Using your favourite ground-based toolkits to do space-based analysis!

Big Picture

- A number of people are starting to get involved in LISA, who have extensive experience in LVK.
- When we started getting involved ~4 years ago, very little of the LISA codebase was publicly available and it was hard to understand what already existed.
- This motivated us with a simple thought: People are familiar with toolkits used for LVK analyses (PyCBC and Bilby). What if these tools can also be used for LISA (and Taiji,TianQin,...) analyses?
- The LISA analysis toolkit landscape looks a lot better today, with much more code open-source, and remaining close-source packages also considering the move.
- However, we still feel there is a strong benefit that tools people are familiar with can also be used for LISA analysis.
- We'll discuss PyCBC here, and Charlie will talk about the overlapping effort to do the same within Bilby.

LISA with PyCBC

the PyCBC LISA team

Alex Nitz

Ian Harry

Connor Weaving

Shichao Wu

Collin Capano

Gareth C. Davies

Michael J. Williams

Han Wang

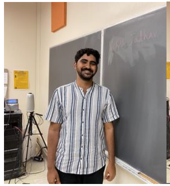
Laura Nuttall

Tito Dal Canton

Charlie Hoy

Xisco Jimenez Forteza

Alex Correia



Vikas Jadhav Y

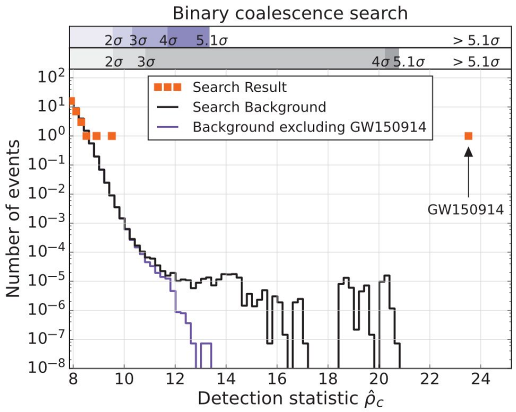
Labani Roy

Lalit Pathak

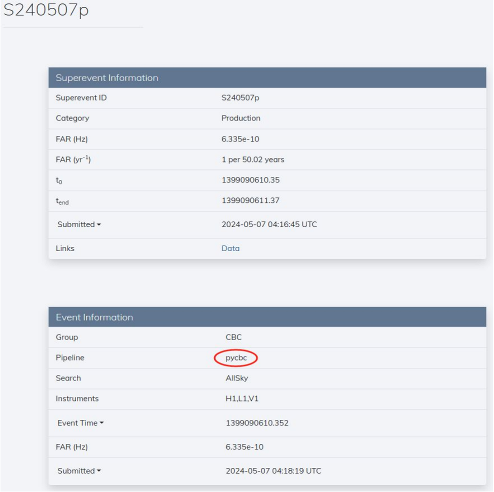
Abhishek Sharma

PyCBC

- PyCBC was used to make the “5-sigma significance plot” for GW150914
- PyCBC is used by LVK Collaboration to routinely find new CBC signals
- PyCBC is already heavily used in studies for next-generation ground-based detectors, such as ET and CE
- Now PyCBC and PyCBC Inference, can be or will be used for LISA, TianQin, Taiji, and DECIGO



10.1103/PhysRevLett.116.061102
10.1103/PhysRevD.93.122003



S240507p

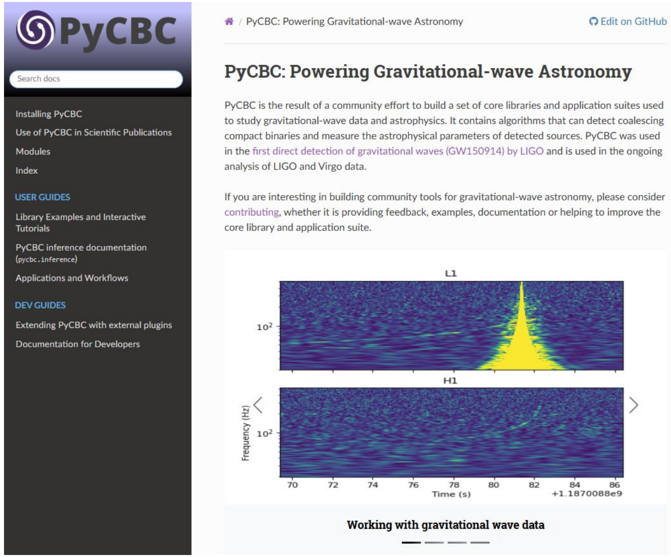
Superevent Information	
Superevent ID	S240507p
Category	Production
FAR (Hz)	6.335e-10
FAR (yr^{-1})	1 per 50.02 years
t_0	1399090610.35
t_{end}	1399090611.37
Submitted	2024-05-07 04:16:45 UTC
Links	Data

Event Information	
Group	CBC
Pipeline	pycbc
Search	AllSky
Instruments	H1,L1,V1
Event Time	1399090610.352
FAR (Hz)	6.335e-10
Submitted	2024-05-07 04:18:19 UTC

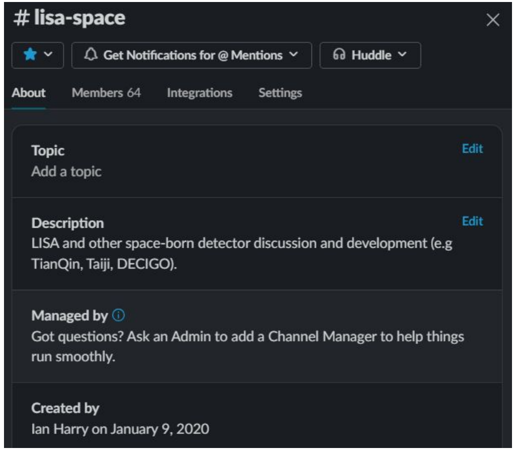
GraceDB | S240507p (ligo.org)

Core Principles

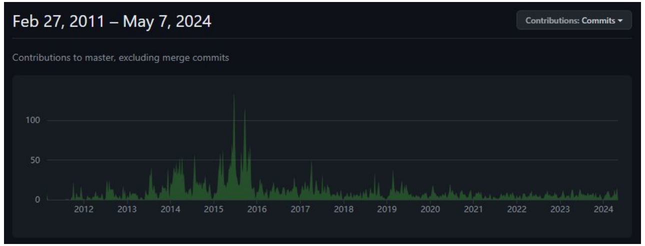
- Friendly and Easy-to-use
- Community open development model
- Extensibility
- Well-documented



https://pycbc.org/pycbc/latest/html/index.html



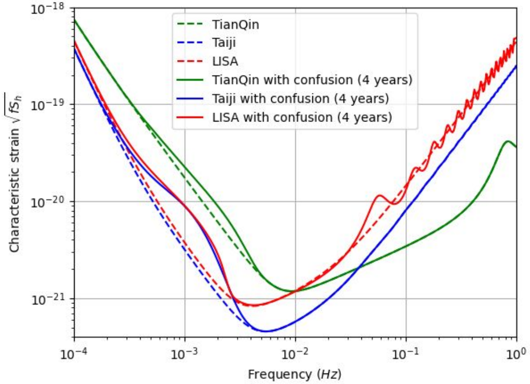
https://gw-astro.slack.com/archives/CSJ4B9022

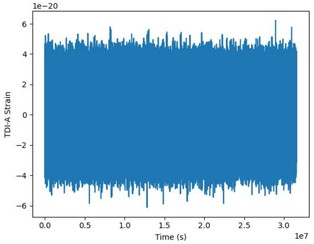


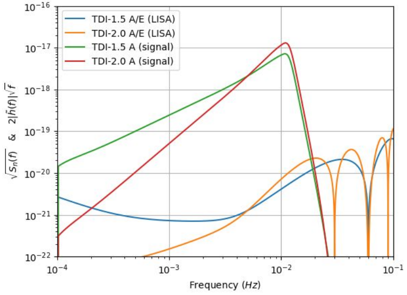
https://github.com/gwastro/pycbc/graphs/contributors

Key Packages

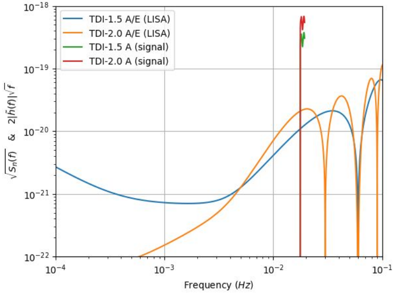
- psd: supports sensitivity curves and TDI-1.5/2.0 PSDs for LISA, Taiji, TianQin with or without DWD confusion noise
- noise: simulates (non-)stationary noise from psd module
- waveform: TD or FD, with or without detector response
- detector: flexible site and orientation for ground-based detectors, currently only supports LISA response for space-borne
- inject: inject SOBHB or SMBHB signals into noise
- inference: performs Bayesian inference
- coordinates: easily convert between SSB/LISA/GEO frame
- distributions: built-in or external prior
- sampler: Emcee, PTEmcee, Dynesty, Ultranest, Epsie, cpnest, Multinest, Snowline, nessai
- population: population inference







SMBHB



SOBHB

PyCBC waveform package

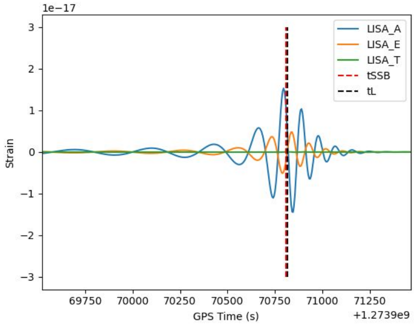
- The standard easy-to-use python interface for waveform generation
  - Provides the high level easy-to-use interface for general users and higher level codes
  - Supports any underlying waveform generation code with a common standardized interface
  - Example of supported waveform sources (that I know of)
    - BBHx
    - TEOBNRResumS
    - GWSurrogate
    - SEOBNRE
    - and naturally lalsimulation
    - Commonly used for non-GR waveform modifications (interface is very extensible)
      - full waveform BH spectroscopy
      - birefringence

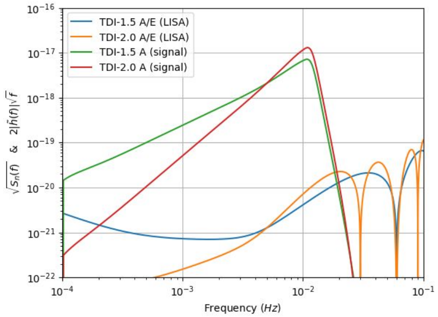
https://pycbc.org/pycbc/latest/html/waveform.html https://pycbc.org/pycbc/latest/html/waveform_plugin.html

How to generate LISA waveforms in PyCBC

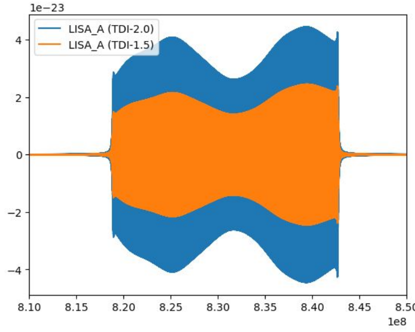
```
import matplotlib.pyplot as plt
from pycbc.waveform import get_td_det_waveform_from_fd_det
from pycbc.coordinates import TIME_OFFSET_20_DEGREES, lisa_to_ssb


# set parameters
params = {}
params['tdi'] = 1.5
params['ref_frame'] = 'LISA'
params['approximant'] = 'BBHX_PhenomD'
params['coa_phase'] = 0.0
params['mass1'] = 1e6
params['mass2'] = 5e5
params['spin1z'] = 0.0
params['spin2z'] = 0.0
params['distance'] = 410
params['inclination'] = 0.0
params['eclipticlongitude'] = 5.4
params['eclipticlatitude'] = 0
params['polarization'] = 0.0
params['tc'] = 1273970818
params['t_obs_start'] = 31558149.763545603
params['f_lower'] = 1e-4
params['f_ref'] = 1e-4
params['f_final'] = 0.1
params['delta_t'] = 1/0.2
params['t_offset'] = TIME_OFFSET_20_DEGREES

# generate time-domain TDI waveform
bbhx_td = get_td_det_waveform_from_fd_det(ifos=['LISA_A','LISA_E','LISA_T'], **params)
# get the merge time in SSB frame as a comparison
tSSB, _, _, _ = lisa_to_ssb(params['tc'], params['eclipticlongitude'],
                            params['eclipticlatitude'], params['polarization'],
                            params['t_offset'])

plt.plot(bbhx_td['LISA_A'].sample_times, bbhx_td['LISA_A'], label='LISA_A')
plt.plot(bbhx_td['LISA_E'].sample_times, bbhx_td['LISA_E'], label='LISA_E')
plt.plot(bbhx_td['LISA_T'].sample_times, bbhx_td['LISA_T'], label='LISA_T')

plt.vlines(x=tSSB, ymin=-3e-17, ymax=3e-17, colors='red', linestyles='dashed', label='tSSB')
plt.vlines(x=params['tc'], ymin=-3e-17, ymax=3e-17, colors='k', linestyles='dashed', label='tL')
plt.xlim(params['tc']-1296, params['tc']+648)
plt.xlabel("GPS Time (s)")
plt.ylabel("Strain")
plt.legend()
plt.show()
```

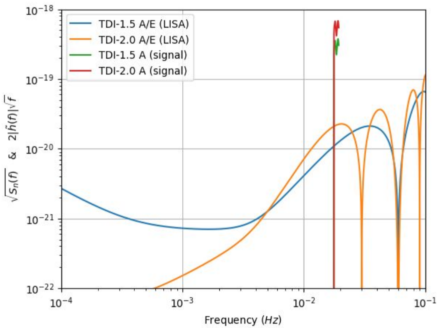
IMR SMBHB waveform





narrow band SOBHB waveform





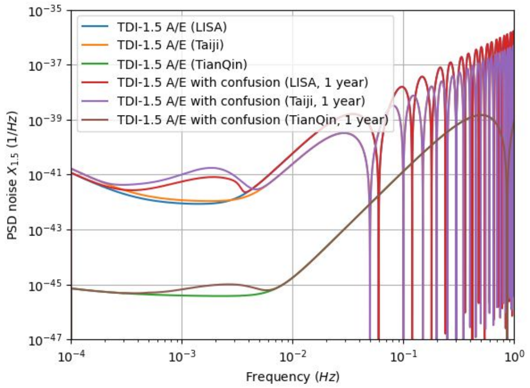
Getting PyCBC to use some new waveform model

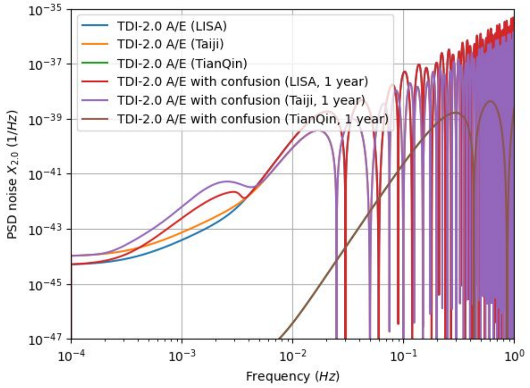
- PyCBC supports custom “plugin” waveforms
  - waveforms: write your own code that will generate any waveform
- You create your own package that defines your waveform & install it
- PyCBC will detect your waveform at run time and allow you to use it
- No changes to the PyCBC source code are needed
  - Custom packages just need to use API PyCBC understands
- You can publish/distribute your custom package independently
- For details, see the PyCBC Tutorials <https://github.com/gwastro/PyCBC-Tutorials/tree/master>
  - Tutorial 7: custom waveforms

Available PSDs for space-borne detectors

Here you can find all implemented PSDs and sensitivity curves:
https://pycbc.org/pycbc/latest/html/pycbc.psd.html#module-pycbc.psd.analytical_space

```python
import numpy as np
import matplotlib.pyplot as plt
from pycbc.psd.analytical_space import *

flow = 1e-4
delta_f = 1.0 / (3600*24*31)
fs = 2
flen = int(fs/delta_f)//2 + 1

psd_tdi_1p5_lisa_A = analytical_psd_lisa_tdi_AE(flen, delta_f, flow,
                len_arm=2.5e9, acc_noise_level=3e-15, oms_noise_level=15e-12, tdi='1.5')
psd_tdi_1p5_taiji_A = analytical_psd_taiji_tdi_AE(flen, delta_f, flow,
                len_arm=3e9, acc_noise_level=3e-15, oms_noise_level=8e-12, tdi='1.5')
psd_tdi_1p5_tianqin_A = analytical_psd_tianqin_tdi_AE(flen, delta_f, flow,
                len_arm=np.sqrt(3)*1e8, acc_noise_level=1e-15, oms_noise_level=1e-12, tdi='1.5')
psd_tdi_1p5_lisa_A_confusion_1 = analytical_psd_lisa_tdi_AE_confusion(flen, delta_f, flow,
                len_arm=2.5e9, acc_noise_level=3e-15, oms_noise_level=15e-12, duration=1.0, tdi="1.5")
psd_tdi_1p5_taiji_A_confusion_1 = analytical_psd_taiji_tdi_AE_confusion(flen, delta_f, flow,
                len_arm=3e9, acc_noise_level=3e-15, oms_noise_level=8e-12, duration=1.0, tdi="1.5")
psd_tdi_1p5_tianqin_A_confusion_1 = analytical_psd_tianqin_tdi_AE_confusion(flen, delta_f, flow,
                len_arm=np.sqrt(3)*1e8, acc_noise_level=1e-15, oms_noise_level=1e-12, duration=1.0, tdi="1.5")

plt.loglog(psd_tdi_1p5_lisa_A.sample_frequencies, psd_tdi_1p5_lisa_A, label='TDI-1.5 A/E (LISA)')
plt.loglog(psd_tdi_1p5_taiji_A.sample_frequencies, psd_tdi_1p5_taiji_A, label='TDI-1.5 A/E (Taiji)')
plt.loglog(psd_tdi_1p5_tianqin_A.sample_frequencies, psd_tdi_1p5_tianqin_A, label='TDI-1.5 A/E (TianQin)')

plt.loglog(psd_tdi_1p5_lisa_A_confusion_1.sample_frequencies, psd_tdi_1p5_lisa_A_confusion_1,
            label='TDI-1.5 A/E with confusion (LISA, 1 year)')
plt.loglog(psd_tdi_1p5_taiji_A_confusion_1.sample_frequencies, psd_tdi_1p5_taiji_A_confusion_1,
            label='TDI-1.5 A/E with confusion (Taiji, 1 year)')
plt.loglog(psd_tdi_1p5_tianqin_A_confusion_1.sample_frequencies, psd_tdi_1p5_tianqin_A_confusion_1,
            label='TDI-1.5 A/E with confusion (TianQin, 1 year)')

plt.xlabel(r'Frequency ($Hz$)')
plt.ylabel(r'PSD noise $X_{1.5}$ ($1/{Hz}$)')
plt.xlim([1e-4, 1])
plt.ylim([1e-47, 1e-35])
plt.legend(loc="upper left")
plt.grid()
plt.show()
```





Bayesian inference with PyCBC

- Parameter estimation is performed using the inference module in PyCBC
- Ties together the waveform generation, data conditioning, stochastic sampling in order to produce posterior distributions on parameters
  - External samplers are used (e.g., dynesty)
- Several “Model” classes supported:
  - Gaussian Noise, Marginalized Phase, Marginalized Polarization, Marginalized Time, Marginalized Higher Mode Phase, Heterodyne / Relative Binning, Gated Gaussian Noise, Hierarchical, Multiple Signal, Multiband, custom user-defined models
- Models make different assumptions about waveforms and detectors analyzed. Allows for analyzing different detectors, waveform complexity, tests of GR, etc. under single framework.
- In principle these can also work for space-based detectors.
- Custom models supported using similar API as for waveforms
  - https://colab.research.google.com/github/gwastro/pycbc-tutorials/blob/master/tutorial/inference_9_AddingCustomModels.ipynb

Using PyCBC to perform LISA analysis

Screenshots from https://pycbc.org/pycbc/latest/html/inference/examples/lisa_smbhb_inj_pe.html

First, we use the following configuration file to define the parameters of our SMBHB injection, we use the same parameters from the SMBHB signal in LISA parameter estimation for simulated SMBHB from LDC example:

```
[variable_params]

[static_params]
; This assumes all those values are in LISA frame.
; You can set "ref_frame = SSB", but then you should also add it to
; "static_params" section in PE .ini file.
ref_frame = LISA
approximant = BBHX_PhenomD
; You can use "1.5" or "2.0" for TDI.
; Please use the same TDI version for PSD and static_params in the PE .ini file.
tdi = 1.5
mass1 = 1015522.4376
mass2 = 796849.1091
spin1z = 0.597755394865021
spin2z = 0.36905807298613247
distance = 17758.367941273442
inclination = 1.5970175301911231
coa_phase = 4.275929308696054
eclipticlongitude = 5.4431083771985165
eclipticlatitude = -1.2734504596198182
polarization = 0.22558110042980073
tc = 4799624.274911478
t_obs_start = 31536000
; Put LISA behind the Earth by ~20 degrees.
t_offset = 7365189.431698299
f_lower = 1e-4
f_ref = 1e-4
f_final = 0.1
```

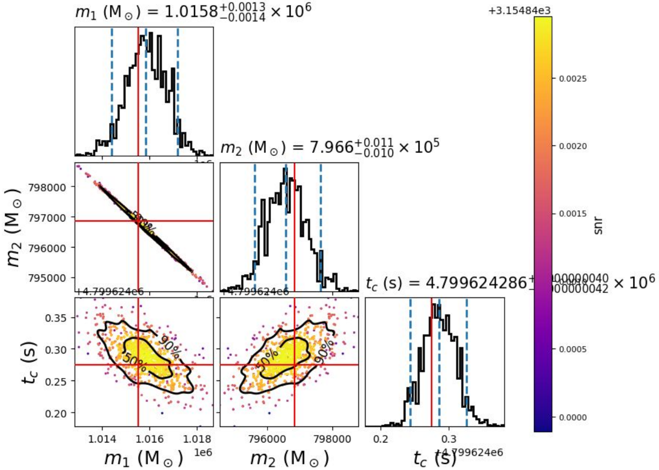
Download

Then we run the following bash script to create a .hdf file that contains same information:

```
#!/bin/sh
pycbc_create_injections --verbose \
        --config-files injection_smbhb.ini \
        --ninjections 1 \
        --seed 10 \
        --output-file injection_smbhb.hdf \
        --variable-params-section variable_params \
        --static-params-section static_params \
        --dist-section prior \
        --force
```

```
[data]
instruments = LISA_A LISA_E LISA_T
trigger-time = 4800021.15572853
analysis-start-time = -4800021
analysis-end-time = 26735979
pad-data = 0
sample-rate = 0.2
fake-strain = LISA_A:analytical_psd_lisa_tdi_AE LISA_E:analytical_psd_lisa_tdi_AE LISA_T:analytical_psd_
; fake-strain-extra-args = LISA_A:len_arm:2.5e9 LISA_A:acc_noise_level:2.4e-15 LISA_A:oms_noise_level:7.
fake-strain-extra-args = len_arm:2.5e9 acc_noise_level:2.4e-15 oms_noise_level:7.9e-12 tdi:1.5
fake-strain-seed = LISA_A:100 LISA_E:150 LISA_T:200
fake-strain-flow = 0.0001
fake-strain-sample-rate = 0.2
fake-strain-filter-duration = 31536000
psd-estimation = median-mean
psd-inverse-length = 267840
invpsd-trunc-method = hann
psd-segment-length = 267840
psd-segment-stride = 133920
psd-start-time = -4800021
psd-end-time = 26735979
channel-name = LISA_A:LISA_A LISA_E:LISA_E LISA_T:LISA_T
injection-file = injection_smbhb.hdf

[model]
name = relative
low-frequency-cutoff = 0.0001
high-frequency-cutoff = 0.1
epsilon = 0.01
mass1_ref = 1015522.4376
mass2_ref = 796849.1091
tc_ref = 4799624.274911478
spin1z_ref = 0.597755394865021
spin2z_ref = 0.36905807298613247

[variable_params]
mchirp =
q =
tc =

[static_params]
; Change it to "ref_frame = SSB", if you use SSB frame in injection file.
ref_frame = LISA
approximant = BBHX_PhenomD
; You can use "1.5" or "2.0" for TDI.
; Please use the same TDI version for PSD and injection file.
tdi = 1.5
coa_phase = 4.275929308696054
eclipticlongitude = 5.4431083771985165
eclipticlatitude = -1.2734504596198182
polarization = 0.22558110042980073
spin1z = 0.597755394865021
spin2z = 0.36905807298613247
distance = 17758.367941273442
inclination = 1.5970175301911231
t_obs_start = 31536000
f_lower = 1e-4
; Put LISA behind the Earth by ~20 degrees.
t_offset = 7365189.431698299

[prior-mchirp]
name = uniform
min-mchirp = 703772.7245316936
max-mchirp = 860166.6633165143

[prior-q]
name = uniform
min-q = 1.1469802543574181
max-q = 1.401864755325733

[prior-tc]
name = uniform
min-tc = 4798221.15572853
max-tc = 4801821.15572853

[waveform_transforms-mass1+mass2]
name = mchirp_q_to_mass1_mass2

[sampler]
name = dynesty
dlogz = 0.1
nlive = 150
```

Heterodyne PE example for SMBHB

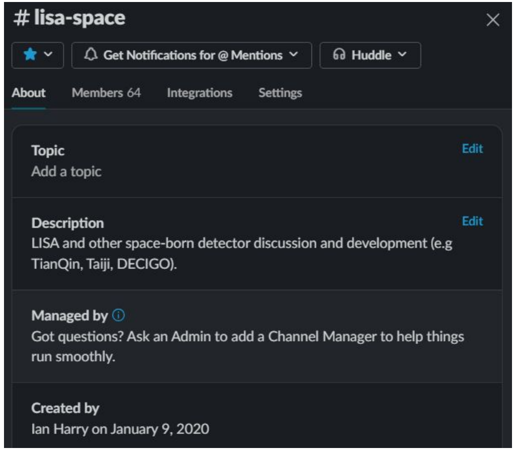


Ongoing projects in PyCBC-Space

- LISA+3G multiband parameter estimation of SOBHB
- Pre-merger detection and inference for SMBHB in LISA
- Higher order modes parameter estimation for SMBHB by LISA
- Parameter estimation includes eccentricity of SOBHB
- More space-borne detectors support
- More flexible detector response models
- Support for FastLISAResponse package
- Support for FastEMRIWaveforms package
- Galactic BNS detectability by LISA
- Line-of-sight acceleration of SOBHB near SMBH
- Meshfree likelihood model for SMBHB

How to join us?

- Scan the QR code below to join our “lisa-space” slack channel. We have our monthly Zoom telecon here. You can also discuss anything related to PyCBC and LISA/Taiji/TianQin/DECIGO in the channel.



https://gw-astro.slack.com/archives/CSJ4B9022