`bilby` in space: Bayesian inference for transient gravitational-wave signals observed with LISA

Charlie Hoy, Laura K. Nuttall

Connecting LVK/LISA Waveform Infrastructures

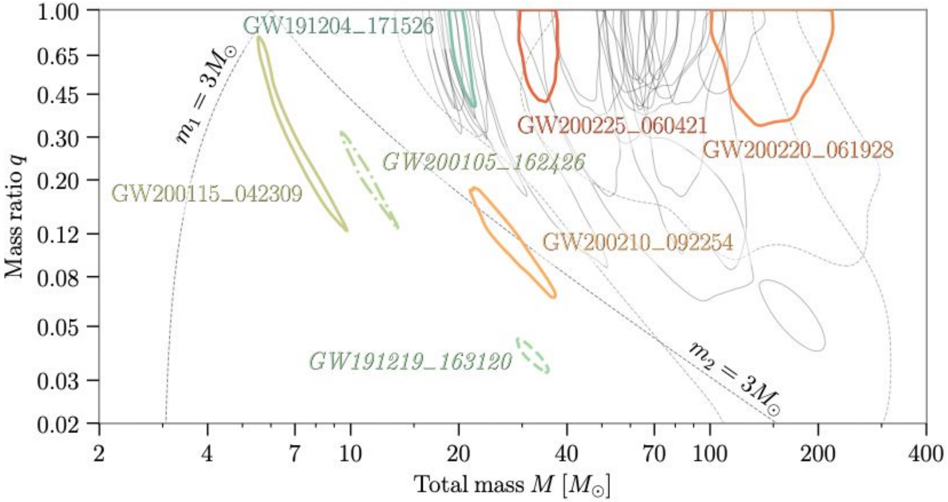
08/05/2024

MNRAS 529,3,3052–3059 (2024)

bilby

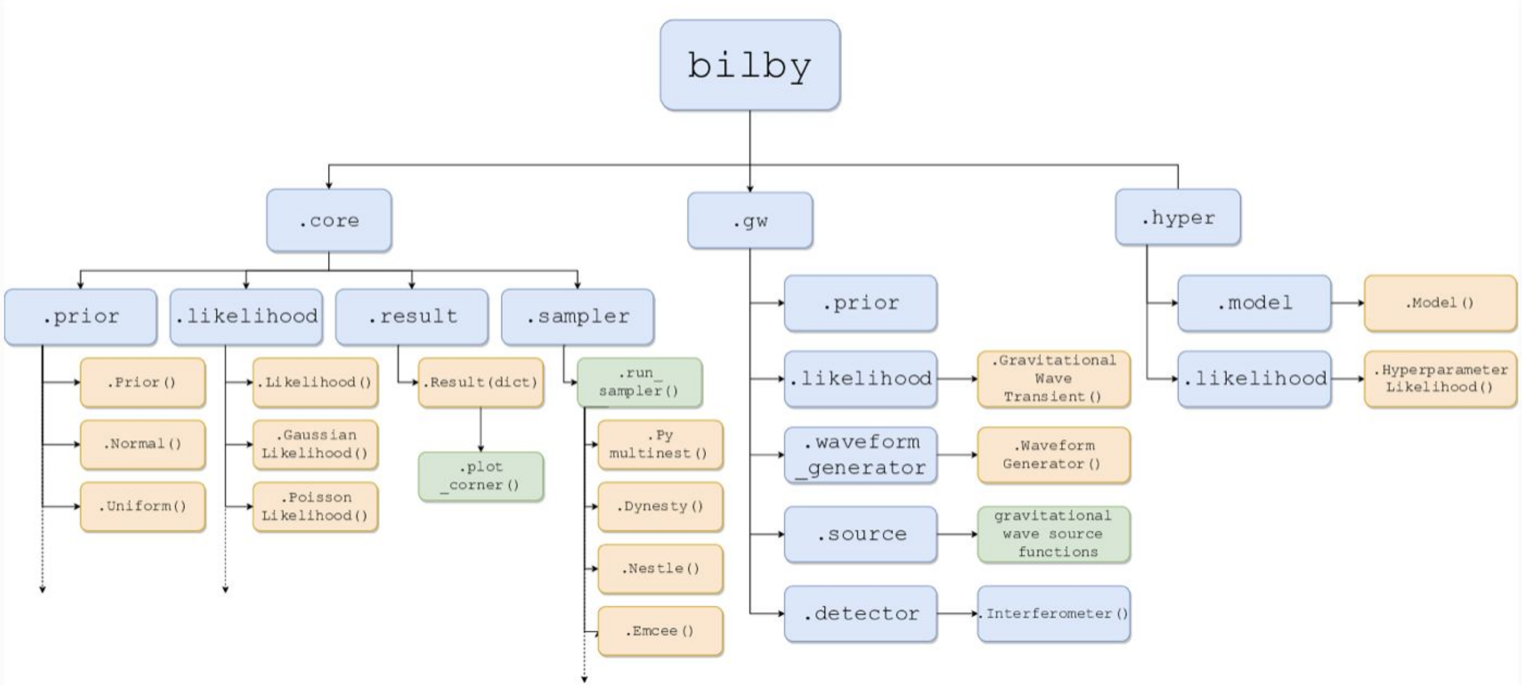
"Bilby: a user-friendly Bayesian inference library. The aim of bilby is to provide a user-friendly interface to perform parameter estimation"

Ashton *et al.* ApJS 241 27 (2019)



Abbott *et al.* PRX 13, 041039 (2023)

bilby



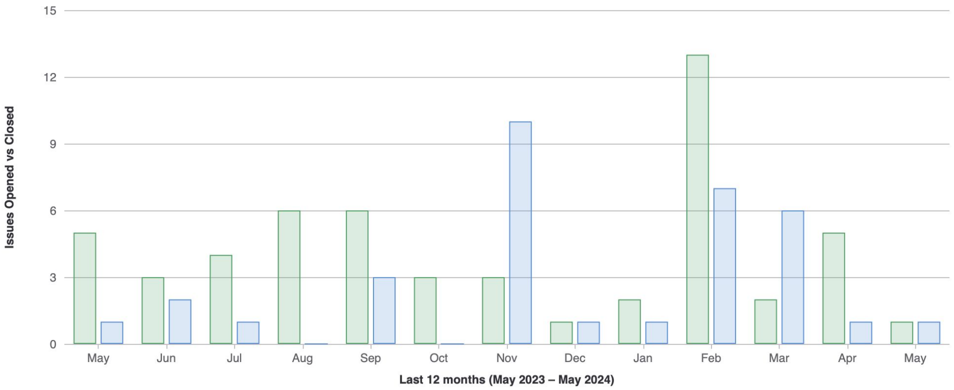
https://lscsoft.docs.ligo.org/bilby/code-overview.html

Active use and development

Bilby: A user-friendly Bayesian inference library for gravitational-wave astronomy

Gregory Ashton,[1,2,*] Moritz Hübner,[1,2,†] Paul D. Lasky,[1,2,‡] Colm Talbot,[1,2,§] Kendall Ackley,[1,2] Sylvia Biscoveanu,[3,1,2] Qi Chu,[4,5] Atul Divarkala,[6,1,2] Paul J. Easter,[1,2] Boris Goncharov,[1,2] Francisco Hernandez Vivanco,[1,2] Jan Harms,[7,8] Marcus E. Lower,[9,10,1] Grant D. Meadors,[1,2] Denyz Melchor,[11,1,2] Ethan Payne,[1,2] Matthew D. Pitkin,[12] Jade Powell,[9,10] Nikhil Sarin,[1,2] Rory J. E. Smith,[1,2] and Eric Thrane[1,2]

Cited more than 700 times



Issues are regularly opened and closed.
~ 100 contributors worldwide

https://git.ligo.org/lscsoft/bilby/-/analytics/issues_analytics

Interface to multiple samplers

Nested Samplers

- Dynesty: `bilby.core.sampler.dynesty.Dynesty`
- Nestle `bilby.core.sampler.nestle.Nestle`
- CPNest `bilby.core.sampler.cpnest.Cpnest`
- PyMultiNest `bilby.core.sampler.pymultinest.Pymultinest`
- PyPolyChord `bilby.core.sampler.polychord.PyPolyChord`
- UltraNest `bilby.core.sampler.ultranest.Ultranest`
- DNest4 `bilby.core.sampler.dnest4.DNest4`
- Nessai `bilby.core.sampler.nessai.Nessai`

MCMC samplers

- bilby-mcmc `bilby.bilby_mcmc.sampler.Bilby_MCMC`
- emcee `bilby.core.sampler.emcee.Emcee`
- ptemcee `bilby.core.sampler.ptemcee.Ptemcee`
- pymc `bilby.core.sampler.pymc.Pymc`
- zeus `bilby.core.sampler.zeus.Zeus`

Bilby interfaces with a large (and growing) number of off-the-shelf samplers, but Dynesty has been extensively used in other studies. For details see: https://lscsoft.docs.ligo.org/bilby/samplers.html

bilby_lisa

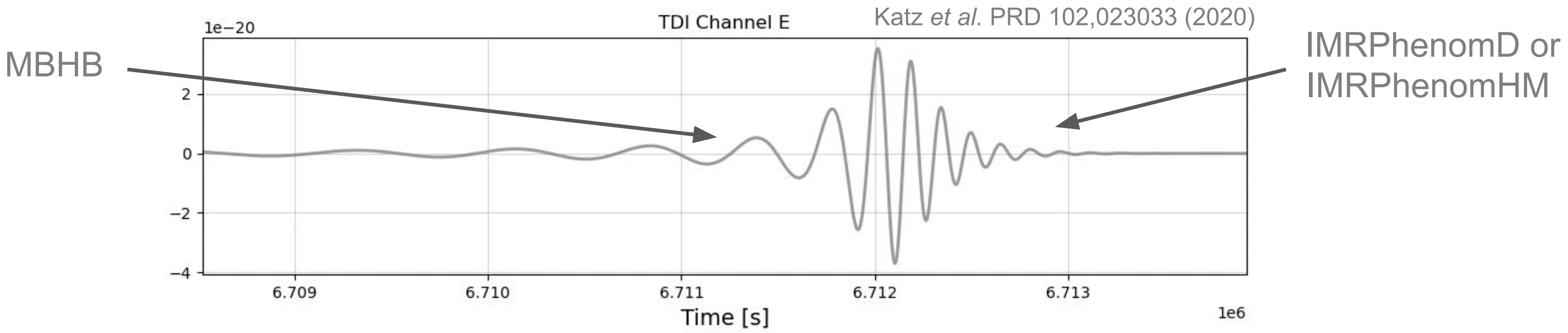
- bilby_lisa is an extension of bilby, bilby_pipe, parallel_bilby which adds functionality for LISA data analysis

- bilby_lisa adds the LISA detector, and custom source models to bilby

- All the functionality in bilby can be used with bilby_lisa, e.g. marginalizations (phase, time, distance) and accelerated likelihoods (heterodyned, ROQ, multibanded)

Implemented LISA models

`bilby_lisa` can generate LISA waveforms via BBHx, a package that generates GW polarizations, projects them onto LISA to form an arm response and generates waveforms in the TDI channels, via: `bilby_lisa.source.lisa_binary_black_hole`.



Interfacing with other models

1 — Generate hp/hc

Generate hp/hc with LALSimulation via the existing bilby.gw.source.lal_binary_black_hole.

2a — Apply LISA response

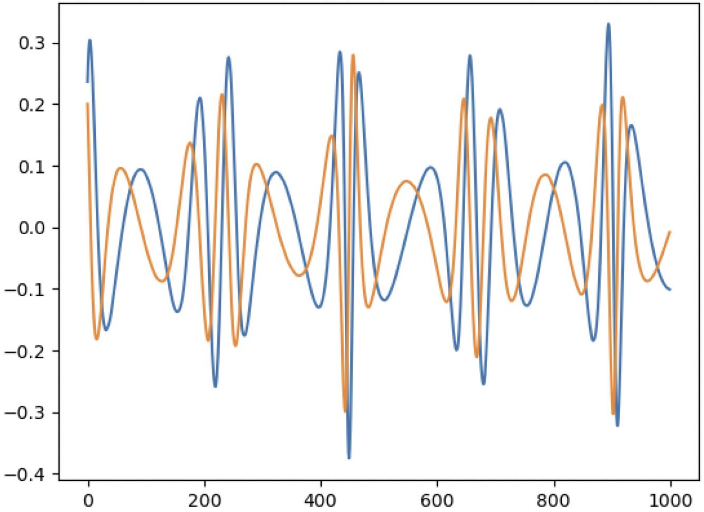
Use functionality in BBHx to apply LISA response to hp/hc from LALSimulation

2b — Return TDI representation

Use functionality in BBHx to return TDI representation for hp/hc from LALSimulation

`bilby` already interfaces with the "new waveforms interface" (see Frank Ohme's talk 07/05/2024) via: `bilby.gw.source.gwsignal_binary_black_hole`. `bilby_lisa` could also interface with the "new waveforms interface" in the above workflow. Other models can be added to the `bilby_lisa` source code, or via plugins (I am happy to add this functionality if useful).

Interfacing with other models



I am working with Jonathan Thompson to interface with `FastEMRIWaveforms` (see Lorenzo Speri's talk on 08/05/2024) via:

`bilby_lisa.source.lisa_extreme_mass_ratio_inspiral`

Generating waveforms

Or create your favourite source model!

```
params = {} # dictionary of parameters

generator = bilby.gw.waveform_generator.WaveformGenerator(
    duration=31536000.0, sampling_frequency=sampling_frequency,
    start_time=0., frequency_domain_source_model=bilby_lisa.source.lisa_binary_black_hole,
    waveform_arguments={
        "waveform_approximant": “BBHx_IMRPhenomD”, "reference_frame": "LISA",
        "ifos": ["LISA_A", "LISA_E", "LISA_T"]
    }
)
ht = generator.frequency_domain_strain(params)["LISA_A"]
freqs = generator.frequency_array

h = generator.time_domain_strain(params)["LISA_A"]
times = generator.time_array
```

pip install bilby_lisa

bilby LISA 1.0.0a4.dev2 documentation

Search

v: latest

bilby LISA 1.0.0a4.dev2 documentation

Contents:

Installation instructions

`bilby_lisa` can be installed through a variety of methods, see below. Independent of the method chosen, we recommend installing `bilby_lisa` within a conda environment. For speed, we recommend creating an environment with mamba. `bilby_lisa` can be installed with,

mamba | PyPI | From source

```
$ mamba env create --name bilby-lisa --file environment.yml
```

where `environment.yml` can be downloaded `here`.

Warning

As part of this installation, non-released versions of `bilby`, `bilby_pipe` and `parallel_bilby` were installed. This is because we are waiting for required code to be merged into the main `bilby`, `bilby_pipe` and `parallel_bilby` code bases. Please see the following merge requests for details:

- bilby!1314
- bilby_pipe!583
- bilby_pipe!586
- parallel_bilby!137

Public documentation available at:

https://bilby-lisa.readthedocs.io

Data analysis with `bilby_lisa`

```
################################################################################
## Detector arguments
################################################################################
detectors=[LISA]
tdi=[A, E, T]
duration=2628000.0 # Approximately 30 days of data

################################################################################
## Job submission arguments
################################################################################
analysis_executable_parser=bilby_lisa.bilby_pipe.create_parser

################################################################################
## Waveform arguments
################################################################################
waveform-approximant=BBHx_IMRPhenomD
waveform-arguments-dict={'reference_frame':'LISA', t_obs_start:0.8}
frequency-domain-source-model=bilby_lisa.source.lisa_binary_black_hole
```

Or create your favourite source model!

Data analysis with `bilby_lisa`

bilby LISA 1.0.0 documentation

Search

Reproducing our analysis

Once `bilby_lisa` has been installed (see here for details), our analyses can be reproduced by interfacing with bilby_pipe, a Python package for automating transient gravitational-wave parameter estimation analyses, or parallel_bilby, a Python package for launching gravitational-wave parameter estimation analyses on a large number of cores. In Section 3.1 of BILBY in space: Bayesian inference for transient gravitational-wave signals observed with LISA we analysed a zero-noise injection through `bilby_pipe` (when evaluating the heterodyned likelihood), and `parallel_bilby` (when evaluating the full Whittle likelihood). Although `bilby_pipe` could have been used for the full Whittle likelihood analysis, we used `parallel_bilby` in order to reduce wall time. To reproduce these analyses, the following configuration files can be used,

`bilby_pipe` `parallel_bilby`

```
#################################################################
## Data generation arguments
#################################################################
```

https://icg-gravwaves.github.io/bilby_lisa_paper/

for further details

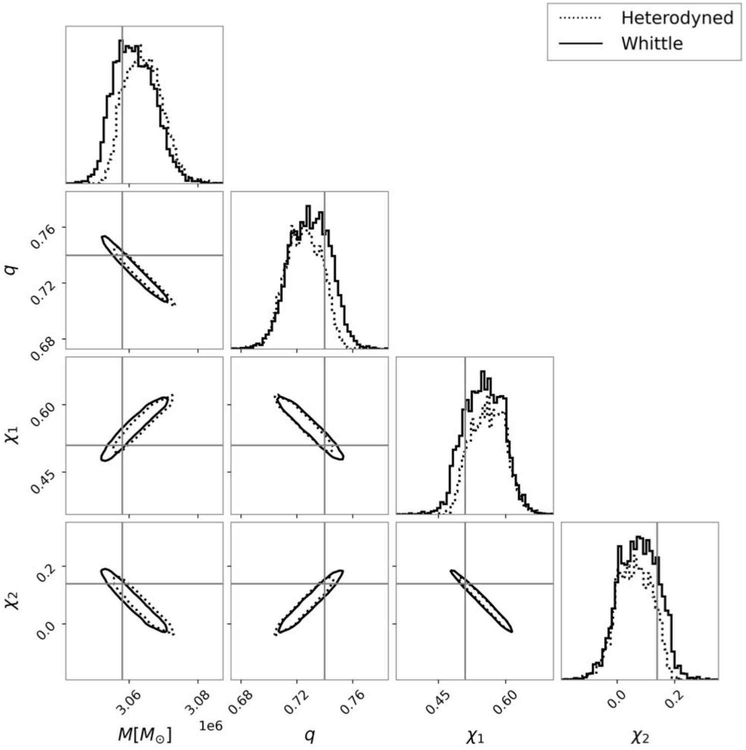
A slightly different configuration file is needed for `parallel_bilby`

Demonstration: Zero noise injections

Performed full nested sampling with the dynesty sampler via `bilby` and `parallel_bilby` (a parallelized version of `bilby`)

We recovered the injected values to good accuracy.

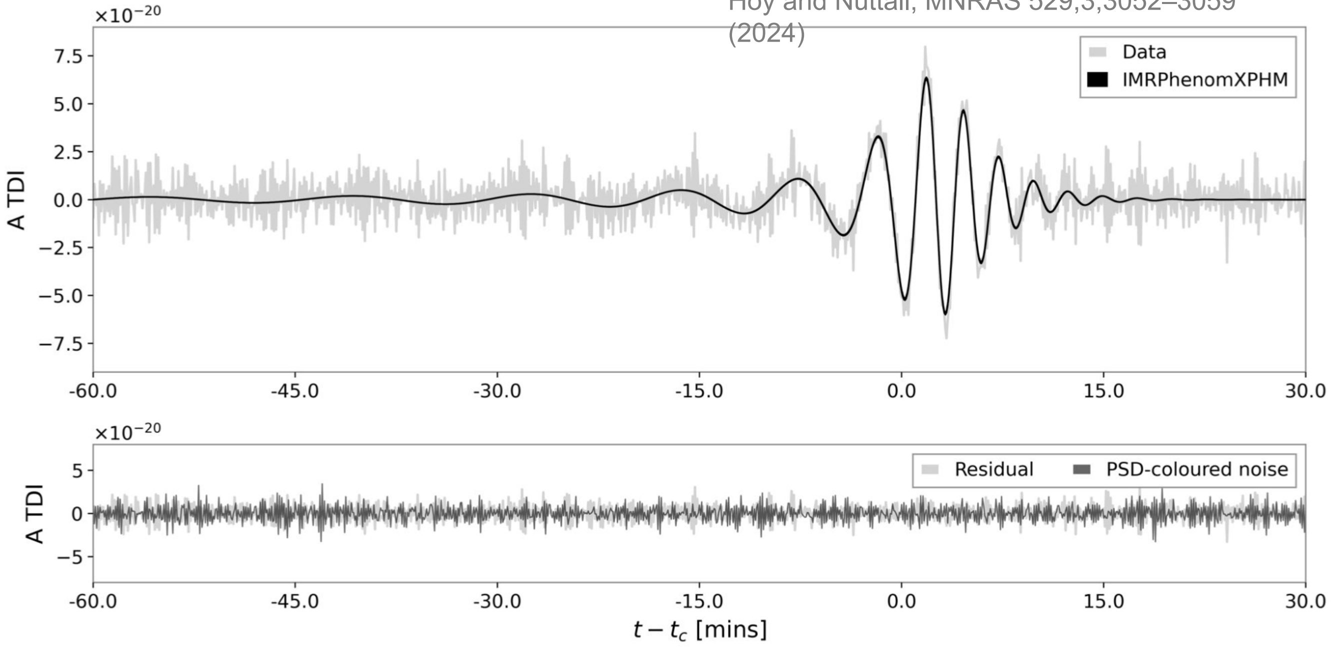
Full Whittle (Heterodyned) analysis performed on 512 (32) CPUs with total sampling time: 80 (14) hours.



Hoy and Nuttall, MNRAS 529,3,3052–3059 (2024)

Demonstration: Gaussian noise

Hoy and Nuttall, MNRAS 529,3,3052–3059 (2024)



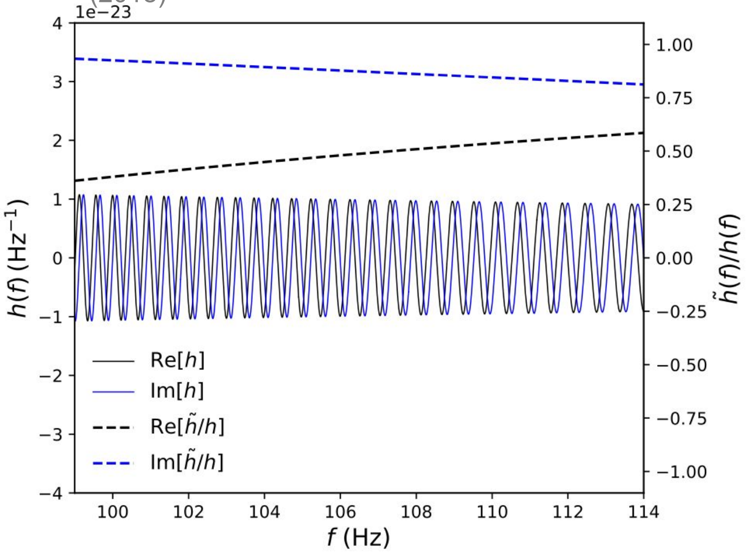
Summary

- `bilby` is the main workhorse of the LIGO-Virgo-KAGRA collaboration for gravitational-wave Bayesian inference, and it is actively developed/maintained.
- `bilby_lisa` is an extension of `bilby` to perform Bayesian inference for transient gravitational-wave signals observed with LISA
- `bilby_lisa` has the ability to interface with a range of samplers (although only tested with dynesty) and likelihood optimisations (although only tested with the whittle and heterodyned likelihoods). In principle these should work with LISA.
- Complements work done by PyCBC: It is good to have multiple independent implementations to check for systematics etc. There is also a strong benefit to use tools that are already being used for gravitational-wave Bayesian inference.

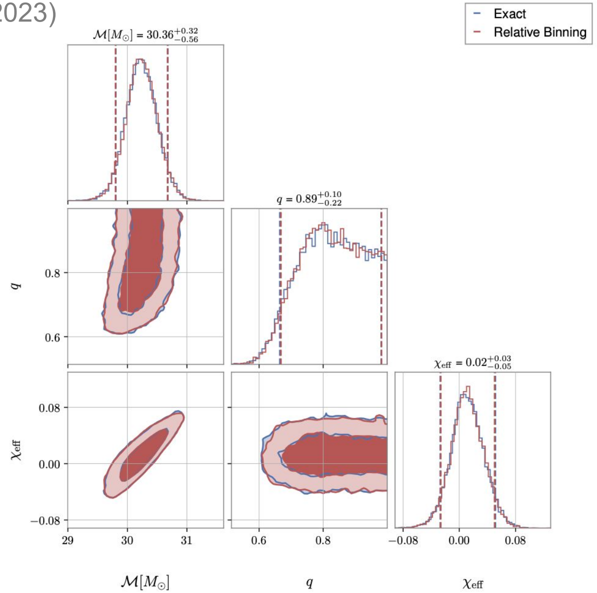
Questions

Likelihoods - Heterodyned

Plot taken Zackay *et al.*, arXiv:1806.08792 (2018)



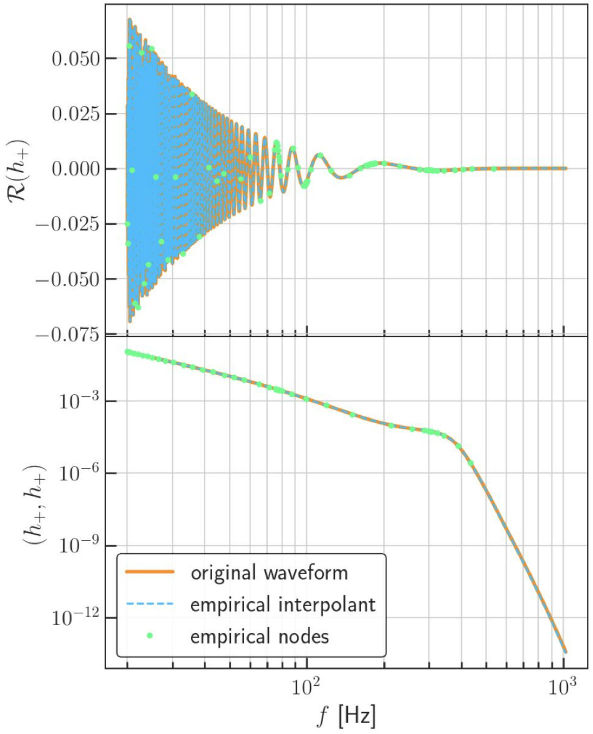
Plot taken Krishna *et al.*, arXiv:2312.06009 (2023)

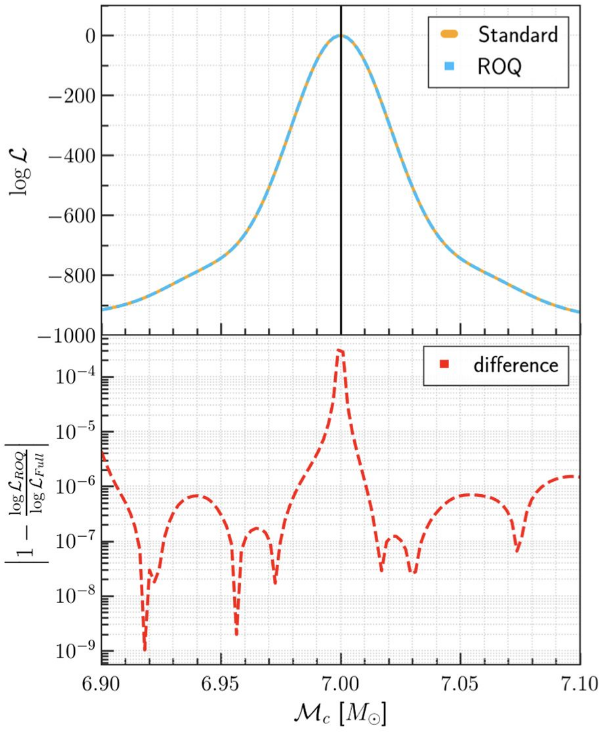


The heterodyned likelihood assumes that the ratio of two points is a smoothly varying function. Summary data for a fiducial point can then be pre-computed. This likelihood requires knowing a fiducial point *a priori*.

Likelihoods - ROQ

Plots taken Qi *et al.*, PRD 104, 063031 (2021)

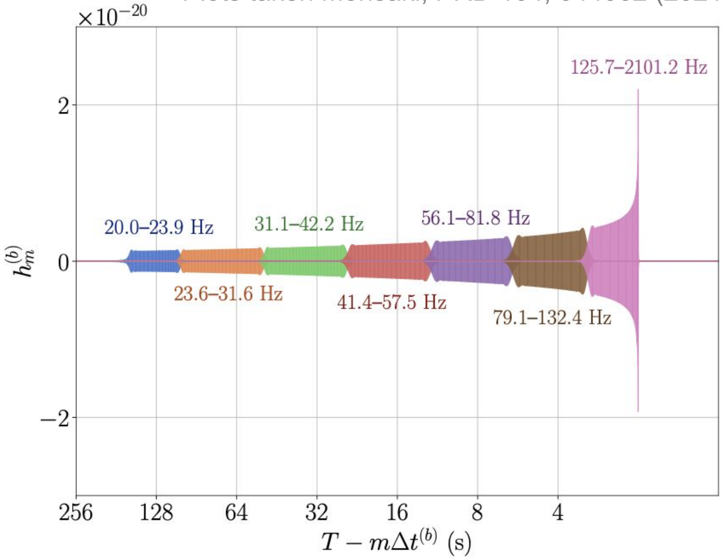


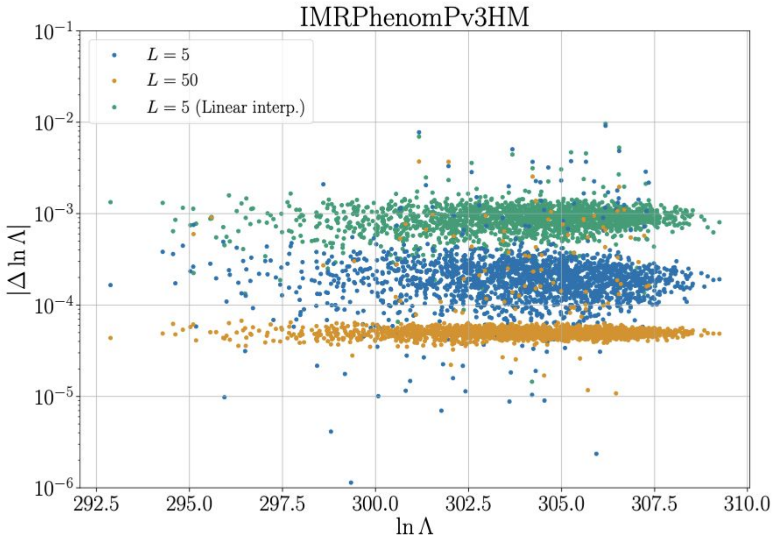


The ROQ likelihood works by identifying a reduced basis that accurately describes the waveform model over a reduced parameter space. This likelihood requires pre-computing the appropriate basis.

Likelihoods - Multiband

Plots taken Morisaki, PRD 104, 044062 (2021)





The multibanded likelihood splits the waveform into multiple frequency bands, and uses a varying `delta_f` in each band. This likelihood does not require any pre-calculations.