The Astrophysical Journal Supplement Series, 226:10 (8pp), 2016 September
doi:10.3847/0067-0049/226/1/10

SUPPLEMENT: "GOING THE DISTANCE: MAPPING HOST GALAXIES OF LIGO AND VIRGO SOURCES IN THREE DIMENSIONS USING LOCAL COSMOGRAPHY AND TARGETED FOLLOW-UP" (2016, ApJL, 829, L15)

Leo P. Singer[1,13], Hsin-Yu Chen[2], Daniel E. Holz[2], Will M. Farr[3], Larry R. Price[4], Vivien Raymond[4,5], S. Bradley Cenko[1,6], Neil Gehrels[1], John Cannizzo[1], Mansi M. Kasliwal[7], Samaya Nissanke[8], Michael Coughlin[9], Ben Farr[2], Alex L. Urban[10], Salvatore Vitale[11], John Veitch[3], Philip Graff[12], Christopher P. L. Berry[3], Satya Mohapatra[11], and Ilya Mandel[3]

[1] Astroparticle Physics Laboratory, NASA Goddard Space Flight Center, Mail Code 661, Greenbelt, MD 20771, USA
[2] Department of Physics, Enrico Fermi Institute, and Kavli Institute for Cosmological Physics, University of Chicago, Chicago, IL 60637, USA
[3] School of Physics and Astronomy, University of Birmingham, Birmingham B15 2TT, UK
[4] LIGO Laboratory, California Institute of Technology, Pasadena, CA 91125, USA
[5] Albert-Einstein-Institut, Max-Planck-Institut für Gravitationsphysik, D-14476 Potsdam-Golm, Germany
[6] Joint Space-Science Institute, University of Maryland, College Park, MD 20742, USA
[7] Cahill Center for Astrophysics, California Institute of Technology, Pasadena, CA 91125, USA
[8] Institute of Mathematics, Astrophysics and Particle Physics, Radboud University, Heyendaalseweg 135, 6525 AJ Nijmegen, The Netherlands
[9] Department of Physics and Astronomy, Harvard University, Cambridge, MA 02138, USA
[10] Leonard E. Parker Center for Gravitation, Cosmology, and Astrophysics, University of Wisconsin–Milwaukee, Milwaukee, WI 53201, USA
[11] LIGO Laboratory, Massachusetts Institute of Technology, 185 Albany Street, Cambridge, MA 02139, USA
[12] Department of Physics, University of Maryland, College Park, MD 20742, USA
Received 2016 May 13; accepted 2016 August 5; published 2016 September 21

ABSTRACT

This is a supplement to the Letter of Singer et al., in which we demonstrated a rapid algorithm for obtaining joint 3D estimates of sky location and luminosity distance from observations of binary neutron star mergers with Advanced LIGO and Virgo. We argued that combining the reconstructed volumes with positions and redshifts of possible host galaxies can provide large-aperture but small field of view instruments with a manageable list of targets to search for optical or infrared emission. In this Supplement, we document the new HEALPix-based file format for 3D localizations of gravitational-wave transients. We include Python sample code to show the reader how to perform simple manipulations of the 3D sky maps and extract ranked lists of likely host galaxies. Finally, we include mathematical details of the rapid volume reconstruction algorithm.

Key words: catalogs – galaxies: distances and redshifts – gravitational waves – surveys

1. OUTLINE OF THIS SUPPLEMENT

In Singer et al. (2016), we discussed the measurement of luminosity distances of compact binary coalescence (CBC) events using the Advanced Laser Interferometer Gravitational-wave Observatory (LIGO) and Virgo ground-based interferometric gravitational-wave (GW) detectors. In the Letter, an algorithm was introduced for rapidly extracting directionally dependent distance estimates from GW observations and illustrated the typical 3D shape of GW volume reconstructions during early Advanced LIGO. Finally, we argued that the 3D structure and distance information can be leveraged to guide searches of likely nearby host galaxies for X-ray, optical, and infrared counterparts of binary neutron star (BNS) mergers.

This Supplement provides the following supporting material. First, in Section 2, we document a file format for the rapid transmission of 3D volume reconstructions in GW alerts. It is based on and is backward-compatible with the 2D localization formation that we introduced in Singer et al. (2014) and that was employed in GW alerts that were sent in Advanced LIGO's first observing run (O1; Abbott et al. 2016). Second, in Section 3, we describe the online data release, which provides a browsable collection of simulated 3D sky maps. Third, in Section 4, we provide a Python primer for performing basic operations on 3D sky maps, all the way through selecting a list of the most likely host galaxies. In Section 5, we provide additional details of the position reconstruction algorithm. Finally, in Section 6, we show that the algorithm produces faithful representations of the full 3D probability distributions.

The reader who is interested in leveraging GW distance information for planning electromagnetic (EM) follow-up observations or performing archival research needs only consult Sections 2 and 4.

2. 3D LOCALIZATION FILE FORMAT

The 3D localization for a single GW candidate is stored as a Flexible Image Transport System (FITS; Wells et al. 1981) file. The FITS file contains a single binary table (Cotton et al. 1995) that represents a Hierarchical Equal Area isoLatitude Pixelization (HEALPix; Górski et al. 2005) all-sky image. The table has four floating-point columns, listed in Table 1, which represent four channels of the HEALPix image. The first column, PROB, is simply the probability that the source is contained within the pixel i that is centered on the direction $\boldsymbol{n}_i$, the same as in the 2D localization format. The second and third columns, DISTMU and DISTSTD, are the ansatz location and scale parameters, respectively. The fourth column, DISTNORM, is the ansatz normalization coefficient, included for convenience.

In pixels on the sky that contain very little probability, sometimes the conditional distance distribution cannot be represented using the ansatz. This is signaled by DISTMU $= \infty$, DISTSIGMA $= 1$, and DISTNORM $= 0$.

[13] NASA Postdoctoral Program Fellow.

Table 1
HEALPix Columns

FITS Name	Symbol	Units	Description
`PROB`	ρ_i	pixel^{-1}	Probability that the source is contained in pixel i, centered on the direction $\boldsymbol{n}_i$
`DISTMU`	$\hat{\mu}_i$	Mpc	Ansatz location parameter of conditional distance distribution in direction $\boldsymbol{n}_i$, or ∞ if invalid
`DISTSIGMA`	$\hat{\sigma}_i$	Mpc	Ansatz scale parameter of conditional distance distribution in direction $\boldsymbol{n}_i$, or 1 if invalid
`DISTNORM`	$\hat{N}_i$	Mpc^{-2}	Ansatz normalization coefficient, or 0 if invalid

Table 2
Example FITS Header

Key	Value	Comment
	HDU 0	
SIMPLE	T	conforms to FITS standard
BITPIX	8	array data type
NAXIS	0	number of array dimensions
EXTEND	T	
	HDU 1	
XTENSION	'BINTABLE'	binary table extension
BITPIX	8	array data type
NAXIS	2	number of array dimensions
NAXIS1	16384	length of dimension 1
NAXIS2	3072	length of dimension 2
PCOUNT	0	number of group parameters
GCOUNT	1	number of groups
TFIELDS	4	number of table fields
TTYPE1	'PROB'	
TFORM1	'1024E'	
TUNIT1	'pix-1'	
TTYPE2	'DISTMU'	
TFORM2	'1024E'	
TUNIT2	'Mpc'	
TTYPE3	'DISTSIGMA'	
TFORM3	'1024E'	
TUNIT3	'Mpc'	
TTYPE4	'DISTNORM'	
TFORM4	'1024E'	
TUNIT4	'Mpc-2'	
PIXTYPE	'HEALPIX'	HEALPIX pixelisation
ORDERING	'NESTED'	Pixel ordering scheme, either RING or NESTED
COORDSYS	'C'	Ecliptic, Galactic or Celestial (equatorial)
EXTNAME	'xtension'	name of this binary table extension
NSIDE	512	Resolution parameter of HEALPIX
FIRSTPIX	0	First pixel # (0 based)
LASTPIX	3145727	Last pixel # (0 based)
INDXSCHM	'IMPLICIT'	Indexing: IMPLICIT or EXPLICIT
OBJECT	'coinc_event:coinc_event_id:18951'	Unique identifier for this event
INSTRUME	'H1, L1'	Instruments that triggered this event
DATE-OBS	'2010-09-03T06:12:26.60324'	UTC date of the observation
MJD-OBS	55442.2586412414	modified Julian date of the observation
DATE	'2015-04-13T10:17:11'	UTC date of file creation
CREATOR	'bayestar_localize_coincs.py'	Program that created this file
DISTMEAN	68.54061620909769	Posterior mean distance in Mpc
DISTSTD	17.14006463067744	Posterior standard deviation of distance in Mpc

The FITS header, an example of which is shown in Table 2, provides metadata including the UTC time of the GW trigger and the list of GW instruments that contributed to the localization. The header also provides values for `DISTMEAN` and `DISTSTD`, respectively, being the posterior mean and standard deviation of distance marginalized over the whole sky.

3. DATA RELEASE

An online data release provides a browsable catalog of simulated 3D GW localizations. One may select events from O1 or Advanced LIGO's second observing run (O2). Events may be sorted by detector network (a one- or two-letter combination consisting of "H" for LIGO Hanford Observatory, "L" for LIGO Livingston Observatory, "V" for Virgo), 90% credible volume in Mpc^3, 90% credible area in deg^2, or supernova (SN). For each event, a BAYESian TriAngulation and Rapid localization (BAYESTAR) or LALInference FITS file may be downloaded. A screen shot of the data release is shown in Figure 1.

4. PYTHON EXAMPLE CODE

In this section, we provide some Python sample code to perform some simple manipulations of 3D sky maps. The triple greater-than signs (>>>) and triple-dots (...) are the Python interactive prompt; the reader should type everything on the line *after* these.

4.1. Python Environment

These examples will work in Python 2.7 and later on Linux or UNIX systems. If the reader does not already have a Python environment of preference, we suggest the Anaconda Python distribution[14] for desktop use or the lightweight Miniconda variant[15] for computing clusters. Only the Astropy, Healpy, and Numpy packages are essential for working with the 3D localizations, but the examples below will also use Matplotlib, Scipy, and Astroquery. All of these packages can be installed with Pip[16]:

```
$ pip install astropy astroquery healpy matplotlib scipy
```

4.2. Reading Sky Maps

For all of the samples below, start by importing the Healpy for working with HEALPix files, the Numpy for vector operations, Matplotlib for plotting, and Scipy for probability functions:

```
$ python
>>> import healpy as hp
>>> import numpy as np
>>> from matplotlib import pyplot as plt
>>> from scipy.stats import norm
```

Next, select download an example sky map from the data release. In this example, we use the simulated event that is shown in Figures 1 and 2 of Singer et al. (2016). A convenient way to download it is using Astropy's `download_file` utility, which will retrieve the file and cache it locally:

```
>>> from astropy.utils.data import download_file
>>> url
= ('https://dcc.ligo.org/P1500071/public'
... + '/18951_bayestar.fits.gz')
>>> filename = download_file(url, cache = True)
```

The new 3D localization format is backward-compatible with the 2D format introduced in Singer et al. (2014). By default, when we read the HEALPix file with Healpy (or any other common-place HEALPix library or tool), we get just the first layer, the probability sky map:

```
>>> prob = hp.read_map(filename)
```

To read both the probability layer and the three additional distance layers, we need to pass the optional `field =` parameter to Healpy:

```
>>> prob, distmu, distsigma, distnorm = hp.read_map(
... filename, field = [0, 1, 2, 3])
```

or slightly more concisely:

```
>>> prob, distmu, distsigma, distnorm = hp.read_map(
... filename, field = range(4))
```

Last, it will be useful for subsequent Healpy calls to have the HEALPix resolution on hand:

```
>>> npix = len(prob)
>>> npix
3145728
>>> nside = hp.npix2nside(npix)
>>> nside
512
```

4.3. 2D Probability in a Given Line of Sight

In this example, we compute the 2D probability per steradian or per deg^2 that the source is in a given direction. Let's take as an example the following equatorial coordinates:

```
>>> ra, dec = 137.8, -39.9
```

which, coincidentally, happen to be the true simulated position to the source.

Healpy uses "physicist's" spherical coordinates (θ, ϕ), with $\theta \in [0, \pi]$ being the colatitude from the north celestial pole in radians, and $\phi \in [0, 2\pi)$ being the right ascension in radians. We convert

```
>>> theta = 0.5 * np.pi - np.deg2rad(dec)
>>> phi = np.deg2rad(ra)
```

Next, we use Healpy to look up the index of the HEALPix pixel that contains that direction:

```
>>> ipix = hp.ang2pix(nside, theta, phi)
>>> ipix
2582288
```

Healpy will tell us the area per pixel in steradians at the current HEALPix resolution:

```
>>> pixarea = hp.nside2pixarea(nside)
>>> pixarea
3.994741635118857e-06
```

or in deg^2:

```
>>> pixarea_deg2 = hp.nside2pixarea(nside, degrees = True)
>>> pixarea_deg2
0.013113963206424481
```

All that is left to do is look up the probability contained within pixel `ipix` and (if desired) divide by the area per pixel to obtain the probability per steradian:

```
>>> dp_dA = prob[ipix] / pixarea
>>> dp_dA
7.4387317043042076
```

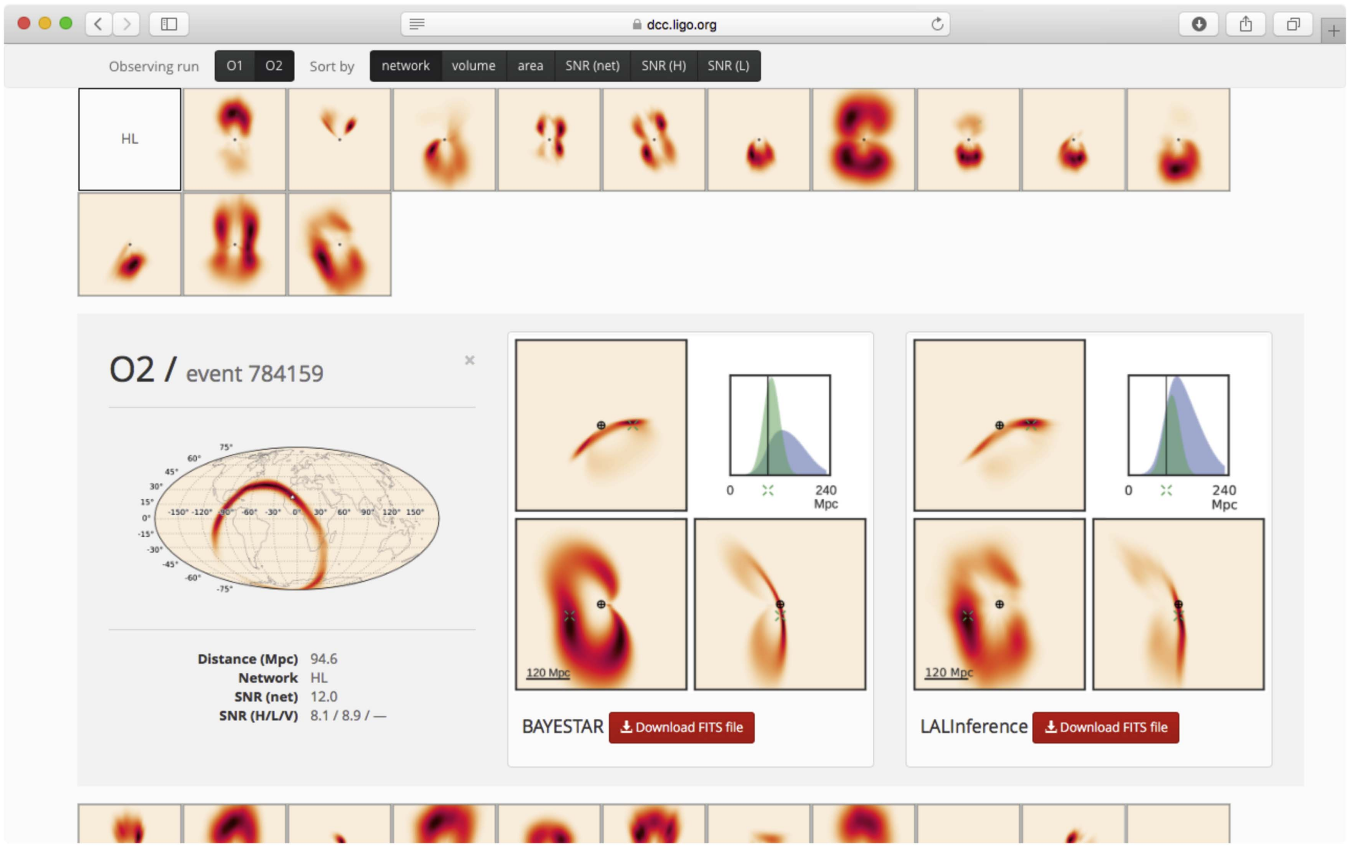
or the probability per deg^2:

```
>>> dp_dA_deg2 = prob[ipix] / pixarea_deg2
>>> dp_dA_deg2
0.0022659672582507331
```

4.4. Conditional Distance Distribution along a Line of Sight

Next, we calculate the conditional distance distribution along a given line of sight, which is the probability per unit distance under the assumption that the source is in a given direction. We will use the same sky position as in the example above. We lay

[14] https://www.continuum.io/anaconda
[15] http://conda.pydata.org/miniconda.html
[16] https://pip.pypa.io



Distance (Mpc) 94.6
Network HL
SNR (net) 12.0
SNR (H/L/V) 8.1 / 8.9 / —

Figure 1. Screen shot of data release page.

out a grid in distance along that line of sight:

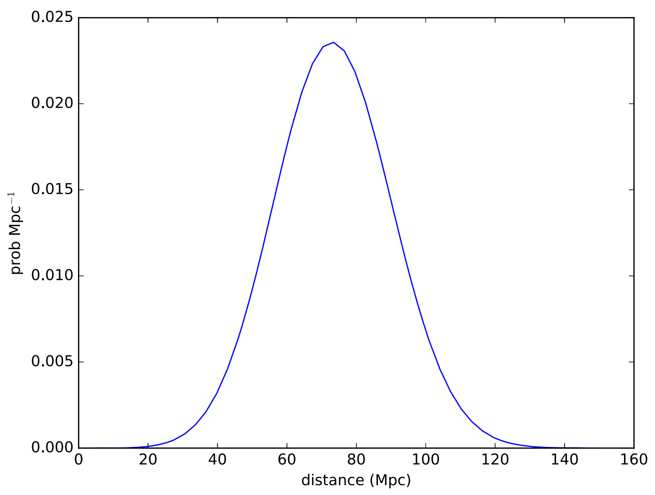
```
>>> r = np.linspace(0, 150)
```

Then, we plug everything into the ansatz distribution:

```
>>> dp_dr = r ** 2 * distnorm[ipix] * norm(
... distmu[ipix], distsigma[ipix]).pdf(r)
```

Finally, we plot the result:

```
>>> plt.plot(r, dp_dr)
>>> plt.xlabel('distance_(Mpc)')
>>> plt.ylabel('prob_Mpc$^{-1}$')
>>> plt.show()
```



4.5. Probability per Unit Volume at a Point

Now, we calculate the probability density per Mpc^3 at a point. We will use the same right ascension and declination as above and a distance of 74.8 Mpc:

```
>>>r = 74.8
```

Finally,

```
>>> dp_dV = prob[ipix] * distnorm[ipix] * norm(
... distmu[ipix], distsigma[ipix]).pdf(r) / pixarea
>>> dp_dV
3.1173200109121657e-05
```

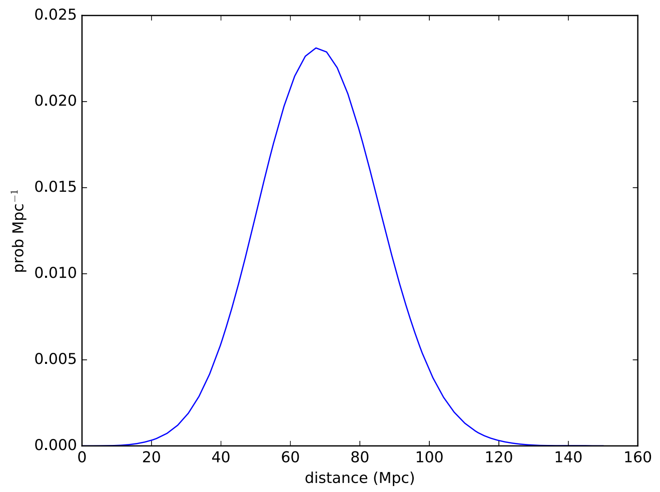
4.6. Marginal Distance Distribution Integrated over the Sky

As our next example, we compute the marginal distance distribution, the probability density per unit distance integrated over the entire sky:

```
>>> r = np.linspace(0, 150)
>>> dp_dr = [np.sum(prob * rr ** 2 * distnorm
... * norm(distmu, distsigma).pdf(rr)) for rr in r]
```

Finally, we plot the result:

```
>>> plt.plot(r, dp_dr)
>>> plt.xlabel('distance_(Mpc)')
>>> plt.ylabel('prob_Mpc$^{-1}$')
>>> plt.show()
```



4.7. Ranked List of Galaxies

As our final example, we will generate a ranked list of galaxies. For the purpose of this demonstration, we will use the 2MASS Redshift Survey (2MRS; Huchra et al. 2012) because it is a flux-limited all-sky spectroscopic redshift catalog. This greatly simplifies the issues of completeness, sky coverage, and accuracy of redshift estimates. First, download the entire catalog from VizieR (Ochsenbein et al. 2000) using Astroquery:

```
>>> from astroquery.vizier import Vizier
>>> Vizier.ROW_LIMIT = -1
>>> cat, = Vizier.get_catalogs('J/ApJS/199/26/table3')
```

According to Tully (2015), the 2MRS luminosity function is well fit by a Schechter function with a cutoff absolute magnitude of $M_K^* = -23.55$ and a power-law index of $\alpha_K = -1$. We find the maximum absolute magnitude $M_K^{\max}$ for a completeness fraction of 0.5:

```
>>> from scipy.special import gammaincinv
>>> completeness = 0.5
>>> alpha = -1.0
>>> MK_star = -23.55
>>> MK_max = MK_star + 2.5 * np.log10(
... gammaincinv(alpha + 2, completeness))
>>> MK_max
-23.947936347387156
```

We select only galaxies with positive redshifts and absolute magnitudes greater than $M_K^{\max}$:

```
>>> from astropy.cosmology import WMAP9 as cosmo
>>> from astropy.table import Column
>>> import astropy.units as u
>>> import astropy.constants as c
>>> z = (u.Quantity(cat['cz']) / c.c).to(
... u.dimensionless_unscaled)
>>> MK = cat['Ktmag'] - cosmo.distmod(z)
>>> keep = (z > 0) & (MK < MK_max)
>>> cat = cat[keep]
>>> z = z[keep]
```

Then, we calculate the luminosity distance and HEALPix index of each galaxy:

```
>>> r = cosmo.luminosity_distance(z).to('Mpc').value
>>> theta = 0.5 * np.pi - cat['_DEJ2000'].to('rad').value
>>> phi = cat['_RAJ2000'].to('rad').value
>>> ipix = hp.ang2pix(nside, theta, phi)
```

We find the probability density per unit volume at the position of each galaxy:

```
>>> dp_dV = prob[ipix] * distnorm[ipix] * norm(
... distmu[ipix], distsigma[ipix]).pdf(r) / pixarea
```

Finally, we sort the galaxies by descending probability density and take the top 50:

```
>>> top50 = cat[np.flipud(np.argsort(dp_dV))][:50]
>>> top50['_RAJ2000', '_DEJ2000', 'Ktmag']
<Table masked = True length = 50>
  _RAJ2000    _DEJ2000    Ktmag
    deg         deg        mag
  float64     float64    float32
----------- ----------- --------
  344.01190    36.36136    8.772
  343.81122    36.67177    9.958
  137.19089   -38.60788    9.566
  334.86545    29.39581    9.835
  359.81589    46.88923    9.307
    0.00695    47.27456    9.499
        ...         ...      ...
  123.16494   -16.05073    9.727
  341.26642    33.99616    9.799
  339.33075    34.44790    9.204
  137.27219   -35.90259   10.822
  188.13953   -68.53886    9.609
  339.01483    33.97575   10.032
```

5. VOLUME RECONSTRUCTION ALGORITHM

The volume reconstruction algorithm consists of a computationally trivial postprocessing stage that is added to the two established LIGO/Virgo methods for localization of CBC events, the BAYESTAR rapid triangulation code (Singer et al. 2014; Singer 2015), and the LALInference parameter estimation pipeline (Aasi et al. 2013).

The conditional mean and standard deviation of distance are extracted from BAYESTAR as described in Section 5.1 and from LALInference as explained in Section 5.2 below. Then, the mean and standard deviation are converted to the ansatz parameters as described in Section 5.3.

5.1. Volume Reconstruction in BAYESTAR

BAYESTAR (Singer et al. 2014; Singer 2015) is a rapid position reconstruction algorithm for BNS mergers. Its inputs are a trio of numbers for each detector: the matched-filter estimates of the arrival time, phase, and amplitude at each GW site. It marginalizes over polarization angle, inclination angle, coalescence time, and distance by performing low-order Gaussian quadrature integration in a series of nested loops. The output is a HEALPix all-sky map of posterior probability, consisting of N_{pix} equal-area pixels.

The BAYESTAR distance prior is a power law of r^k, with k being supplied by the user and normally set to $k = 2$ for a spatially homogeneous source population. To evaluate the distances, we run BAYESTAR two more times, with $k' = k + 1$ and $k'' = k + 2$. The resulting three sky maps are denoted $\langle 1 \rangle$, $\langle r \rangle$, and $\langle r^2 \rangle$. The HEALPix-sampled marginal sky posterior $\hat{\rho}$, conditional mean distance $\hat{m}$, and conditional standard deviation of distance $\hat{s}$ are then given by

$$\hat{\rho} = \langle 1 \rangle, \tag{1}$$

$$\hat{m} = \langle r \rangle / \langle 1 \rangle, \text{ and} \tag{2}$$

$$\hat{s} = \sqrt{\langle r^2 \rangle / \langle 1 \rangle - \hat{m}^2}. \tag{3}$$

Finally, the moments $\hat{m}$ and $\hat{s}$ are converted to the ansatz parameters $\hat{\mu}$, $\hat{\sigma}$, and $\hat{N}$ using the procedure described in Section 5.3 below.

BAYESTAR takes about a minute to run (Singer 2015); the conventional one-dimensional sky map is ready with a response time of a few minutes. Since we will now run BAYESTAR three times, the total number of computations will increase by about a factor of 3. Fortunately, since BAYESTAR is able to make effective use of many CPU cores, we can offset the modest increase in computational cost by moving the analysis to a machine with more cores, resulting in a negligible overall change in running time.

5.2. Volume Reconstruction in LALInference

LALInference (Aasi et al. 2013) is the Advanced LIGO Bayesian parameter estimation library. It includes several algorithms that perform full modeling of the GW signal and stochastic sampling of the CBC parameter space. The inputs to LALInference are the GW time series from all of the detectors. The output is a cloud of sample points drawn from the GW posterior.

The samples are converted to a smooth multidimensional probability distribution by clustering them into N disjoint sets, each consisting of N_i spatially neighboring points, and building a kernel density estimator (KDE) for each cluster.[17] In Cartesian coordinates, the smoothed distribution is given by a double sum over the clusters and the samples within each cluster:

$$p(\boldsymbol{x}) = \sum_{i=1}^{N} W_i\, |2\pi \boldsymbol{C}_i|^{-1/2} N_i^{-1} \sum_{j=1}^{N_i} \exp\left[-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{X}_{ij})^{\mathsf{T}} \boldsymbol{C}_i^{-1} (\boldsymbol{x} - \boldsymbol{X}_{ij})\right]. \tag{4}$$

Here, W_i is a weight associated with cluster i, and each cluster is described by its KDE covariance $\boldsymbol{C}_i$ and N_i samples $\boldsymbol{X}_{ij}$. The $|\ldots|$ denotes the matrix determinant.

The stochastic sampling takes hours to weeks depending on the sophistication of the waveform models that are used and on the treatment of uncertainty in detector calibration. It takes up to tens of minutes to build the KDE.

We can exactly calculate the conditional mean and standard deviation of the distance for the KDE posterior. First, we evaluate Equation (4) at the position $\boldsymbol{x} = r\boldsymbol{n}$:

$$p(r\boldsymbol{n}) = \sum_{i=1}^{N} (2\pi c_i)^{-1/2} N_i^{-1} \sum_{j=1}^{N_i} w_{ij} \exp\left[\frac{-(r - x_{ij})^2}{2c_i}\right], \tag{5}$$

with

$$c_i = (\boldsymbol{n}^{\mathsf{T}} \boldsymbol{C}_i^{-1} \boldsymbol{n})^{-1}, \tag{6}$$

$$x_{ij} = (\boldsymbol{n}^{\mathsf{T}} \boldsymbol{C}_i^{-1} \boldsymbol{X}_{ij}) c_i, \text{ and} \tag{7}$$

$$w_{ij} = \frac{1}{2\pi}\sqrt{\frac{c_i}{|\boldsymbol{C}_i|}} \exp\left[\frac{1}{2}\left(\frac{x_{ij}{}^2}{c_i} - \boldsymbol{X}_{ij}^{\mathsf{T}} \boldsymbol{C}_i^{-1} \boldsymbol{X}_{ij}\right)\right] W_i. \tag{8}$$

We compute the integrals of 1, r, and r^2, weighted by $p(r\boldsymbol{n})r^2$:

$$\langle 1 \rangle = \int_0^\infty p(r\boldsymbol{n}) r^2\, dr = \sum_{ij} w_{ij} \langle 1_{ij} \rangle / N_i, \tag{9}$$

$$\langle r \rangle = \int_0^\infty p(r\boldsymbol{n}) r^3\, dr = \sum_{ij} w_{ij} \langle r_{ij} \rangle / N_i, \tag{10}$$

$$\langle r^2 \rangle = \int_0^\infty p(r\boldsymbol{n}) r^4\, dr = \sum_{ij} w_{ij} \langle r^2 \rangle_{ij} / N_i, \tag{11}$$

with

$$\langle 1 \rangle_{ij} = (x_{ij}{}^2 + c_i)a + x_{ij} b, \tag{12}$$

$$\langle r \rangle_{ij} = (x_{ij}{}^3 + 3x_{ij}c_i)a + (x_{ij}{}^2 + 2c_i)b, \tag{13}$$

$$\langle r^2 \rangle_{ij} = (x_{ij}{}^4 + 6x_{ij}{}^2 c_i + 3c_i{}^2)a + (x_{ij}{}^3 + 5x_{ij}c_i)b, \tag{14}$$

$$a = \frac{1}{2}\left(1 + \operatorname{erf}\left[\frac{x_{ij}}{\sqrt{2c_i}}\right]\right), \text{ and} \tag{15}$$

$$b = \sqrt{\frac{c_i}{2\pi}} \exp\left[\frac{-x_{ij}{}^2}{2c_i}\right]. \tag{16}$$

Then $\langle 1 \rangle$, $\langle r \rangle$, and $\langle r^2 \rangle$ are converted to $\hat{\rho}$, $\hat{m}$, and $\hat{s}$ using Equations (1)–(3). Finally, $\hat{m}$ and $\hat{s}$ are converted to $\hat{\mu}$, $\hat{\sigma}$, and $\hat{N}$ using the procedure described in Section 5.3 below.

5.3. Method of Moments

For both BAYESTAR and LALInference, the parameters of the ansatz distribution are extracted using the method of moments. The ansatz is that the conditional distribution of distance is described by the function

$$p(r|\boldsymbol{n}) = \frac{N(\boldsymbol{n})}{\sqrt{2\pi}\,\sigma(\boldsymbol{n})} \exp\left[-\frac{(r - \mu(\boldsymbol{n}))^2}{2\sigma(\boldsymbol{n})^2}\right] r^2 \quad \text{for } r \geqslant 0. \tag{17}$$

The nth moment of the distance ansatz is

$$\overline{r^n}(\mu, \sigma) = \int_0^\infty \frac{N}{\sqrt{2\pi}\,\sigma} \exp\left[-\frac{(r - \mu)^2}{2\sigma^2}\right] r^{2+n}\, \mathrm{d}r. \tag{18}$$

The conditional mean and standard deviation of the ansatz distribution are

$$m(\mu, \sigma) = \bar{r}(\mu, \sigma) \text{ and} \tag{19}$$

$$s(\mu, \sigma) = \sqrt{\overline{r^2}(\mu, \sigma) - \bar{r}^2(\mu, \sigma)}. \tag{20}$$

Our task is, given the conditional mean $\hat{m}$ and standard deviation $\hat{s}$ as measured from the actual posterior probability distribution, to numerically solve the following system of equations for $\hat{\mu}$ and $\hat{\sigma}$:

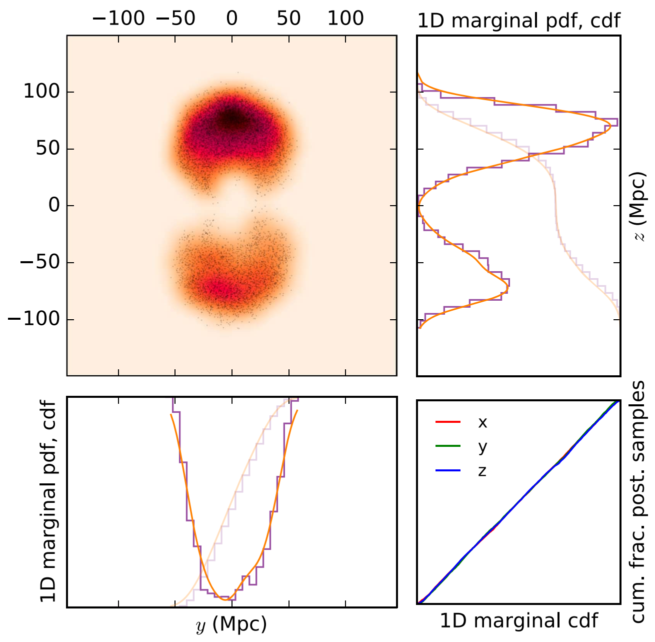
$$\hat{m} = \bar{r}(\hat{\mu}, \hat{\sigma}) \text{ and} \tag{21}$$

$$\hat{s} = \sqrt{\overline{r^2}(\hat{\mu}, \hat{\sigma}) - \bar{r}^2(\hat{\mu}, \hat{\sigma})}. \tag{22}$$

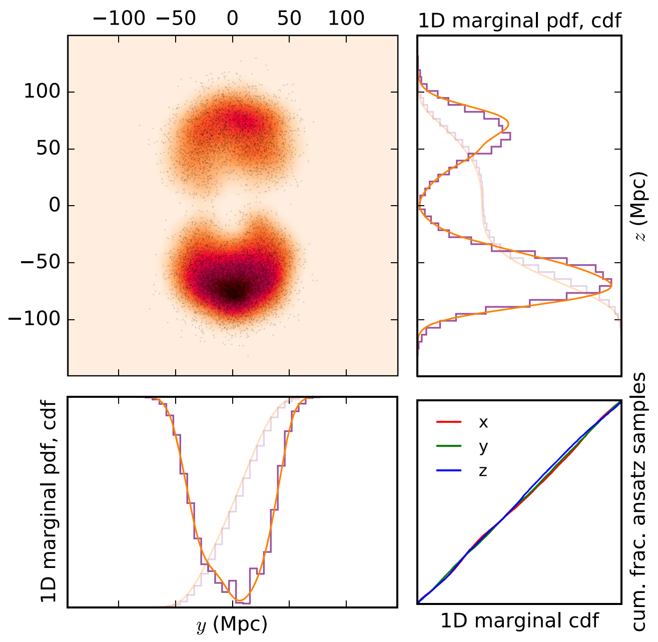
We can reduce this to a single equation by defining $z = \mu/\sigma$ and $\hat{z} = \hat{\mu}/\hat{\sigma}$. With this substitution, the moments can be written as

$$\overline{r^n} = NQ(-z)\sigma^{2+n} x_{2+n}(z),$$

[17] https://github.com/farr/skyarea

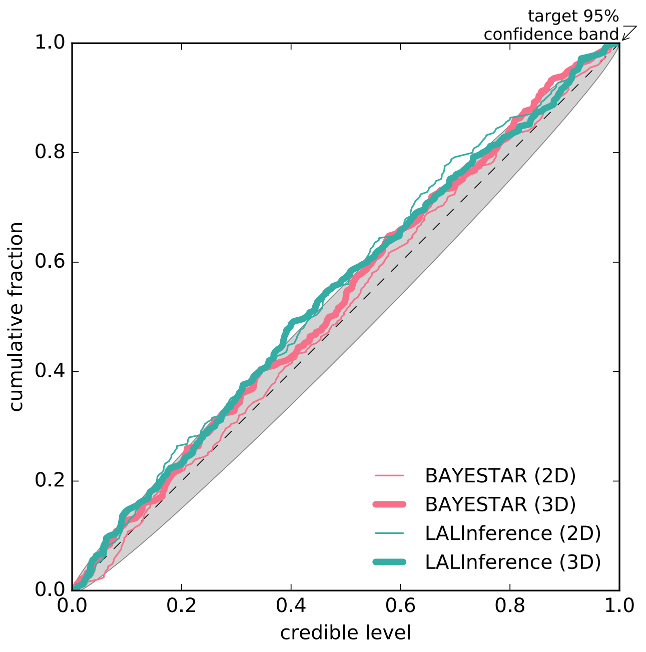


(a) KDE versus posterior samples

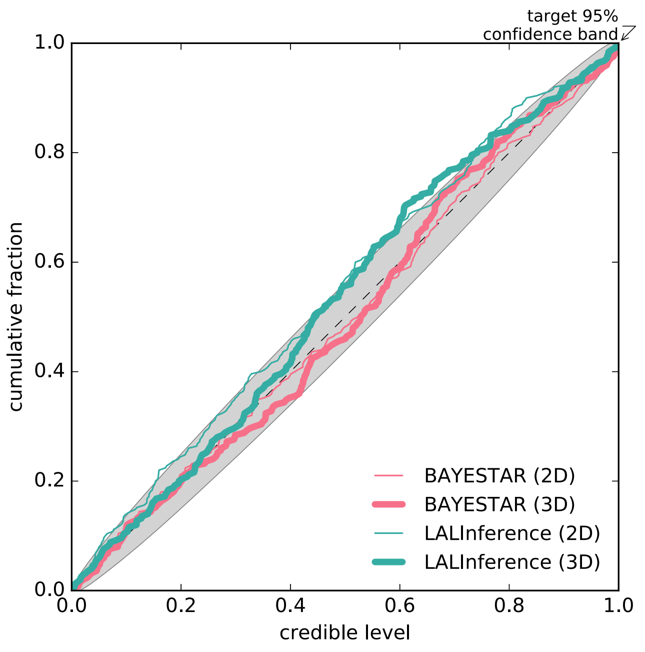


(b) KDE versus samples from ansatz

Figure 2. Comparison between the posterior sample chain, the KDE, and the ansatz distribution. In panel (a), the heat map shows a 2D projection of the KDE. The black dots are the LALInference posterior samples from which the KDE was built. The top right and bottom left plots show the 1D projections. The dark, smooth, orange line is the 1D marginal KDE and the dark, purple, stepped line is a 1D histogram of the LALInference posterior samples. The faint lines of the corresponding colors and styles are the respective 1D cumulative distributions. The bottom right plot is a *P*–*P* plot of the 1D cumulative distribution of the KDE vs. the 1D cumulative histogram of the posterior samples. Panel (b) is the same as panel (a), except that samples from the ansatz distribution are substituted for samples from the posterior.



(a) Ensemble *P*–*P* plot (O1)



(b) Ensemble *P*–*P* plot (O2)

Figure 3. Ensemble *P*–*P* plot test for the 250 events from the O1 scenario (a) and the 250 events from the O2 scenario (b).

with

$$x_2(z) = z^2 + 1 + zH(-z),$$
$$x_3(z) = z^3 + 3z + (z^2 + 2)H(-z), \text{ and}$$
$$x_4(z) = z^4 + 6z^2 + 3 + (z^3 + 5z)H(-z).$$

The function $Q(x) = \mathrm{erfc}(x/\sqrt{2})/2$ is the upper tail of the normal distribution, $P(x) = \exp(-x^2/2)/\sqrt{2\pi}$ is the normal distribution function, and $H(x) = P(x)/Q(x)$ is the hazard function. Then Equations (21) and (22) become

$$f(\hat{z}) = \left(1 + \left(\frac{\hat{s}}{\hat{m}}\right)^2\right)x_3(\hat{z})^2 - x_2(\hat{z})x_4(\hat{z}) = 0. \qquad (23)$$

The derivative of the left-hand side, $f'(\hat{z})$, is given by

$$f'(\hat{z}) = 2\left(1 + \left(\frac{\hat{s}}{\hat{m}}\right)^2\right)x_3(\hat{z})x_3'(\hat{z}) - x_2(\hat{z})x_4'(\hat{z}) - x_2'(\hat{z})x_4(\hat{z}) \qquad (24)$$

with

$$x_2'(z) = 2z + H(-z) + z\partial_z H(-z),$$
$$x_3'(z) = 3z^2 + 3 + 2zH(-z) + (z^2 + 2), \partial_z H(-z)$$
$$x_4'(z) = 4z^3 + 12z + (3z^2 + 5)H(-z) + (z^3 + 5z)\partial_z H(-z),$$

and $\partial_z H(-z) = -H(-z)(z + H(-z))$.

We solve Equations (23) and (24) for $\hat{z}$ using Steffensen's method[18], an accelerated Newton solver. Starting from an initial value of $\hat{z}_0 = \hat{m}/\hat{s}$, the solution converges to machine precision in 10 iterations.

Finally, we calculate $\hat{\mu}$, $\hat{\sigma}$, and N as follows:

$$\hat{\sigma} = mx_2(\hat{z})/x_3(\hat{z}),$$
$$\hat{\mu} = \hat{\sigma}\hat{z},$$
$$\hat{N} = (Q(-\hat{z})\hat{\sigma}^2 x_2(\hat{z}))^{-1}.$$

In the rare event that the solution does not converge, or yields an invalid value such that $\overline{r^2}(\hat{\mu}, \hat{\sigma}) - \bar{r}^2(\hat{\mu}, \hat{\sigma}) < 0$, we set

$$\hat{\mu} = \infty,$$
$$\hat{\sigma} = 1,$$
$$\hat{N} = 0.$$

6. FAITHFULNESS

The ansatz guarantees that the first two moments of distance are exactly reproduced along all lines of sight (LOSs). However, we must ask how accurately the ansatz represents the 3D posterior as a whole. The *P–P* plot graphical test, popularized in the GW parameter estimation literature by Sidery et al. (2014), compares two populations by plotting their cumulative distributions against each other. If the two distributions match, then the result should be a diagonal line.

In our case, we compare the KDE to the LALInference posterior samples by projecting both the KDE and the posterior samples along the distribution's three principal axes, yielding three *P–P* tests. As shown in Figure 2(a), the plot is nearly diagonal, indicating that the KDE is a faithful representation of the posterior samples. We then compare the KDE with the ansatz by drawing samples from the ansatz distribution (Figure 2(b)). Some deviation is perceptible; in the most extreme cases we find a maximum difference in credible levels of about 5%. *P–P* tests of the conditional distance distribution itself along individual LOSs generally also agree within 5% or better, except in directions of low probability (small ρ_i).

We test the statistical self-consistency of the entire ensemble of simulated events in Figure 3. Here, we show a cumulative histogram of the number of simulated events whose true 2D and 3D coordinates are found within a given credible level. We find that both the 2D sky maps and the 3D ansatz are self-consistent within a binomial 95% tolerance band due to the finite sample size of 250 events.

Our interpretation is that the ansatz is a reasonable approximation of the full 3D posterior, in the sense that a stated 50% credible volume has a 50% ± 5% chance of containing the source. The most obvious alternative to the ansatz is a densely sampled 3D grid, or a stack of 2D sky maps for a series of distance shells. Either would be just as conceptually simple, but computationally cumbersome due to size. The KDE is an accurate representation of the posterior, but is expensive to evaluate because it is a sum of 10^4–10^5 Gaussians. As a rapidly available data product and as a tool for real-time observation planning, the ansatz distribution is a reasonable compromise.

We thank the Aspen Center for Physics and NSF grant #1066293 for hospitality during the conception, writing, and editing of this paper. We thank P Shawhan and F Tombesi for detailed feedback on the manuscript. The online data release is available at https://dcc.ligo.org/P1500071/public/html. This is LIGO document P1500071-v7.

Software: Astropy (Robitaille et al. 2013), GNU Scientific Library (Galassi & Gough 2009), HEALPix (Górski et al. 2005), Matplotlib (Hunter 2007), Yt (Turk et al. 2011).

REFERENCES

Aasi, J., Abadie, J., Abbott, B. P., et al. 2013, PhRvD, 88, 062001
Abbott, B. P., Abbott, R., Abbott, T. D., et al. 2016, ApJ, 826, L13
Cotton, W. D., Tody, D., & Pence, W. D. 1995, A&AS, 113, 159
Galassi, M., & Gough, B. 2009, GNU Scientific Library: Reference Manual, GNU manual (UK: Network Theory)
Górski, K. M., Hivon, E., Banday, A. J., et al. 2005, ApJ, 622, 759
Huchra, J. P., Macri, L. M., Masters, K. L., et al. 2012, ApJS, 199, 26
Hunter, J. D. 2007, CSE, 9, 90
Ochsenbein, F., Bauer, P., & Marcout, J. 2000, A&AS, 143, 23
Robitaille, T. P., Tollerud, E. J., Greenfield, P., et al. 2013, A&A, 558, A33
Sidery, T., Aylott, B., Christensen, N., et al. 2014, PhRvD, 89, 084060
Singer, L. P. 2015, PhD thesis, California Institute of Technology
Singer, L. P., Chen, H.-. Y., Holz, D. E., et al. 2016, ApJL, 829, L15
Singer, L. P., Price, L. R., Farr, B., et al. 2014, ApJ, 795, 105
Tully, R. B. 2015, AJ, 149, 171
Turk, M. J., Smith, B. D., Oishi, J. S., et al. 2011, ApJS, 192, 9
Wells, D. C., Greisen, E. W., & Harten, R. H. 1981, A&AS, 44, 363

[18] The `gsl_root_fdfsolver_steffenson` method in the GNU Scientific Library.