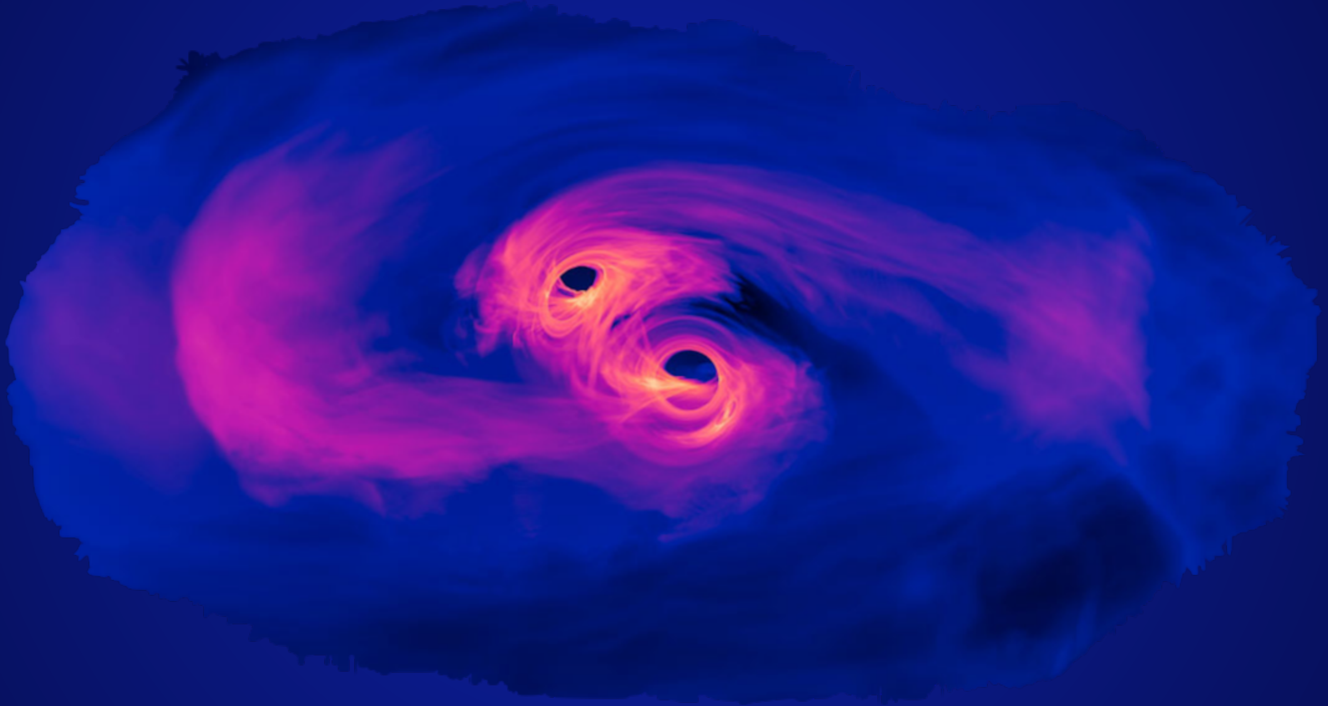


NSF HDR A3D3: Detecting Anomalous Gravitational Wave Signals

Presented by: Eric A. Moreno

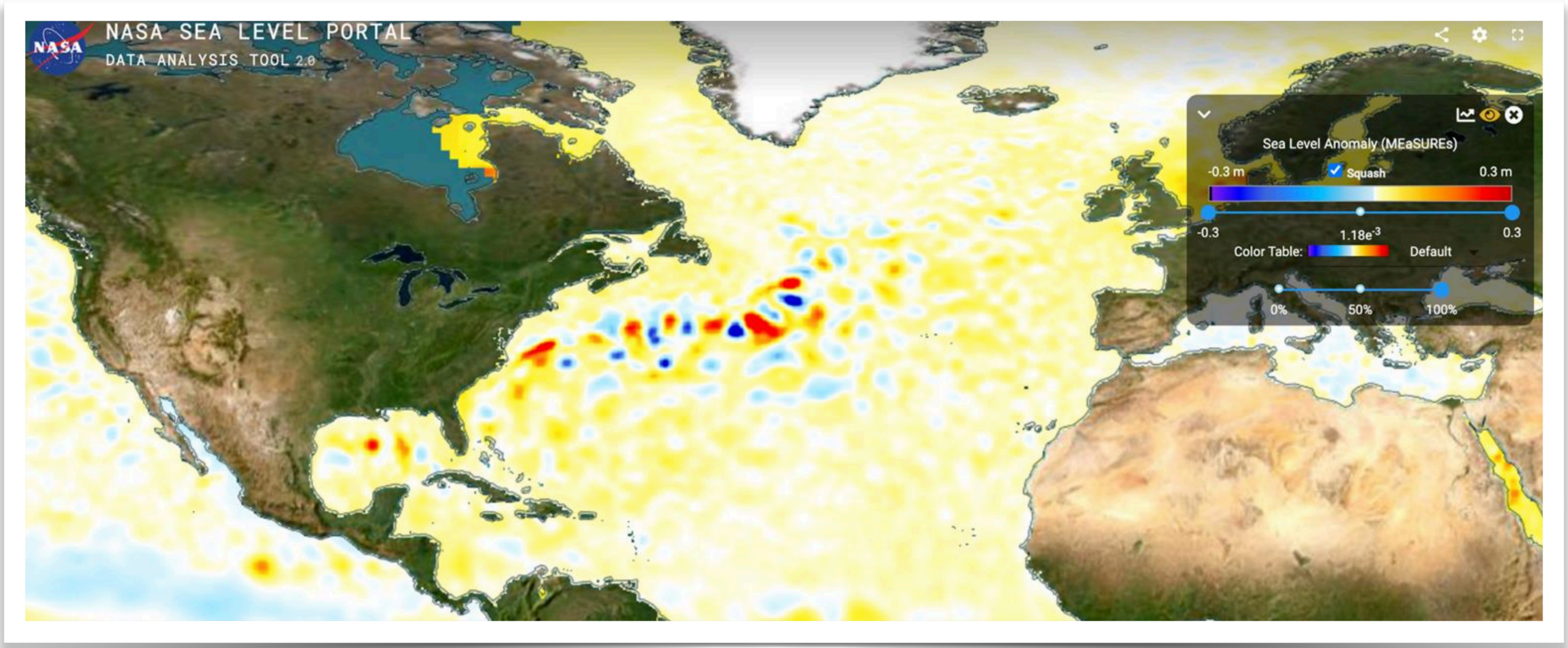
Made by: Katya Govorkova, yuan–Tang Chou, Phil Harris

Don’t forget to check out the other topics…

DETECTING ANOMALOUS SEA LEVEL RISE EVENTS

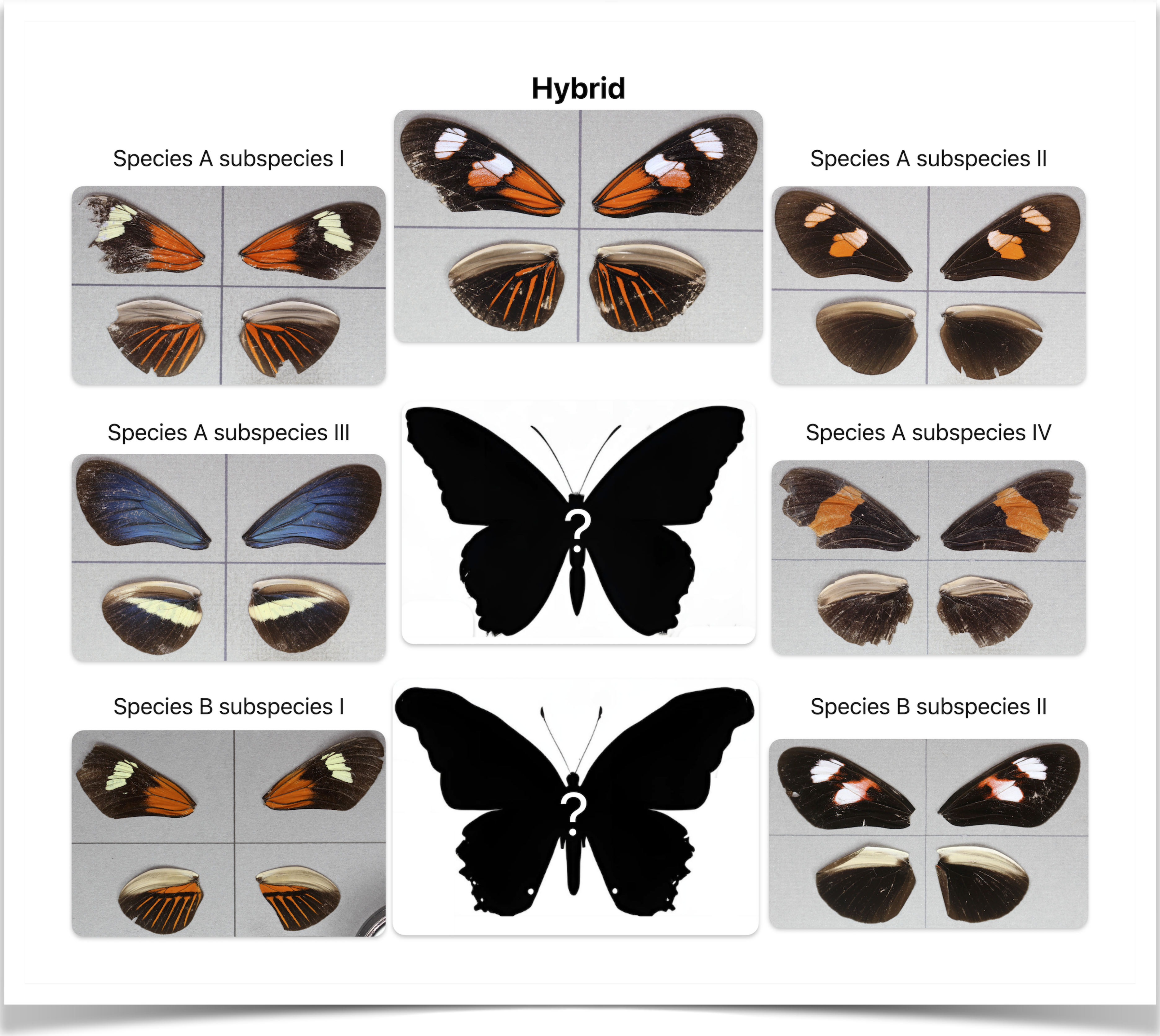
iHARP CodaBench

- **Challenge is to detect anomalous flooding events along the US East Coast** with the maps of sea level over the North Atlantic
- Provided with daily satellite sea level anomaly data over the North Atlantic for the past 30 years
- Provided with dates of anomalous flooding along US East coast stations for the past 30 years



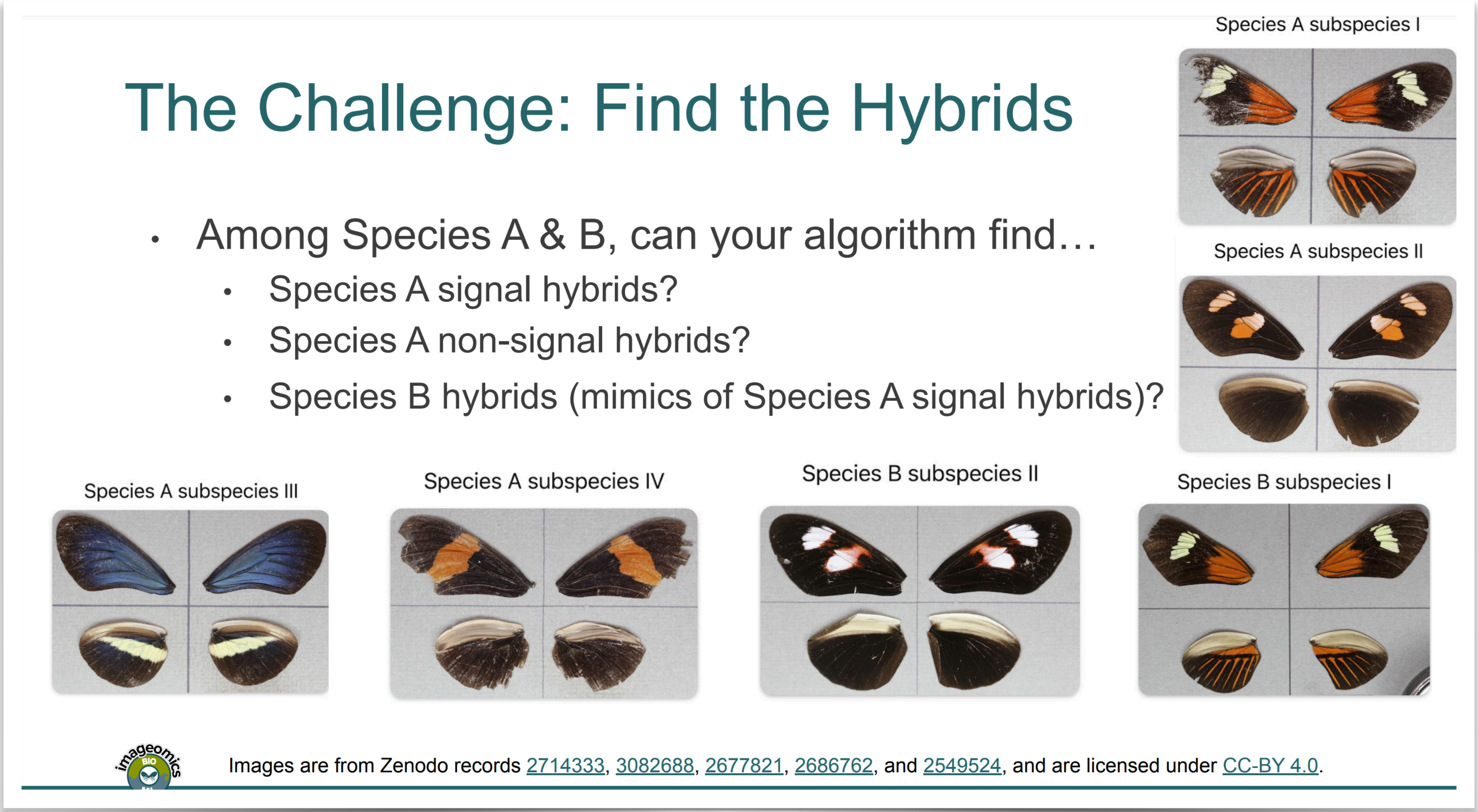
BUTTERFLY HYBRID DETECTION

Butterfly CodaBench



BUTTERFLY HYBRID DETECTION

Butterfly CodaBench



The Challenge: Find the Hybrids

- Among Species A & B, can your algorithm find…
  - Species A signal hybrids?
  - Species A non-signal hybrids?
  - Species B hybrids (mimics of Species A signal hybrids)?

Species A subspecies I

Species A subspecies II

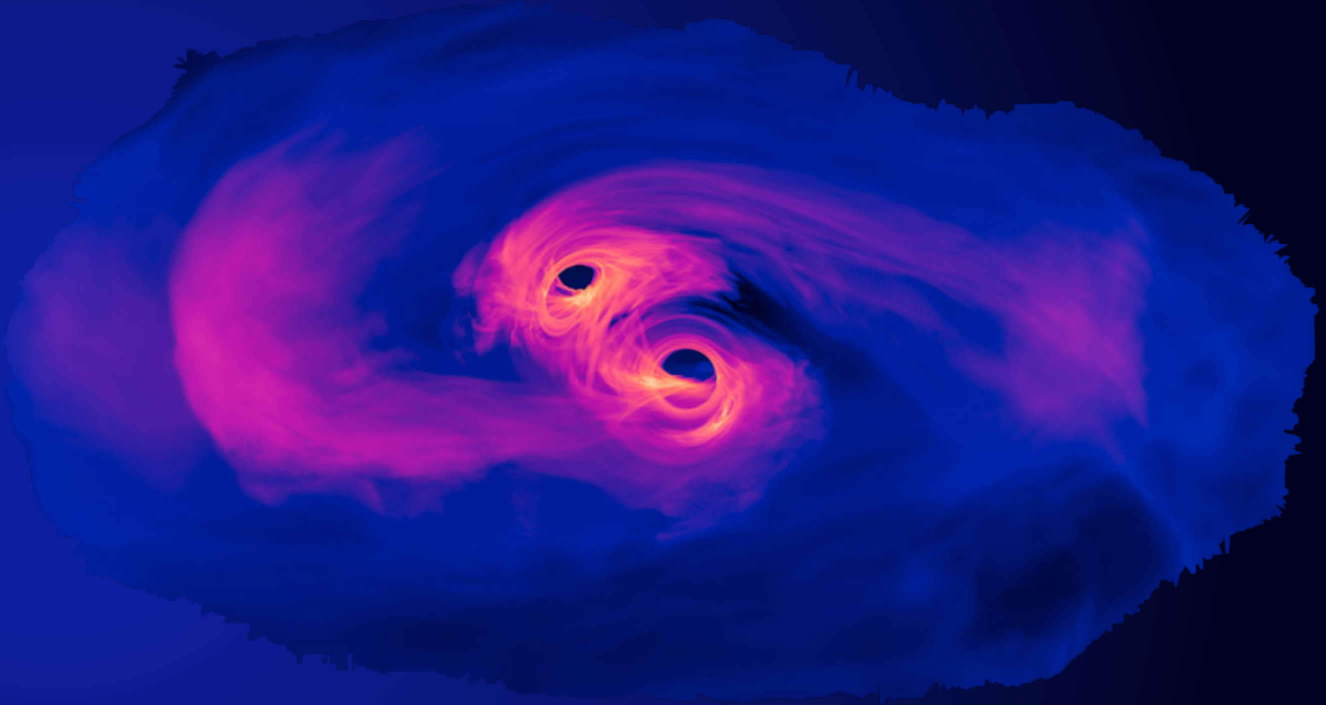
Species A subspecies III

Species A subspecies IV

Species B subspecies II

Species B subspecies I

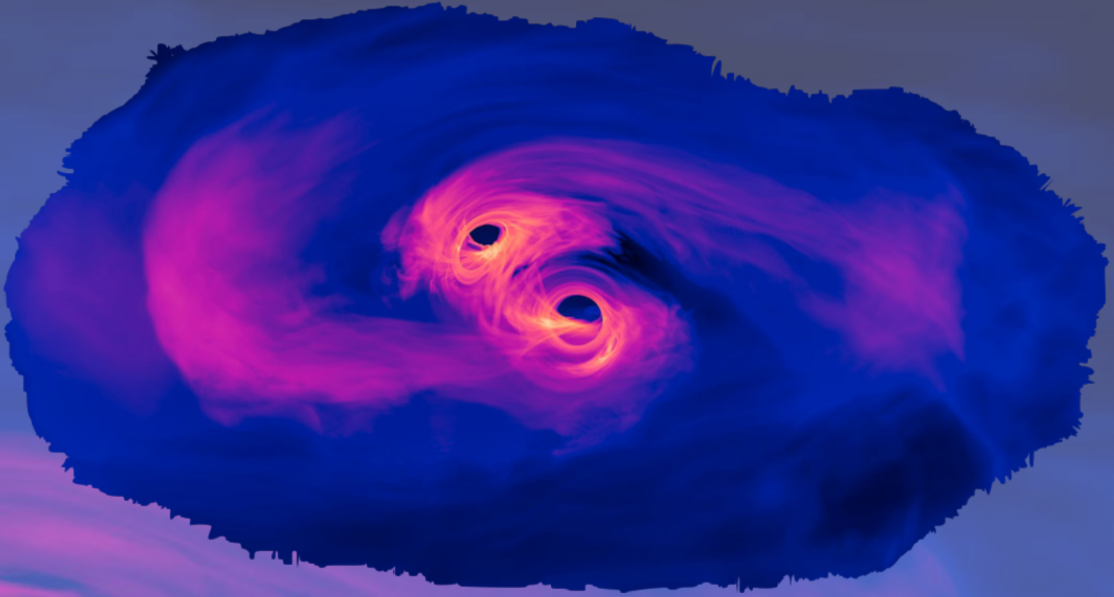
Images are from Zenodo records 2714333, 3082688, 2677821, 2686762, and 2549524, and are licensed under CC-BY 4.0.

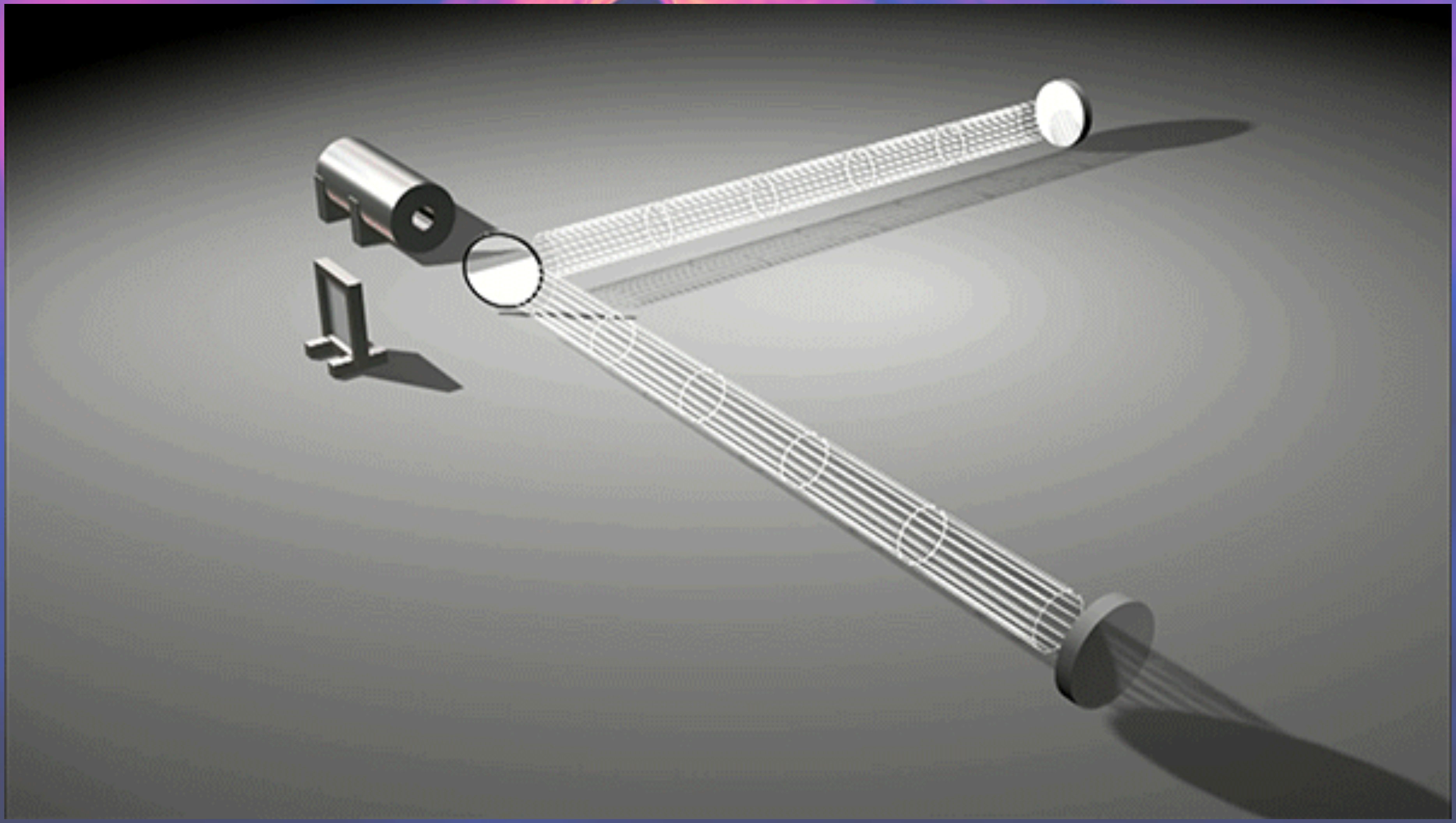


Detecting Anomalous Gravitational Wave Signals

Gravitational Waves and Their Detection

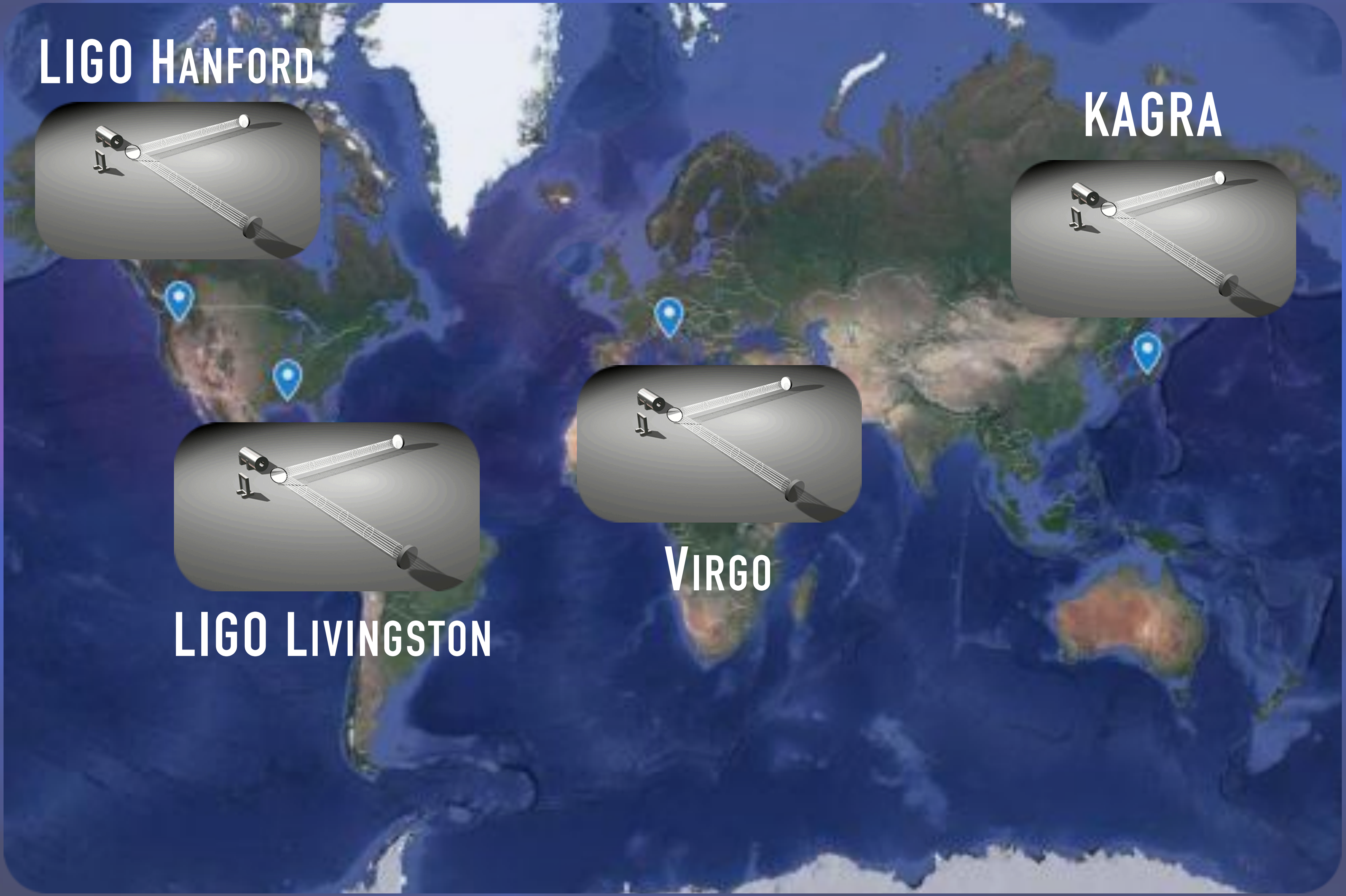
Accelerating masses produce deformations in space time that we can detect via interferometry





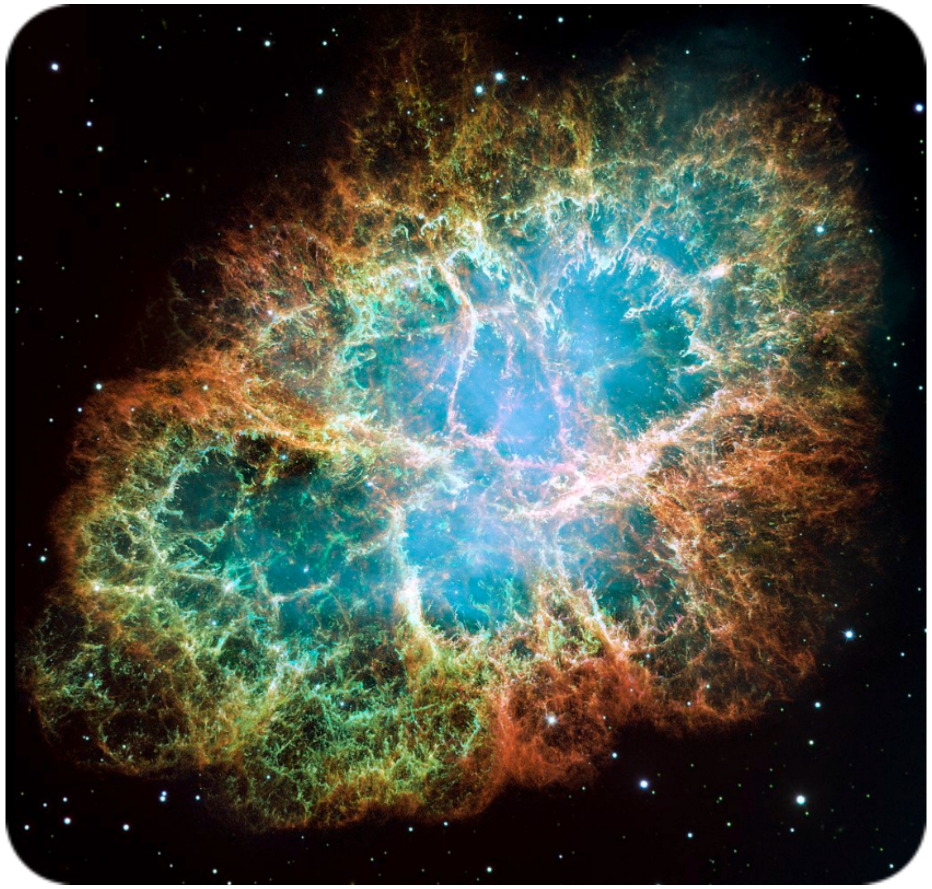
The LIGO-Virgo-KAGRA Collaboration

A signal will appear in at least two interferometers, with the time delay because of the distance between the detectors

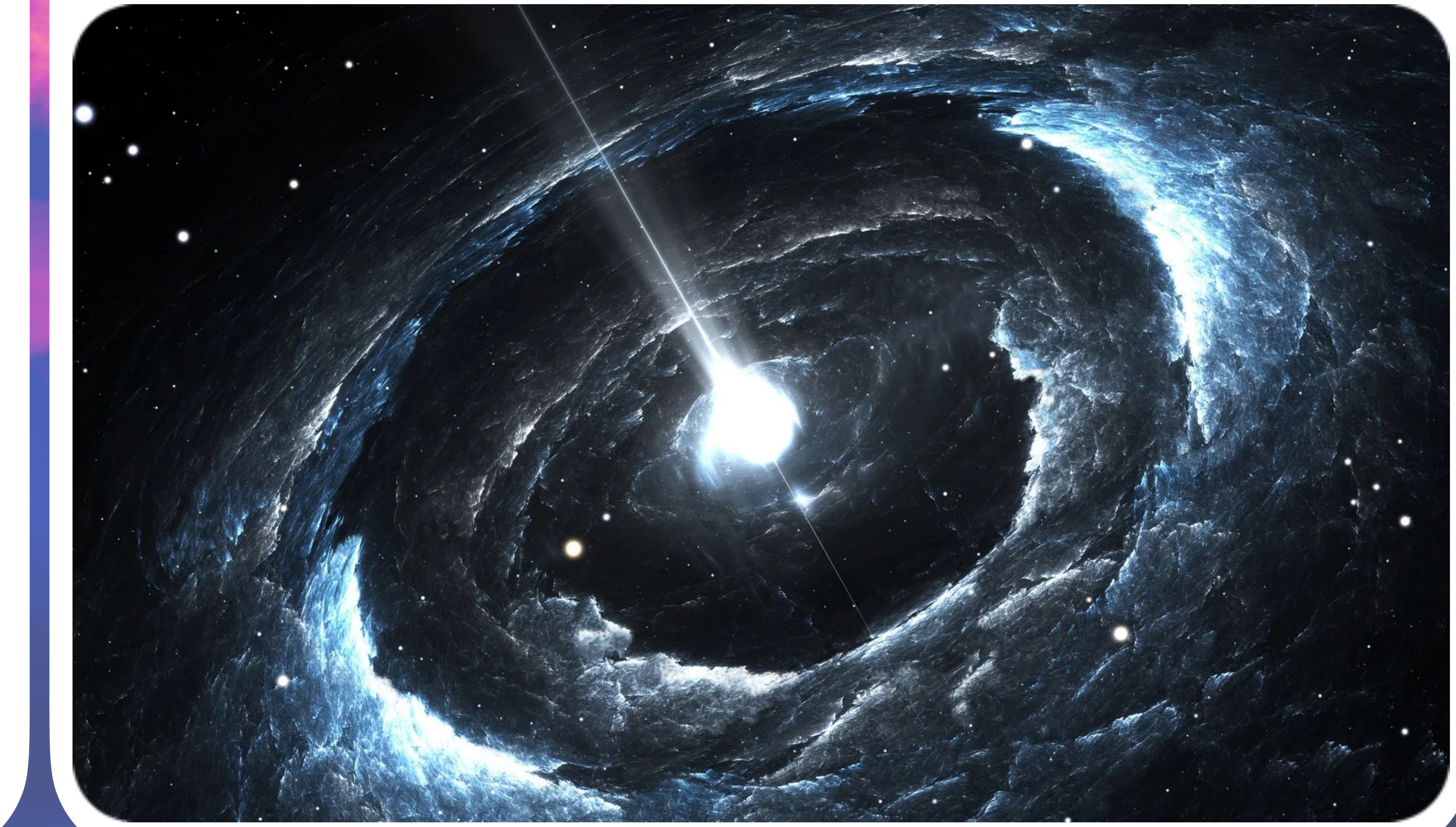


Known "Unknowns" possible signal sources that are poorly modelled and therefore cannot be easily detected using the match filtering pipeline

Core-collapse supernova (CCSN)



Neutron Star Glitches



GWAK Anomalous Gravitational Wave Sources

Unknown "Unknowns" new, unexpected GW sources

We refer to them as anomalous and aim to develop a semi-supervised approach which would let us to discover anomalous signals without explicit modelling

GRAVITATIONAL-WAVE DATA PREPROCESSING

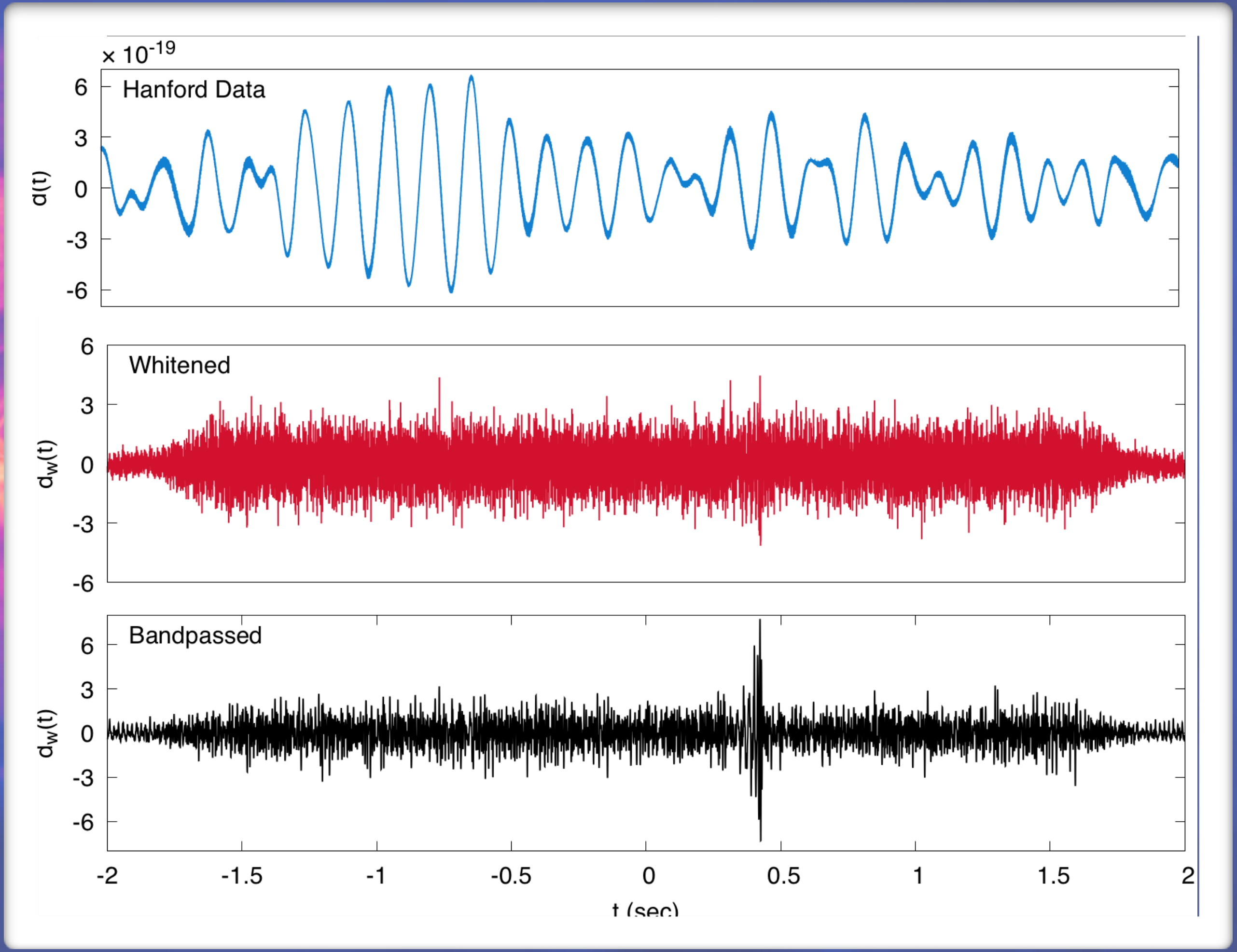
CONTINUOUS TIME SERIES 4096 HZ

WHITENING

IS TRANSFORMING THE DATA SO THAT IT HAS A FLAT (UNIFORM) POWER SPECTRAL DENSITY, MAKING DIFFERENT FREQUENCY COMPONENTS COMPARABLY SCALED FOR MORE EFFECTIVE SIGNAL DETECTION

BANDPASSING 30 HZ < X < 1500 HZ

IS A FILTERING TECHNIQUE THAT ISOLATES THE FREQUENCY RANGE WHERE GRAVITATIONAL WAVE SIGNALS ARE EXPECTED, REMOVING BOTH LOW-FREQUENCY NOISE AND HIGH-FREQUENCY COMPONENTS OUTSIDE THE SIGNAL BAND



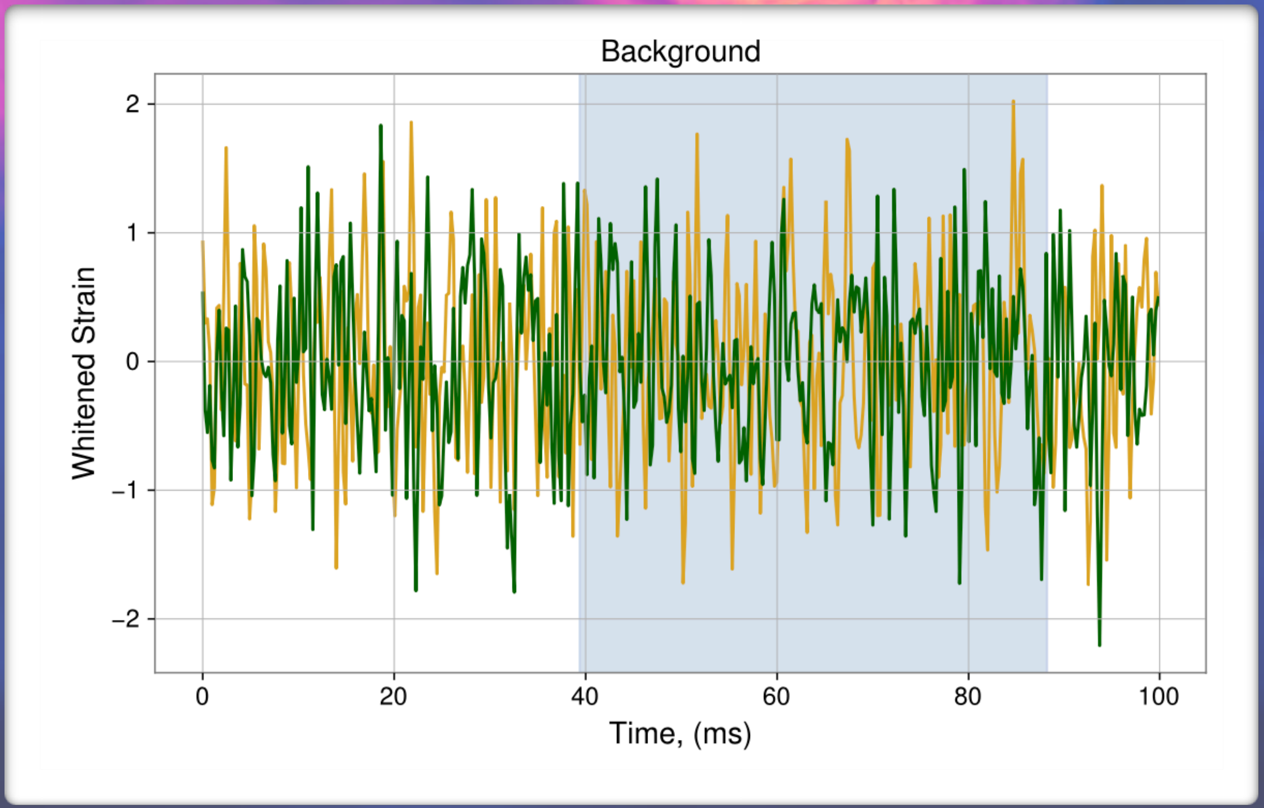
Background dataset

K.Govorkova et al MLST
10.1088/2632-2153/ad3a31

Sampling rate is 4096 Hz, meaning there are 4096 data points recorded every second

The data is divided into segments of 50 milliseconds each, which contains 200 data points (50 milliseconds * 4096 samples/second = 200 samples)

The dimension of the input data is (N, 200, 2), where N represents the number of data segments. The last dimension of 2 corresponds to the data streams from the two LIGO interferometers in Hanford, Washington, and Livingston, Louisiana



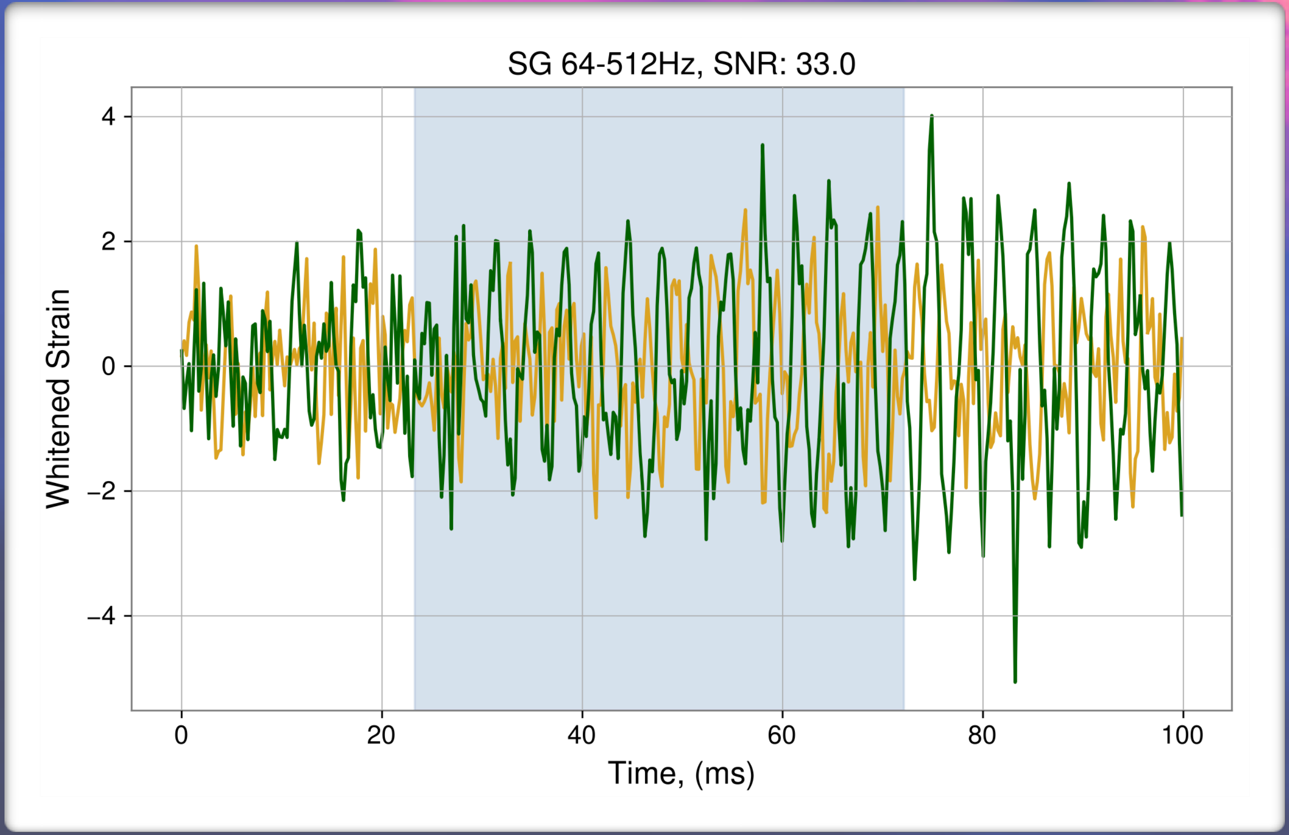
Signal datasets

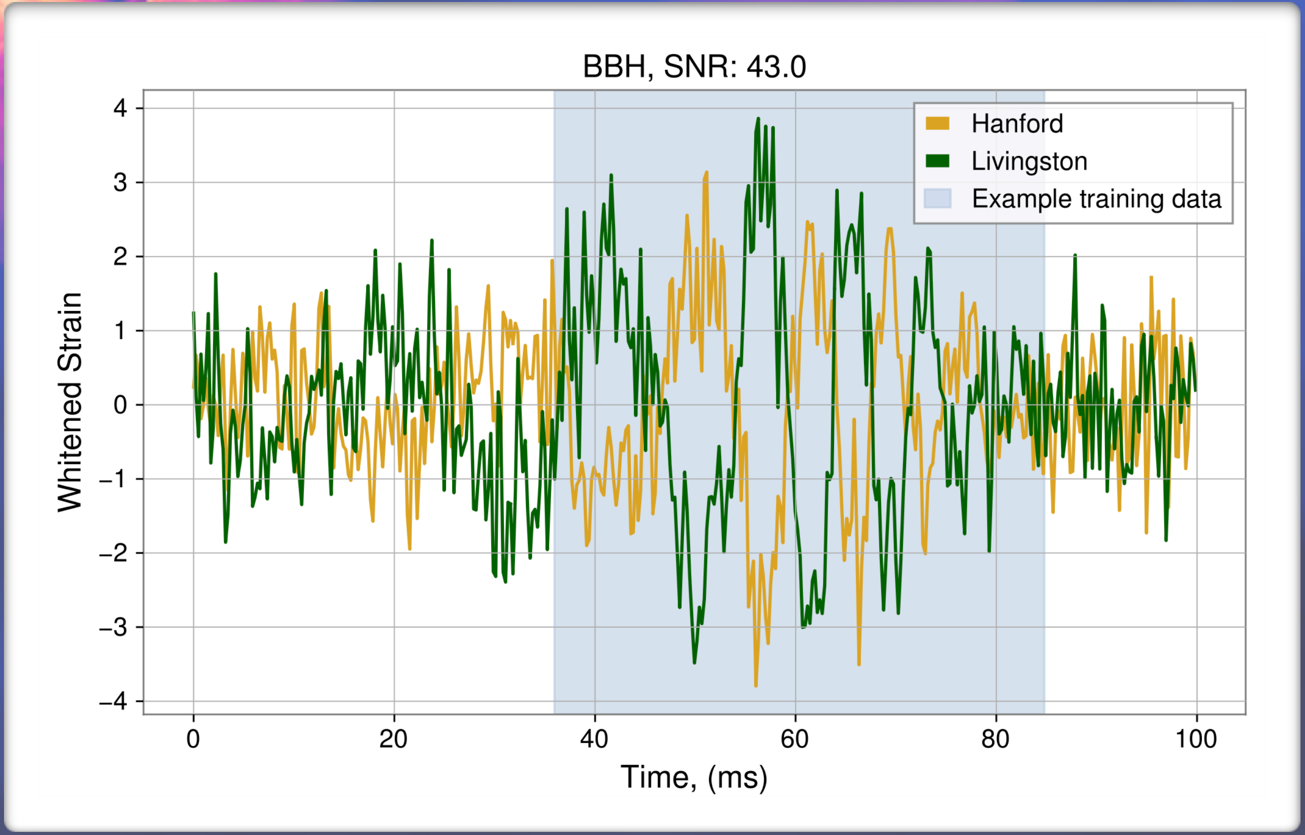
K.Govorkova et al MLST
10.1088/2632-2153/ad3a31

Sampling rate is 4096 Hz, meaning there are 4096 data points recorded every second

The data is divided into segments of 50 milliseconds each, which contains 200 data points (50 milliseconds * 4096 samples/second = 200 samples)

The dimension of the input data is (N, 200, 2), where N represents the number of data segments. The last dimension of 2 corresponds to the data streams from the two LIGO interferometers in Hanford, Washington, and Livingston, Louisiana





Example submission

```
import tensorflow as tf
import os

class Model:
    def __init__(self):
      # You could include a constructor to initialize your model here, but all calls will be made to the load method
        self.clf = None

    def predict(self, X):
      # This method should accept an input of any size (of the given input format) and return predictions appropriately
        b = self.clf.predict(X)

        return [i[0] for i in b]

    def load(self):
      # This method should load your pre-trained model from wherever you have it saved
        with open(os.path.join(os.path.dirname(__file__), 'config.json'), 'r') as file:
            for line in file:
                self.clf = tf.keras.models.model_from_json(line)
        self.clf.load_weights(os.path.join(os.path.dirname(__file__), 'model.weights.h5'))
```

Resources

- The notebook with example https://colab.research.google.com/drive/1hatkYT5Xq6qauDXY6xFrfnGzB66QPsV8?usp=sharing

- The paper with more details and our algorithm MLST 10.1088/2632-2153/ad3a31

- Challenge page with details about the dataset https://www.codabench.org/competitions/2626/

- Any questions should be submitted as a GitHub issue https://github.com/a3d3-institute/HDRchallenge/issues