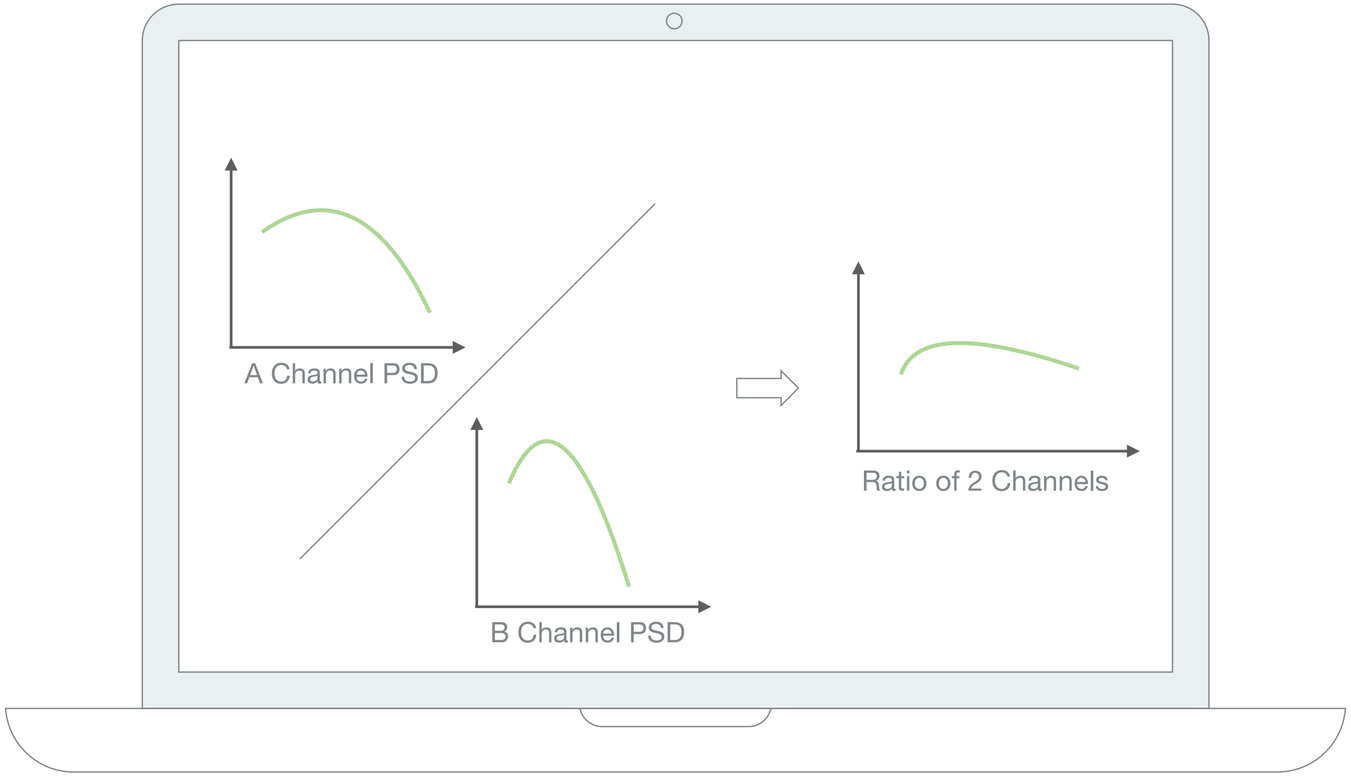
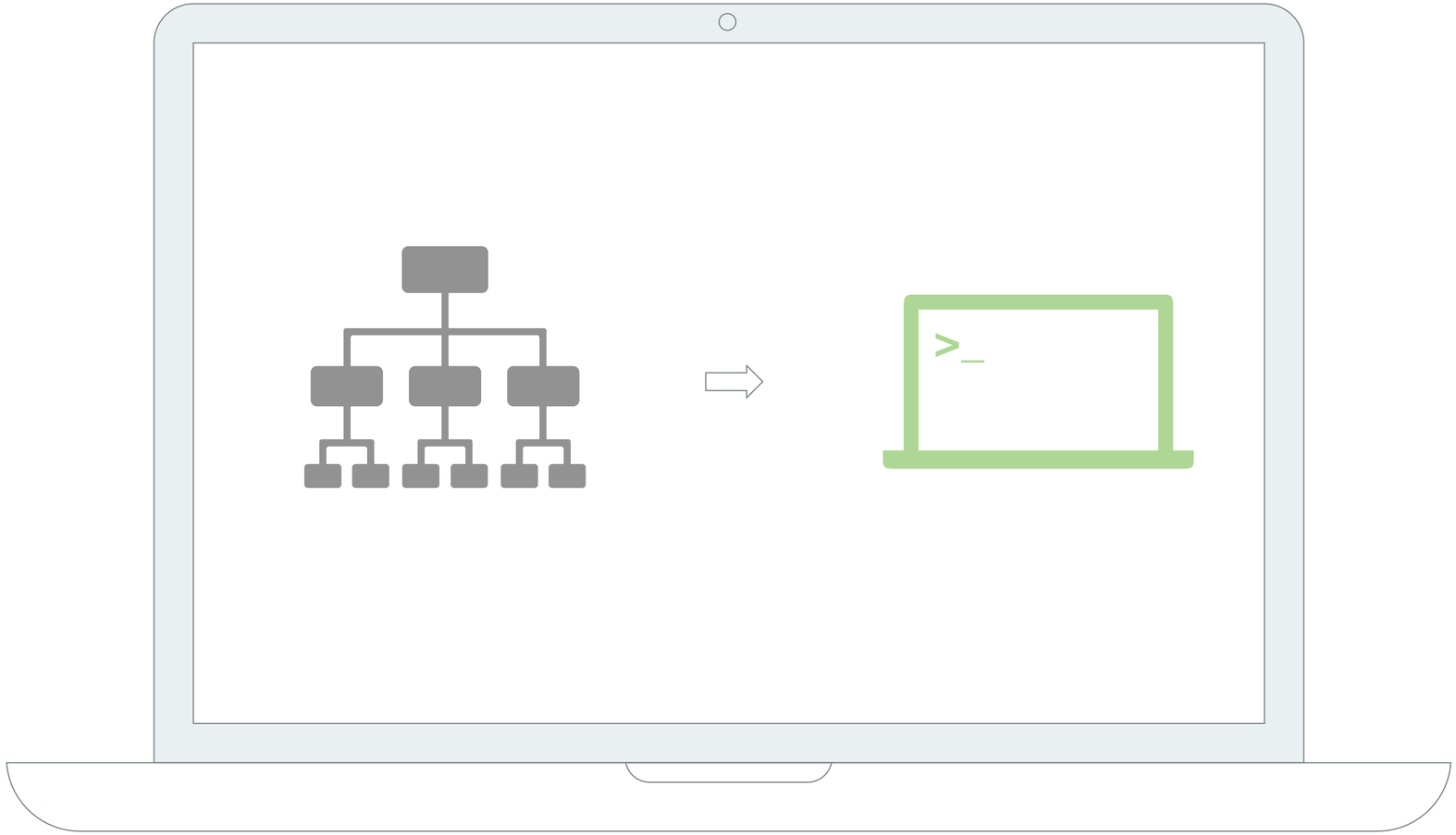
Report for Ratio plot of two channels



- **Requirement**
  - Calculate the ratio of PSD spectrum of two channels
- **Problem**
  - The resolution of each channel is different.
- **Assignment**
  - Interpolation of PSD data for each channel

>>> Code concept structure



Cache read on KISTI cluster

↓

Convert TimeSeries to PSD FrequencySeries

↓

Comparing the maximum value of frequency

↓

Adjust the frequency range to the smaller maximum value

↓

Interpolation Process

>>> Code concept structure

Cache read on KISTI cluster

↓

Convert TimeSeries to PSD FrequencySeries

↓

Comparing the maximum value of frequency

↓

Adjust the frequency range to the smaller maximum value

↓

Interpolation Process

>>> Code concept structure

Cache read on KISTI cluster

↓

Convert TimeSeries to PSD FrequencySeries

↓

Comparing the maximum value of frequency

↓

Adjust the frequency range to the smaller maximum value

↓

Interpolation Process

>>> Code concept structure

Cache read on KISTI cluster

↓

Convert TimeSeries to PSD FrequencySeries

↓

Comparing the maximum value of frequency

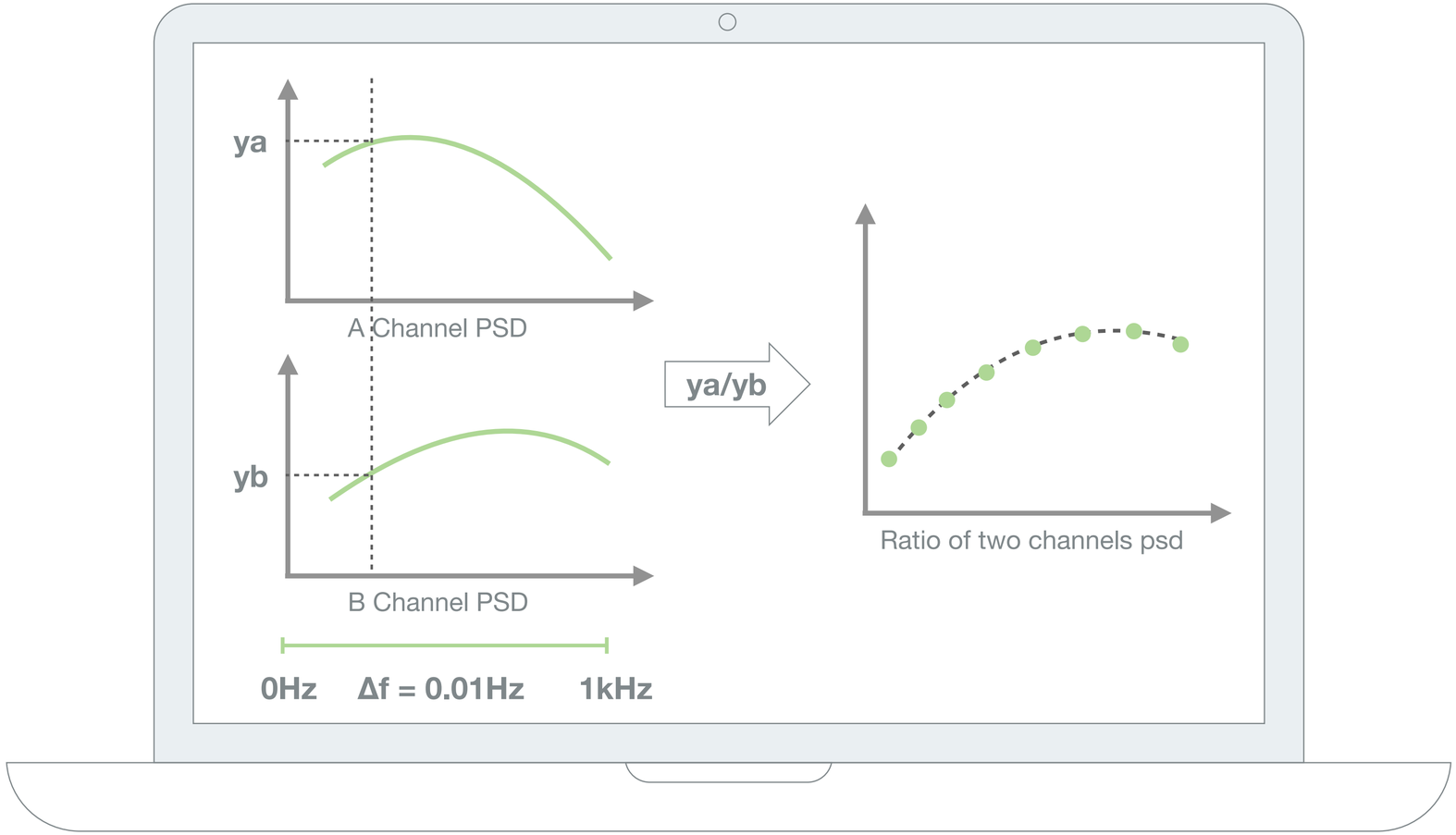
↓

Adjust the frequency range to the smaller maximum value

↓

Interpolation Process

>>> Code concept structure



Cache read on KISTI cluster

↓

Convert TimeSeries to PSD FrequencySeries

↓

Comparing the maximum value of frequency

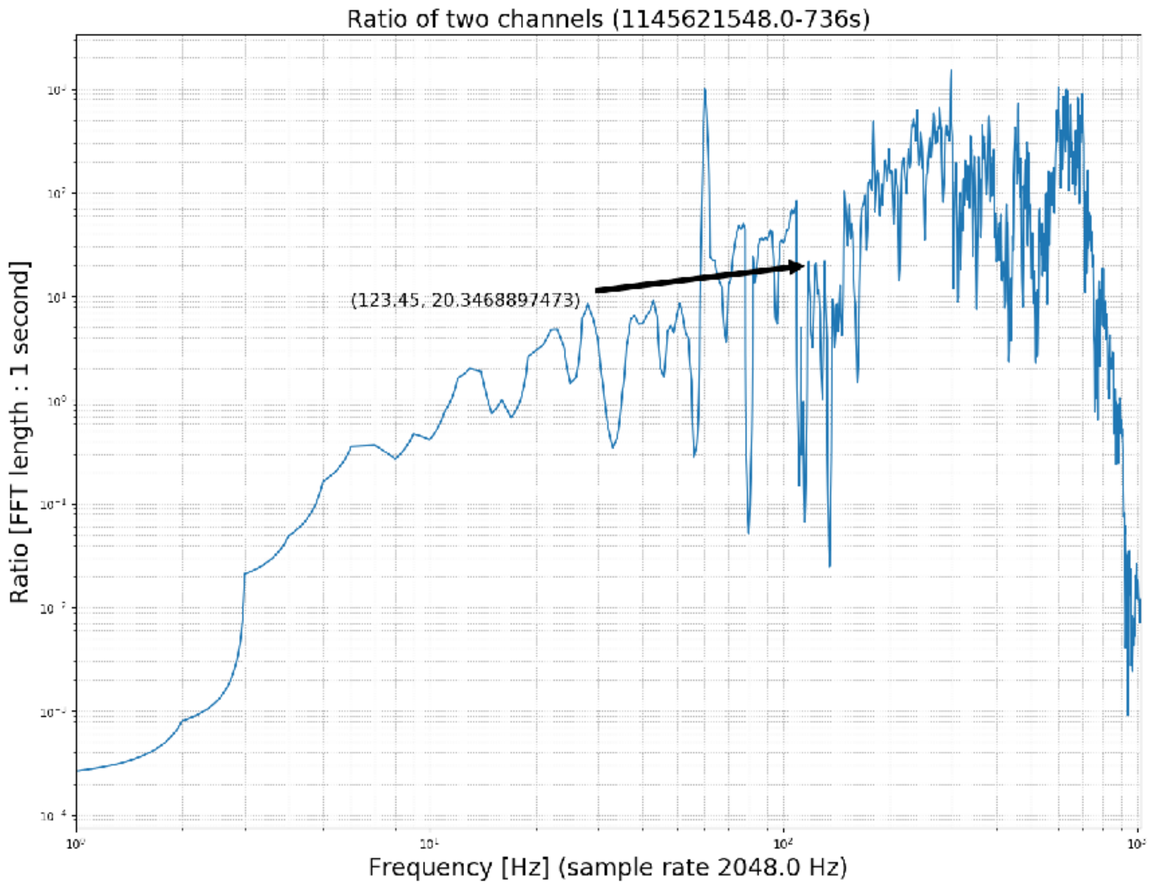
↓

Adjust the frequency range to the smaller maximum value

↓

Interpolation Process

>>> Output Ratio of two channels plot



Start GPS time : 1145621548.0 **Duration :** 736s

A Channel : K1:VIS-EY_TM_OPLEV_SUM_OUT_DQ

A Channel : K1:LSC-MICH_CTRL_CAL_OUT_DQ

Cache read on KISTI cluster

↓

Convert TimeSeries to PSD FrequencySeries

↓

Comparing the maximum value of frequency

↓

Adjust the frequency range to the smaller maximum value

↓

Interpolation Process

>>> Brief View Code structure

Read TimeSeries data from Cache

```
from gwpy.timeseries import TimeSeries

data1 = TimeSeries.read(cache, ach , gst, get, format='lalframe')
data2 = TimeSeries.read(cache, bch , gst, get, format='lalframe')
```

Convert TimeSeries to PSD FrequencySeries

```
data1_psd = data1.psd(1)
data2_psd = data2.psd(1)        # FFT 1 Second
```

>>> Brief View Code structure

PSD data type

```
>>> type(data1_psd)
gwpy.frequencyseries.core.FrequencySeries
```

Change gwpy.FrequencySeries data type to numpy array

```
data1_psd_xlist = []
data1_psd_ylist = []

for i in range(len(data1_psd)) :
    data1_psd_ylist.append(float(str(data1_psd[i])[:-7]))
    data1_psd_xlist.append(i)

import numpy as np

data1_psd_xarr = np.array(data1_psd_xlist)
data1_psd_yarr = np.array(data1_psd_ylist)
```

>>> Brief View Code structure

Interpolation Process

```
from scipy.interpolate import interp1d

g1 = interp1d(data1_psd_xarr, data1_psd_yarr)
g2 = interp1d(data2_psd_xarr, data2_psd_yarr)
```

>>> Brief View Code structure

```
h_xlist = []
h_ylist = []

if len(data1_psd_xarr) >= len(data2_psd_xarr) :
    for a in range(100*(len(data2_psd_xarr)-1)) :
        h_xlist.append(0.01*a)
        h_ylist.append( float(g1(0.01*a)) / float(g2(0.01*a)) )
    hx = np.array(h_xlist)
    hy = np.array(h_ylist)

else :
    for b in range(100*(len(data1_psd_xarr)-1)) :
        h_xlist.append(0.01*b)
        h_ylist.append( float(g1(0.01*b)) / float(g2(0.01*b)) )
    hx = np.array(h_xlist)
    hy = np.array(h_ylist)

h = interp1d(hx, hy)                    # Interpolated ratio function
```

>>> How to use

```
$ ls
kagra_compare_freq.py    run_plot.sh

$ python kagra_compare_freq.py
-ach K1:*** -bch K1:*** -s *** -d 600
-f 123.45
```

- **Argument Parser**

-ach :: Enter first channel name to compare

-bch :: Enter second channel name to compare

-s :: Enter start GPS time

-d :: Enter time duration from star GPS time

-f :: Enter frequency to obtain amplitude value

>>> Current limitations and future assignment

```
$ ls
kagra_compare_freq.py    run_plot.sh

$ python kagra_compare_freq.py
-ach K1:*** -bch K1:*** -s *** -d 600
-f 123.45
```

- **Limitations**
  - Currently **only available** on KISTI Cluster saved iKAGRA data
- **Assignment**
  - **Cache** is required to access data on KAGRA NDS