









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^{1,2,3},
Qi Chu^{4,5}, Atul Divakarla^{1,2,6}, Paul J. Easter^{1,2}, Boris Goncharov^{1,2}, Francisco Hernandez Vivanco^{1,2}, Jan Harms^{7,8},
Marcus E. Lower^{1,9,10} , Grant D. Meadors^{1,2}, Denyz Melchor^{1,2,11}, Ethan Payne^{1,2}, Matthew D. Pitkin¹² , Jade Powell^{9,10},
Nikhil Sarin^{1,2} , Rory J. E. Smith^{1,2}, and Eric Thrane^{1,2}

¹ School of Physics and Astronomy, Monash University, Clayton, VIC 3800, Australia; greg.ashton@monash.edu, moritz.huebner@monash.edu,
paul.lasky@monash.edu, colm.talbot@monash.edu

² OzGrav: The ARC Centre of Excellence for Gravitational Wave Discovery, Clayton, VIC 3800, Australia

³ LIGO, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

⁴ OzGrav: The ARC Centre of Excellence for Gravitational Wave Discovery, Crawley, WA 6009, Australia

⁵ School of Physics, University of Western Australia, Crawley, WA 6009, Australia

⁶ Department of Physics, University of Florida, 2001 Museum Road, Gainesville, FL 32611-8440, USA

⁷ Gran Sasso Science Institute (GSSI), I-67100 L'Aquila, Italy

⁸ INFN, Laboratori Nazionali del Gran Sasso, I-67100 Assergi, Italy

⁹ Centre for Astrophysics and Supercomputing, Swinburne University of Technology, Hawthorn, VIC 3122, Australia

¹⁰ Ozgrav, Swinburne University of Technology, Hawthorn, VIC 3122, Australia

¹¹ California State University Fullerton, Fullerton, CA 92831, USA

¹² SUPA, School of Physics & Astronomy, University of Glasgow, Glasgow G12 8QQ, UK

Received 2018 November 20; revised 2019 February 11; accepted 2019 February 11; published 2019 April 1

Abstract

Bayesian parameter estimation is fast becoming the language of gravitational-wave astronomy. It is the method by which gravitational-wave data is used to infer the sources' astrophysical properties. We introduce a user-friendly Bayesian inference library for gravitational-wave astronomy, BILBY. This PYTHON code provides expert-level parameter estimation infrastructure with straightforward syntax and tools that facilitate use by beginners. It allows users to perform accurate and reliable gravitational-wave parameter estimation on both real, freely available data from LIGO/Virgo and simulated data. We provide a suite of examples for the analysis of compact binary mergers and other types of signal models, including supernovae and the remnants of binary neutron star mergers. These examples illustrate how to change the signal model, implement new likelihood functions, and add new detectors. BILBY has additional functionality to do population studies using hierarchical Bayesian modeling. We provide an example in which we infer the shape of the black hole mass distribution from an ensemble of observations of binary black hole mergers.

Key words: gravitational waves – methods: data analysis – methods: statistical – stars: black holes – stars: neutron

1. Introduction

Bayesian inference underpins gravitational-wave science. Following a detection, Bayesian parameter estimation allows one to estimate the properties of a gravitational-wave source, for example, the masses and spins of the components in a binary merger (e.g., Abbott et al. 2016a, 2016c, 2016e, 2018b, 2018c). If the detection involves neutron stars, Bayesian parameter estimation is used to study the properties of matter at nuclear densities via the signature of tidal physics imprinted on the gravitational waveform (Abbott et al. 2017g, 2018b, 2018c). The posterior probability distributions of source parameters such as inclination angle can be used, in turn, to make inferences about electromagnetic phenomena such as gamma-ray bursts (e.g., Abbott et al. 2017c). Such parameter estimation is also used to measure cosmological parameters such as the Hubble constant (Abbott et al. 2017a). By combining data from multiple detections, Bayesian inference is used to understand the population properties of gravitational-wave sources (e.g., Abbott et al. 2016a; Farr et al. 2018; Smith & Thrane 2018; Talbot & Thrane 2018; Taylor & Gerosa 2018; Wysocki et al. 2018; Roulet & Zaldarriaga 2019; and references therein), which are providing insights into stellar astrophysics. By extending the gravitational-wave signal model, Bayesian inference is used to test general relativity

and look for evidence of new physics (Abbott et al. 2016f, 2017c, 2017f, 2018a, 2018d)

The field of gravitational-wave astronomy is growing rapidly. We have entered the “open-data era,” in which gravitational-wave data has become publicly available (Vallisneri et al. 2015). Since Bayesian parameter estimation is central to gravitational-wave science, there is a need for a robust, user-friendly code that can be used by both gravitational-wave novices and experts alike.

The primary tool currently used by the LIGO and Virgo collaborations for parameter estimation of gravitational-wave signals is LALINFERENCE (Veitch et al. 2015). This pioneering code enabled the major gravitational-wave discoveries achieved during the first two LIGO observing runs (e.g., Abbott et al. 2016a, 2016c, 2016e, 2018b, 2018c). The code itself is now almost a decade old, and years of development have made it hard for beginners to learn and difficult for experts to modify and adapt to new challenges. More recently, PYCBC INFERENCE (Biwer et al. 2019), a modern, PYTHON-based tool kit designed for compact binary coalescence parameter estimation, was released. This package provides access to several different samplers and builds on the PYCBC package (Nitz et al. 2018), an open-source tool kit for gravitational-wave astronomy.

We introduce BILBY, a user-friendly parameter estimation code for gravitational-wave astronomy. BILBY provides expert-level

parameter estimation infrastructure with straightforward syntax and tools that facilitate use by beginners. For example, with minimal user effort, users can download and analyze publicly available LIGO and Virgo data to obtain posterior distributions for the astrophysical parameters associated with recent detections of binary black holes (Abbott et al. 2016a, 2016b, 2016d, 2017d, 2017g, 2017f) and the binary neutron star merger (Abbott et al. 2017g).

One key functional difference between BILBY and LALINFERENCE/PYCBC INFERENCE is its modularity and adaptability. The core library is not specific to gravitational-wave science and has uses outside of the gravitational-wave community. Ongoing projects include astrophysical inference in multimessenger astronomy, pulsar timing, and X-ray observations of accreting neutron stars. The gravitational-wave-specific library is also built in a modular way, enabling users to easily define their own waveform models, likelihood functions, etc. This implies that BILBY can be used for more than studying compact binary coalescences; see Section 5. The modularity further ensures that the code will be sufficiently extensible to suit the future needs of the gravitational-wave community. Moreover, we believe that the wider astrophysics inference community will find the code useful by virtue of having a common interface and ideas that can be easily adapted to a range of inference problems.

The remainder of this paper is structured to highlight the versatile yet user-friendly nature of the code. To that end, the paper is example-driven. We assume familiarity with the mathematical formalism of Bayesian inference and parameter estimation (priors, likelihoods, evidence, etc.), as well as with gravitational-wave data analysis (antenna-response functions, power spectral densities, etc.). Readers looking for an introduction to Bayesian inference in general are referred to Skilling (2004), while gravitational-wave-specific introductions to inference can be found in Veitch et al. (2015) and Thrane & Talbot (2018). Section 2 describes the BILBY design philosophy, and Section 3 provides an overview of the code, including installation instructions in Section 3.1. Subsequent sections show worked examples. The initial examples are the sort of simple calculations that we expect will be of interest to most casual readers. Subsequent sections deal with increasingly complex applications that are more likely of interest to specialists.

The worked examples are as follows. Section 4 is devoted to compact binary coalescences. In Section 4.1, we carry out parameter estimation with publicly available data to analyze GW150914, the first-ever gravitational-wave event. In Section 4.2, we study a simulated binary black hole signal added to Monte Carlo noise. In Section 4.3, we study the matter effects encoded in the gravitational waveforms of a binary neutron star inspiral. In Section 4.4, we show how it is possible to add more sophisticated gravitational-waveform phenomenology, for example, by including memory, eccentricity, and higher-order modes. In Section 4.5, we study an extended gravitational-wave network with a hypothetical new detector.

Section 5 is devoted to signal models for sources that are not compact binary coalescences. In Section 5.1, we perform model selection for gravitational waves from a core-collapse supernova. In Section 5.2, we study the case of a post-merger remnant. Section 6 is devoted to hyperparameterization, a technique used to study the population properties of an

ensemble of events. Closing remarks are provided in Section 8.

2. BILBY Design Philosophy

Three goals guide the design choices of BILBY. First, we seek to provide a parameter estimation code that is sufficiently powerful to serve as a workhorse for expert users. Second, we aim to make the code accessible for novices, lowering the bar to work on gravitational-wave inference. Third, we desire to produce a code that will age gracefully; advances in gravitational-wave astronomy and Bayesian inference can be incorporated straightforwardly without resort to inelegant work-arounds or massive rewrites. To this end, we adhere to a design philosophy, which we articulate with four principles.

1. **Modularity.** Wherever possible, we seek to modularize the code and follow the abstraction principle (Pierce 2002), reducing the amount of repeated code and easing development. For example, the sampler is a modularized object, so if a problem is initially analyzed using the `PyMultiNest` (Buchner et al. 2014) sampler, for example, one can easily switch to the `emcee` (Foreman-Mackey et al. 2013) sampler or even a custom-built gravitational-wave sampler. For example, BILBY accesses samplers through a common interface; as a result, it is trivial to easily switch between samplers to compare performance or check convergence issues.
2. **Consistency.** We enforce strict style guidelines, including adherence to the PEP8 style guide for PYTHON.¹³ As a result, the code is relatively easy to follow and intuitive. In order to maintain the integrity of the code while responding to the needs of a large and active user base, we employ GITLAB’s merge-request feature. Updates require approval by two experts. The PEP8 protocol is enforced using continuous integration.
3. **Generality.** Wherever possible, we keep the code as general as possible. For example, the gravitational-wave package is separate from the package that passes the likelihood and prior to the sampler. This generality provides flexibility. For example, in Section 6, we show how BILBY can be used to carry out population inference, even though the likelihood function is completely different from the one used for gravitational-wave parameter estimation. Moreover, a general design facilitates the transfer of ideas into and out of gravitational-wave astronomy from the greater astrostatistics community.
4. **Usability.** We observe that, historically, people find it difficult to get started with gravitational-wave inference. In order to lower the bar, we endeavor to make basic things doable with very few lines of code. We provide a large number of tutorials that can serve as a blueprint for a large variety of real-world problems. Finally, we endeavor to follow the advice of the PEP20 style guide for PYTHON¹⁴: “There should be one—and preferably only one—obvious way to do it.” In other words, once users are familiar with the basic layout of BILBY, they can intuit where to look if they want to, for example, add

¹³ <https://www.python.org/dev/peps/pep-0008/>

¹⁴ <https://www.python.org/dev/peps/pep-0020/>

a new detector (see Section 4.5) or include nonstandard polarization modes.

3. Code Overview

3.1. Installation

BILBY is open-source, MIT-licensed, and written in PYTHON. The simplest installation method is through PyPI.¹⁵ The following command installs from the command line:

```
$ pip install bilby.
```

This command downloads and installs the package and dependencies. The source code can be obtained from the git repository,¹⁶ which also houses an issue tracker and merge-request tool for those wishing to contribute to code development. This repository is mirrored on github¹⁷ and is archived in Zenodo (doi:10.5281/zenodo.2602178). Documentation about code installation, functionality, and user syntax is also provided.¹⁸ Scripts to run all examples presented in this work are provided in the git repository.

3.2. Packages

BILBY has been designed such that logical blocks of code are separated, and, wherever possible, code is abstracted away to allow future reuse by other models. At the top level, BILBY has three packages: `core`, `gw`, and `hyper`. The `core` package contains the key functionalities. It passes the user-defined priors and likelihood function to a sampler, harvests the posterior samples and evidence calculated by the sampler, and returns a `result` object providing a common interface to the output of any sampler along with information about the inputs. The `gw` package contains gravitational-wave-specific functionality, including waveform models, gravitational-wave-specific priors, and likelihoods. The `hyper` package contains functionality for the hierarchical Bayesian inference (see Section 6). A flowchart showing the dependency of different packages and modules is available on the git repository.¹⁹

3.2.1. The `core` Package

The `core` package provides all of the code required for general problems of inference. It provides a unified interface to several different samplers listed below, standard sets of priors including arbitrary user-defined options, and a universal result object that stores all important information from a given simulation.

Prior and likelihood functions are implemented as classes, with a number of standard types implemented in the `core` package: e.g., the Normal, Uniform, and LogUniform priors and GaussianLikelihood, PoissonLikelihood, and ExponentialLikelihood likelihoods. One can write their own custom prior and likelihood functions by writing a new class that inherits from the parent Prior or Likelihood, respectively. The user only needs to define how the new prior or likelihood is instantiated and calculated, with all other housekeeping logic being abstracted away from the user.

The prior and likelihood are passed to the function `run_sampler`, which allows the user to quickly change the sampler method between any of the prewrapped samplers and define specific run-time requirements, such as the number of live points, number of walkers, etc. Prepackaged samplers include Markov chain Monte Carlo ensemble samplers `emcee` (Foreman-Mackey et al. 2013), `ptemcee` (Vousden et al. 2016), and `PyMC3` (Salvatier et al. 2016) and nested samplers (Skilling 2004, 2006) `MultiNest` (Feroz & Hobson 2008; Feroz et al. 2009, 2013; through the PYTHON implementation `pyMultiNest` of Buchner et al. 2014), `Nestle`,²⁰ `Dynesty`,²¹ and `CPNest` (Veitch et al. 2017). Only the `Dynesty` sampler comes installed as part of the standard BILBY installation process; other samplers can be installed by following the instructions in the documentation.²² The `Sampler` class again allows users to specify their own sampler by following the other examples.

Despite the choice of sampler, the output from BILBY is universal: an `hdf5` file that contains all output, including posterior samples, likelihood calculations, injected parameters, evidence calculations, etc. The `Result` object can be used to load in these output files and perform common operations, such as generating corner plots and creating plots of the data and maximum posterior fit.

3.2.2. The `gw` Package

The `gw` package provides the core functionality for parameter estimation specific to transient gravitational waves. Building on the `core` package, this provides prior specifications unique to such problems, e.g., a prior that is uniform in comoving volume distance, as well as the standard likelihood used when studying gravitational-wave transients (e.g., see Veitch et al. 2015 and Equation (1)), defined as the `GravitationalWaveTransient` class. The `gw` package also provides an implementation of current gravitational-wave detectors in the `detector` module, including their location and orientation, as well as different noise power spectral densities for both current and future instruments. Standard waveform approximants are also included in the `source` module and are handled through the `LALSIMULATION` package (LIGO Scientific Collaboration 2018).

The `gw` package also contains a set of tools to load, clean, and analyze gravitational-wave data. Many of these functions are built on the `GWpy` (Macleod et al. 2018) code base, which is contained within `bilby.gw.detector` and primarily accessed by instantiating a list of `Interferometer` objects. This functionality also allows users to implement their own gravitational-wave detector by instantiating a new `Interferometer` object; we show an explicit example of this in Section 4.5.

The packages mentioned above are not installed with BILBY by default; instructions for installing them can be found in the documentation.²³ We have established a `singularity`²⁴ container that has all dependencies, including `GWpy`, `LALSuite`, and all supported samplers. Instructions for using the `singularity` container can be found in the documentation.²⁵

¹⁵ <https://pypi.org/project/BILBY/>

¹⁶ <https://git.ligo.org/lscsoft/bilby/>

¹⁷ <https://github.com/lscsoft/bilby/>

¹⁸ <https://lscsoft.docs.ligo.org/bilby/>

¹⁹ <https://lscsoft.docs.ligo.org/bilby/code-overview.html>

²⁰ <http://kylebarbary.com/nestle/>

²¹ <https://github.com/joshspeagle/dynesty>

²² <https://lscsoft.docs.ligo.org/bilby/samplers.html#installing-samplers>

²³ <https://lscsoft.docs.ligo.org/bilby/installation.html>

²⁴ <https://www.sylabs.io/singularity/>

²⁵ <https://lscsoft.docs.ligo.org/bilby/containers.html>

Table 1
Default Binary Black Hole Priors

Variable	Unit	Prior	Minimum	Maximum
$m_{1,2}$	M_{\odot}	Uniform	5	100
$a_{1,2}$...	Uniform	0	0.8
$\theta_{1,2}$	rad.	sin	0	π
$\delta\phi, \phi_{\text{JL}}$	rad.	Uniform	0	2π
d_L	Mpc	Comoving	10^2	5×10^3
R.A.	rad.	Uniform	0	2π
Decl.	rad.	cos	$-\pi/2$	$\pi/2$
ι	rad.	sin	0	π
ψ	rad.	Uniform	0	π
ϕ_c	rad.	Uniform	0	2π

Note. The intrinsic variables are the two black hole masses $m_{1,2}$, their dimensionless spin magnitudes $a_{1,2}$, the tilt angle between their spins and the orbital angular momentum $\theta_{1,2}$, and the two spin vectors describing the azimuthal angle separating the spin vectors $\delta\phi$ and the cone of precession about the system’s angular momentum ϕ_{JL} . The extrinsic parameters are the luminosity distance d_L , the R.A. and decl., the inclination angle between the observers line of sight and the orbital angular momentum ι , the polarization angle ψ , and the phase at coalescence ϕ_c . The phase, spins, and inclination angles are all defined at some reference frequency. We do not set a default prior for the coalescence time t_c . Here “sin” and “cos” priors are uniform in cosine and sine, respectively, and “comoving” implies uniform in comoving volume.

3.3. The *hyper* Package

The *hyper* package contains all required functionality to perform hierarchical Bayesian inference of populations. This includes both a `Model` module and a `HyperparameterLikelihood` class. This entire package is discussed in more detail in Section 6.

4. Compact Binary Coalescence

In this section, we show a suite of BILBY examples analyzing binary black hole and binary neutron star signals.

```
>>> import bilby
>>> interferometers = bilby.gw.detector.get_event_data("GW150914")
```

We employ a standard Gaussian noise likelihood \mathcal{L} for strain data d given source parameters θ (van der Sluys et al. 2008a, 2008b; Veitch & Vecchio 2008),

$$\ln \mathcal{L}(d|\theta) = -\frac{1}{2} \sum_k \left\{ \frac{[d_k - \mu_k(\theta)]^2}{\sigma_k^2} + \ln(2\pi\sigma_k^2) \right\}, \quad (1)$$

where k is the frequency bin index, σ is the noise amplitude spectral density, and $\mu(\theta)$ is the waveform. The waveform is a function of the source parameters θ , which consist of (at least) eight intrinsic parameters (primary mass m_1 , secondary mass m_2 , primary spin vector \mathbf{S}_1 , and secondary spin vector \mathbf{S}_2) and

seven extrinsic parameters (luminosity distance d_L , inclination angle ι , polarization angle ψ , time of coalescence t_c , phase of coalescence ϕ_c , and R.A. and decl.). Table 1 shows the default priors implemented for binary black hole systems. We show how these priors can be called in Sections 4.1 and 4.2. Unless otherwise specified, $\mu(\theta)$ is given using the `IMRPhenomP` approximant (Schmidt et al. 2012). However, the approximant can be easily changed; see Sections 4.2 and 4.3. Moreover, it is relatively simpler to sample in different parameters than those listed above (e.g., chirp mass and mass ratio instead of m_1 and m_2); examples for doing this are provided in the `git` repository.

4.1. GW150914: The Onset of Gravitational-wave Astronomy

The first direct detection of gravitational waves occurred on the 2015 September 14, when the two LIGO detectors (Aasi et al. 2015) in Hanford, Washington, and Livingston, Louisiana, detected the coalescence of a binary black hole system (Abbott et al. 2016d). The gravitational waves swept through the two detectors with a $6.9_{-0.4}^{+0.5}$ ms time difference that, when combined with polarization information, allowed for a sky location reconstruction covering an annulus of 590 deg^2 (Abbott et al. 2016d). The initially published masses of the colliding black holes were given as 36_{-4}^{+5} and 29_{-4}^{+4} (Abbott et al. 2016e). Subsequent analyses with more accurate precessing waveforms constrained the masses to be 35_{-3}^{+5} and 30_{-3}^{+4} at 90% confidence (Abbott et al. 2016c). The distance to the source is determined to be 440_{-180}^{+160} Mpc (Abbott et al. 2016c).

In this example, we use BILBY to reproduce the parameter estimation results for GW150914. The data for published LIGO/Virgo events are made available through the Gravitational Wave Open Science Center (Vallisneri et al. 2015). Built-in BILBY functionality downloads and parses this data. We begin with the following two lines:

The first line of code imports the BILBY code base into the PYTHON environment. The second line returns a set of objects that contain the relevant data segments and associated data products relevant for the analysis for both the LIGO Hanford and Livingston detectors. By default, BILBY downloads and windows the data. A local copy of the data is saved, along with diagnostic plots of the gravitational-wave strain amplitude spectral density.

In addition to the data, the two key ingredients for any Bayesian inference calculation are the likelihood and the prior. Default sets of priors can be called from the `gw.prior` module, and we also employ the default Gaussian noise likelihood (Equation (1)):

```
>>> prior = bilby.gw.prior.BBHPriorDict(filename = "GW150914.prior")
>>> likelihood = bilby.gw.likelihood.get_binary_black_hole_likelihood(interferometers)
```

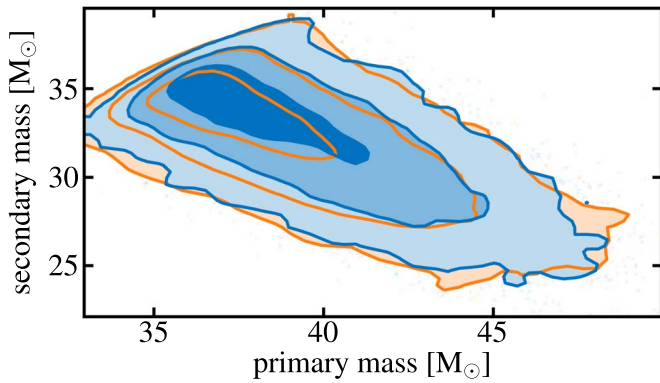


Figure 1. Marginalized posterior source-mass distributions for the first binary black hole merger detected by LIGO, GW150914. We show the posterior distributions recovered using BILBY (blue) and LALINFERENCE (orange) using open data from the Gravitational Wave Open Science Center (Vallisneri et al. 2015). The five lines of BILBY code required for reproducing the posteriors are shown in Section 4.1.

The above code calls the GW150914 prior, which differs from the priors described in Table 1 in two main ways. First, to speed up the running of the code, it restricts the mass priors to between 30 and 50 M_{\odot} for the primary mass and 20 and 40 M_{\odot} for the secondary mass. Moreover, this prior call restricts the time of coalescence to 0.1 s before and after the known coalescence time. One can revert to the priors in Table 1 by replacing the above file call with `filename="binary_black_holes.prior"`, but this would require separately setting a prior for the coalescence time. We show how this can be done in Section 4.2.

The next step is to call the sampler:

```
>>>result = bilby.core.sampler.run_sampler(likelihood,prior).
```

This line performs parameter inference using the sampler default Dynesty, with a default 500 live points. This number can be increased by passing the `nlive=` keyword argument to `run_sampler()`. The sampler returns a list of posterior samples, the Bayesian evidence, and metadata, which are stored in an `hdf5` file. One may plot a corner plot showing the posterior distribution for all parameters in the model using the command

```
>>>result.plot_corner().
```

The above example code produces posterior distributions that, by eye, agree reasonably well with the parameter uncertainty associated with the published distributions for GW150914. The shape of the likelihood for the extrinsic parameters presents significant challenges for samplers due to strong degeneracies between different sky locations, distances, inclination angles, and polarization angles (see, e.g., Farr et al. 2014a; Raymond & Farr 2014). For more accurate results, we use the nested sampling package CPNEST (Veitch et al. 2017), which is invoked by changing the `run_sampler` function above to include the additional argument `sampler='cpnest'`. We also change the number of live points by adding `nlive=5000` to the same function and specify a keyword argument, `maxmcmc=5000`, which is the maximum number of steps the sampler takes before accepting a new sample. To resolve the issue with the phase at coalescence, we analytically

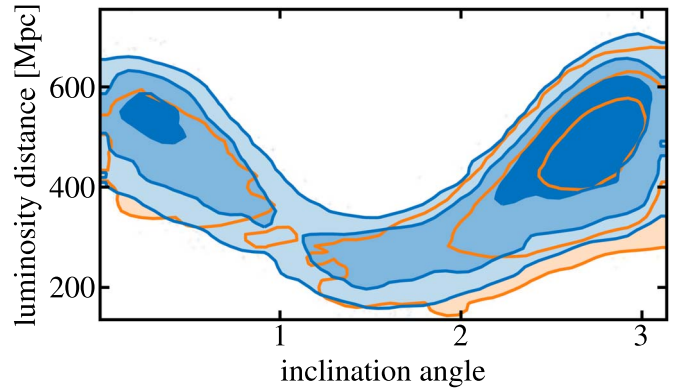


Figure 2. Marginalized posterior distributions on the binary inclination angle and luminosity distance for the first binary black hole merger detected by LIGO, GW150914. We show the posterior distributions recovered using BILBY (blue) and LALINFERENCE (orange) using open data from the Gravitational Wave Open Science Center (Vallisneri et al. 2015).

marginalize over this parameter (Farr 2014) by adding the optional `phase_marginalization=True` argument to the instantiation of the likelihood. BILBY has built-in analytic marginalization procedures for the time of coalescence (Farr 2014) and distance (Singer & Price 2016; Singer et al. 2016), which can both be invoked using `time_marginalization=True` and `distance_marginalization=True`, respectively. These decrease the run time of the code by minimizing the dimensionality of the parameter space. Posterior distributions can still be determined for these parameters by reconstructing them analytically from the full set of posterior samples (e.g., see Thrane & Talbot 2018).

Using BILBY, we can plot marginalized distributions by simply passing the `plot_corner` function the optional `parameters=...` argument. In Figure 1, we show the marginalized, two-dimensional posterior distribution for the masses of the two black holes as calculated using the above BILBY code (shown in blue). In orange, we show the LIGO posterior distributions from Abbott et al. (2016a), calculated using the LALINFERENCE software (Veitch et al. 2015) and hosted at the Gravitational Wave Open Science Center (Vallisneri et al. 2015).

In Figure 2, we show the marginalized posterior distribution of the luminosity distance and inclination angle, where the BILBY posteriors are again shown in blue and the LALINFERENCE posteriors in orange. Figure 3 shows the sky localization uncertainty for both BILBY and LALINFERENCE.

The above example does not make use of detector calibration uncertainty, which is an important feature in LIGO data analysis. Such calibration uncertainty is built into BILBY using the cubic spline parameterization (Farr et al. 2014b), with example usage in the BILBY repository.

4.2. Binary Black Hole Merger Injection

BILBY supports both the analysis of real data, as in the previous section, and the ability to inject simulated signals into Monte Carlo data. In the following two sections, we inject a binary black hole signal and a binary neutron star

signal, respectively, showing how one can easily inject and recover signals and their astrophysical properties.

In this first example,²⁶ we create a binary black hole signal with parameters similar to GW150914 (Abbott et al. 2016e), albeit at a luminosity distance of $d_L = 2$ Gpc (see $d_L \approx 400$ Mpc for GW150914). We inject the signal into a network of LIGO-Livingston, LIGO-Hanford (Aasi et al. 2015), and Virgo interferometers (Acernese et al. 2015), each operating at design sensitivity. When doing examples of this nature, it is time-intensive to sample over all 15 parameters in the waveform model. Therefore, to get quick results that can be run on a laptop, we only sample over four parameters in the waveform model: the two black hole masses $m_{1,2}$, the luminosity distance d_L , and the inclination angle ι . BILBY supports simple functionality to limit or extend the number of parameters included in the likelihood calculation, as shown below.

We begin by setting up a `WaveformGenerator` object using a frequency domain strain model that takes the signal injection parameters and specific waveform arguments, such as the waveform approximant, as arguments. The `WaveformGenerator` also takes data duration and sampling frequency as input parameters. With the source model defined, we now instantiate an `interferometer` object that takes the strain signal from the `WaveformGenerator` and injects it into a noise realization of the three interferometers. One could choose to do a zero-noise simulation by simply including the flag `zero_noise=True`.

Priors are set up as in the previous open-data example, except we call the `binary_black_holes.prior` file instead of the specific prior file for GW150914. Moreover, to hold all but four of the parameters fixed, we set the value of the prior for those other parameters to the injection value. For example, setting

```
>>>prior['a_1'] = 0
```

sets the prior on the dimensionless spin magnitude of the primary black hole to a δ -function at zero.

In general, we can change the prior for any parameter with one line of code. For example, to change the prior on the primary mass to be uniform between $m_1 = 25$ and $35 M_\odot$, say, one includes

```
>>>prior['mass_1'] = bilby.core.prior.Uniform(minimum = 25, maximum = 35, unit = r'$M_\odot$').
```

BILBY knows about many different types of priors that can all be called in this way.

For this example, we are also required to define priors on the coalescence time, which we define to be a uniform prior with minimum and maximum 1 s either side of the injection time.

The likelihood is again set up similarly to the open-data example of Section 4.1, although this time, we must pass the `interferometer`, `waveform_generator`, and `prior`. Finally, the sampler can be called in the same way as Section 4.1; for this example, we use the `pyMultiNest` nested sampler (Buchner et al. 2014).

²⁶ This example is found in the BILBY git repository at https://git.ligo.org/lscsoft/bilby/blob/master/examples/injection_examples/fast_tutorial.py.

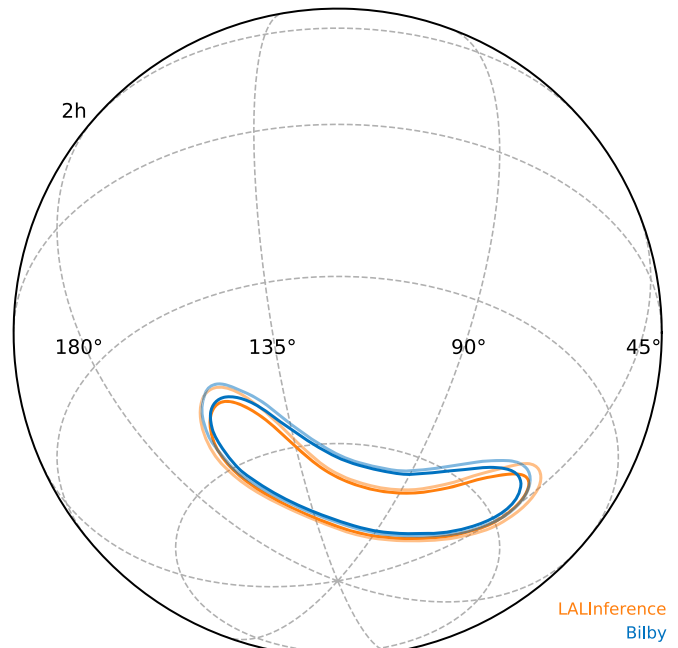


Figure 3. Sky localization uncertainty for GW150914. The blue marginalized posterior distributions are those recovered using BILBY, and the orange are those recovered using LALINFERENCE, using open data from the Gravitational Wave Open Science Center (Vallisneri et al. 2015).

Figure 4 shows the recovered posterior distributions (blue) and injected parameter values (orange). For this example, using the `PyMultiNest` (Buchner et al. 2014) nested sampling package with 6000 live points took approximately 30 minutes on a laptop to fully sample the four-dimensional parameter space. The parameters in Figure 4 are recovered well with the usual degeneracy present between the luminosity distance and inclination angle of the source, d_L and ι , respectively.

4.3. Measuring Tidal Effects in Binary Neutron Star Coalescences

The first detection of binary neutron star coalescence GW170817 was a landmark event signaling the beginning of multimessenger gravitational-wave astronomy (Abbott et al. 2017g, 2017h). Gravitational-wave parameter estimation of the

inspiral is what ultimately determined that both objects were likely neutron stars and provides the best-yet constraints on the nuclear equation of state of matter at supranuclear densities (Abbott et al. 2017c, 2017g, 2017h).

One of the key measurements in determining the equation of state from binary neutron star coalescences is that of the tidal parameters. The dimensionless tidal deformability

$$\Lambda = \frac{2k_2}{3} \left(\frac{c^2 R}{Gm} \right)^5 \quad (2)$$

is a fixed parameter for a given equation of state and neutron star mass. Here k_2 is the second Love number, and R and m are the neutron star radius and mass, respectively. The binary

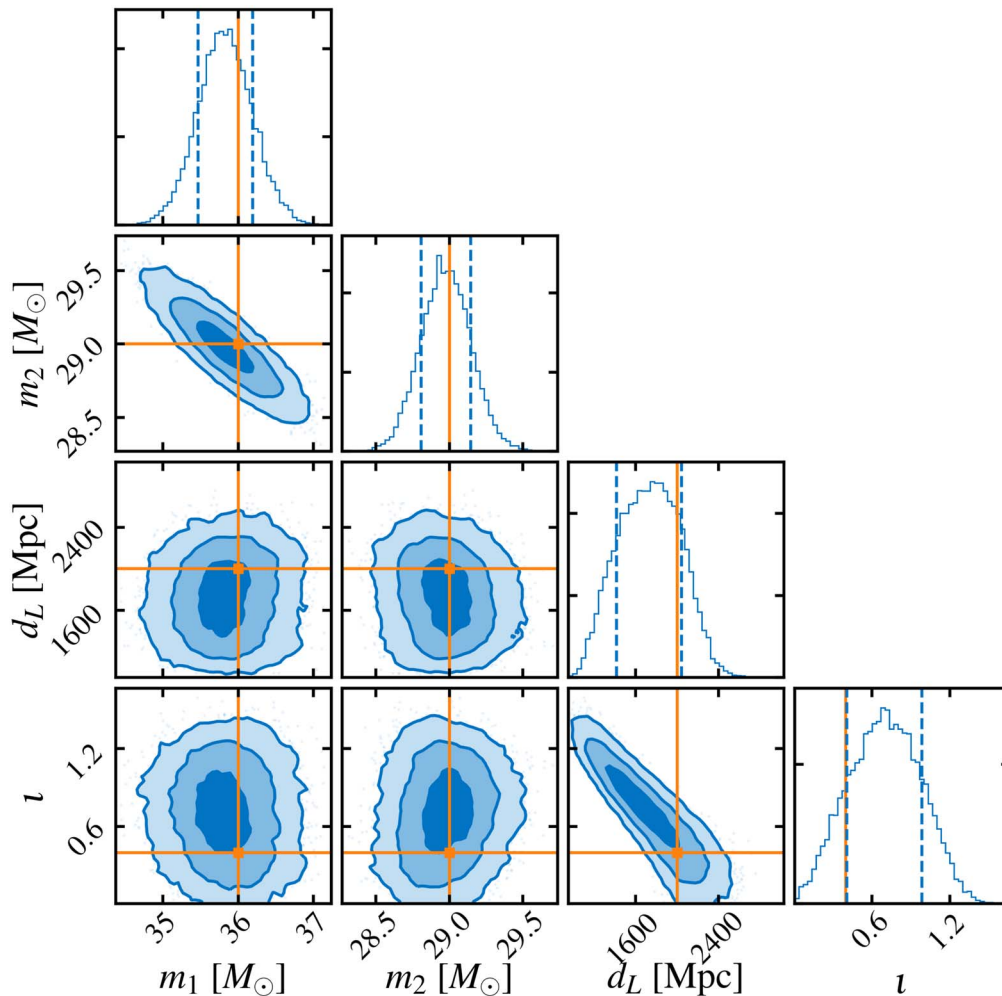


Figure 4. Injecting and recovering a binary black hole gravitational-wave signal with BILBY. We inject a signal into a three-detector network of LIGO-Livingston, LIGO-Hanford, and Virgo and perform parameter estimation. The posterior distributions are shown in blue and the injected values in orange. To speed up the simulation, we only search over the two black hole masses m_1 and m_2 , the luminosity distance d_L , and the inclination angle i .

neutron star merger GW170817 provided constraints of $\Lambda_{1,4} = 190^{+390}_{-120}$ (Abbott et al. 2018b; De et al. 2018), where the subscript denotes that this is the estimate on Λ assuming a $1.4 M_\odot$ neutron star, and the uncertainty is the 90% credible interval.

BILBY can be used to study neutron star coalescences in both real and simulated data. We inject a binary neutron star signal using the `TaylorF2` waveform approximant into a three-detector network of the two LIGO detectors and Virgo, all operating at design sensitivity.²⁷ Our injected signal is an $m_1 = 1.3 M_\odot$, $m_2 = 1.5 M_\odot$ binary at $d_L = 50$ Mpc with dimensionless spin parameters $a_{1,2} = 0.02$ and tidal deformabilities $\Lambda_{1,2} = 400$. Setting up such a system in BILBY is equivalent to doing the binary black hole injection study of Section 4.2, except we call the `lal_binary_neutron_star` source function, which requires the additional $\Lambda_{1,2}$ arguments. We also have specific binary neutron star priors; the default set can be called using

```
>>> priors = bilby.gw.prior.BNSPriorDict().
```

²⁷ This example is found in the BILBY git repository at https://git.ligo.org/lscsoft/bilby/blob/master/examples/injection_examples/binary_neutron_star_example.py.

Table 2
Default Binary Neutron Star Priors

Variable	Unit	Prior	Minimum	Maximum
$m_{1,2}$	M_\odot	Uniform	1	2
$a_{1,2}$...	Uniform	-0.05	0.05
$\Lambda_{1,2}$...	Uniform	0	3000
d_L	Mpc	Comoving	10	500
R.A.	rad.	Uniform	0	2π
Decl.	rad.	cos	$-\pi/2$	$\pi/2$
ι	rad.	sin	0	π
ψ	rad.	Uniform	0	π
ϕ_c	rad.	Uniform	0	2π

Note. Here $\Lambda_{1,2}$ are the tidal deformability parameters of the primary and secondary neutron star defined in Equation (2). For other variable definitions, see Table 1. Note that our commonly used waveform approximant does not allow misaligned neutron star spins, implying that we do not require priors on those spin parameters.

The standard set of binary neutron star priors is shown in Table 2. In this example, we use the `Dynesty` sampler.

The tidal deformability parameters Λ_1 and Λ_2 are known to be highly correlated. The terms that appear explicitly due to the tidal corrections in the phase evolution are instead $\tilde{\Lambda}$ and $\delta\tilde{\Lambda}$

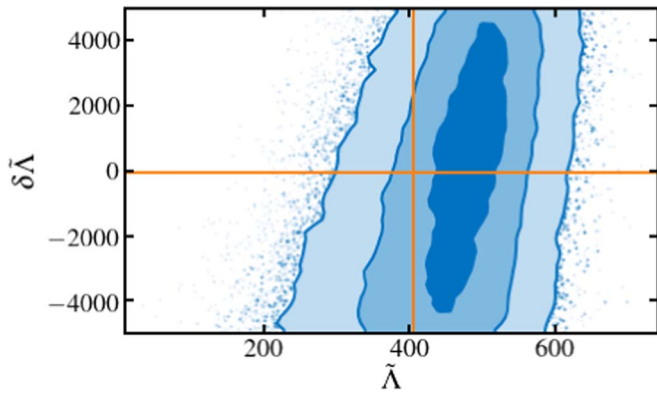


Figure 5. Injecting and recovering a binary neutron star gravitational-wave signal with BILBY. We inject a signal into the three-detector network and show here only the marginalized two-dimensional posterior on the two tidal deformability parameters (blue), with the injected values shown in orange.

(Flanagan & Hinderer 2008; for definitions of these parameters, see Equations (14) and (15) of Lackey & Wade 2015). We therefore sample in $\tilde{\Lambda}$ and $\delta\tilde{\Lambda}$ instead of Λ_1 and Λ_2 . Although we sample in all binary neutron star parameters, we show only the two-dimensional marginalized posterior distribution for $\tilde{\Lambda}$ and $\delta\tilde{\Lambda}$ in Figure 5. The corresponding injected values of $\tilde{\Lambda}$ and $\delta\tilde{\Lambda}$ are shown as the orange vertical and horizontal lines, respectively.

4.4. Implementing New Waveforms

The preceding subsections have only given the flavor of what can be achieved with BILBY for compact binary coalescences. It is trivial to implement more complex signal models that include, for example, higher-order modes, eccentricity, gravitational-wave memory, nonstandard polarizations. Examples showing different signal models are included in the `git` repository.²⁸ BILBY has already been used in one such application: testing how well the orbital eccentricity of binary black hole systems can be measured with Advanced LIGO and Advanced Virgo (Lower et al. 2018). An example script reproducing those results can be found in the `git` repository.

If a signal model exists in the LAL software,²⁹ then calling that signal model and defining which parameters to include in the sampler is as simple as the above examples. In Section 5, we also show how to include a user-defined source model. Moreover, one is free to define and sample models in either the time or frequency domain. We include examples for both cases in the `git` repository. The latter case of using a time-domain source model requires doing little more than selecting the argument `time_domain_source_model` in the `WaveformGenerator`, rather than selecting `frequency_domain_source_model`.

Of course, one may also want to set up the injection and sampler using two different waveform models, for example, to inject a numerical relativity signal into Monte Carlo data and recover it with a waveform approximant (see also Section 5.1). This is possible by simply instantiating two `WaveformGenerators`, injecting with one and passing the other to the likelihood.

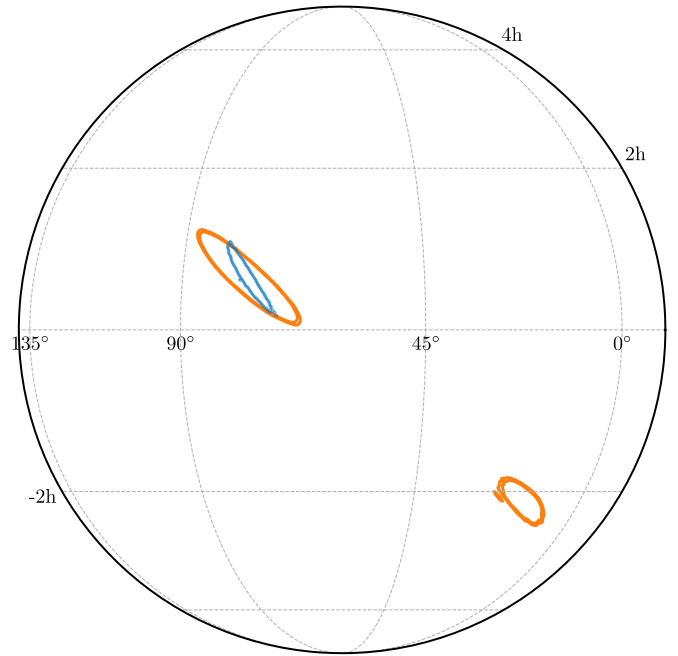


Figure 6. Sky location uncertainty when including a gravitational-wave detector in Gingin, Australia. Shown are the sky localizations (marginalized two-dimensional posterior distributions) for an injected binary black hole signal using a two-detector network of gravitational-wave interferometers Hanford and Livingston (orange) and a three-detector network that also includes the Australian detector (blue).

4.5. Adding Detectors to the Network

The full network of ground-based gravitational-wave interferometers will soon consist of the two LIGO detectors in the US, Virgo, LIGO-India (Iyer et al. 2011), and the KAGRA detector in Japan (Aso et al. 2013), all of which are implemented in BILBY. A gravitational-wave interferometer is specified by its geographic coordinates, orientation, and noise power spectral density. By default, BILBY includes descriptions of current detectors, including LIGO, Virgo, and KAGRA, as well as proposed future detectors A+ (Miller et al. 2015), Cosmic Explorer (Abbott et al. 2017b), and the Einstein Telescope (Punturo et al. 2010). It is also possible to define new detectors, which is useful for developing the science case for proposals and to optimize the design and placement of new detectors. Among other things, this can be used in developing the science case for interferometer design and placement.

BILBY provides a common interface to define detectors by their geometry, location, and frequency response. By way of example, we place a new 4 km arm interferometer in the Shire of Gingin, located outside of Perth, Australia, the current location of the Australian International Gravitational Observatory. We assume a futuristic network configuration of the Australian Observatory, together with the two LIGO detectors in Hanford and Livingston, all operating at A+ sensitivity (Miller et al. 2015). We generate A+ power spectral densities in the same script used to run BILBY by using the PYGWINC software,³⁰ which creates an array containing the frequency and noise power spectral density³¹ (one could equally use more sophisticated software such as FINESSE (Brown & Freise 2014)

²⁸ <https://git.ligo.org/lscsoft/bilby/>

²⁹ <https://wiki.ligo.org/DASWG/LALSuite>

³⁰ <https://git.ligo.org/gwinc/pygwinc>

³¹ This example is found in the BILBY `git` repository at https://git.ligo.org/lscsoft/bilby/blob/master/examples/injection_examples/australian_detector.py.

to create more detailed interferometer sensitivity curves). We then create a new `Interferometer` object using `bilby.gw.detector.Interferometer()`, which takes numerous arguments, including the position and orientation of the detector, minimum and maximum frequencies, and power or amplitude noise spectral density. The noise spectral density can be passed as an ascii file containing the frequency and spectral noise density. With the new detector defined, one can again calculate a noise realization and signal injection in a manner similar to what is done in Section 4.

In this example, we inject a GW150914-like binary black hole inspiral signal at a luminosity distance of $d_L = 4$ Gpc and recover the masses, sky location, luminosity distance, and inclination angle of the system. In this example, we use the `Nestle` sampler. Figure 6 shows the two-dimensional marginalized posterior for the sky location uncertainty when including (blue) and not including (orange) the Australian detector in Gingin. In this instance, the sky localization uncertainty decreases by approximately a factor of four when including the third detector.

While this example includes three detectors, it is straightforward to extend this analysis to an arbitrary detector network. The likelihood evaluation simply loops over the number of detectors passed to it and multiplies the likelihood for each detector to get a combined likelihood for each point in the parameter space.

5. Alternative Signal Models

Section 4 focuses on compact binary coalescences. However, the `BILBY gw` package enables parameter estimation for any type of signal for which a signal model can be defined. In this section, we show two illustrative examples: the injection and recovery of a core-collapse supernova signal and a much-simplified model of a hypermassive neutron star following a binary neutron star merger. The former example highlights two key pieces of infrastructure: the ability to inject numerical relativity signals and the ability to develop one’s own source model that is not built into `BILBY`. The latter example highlights the use of a different likelihood function that only uses the amplitude of the signal and throws away the phase information.

5.1. Supernovae

Gravitational-wave signals from core-collapse supernovae are complicated and not well understood in terms of their specific phase evolution. Numerous techniques have been developed to deal with both detection and parameter estimation. One such method for the latter problem involves principal component analysis (Logue et al. 2012; Powell et al. 2016, 2017), where the signal is reconstructed using a weighted sum of orthonormal basis vectors. In this example, we inject a gravitational-wave signal from a numerical relativity simulation (Müller et al. 2012) and recover the principal components using `BILBY`.³²

The injection is performed by defining a new signal class that, in this case, simply reads in an ascii text file containing the gravitational-wave strain time series. The injection is then performed in a way akin to the binary black hole and binary neutron star examples in Section 4. We inject signal L15 from

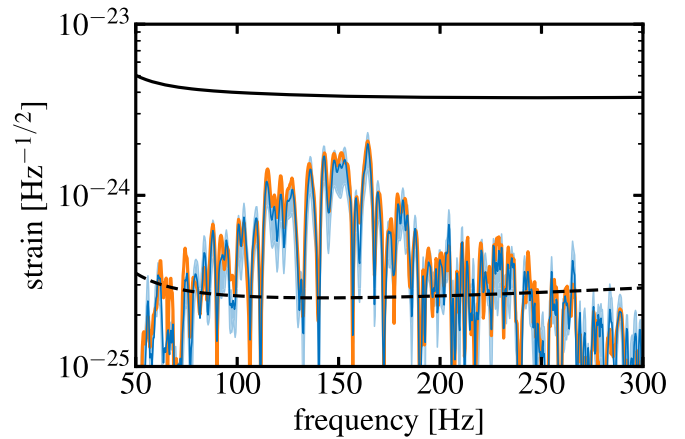


Figure 7. Parameter estimation reconstruction of a numerical relativity supernova signal. A numerical relativity supernova signal (orange) is injected into a three-detector network of the two Advanced LIGO detectors and Advanced Virgo, all operating at design sensitivity. The maximum-likelihood reconstruction of the signal is shown in dark blue, and the light blue band shows the superposition of many reconstructed waveforms from the posterior samples.

Müller et al. (2012), which comes from a three-dimensional simulation of a nonrotating core-collapse supernova with a $15 M_{\odot}$ progenitor star. The signal is injected at a distance of 5 kpc in the direction of the galactic center. The amplitude spectral density of the injected signal is shown in Figure 7 as the orange line.

The signal is reconstructed using principal component analysis, such that the strain is expressed as

$$\tilde{h}(f) = A \sum_{j=1}^k \beta_j U_j(f), \quad (3)$$

where A is an amplitude factor, and β_j and U_j are the complex principal component amplitudes and vectors, respectively. Equation (3) is implemented into `BILBY` as another new signal model that takes the β_j coefficients, luminosity distance (which is a proxy for A), and sky location as inputs. Priors for each of the new parameters are established in the same way as the example with the mass in Section 4.2. In this case, we set $k = 5$ and use uniform priors between -1 and 1 for each of the β_j values.

Figure 7 shows the injected (orange) and recovered (blue) gravitational-wave signal in the frequency domain. The dark blue curve shows the maximum-likelihood curve, and the shaded blue region is a superposition of many reconstructed waveforms from the posterior samples.

5.2. Neutron Star Post-merger Remnant

There are a number of physical scenarios that can occur following the merger of two neutron stars, including the existence of short- or long-lived neutron star remnants. In the early phases post-merger ($\lesssim 1$ s), these neutron stars are highly dynamic and can emit significant gravitational radiation potentially observable by Advanced LIGO and Virgo at a design sensitivity out to ~ 50 Mpc (e.g., Clark et al. 2014 and references therein). While the ultimate fate of binary merger GW170817 is unknown, no gravitational waves from a post-merger remnant were found (Abbott et al. 2017i, 2018c), which

³² This example is found in the `BILBY git` repository at https://git.ligo.org/lscsoft/bilby/blob/master/examples/supernova_example/supernova_example.py.

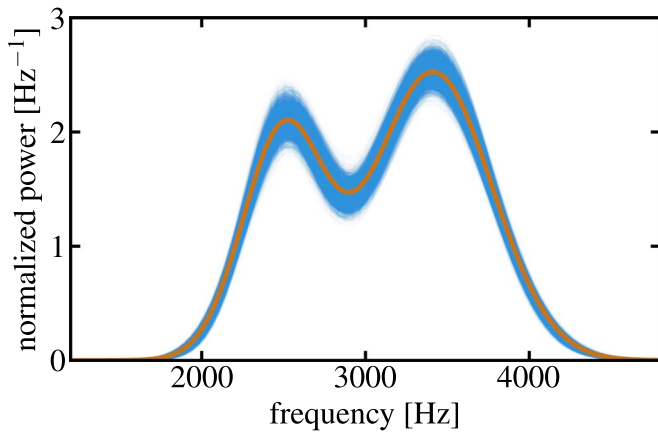


Figure 8. Proxy post-merger gravitational-wave signal from a short-lived neutron star showing the implementation of a different likelihood function in BILBY. The orange curve is an injected, double-peaked Gaussian signal injected into a constant noise realization. The blue band shows the waveform reconstructions from the posterior samples using a power-spectrum likelihood function, i.e., one that only uses the amplitude of the signal and ignores the phase.

is not surprising given that the interferometers were not operating at design sensitivity and the distances involved.

Provided the sensitivity of gravitational-wave interferometers continues to increase, it is possible that a gravitational-wave signal from a post-merger remnant could be detected in the relatively near future. Such a detection would provide an excellent opportunity to understand the nuclear equation of state of matter at extreme densities, as well as the rich physics of these exotic objects (e.g., Shibata & Taniguchi 2006; Baiotti et al. 2008; Read et al. 2013). Parameter inference of such short-lived signals is in its infancy (e.g., see Chatziioannou et al. 2017), largely due to the paucity of reliable waveforms (Clark et al. 2016; Easter et al. 2018). This is an ongoing challenge due to the expensive nature of numerical relativity simulations and the complex physics that must be included in such simulations.

Simple models that provide approximate gravitational-wave signals fit to a handful of numerical relativity waveforms exist (Messenger et al. 2014; Bose et al. 2018; Easter et al. 2018), which may eventually be used for full parameter inference. The phase evolution of such numerical relativity simulations is rapid and very difficult to model (Messenger et al. 2014; Easter et al. 2018). However, it is the frequency content of the signal that carries information about the equation of state and the physics of the remnant (e.g., Takami et al. 2015 and references therein). It is therefore possible that parameter estimation algorithms may require one to throw away information about the phase and only keep amplitude spectral content. Such a process requires a different likelihood function than the one that has been used to this point. This, therefore, provides good motivation for showing how to include a different likelihood function in the BILBY code.

We implement a power-spectral density (“burst”) likelihood,

$$\ln \mathcal{L}(|d| | \theta) = \sum_{i=1}^N \left[\ln I_0 \left(\frac{|\tilde{h}_i(\theta)| |\tilde{d}_i|}{S_n(f_i)} \right) - \frac{|\tilde{h}_i|^2 + |\tilde{d}_i|^2}{2S_n(f_i)} \right] + \ln |\tilde{h}_i(\theta)| - \ln S_n(f_i), \quad (4)$$

where I_0 is the zeroth-order modified Bessel function of the first kind. This requires setting up a new Likelihood class that contains a `log_likelihood` function that reads in the frequency array, noise spectral density, and waveform model and outputs a single likelihood evaluation. Having defined a new likelihood function, one calls the remaining functions in the usual way; the likelihood function is instantiated and passed to the `run_sampler()` command.

We inject a double-peaked Gaussian, shown in Figure 8 as the orange curve. We recover this signal using the same model (with a constant noise spectral density), where we use uniform priors for the amplitudes, widths, and frequencies of each of the peaks. Figure 8 shows the waveform reconstruction for each of the posterior samples, which can be seen to cover the injected signal.

6. Population Inference: Hyperparameterizations

Individual detections of binary coalescences can provide stunning insights into various physical and astrophysical questions. Increased detector sensitivities imply that significantly more events will be detected, enabling statements to also be made about the ensemble properties of populations (e.g., Abbott et al. 2016a; Farr et al. 2018; Smith & Thrane 2018; Talbot & Thrane 2018; Taylor & Gerosa 2018; Wysocki et al. 2018; Roulet & Zaldarriaga 2019; and references therein). Extracting information from a population of events is performed using hierarchical Bayesian inference, where the population is described by a set of hyperparameters, Λ . BILBY has built-in support for calculating Λ from multiple sets of posterior samples from individual events.

BILBY implements the conventional method whereby the posterior samples θ_j^i for each event j are reweighted according to the ratio of the population model prior $\pi(\theta|\Lambda)$ and the sampling prior $\pi(\theta)$ to obtain the hyperparameter likelihood

$$\mathcal{L}(h|\Lambda) = \prod_j \frac{\mathcal{Z}_j}{n_j} \sum_i \frac{\pi(\theta_j^i|\Lambda)}{\pi(\theta_j^i)}. \quad (5)$$

Here \mathcal{Z}_j is the Bayesian evidence for the data given the original model and n_j is the number of posterior samples in the j th event.

The BILBY implementation requires the user to define $\pi(\theta|\Lambda)$ and $\pi(\theta)$, which, along with the set of posterior samples θ_j^i , are passed to the `HyperparameterLikelihood` in BILBY’s `hyper` package. The hyperparameter priors are then set up in the usual way and passed to the standard `run_sampler` function.

As a demonstration³³ of this method, we reproduce the results of Talbot & Thrane (2018), recovering parameters describing a postulated excess of black holes due to pulsational pair-instability supernovae (PPSNe; Heger et al. 2003; Woosley & Heger 2015). The posterior distribution for the hyperparameters determining the abundance and characteristic mass of black holes formed through this mechanism is shown in Figure 9. The hyperparameter λ is the fraction of binaries where the more massive black hole formed through PPSNe, μ_{pp} is the typical mass of these black holes, and σ_{pp} determines the width of the “PPSN graveyard.”

This model contains seven additional hyperparameters describing the remainder of the distribution of black hole

³³ This example is found in the BILBY git repository at https://git.ligo.org/lscsoft/bilby/blob/master/examples/other_examples/hyper_parameter_exam.py.

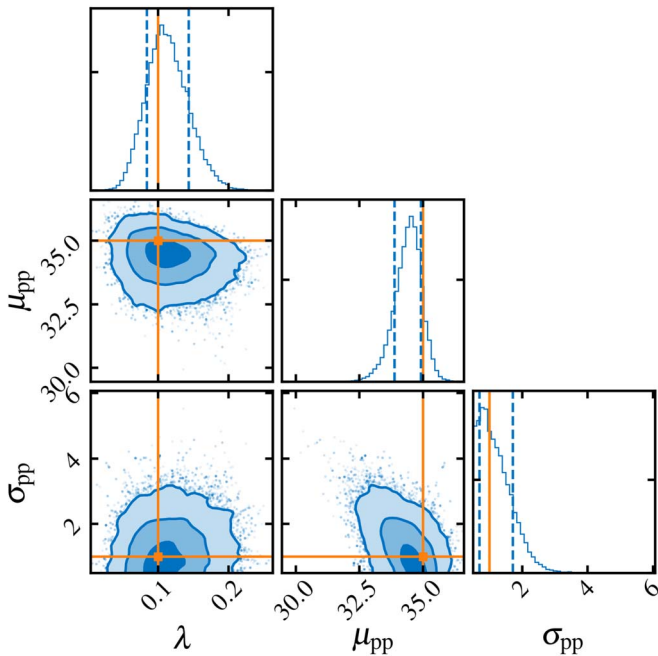


Figure 9. Population modeling with the BILBY hierarchical Bayesian inference module. We show the recovery of parameters describing part of the mass distribution of binary black holes using the model described in Talbot & Thrane (2018). The population parameters are drawn from values shown in orange and the posterior distributions for the hyperparameters shown in blue. Here λ is the fraction of binaries where the more massive black hole formed through PPSNe, and μ_{pp} and σ_{pp} are the typical mass of these black holes and the width of the “PPSN graveyard,” respectively.

masses that we hold fixed for the purposes of this example. Additional hyperparameters may be added straightforwardly.

7. Analysis of Arbitrary Data: An Example

BILBY is more than a tool for gravitational-wave astronomy; it can also be used as a generic and versatile inference package. In the documentation examples, we demonstrate how BILBY can be applied to generic time-domain data from radioactive decay processes. Furthermore, BILBY is currently being used to analyze radio and X-ray data from neutron stars and to study multimessenger signals associated with binary neutron star mergers. Here we show an example that calculates posterior distributions for one of the letters in the BILBY logo.

We import an image file containing the letter, map this to an $x - y$ coordinate system, and sample in both dimensions with likelihood

$$\ln \mathcal{L} \propto \frac{-1}{xy}, \quad (6)$$

assuming uniform priors on both variables. Figure 10 shows the posterior distribution for the “B” in the BILBY logo. All letters are shown in Figure 11, where the axis labels have been removed. The code for making this plot, and all other posterior distributions in the logo, are available with the `git` repository in `sample_logo.py`. Other examples of using BILBY with non-gravitational-wave data can be found in the `git` repository in the `tutorials` subdirectory.³⁴

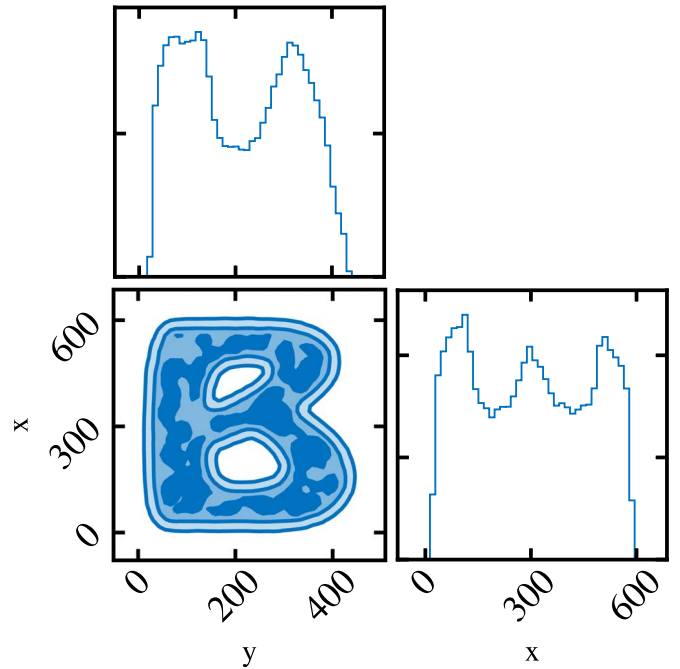


Figure 10. The “B” from the BILBY logo, generated using the BILBY package; see Section 7.



Figure 11. All letters from the BILBY logo, generated using the BILBY package; see Section 7.

8. Conclusion

Gravitational-wave astronomy is fast becoming a data-rich field. With the significantly increased activity in the field, there is a developing need for robust, easy-to-use inference software that is also modular and adaptable. We present BILBY: the Bayesian inference library for gravitational-wave astronomy. BILBY is open-source software that can be used to perform Bayesian inference. It is easily applied to data from LIGO/Virgo, including open data available from the Gravitational Wave Open Science Center. We access and manipulate LIGO data using `GWPY` (Macleod et al. 2018). Alternatively, BILBY may be used to study simulated data. BILBY can also be used to perform hierarchical Bayesian inference for population studies.

We present examples highlighting BILBY’s functionality and usability, including examples using open data from the first gravitational-wave detection, GW150914. Only five lines of code are required to reconstruct the astrophysical parameters of GW150914. One can redo the analysis using different priors, alternative waveform models, and/or a different sampling method with only modest changes. We show how to inject binary black hole and binary neutron star signals into Monte Carlo noise. We show how to define new gravitational-wave detectors.

We emphasize that BILBY is a front-end system that provides a unified interface to a variety of samplers, which are a primary workhorse of Bayesian inference. While numerous off-the-shelf samplers are implemented (see Section 3.2.1), to the best of our knowledge, there is no universal sampling solution to

³⁴ <https://git.ligo.org/lscsoft/bilby/tree/master/examples/tutorials>

gravitational-wave parameter estimation problems. BILBY is therefore only as good as the implemented samplers; initial studies show that CPNest (Veitch et al. 2017), Dynesty, and emcee (Foreman-Mackey et al. 2013; Vousden et al. 2016) sample the extrinsic parameters of binary coalescences more accurately than Nestle and pyMultiNest (Buchner et al. 2014). A systematic comparison of all off-the-shelf and boutique samplers is currently underway using BILBY.

BILBY is designed so as to be applicable to arbitrary signal models, not just compact binary coalescences. To this end, we show two examples: one of an injected numerical relativity supernova waveform that we reconstruct using principal component analysis, and another using a proxy for a neutron star post-merger waveform. The former example highlights how users can include their own signal models to perform both injections and signal recoveries, while the latter example demonstrates the ability to add a likelihood function that is different from the standard gravitational-wave transient likelihood.

We are grateful to John Veitch and Christopher Berry, who provided valuable comments on the manuscript, and also the LIGO/Virgo Parameter Estimation group for insightful discussions. We thank the anonymous referee for valuable comments that have improved the manuscript. This work is supported through Australian Research Council (ARC) Centre of Excellence CE170100004. P.D.L. is supported through ARC Future Fellowship FT160100112 and ARC Discovery Project DP180103155. S.B. is partially supported by the Australian-American Fulbright Commission. M.D.P. is funded by the UK Science & Technology Facilities Council (STFC) under grant ST/N005422/1. E.T. is supported through ARC Future Fellowship FT150100281 and CE170100004. This research has made use of data, software, and/or web tools obtained from the Gravitational Wave Open Science Center (<https://www.gw-openscience.org>), a service of LIGO Laboratory, the LIGO Scientific Collaboration, and the Virgo Collaboration. LIGO is funded by the U.S. National Science Foundation. Virgo is funded by the French Centre National de Recherche Scientifique (CNRS), the Italian Istituto Nazionale della Fisica Nucleare (INFN), and the Dutch Nikhef, with contributions by Polish and Hungarian institutes.

Software: `scipy` (Jones et al. 2001), `numpy` (Oliphant 2006), `pandas` (McKinney 2010), `matplotlib` (Hunter 2007), `corner` (Foreman-Mackey 2016), `healpy` (Górski et al. 2005), `deepdish` (<https://github.com/uchicago-cs/deepdish>), `astropy` (Robitaille et al. 2013; Price-Whelan et al. 2018), LALSIMULATION (LIGO Scientific Collaboration 2018), `GWPY` (Macleod et al. 2018), `Dynesty` (<https://github.com/joshspeagle/dynesty>), `CPNest` (Veitch et al. 2017), `emcee` and `pemcee` (Foreman-Mackey et al. 2013; Vousden et al. 2016), `MultiNest` (Feroz & Hobson 2008; Feroz et al. 2009, 2013), `pyMultiNest` (Buchner et al. 2014), `Nestle` (<http://kylebarbary.com/nestle/>), `singularity` (<https://www.sylabs.io/singularity/>).

ORCID iDs

Paul D. Lasky  <https://orcid.org/0000-0003-3763-1386>
 Colm Talbot  <https://orcid.org/0000-0003-2053-5582>
 Kendall Ackley  <https://orcid.org/0000-0002-8648-0767>
 Marcus E. Lower  <https://orcid.org/0000-0001-9208-0009>
 Matthew D. Pitkin  <https://orcid.org/0000-0003-4548-526X>

Nikhil Sarin  <https://orcid.org/0000-0003-2700-1030>

References

- Aasi, J., Abbott, B. P., Abbott, R., et al. 2015, *CQGrA*, 32, 074001
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016a, *PhRvX*, 6, 041015
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016b, *PhRvL*, 116, 241103
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016c, *PhRvX*, 6, 041014
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016d, *PhRvL*, 116, 061102
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016e, *PhRvL*, 116, 241102
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016f, *PhRvL*, 116, 221101
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017a, *Natur*, 551, 85
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017b, *CQGrA*, 34, 44001
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017c, *ApJL*, 848, L13
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017d, *PhRvL*, 118, 221101
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017e, *ApJL*, 851, L35
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017f, *PhRvL*, 119, 141101
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017g, *PhRvL*, 119, 161101
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017h, *ApJL*, 848, L12
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2017i, *ApJL*, 851, L16
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2018a, *PhRvL*, 120, 031104
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2018b, *PhRvL*, 121, 161101
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2018c, *PhRvX*, 9, 011001
- Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2018d, *PhRvL*, 120, 201102
- Acerese, F., Agathos, M., Agatsuma, K., et al. 2015, *CQGrA*, 32, 024001
- Aso, Y., Michimura, Y., Somiya, K., et al. 2013, *PhRvD*, 88, 043007
- Baiotti, L., Giacomazzo, B., & Rezzolla, L. 2008, *PhRvD*, 78, 084033
- Biwer, C. M., Capano, C. D., De, Soumi, et al. 2019, *PASP*, 131, 024503
- Bose, S., Chakravarti, K., Rezzolla, L., et al. 2018, *PhRvL*, 120, 031102
- Brown, D. D., & Freise, A. 2014, *Finesse 2: Frequency domain Interferometer Simulation Software*, <http://www.gwoptics.org/finesse>
- Buchner, J., Georgakakis, A., Nandra, K., et al. 2014, *A&A*, 564, A125
- Chatziioannou, K., Clark, J. A., Bauswein, A., et al. 2017, *PhRvD*, 96, 124035
- Clark, J., Bauswein, A., Cadonati, L., et al. 2014, *PhRvD*, 90, 062004
- Clark, J. A., Bauswein, A., Stergioulas, N., & Shoemaker, D. 2016, *CQGrA*, 33, 085003
- De, S., Finstad, D., Lattimer, J. M., et al. 2018, *PhRvL*, 121, 091102
- Easter, P. J., Lasky, P. D., & Casey, A. R. 2018, arXiv:1811.11183
- Farr, B., Holz, D. E., & Farr, W. M. 2018, *ApJL*, 854, L9
- Farr, B., Ochsner, E., Farr, W. M., & O'Shaughnessy, R. 2014a, *PhRvD*, 90, 024018
- Farr, W. M. 2014, *Marginalisation of the Time Parameter in Gravitational Wave Parameter Estimation* LIGO Tech. Rep. T1400460-v2 (Alexandria, VA: National Science Foundation), <https://dcc.ligo.org/T1400460-v2/public>
- Farr, W. M., Farr, B., & Littenberg, T. 2014b, *Modelling Calibration Errors in CBC Waveforms* LIGO Tech. Rep. T1400682-v1 (Alexandria, VA: National Science Foundation), <https://dcc.ligo.org/LIGO-T1400682/public>
- Feroz, F., & Hobson, M. P. 2008, *MNRAS*, 384, 449
- Feroz, F., Hobson, M. P., & Bridges, M. 2009, *MNRAS*, 398, 1601
- Feroz, F., Hobson, M. P., Cameron, E., & Pettitt, A. N. 2013, arXiv:1306.2144
- Flanagan, É. É., & Hinderer, T. 2008, *PhRvD*, 77, 021502
- Foreman-Mackey, D. 2016, *JOSS*, 24
- Foreman-Mackey, D., Hogg, D. W., Lang, D., & Goodman, J. 2013, *PASP*, 125, 306
- Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, *ApJ*, 622, 759
- Heger, A., Fryer, C. L., Woosley, S. E., Langer, N., & Hartmann, D. H. 2003, *ApJ*, 591, 288
- Hunter, J. D. 2007, *CSE*, 9, 90
- Iyer, B. R., Souradeep, T., & Unnikrishnan, C. S. 2011, *Proposal of the Consortium for Indian Initiative in Gravitational-wave Observations* LIGO Tech. Rep. M1100296-v2, (Alexandria, VA: National Science Foundation), <https://dcc.ligo.org/LIGO-M1100296/public>
- Jones, E., Oliphant, T. E., Peterson, P., et al. 2001, *SciPy: Open Source Scientific Tools for Python*, <http://www.scipy.org/>
- Lackey, B. D., & Wade, L. 2015, *PhRvD*, 91, 043002
- LIGO Scientific Collaboration 2018, *LIGO Algorithm Library-LALSuite*, <https://lscsoft.docs.ligo.org/lalsuite/lalsimulation>
- Logue, J., Ott, C. D., Heng, I. S., Kamlus, P., & Scargill, J. H. C. 2012, *PhRvD*, 86, 044023
- Lower, M. E., Thrane, E., Lasky, P. D., & Smith, R. 2018, *PhRvD*, 98, 083028
- Macleod, D., Coughlin, S., Urban, A. L., et al. 2018, *gwpw/gwpy: v.0.12.0*, Zenodo, doi:10.5281/zenodo.1346349
- McKinney, W. 2010, in *Proc. 9th Python in Science Conf.*, ed. S. van der Walt & J. Millman, 51

- Messenger, C., Takami, K., Gossan, S., Rezzolla, L., & Sathyaprakash, B. S. 2014, [PhRvX](#), **4**, 041004
- Miller, J., Barsotti, L., Vitale, S., et al. 2015, [PhRvD](#), **91**, 062005
- Müller, E., Janka, H.-T., & Wongwathanarat, A. 2012, [A&A](#), **537**, A63
- Nitz, A., Harry, I., Brown, D., et al. 2018, gwastro/pycbc: PyCBC Release v.1.13.5, Zenodo, doi:[10.5281/zenodo.2581446](#)
- Oliphant, T. E. 2006, *A Guide to NumPy* (Cambridge: MIT Press)
- Pierce, B. 2002, *Types and Programming Languages* (Cambridge, MA: MIT Press)
- Powell, J., Gossan, S. E., Logue, J., & Heng, I. S. 2016, [PhRvD](#), **94**, 123012
- Powell, J., Szczepanczyk, M., & Heng, I. S. 2017, [PhRvD](#), **96**, 123013
- Price-Whelan, A. M., Sipőcz, B. M., Günther, H. M., et al. 2018, [AJ](#), **156**, 123
- Punturo, M., Abernathy, M., Acernese, F., et al. 2010, [CQGra](#), **27**, 194002
- Raymond, V., & Farr, W. M. 2014, arXiv:[1402.0053](#)
- Read, J. S., Baiotti, L., Creighton, J. D. E., et al. 2013, [PhRvD](#), **88**, 044042
- Robitaille, T. P., Tollerud, E. J., Greenfield, P., et al. 2013, [A&A](#), **558**, A33
- Roulet, J., & Zaldarriaga, M. 2019, [MNRAS](#), **484**, 4216
- Salvatier, J., Wiecki, T., & Fonnesbeck, C. 2016, [PeerJ Comp. Sci.](#), **2**, 55
- Schmidt, P., Hannam, M., & Husa, S. 2012, [PhRvD](#), **86**, 104063
- Shibata, M., & Taniguchi, K. 2006, [PhRvD](#), **73**, 064027
- Singer, L. P., Chen, H.-Y., Holz, D. E., et al. 2016, [ApJL](#), **829**, L15
- Singer, L. P., & Price, L. R. 2016, [PhRvD](#), **93**, 024013
- Skilling, J. 2004, in Proc. AIP Conf. 735, *Bayesian Inference and Maximum Entropy Methods in Science and Engineering*, ed. R. Fischer, R. Preuss, & U. von Toussaint (Melville, NY: AIP), 395
- Skilling, J. 2006, [BayAn](#), **1**, 833
- Smith, R., & Thrane, E. 2018, [PhRvX](#), **8**, 021019
- Takami, K., Rezzolla, L., & Baiotti, L. 2015, [PhRvD](#), **91**, 064001
- Talbot, C., & Thrane, E. 2018, [ApJ](#), **856**, 173
- Taylor, S. R., & Gerosa, D. 2018, [PhRvD](#), **98**, 083017
- Thrane, E., & Talbot, C. 2018, arXiv:[1809.02293](#)
- Vallisneri, M., Kanner, J., Williams, R., Weinstein, A., & Stephens, B. 2015, [JPhCS](#), **610**, 012021
- van der Sluys, M., Raymond, V., Mandel, I., et al. 2008b, [CQGra](#), **25**, 184011
- van der Sluys, M. V., Röver, C., Stroer, A., et al. 2008a, [ApJL](#), **688**, L61
- Veitch, J., Del Pozzo, W., Pitkin, M., et al. 2017, johnveitch/cpnest: Minor Optimisation, Zenodo, doi:[10.5281/zenodo.835874](#)
- Veitch, J., Raymond, V., Farr, B., et al. 2015, [PhRvD](#), **91**, 042003
- Veitch, J., & Vecchio, A. 2008, [PhRvD](#), **78**, 022001
- Vousden, W. D., Farr, W. M., & Mandel, I. 2016, [MNRAS](#), **455**, 1919
- Woosley, S. E., & Heger, A. 2015, [ASSL](#), **412**, 199
- Wysocki, D., Gerosa, D., O'Shaughnessy, R., et al. 2018, [PhRvD](#), **97**, 043014