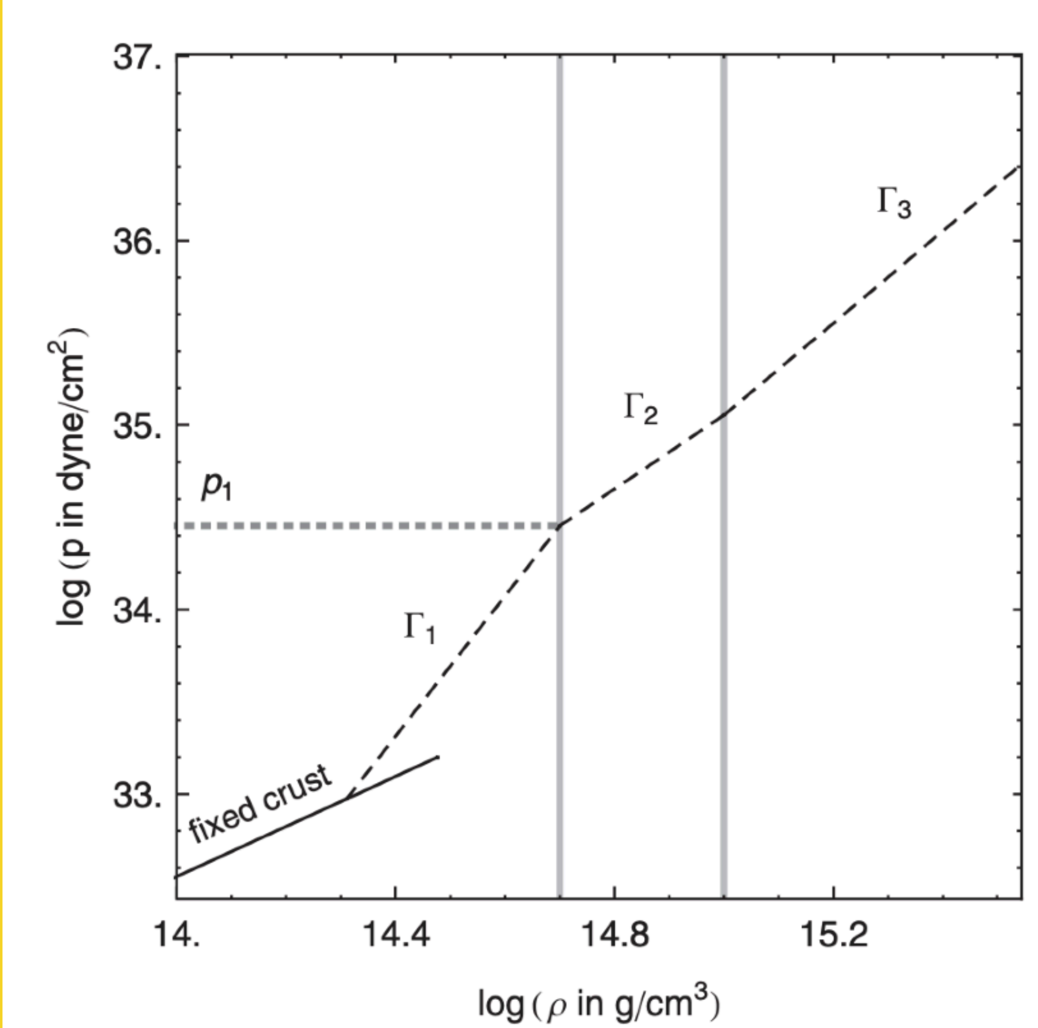
# Sample code (continued)

```
priors = dict(
    logp1=bilby.core.prior.Uniform(34, 34.9, "logp1"),
    gamma1=bilby.core.prior.Uniform(4.0, 6.0, "gamma1"),
    gamma2=bilby.core.prior.Uniform(2.0, 4.0, "gamma2"),
    gamma3=bilby.core.prior.Uniform(1.0, 4.0, "gamma3"),
)
```

```
# for all eos in PRD 79, 124032
# logp1, [33.943,34.858]
# gamma1, [2.013,4.070]
# gamma2, [1.267,3.791]
# gamam3, [1.325,3.660]

# M_max >= 2.0 Msun, eos  in PRD 79, 124032
# logp1, [34.031,34.858]
# gamma1, [2.519,4.070]
# gamma2, [2.246,3.791]
# gamam3, [1.325,3.660]

# M_max >= 2.0 Msun, eos  in PRD 79, 124032
# w/o ALF2, gamma1=4.070, residual=0.043:larger than others
# logp1, [34.031,34.858]
# gamma1, [2.519,3.514]
# gamma2, [2.246,3.791]
# gamam3, [1.325,3.660]
```

Piece-wise polytope: 4 parameters

$$\epsilon(\rho) = (1 + a_i)\rho + \frac{K_i}{\Gamma_i - 1}\rho^{\Gamma_i},$$




Read et al. PRD 79, 124032 (2009)

# Sample code (continued)

```
# load GW170817 eos-insensitive M-Lambda-R data
data = np.loadtxt('../EoS-insensitive_posterior_samples.dat')

m1_source = data[:,0] # Msun
m2_source = data[:,1] # Msun
Lambda1 = data[:,2] # dimensionless
Lambda2 = data[:,3] # dimensionless
radius1 = data[:,4] # km
radius2 = data[:,5] # km

mL1 = np.vstack([m1_source,Lambda1])
mL2 = np.vstack([m2_source,Lambda2])

m_merged = np.concatenate([m1_source,m2_source])
L_merged = np.concatenate([Lambda1,Lambda2])
mL_merged = np.vstack([m_merged,L_merged])


#mL_kernel = stats.gaussian_kde(mL_merged)
#mL_resamples = mL_kernel.resample(size=10)

m_merged = np.concatenate([m1_source,m2_source])
r_merged = np.concatenate([radius1,radius2])
mr_merged = np.vstack([m_merged,r_merged])
mr1 = np.vstack([m1_source, radius1])
mr2 = np.vstack([m2_source, radius2])
```

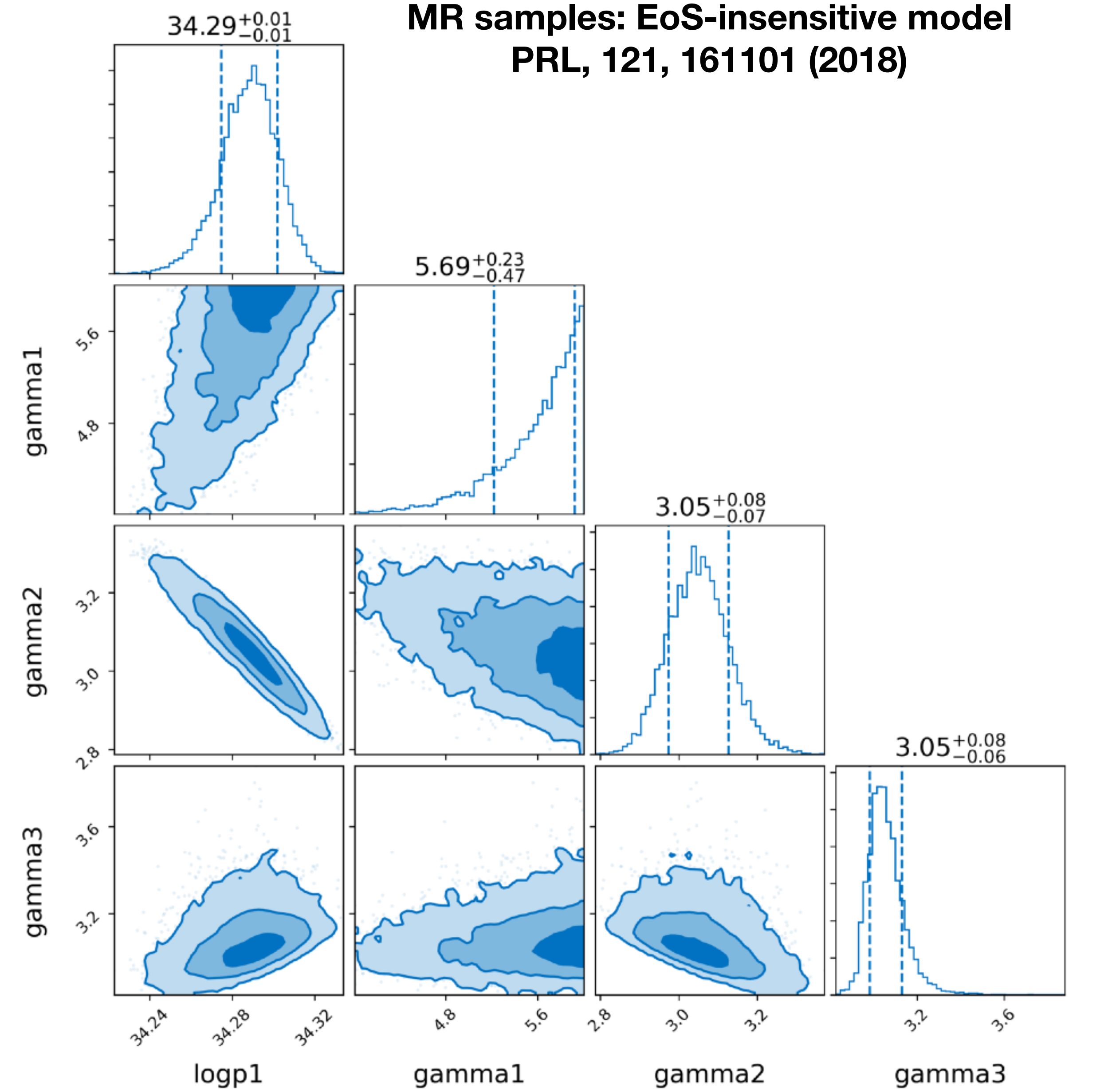
```
likelihood = SimpleGaussianLikelihood(mr_merged)

# And run sampler

result = bilby.run_sampler(
    likelihood=likelihood,
    priors=priors,
    sampler="dynesty",
    nlive=1000,
    outdir=outdir,
    label=label,
    npool=10,
    dlogz=0.1
)

result.plot_corner()
```

# Posteriors w/ Piece-wise Polytropic EoSs

**Reference for Piece-wise Polytropic EoSs : Read et al. PRD 79, 124032 (2009)**

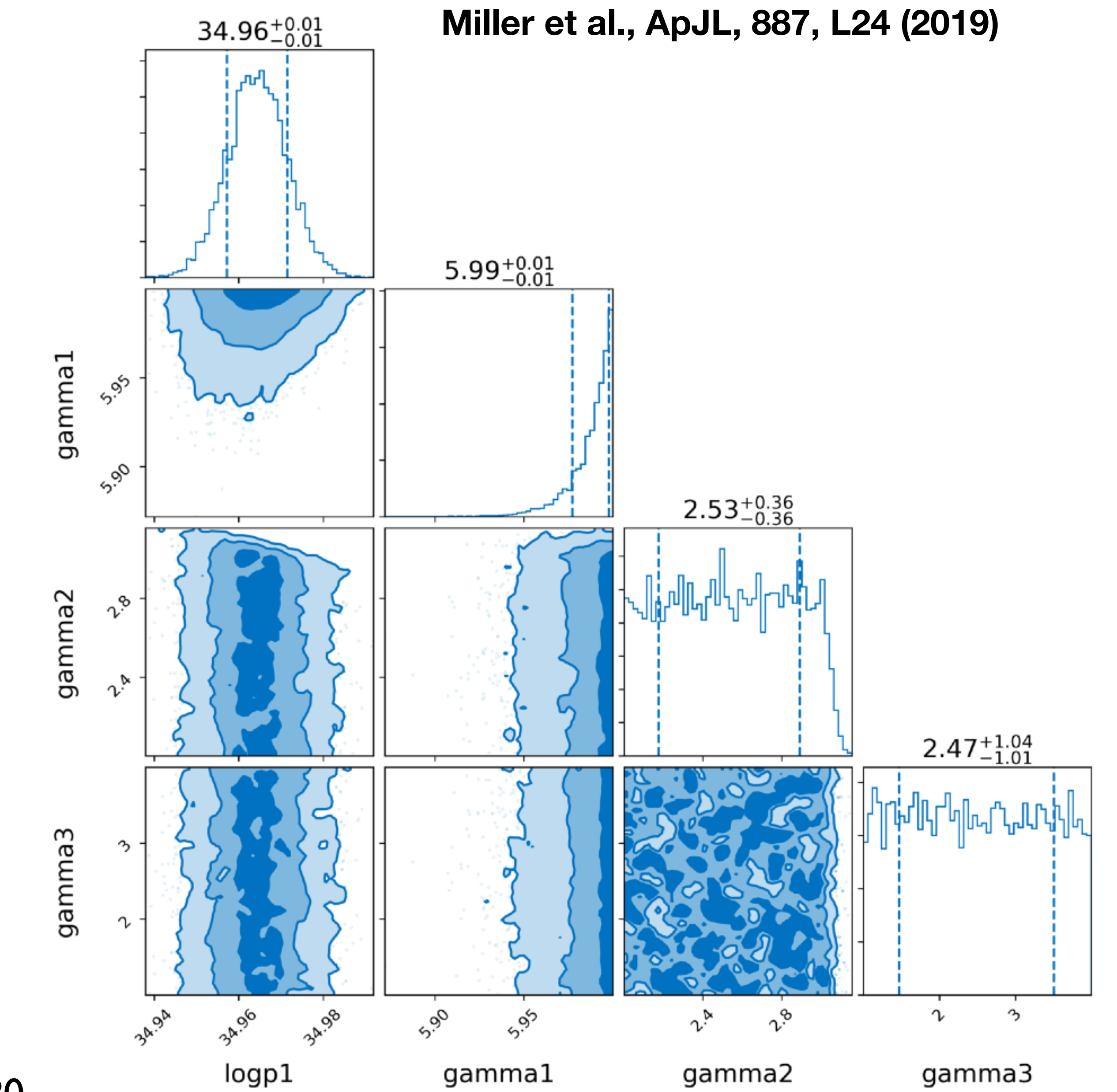
## GW170817

**MR samples: EoS-insensitive model**
**PRL, 121, 161101 (2018)**




## PSR J0030+0451

**MR samples: 2 spot model**
**Miller et al., ApJL, 887, L24 (2019)**



# 참고자료 (1)

1. Pycbc : http://pycbc.org/pycbc/latest/html/index.html
   - https://github.com/gw-odw/odw-2021/blob/master/Tutorials/Day_2/Tuto_2.2_Matched_Filtering_In_action.ipynb
   - https://github.com/gw-odw/odw-2021/blob/master/Tutorials/Day_2/Tuto_2.3_Signal_consistency_and_significance.ipynb
2. Bilby : https://lscsoft.docs.ligo.org/bilby/index.html
   - https://github.com/gw-odw/odw-2021/blob/master/Tutorials/Day_3/Tuto_3.2_Parameter_estimation_for_compact_object_mergers.ipynb
   - https://git.ligo.org/lscsoft/bilby/blob/master/examples/gw_examples/injection_examples/standard_15d_cbc_tutorial.py
   - https://git.ligo.org/lscsoft/bilby/-/blob/master/examples/gw_examples/injection_examples/binary_neutron_star_example.py

# 참고자료 (2)

1. LAL에서의 단위계

- G, gravitational constant = lal.G_SI
- c, speed of light = lal.C_SI
- Msun, solar mass = lal.MSUN_SI

2. Kernel Density Estimation

- https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.gaussian_kde.html

# Jump to Tutorials w/ Colab.

1. https://colab.research.google.com/
2. Search “gw-odw/odw-2022” in GitHub Tab.
   - https://github.com/gw-odw/odw-2022
3. An example of GW150914

★ Save a copy in your google drive

- Go ‘file’ tab > ‘Save a copy in Drive’

★ Email me (ymkim715@gmail.com, or ymkim715@unist.ac.kr) if you have a question after the summer school.

★ Or contact GWOSC team (gwosc@igwn.org)


Thank you for your attention.