Gravitational wave inference at GPU speed: A `bilby`-like nested sampler within `blackjax-ns`

Metha Prathaban,[1★] David Yallup,[2] James Alvey[2,3] and Will Handley[3]
[1] *University of Cambridge, Cambridge, CB3 0HE, UK*
[2]
[3]

Accepted XXX. Received YYY; in original form ZZZ

ABSTRACT

This is a simple template for authors to write new MNRAS papers. The abstract should briefly describe the aims, methods, and main results of the paper. It should be a single paragraph not more than 250 words (200 words for Letters). No references should appear in the abstract.

Key words: keyword1 – keyword2 – keyword3

1 INTRODUCTION

The era of gravitational wave (GW) astronomy, initiated by the landmark detections of the Laser Interferometer Gravitational-Wave Observatory (LIGO) and Virgo, and now advanced by the global network including KAGRA, has revolutionized our view of the cosmos (Abbott et al. 2016, 2017a, 2019, 2024, 2023b,a, 2021, 2017b). Extracting scientific insights from the faint, transient signals buried in detector noise—from measuring the masses and spins of binary black holes to performing precision tests of general relativity—relies critically on the framework of Bayesian inference (Thrane & Talbot 2019). This allows for the estimation of posteriors on source parameters (parameter estimation) and the comparison of competing physical models (model selection).

The process of Bayesian inference in GW astronomy is, however, computationally demanding. Realistic theoretical models for the GW waveform are complex, and the stochastic sampling algorithms required to explore the high-dimensional parameter space can require millions of likelihood evaluations per analysis (Abbott et al. 2020b). The community-standard software tools, the inference library `bilby` (Ashton et al. 2019) paired with a custom implementation of the nested sampler `dynesty` (Speagle 2020), have proven to be a robust and highly effective framework, tuned for the specific needs of GW posteriors. However, this framework is predominantly executed on central processing units (CPUs), making individual analyses time-consuming and creating a significant computational bottleneck. This challenge is set to become acute with the increased data volumes from future observing runs (The LIGO Scientific Collaboration et al. 2015; Acernese et al. 2014; Abbott et al. 2020a) and the advent of next-generation observatories, such as the Einstein Telescope (Branchesi et al. 2023), which promise unprecedented sensitivity and detection volumes (Hu & Veitch 2024).

In response to this challenge, the GW community has begun to leverage the immense parallel processing power of graphics processing units (GPUs). Pioneering work in this domain, such as the `jimgw` codebase (Wong et al. 2023a), has successfully implemented GPU-accelerated Markov Chain Monte Carlo (MCMC) samplers like `flowMC` (Wong et al. 2023b), paired with GPU implementations of waveform models provided by the `ripple` library (Edwards et al. 2024). This work has demonstrated that substantial, order-of-magnitude speedups for GW parameter estimation are achievable. While these MCMC-based approaches excel at rapidly generating samples from the posterior for parameter estimation, they do not directly compute the Bayesian evidence, which remains key for robust model selection.

In this paper, we introduce a GPU-accelerated nested sampling algorithm for gravitational wave data analysis. Our method builds upon the trusted ‘acceptance-walk’ sampler used in the community-standard `bilby` and `dynesty` framework. We leverage the `blackjax ns` sampler, a recent tool designed for vectorized execution on GPUs (Yallup et al. 2025). This sampler, part of the `blackjax` library, is a novel reformulation of the core nested sampling algorithm for massive parallelization (Cabezas et al. 2024).

Instead of its default slice sampler, we developed a custom kernel that implements the ‘acceptance-walk’ method. This ensures our sampler’s logic is almost identical to the `bilby` and `dynesty` implementation, with differences primarily related to parallelization. This approach offers `bilby` users a direct path to GPU-accelerated inference for the most expensive problems in GW astronomy. They can achieve significant speedups while retaining the same robust and trusted algorithm at the core of their analyses.

2 BACKGROUND

2.1 Bayesian inference in GW astronomy

We provide a brief overview of the core concepts of Bayesian inference as applied to GW astronomy. For a more comprehensive treatment, we refer the reader to Skilling (2006); Thrane & Talbot (2019); Veitch et al. (2015); Ashton et al. (2019); Abbott et al. (2020b).

The analysis of GW signals is fundamentally a problem of sta-

[★] E-mail: myp23@cam.ac.uk (MP)

tistical inference, for which the Bayesian framework is the community standard. The relationship between data, d, and a set of source parameters, θ, under a specific hypothesis, H, is given by Bayes' theorem (Bayes 1763):

$$p(\theta|d,H) = \frac{\mathcal{L}(d|\theta,H)\pi(\theta|H)}{Z(d|H)}. \tag{1}$$

Here, the posterior, $p(\theta|d,H)$, is the probability distribution of the source parameters conditioned on the observed data. It is determined by the likelihood, $\mathcal{L}(d|\theta,H)$, which is the probability of observing the data given a specific realisation of the model, and the prior, $\pi(\theta|H)$, which encodes initial beliefs about the parameter distributions.

The denominator is the Bayesian evidence,

$$Z(d|H) = \int \mathcal{L}(d|\theta,H)\pi(\theta|H)d\theta, \tag{2}$$

defined as the likelihood integrated over the entire volume of the prior parameter space.

There are two pillars of Bayesian inference of particular interest in GW astronomy. The first, parameter estimation, seeks to infer the posterior distribution $p(\theta|d,H)$ of the source parameters of a signal or population of signals. The second, model selection, evaluates two competing models, H_1 and H_2, under a fully Bayesian framework by computing the ratio of their evidences, known as the Bayes factor, Z_1/Z_2. This enables principled classification of noise versus true signals, as well as the comparison of different waveform models.

In GW astronomy, the high dimensionality of the parameter space and the computational cost of the likelihood render the direct evaluation of Eq. 1 and Eq. 2 intractable (Abbott et al. 2020b). Analysis therefore relies on stochastic sampling algorithms to numerically approximate the posterior and evidence.

2.2 GPU-accelerated nested sampling

2.2.1 *The nested sampling algorithm*

Nested Sampling (NS) is a Monte Carlo algorithm designed to solve the Bayesian inference problem outlined in Sec. 2.1. A key strength of the NS algorithm is that it directly computes the Bayesian evidence, Z, while also producing posterior samples as a natural by-product of its execution (Skilling 2006).

The algorithm starts by drawing a population of N 'live points' from the prior distribution, $\pi(\theta)$. It then proceeds iteratively. In each iteration, the live point with the lowest likelihood value, $\mathcal{L}_{\min}$, is identified. This point is deleted from the live set and stored. It is then replaced with a new point, drawn from the prior, but subject to the hard constraint that its likelihood must exceed that of the deleted point, i.e., $\mathcal{L}_{\text{new}} > \mathcal{L}_{\min}$. This process systematically traverses nested shells of increasing likelihood, with the sequence of discarded points mapping the likelihood landscape.

The primary computational challenge within the NS algorithm is the efficient generation of a new point from the likelihood-constrained prior (Ashton et al. 2022). The specific method used for this 'inner sampling' task is a critical determinant of the sampler's overall efficiency and robustness.

2.2.2 *GPU architectures for scientific computing*

The distinct architectures of Central Processing Units (CPUs) and Graphics Processing Units (GPUs) offer different advantages for computational tasks. CPUs are comprised of a few powerful cores optimised for sequential task execution and low latency. In contrast, GPUs feature a massively parallel architecture, containing thousands of simpler cores designed for high-throughput computation.

This architecture makes GPUs exceptionally effective for problems that can be expressed in a Single Instruction, Multiple Data (SIMD) paradigm. In such problems, the same operation is performed simultaneously across a large number of data elements, leading to substantial performance gains over serial execution on a CPU. The primary trade-off is that algorithms must be explicitly reformulated to expose this parallelism, as not all computational problems are amenable to vectorization.

2.2.3 *A vectorized formulation of nested sampling*

The iterative, one-at-a-time nature of the traditional NS algorithm is intrinsically serial, making it a poor fit for the parallel architecture of GPUs. To overcome this limitation, Yallup et al. recently developed a vectorized formulation of the NS algorithm, specifically designed for highly parallel execution within the `blackjax` framework (Yallup et al. 2025; Cabezas et al. 2024).

One of the core innovations of this approach is the introduction of batch processing. Instead of replacing a single live point in each iteration, the algorithm removes a batch of k points with the lowest likelihoods simultaneously. The critical step of replacing these points is then parallelized. The algorithm launches k independent sampling processes on the GPU, with each process tasked with finding one new point that satisfies the likelihood constraint, $\mathcal{L} > \mathcal{L}_{\min}$, where $\mathcal{L}_{\min}$ is now the maximum likelihood of the discarded batch.

This reformulation transforms the computationally intensive task of sample generation from a serial challenge into a massively parallel one, thereby leveraging the architectural strengths of the GPU. While the original work proposed a specific inner sampling kernel for this task, the vectorized framework itself is general. It provides a structure within which any suitable inner sampling method can be deployed in parallel.

3 METHODS

3.1 The inner sampling kernel

Several inner sampling methods are implemented within the `bilby` and `dynesty` framework (Ashton et al. 2019; Speagle 2020). In this work, we focus on a GPU-accelerated implementation of the 'acceptance-walk' method, which is a robust and widely used choice for GW analyses.

In the standard CPU-based `dynesty` implementation, the sampler generates a new live point by initiating a Markov Chain Monte Carlo (MCMC) walk from the position of a randomly selected existing live point. The proposal mechanism for the MCMC walk is based on Differential Evolution (DE) (Storn & Price 1997; ter Braak 2006), which uses the distribution of existing live points to inform jump proposals. A new candidate point is generated by adding a scaled vector difference of two other randomly chosen live points to the current point in the chain. Under the default `bilby` configuration, the scaling factor for this vector is chosen stochastically: with equal probability, it is either fixed at 1.0 or drawn from a gamma distribution. This proposed point is accepted if it satisfies the likelihood constraint, $\mathcal{L} > \mathcal{L}_{\min}$, where $\mathcal{L}_{\min}$ is the likelihood of the discarded point being replaced. The walk length is adaptive on a per-iteration basis; the algorithm adjusts the number of MCMC steps dynamically to target a pre-defined number of accepted steps (e.g., 60) for each new live point generated, up to a maximum limit.

This per-iteration adaptive strategy, however, is ill-suited for GPU architectures. The variable walk length required for each parallel sampler would lead to significant thread divergence, where different cores on the GPU finish their tasks at different times, undermining the efficiency of the SIMD execution model. To leverage GPU acceleration effectively, the computational workload must be as uniform as possible across all parallel processes.

Our implementation therefore preserves the core DE proposal mechanism but modifies the walk-length adaptation to be compatible with a vectorized framework. Within the `blackjax ns` sampler, the number of MCMC steps is fixed for all parallel processes within a single batch of live point updates. The adaptive tuning is then performed at the batch level. After a batch of k new points has been generated, we compute the mean acceptance rate across all k walks. The walk length for the *subsequent* batch is then adjusted based on this average rate, using the same logic as `bilby` to target a desired number of accepted proposals.

While this batch-adaptive approach is essential for efficient GPU vectorization, it introduces a trade-off. In the sequential CPU algorithm, an individual MCMC walk that proves to be an outlier with a low acceptance rate will result in a longer walk only for the iteration after it. In our parallel algorithm, if a subset of points in a batch has a low acceptance rate, the average rate will be reduced, causing the walk length for the *entire next batch* to increase. This can sometimes lead to a higher total number of likelihood evaluations compared to the sequential counterpart. Despite this architectural modification, our kernel is designed to be functionally analogous to the trusted `bilby` sampler, operating within the same unit hypercube space and utilizing the same DE proposal strategy to explore the parameter space. Even with the slightly increased number of likelihood evaluations, the GPU implementation is significantly faster than the CPU counterpart.

3.2 Likelihood

To assess the performance of our framework, we employ a standard frequency-domain likelihood. The total speedup in our analysis is achieved by accelerating the two primary computational components of the inference process: the sampler and the likelihood evaluation. The former is parallelized at the batch level as described in Sec. 3.1, while the latter is accelerated using a GPU-native waveform generator.

For this purpose, we generate gravitational waveforms using the `ripple` library, which provides GPU-based implementations of common models (Edwards et al. 2024). This allows the waveform to be calculated in parallel across the frequency domain, enabling massive efficiency gains by ensuring that this calculation does not become a serial bottleneck. To isolate the speedup from this combined GPU-based framework, we deliberately avoid other established acceleration methods like heterodyning (Krishna et al. 2023; Zackay et al. 2018; Leslie et al. 2021; Cornish 2013), though these are available in the `ripple` library too.

For the analyses in this paper, we restrict our consideration to binary black hole systems with aligned spins, for which we use the `IMRPhenomD` waveform approximant (Khan et al. 2016). Further details on the specific likelihood configuration for each analysis, including noise curves and data segments, are provided in Section 4.

[h!]

Table 1. Prior distributions for the parameters of the binary black hole system. The specific ranges for the masses and spins are dependent on the injection and are specified in Section 4.

Parameter	Description	Prior Distribution	Range
$\mathcal{M}_c$	Chirp Mass	Uniform	-
q	Mass Ratio	Uniform	-
$s_{1,z}, s_{2,z}$	Aligned spin components	Uniform	-
d_L	Luminosity Distance	Power Law (2)	[100, 5000] Mpc
$\theta_{\rm JN}$	Inclination Angle	Sine	[0, π]
ψ	Polarization Angle	Uniform	[0, π]
ϕ_c	Coalescence Phase	Uniform	[0, 2π]
t_c	Coalescence Time	Uniform	[-0.1, 0.1] s
α	Right Ascension	Uniform	[0, 2π]
δ	Declination	Cosine	[-π/2, π/2]

3.3 Priors

For this initial study, we adopt a set of standard, separable priors on the source parameters, which are summarized in Table 1. The specific ranges for these priors are dependent on the duration of the signal, and are also given in the table.

As is the default within the `bilby` framework, we sample directly in chirp mass, $\mathcal{M}_c$, and mass ratio, q. We use priors that are uniform in these parameters directly, instead of uniform in the component masses. The aligned spin components, $s_{1,z}$ and $s_{2,z}$, are also taken to be uniform over their allowed range.The coalescence time, t_c, is assumed to be uniform within a narrow window around the signal trigger time.

For the luminosity distance, d_L, we adopt a power-law prior of the form $p(d_L) \propto d_L^2$. This prior corresponds to a distribution of sources that is uniform in a Euclidean universe. While this is a simplification that is less accurate at higher redshifts (Romero-Shaw et al. 2020), it is a standard choice for many analyses and serves as a robust baseline for this work.

These priors were chosen to facilitate a direct, like-for-like comparison against the CPU-based `bilby+dynesty` framework, and in all such comparisons, identical priors were used. The implementation of more astrophysically motivated, complex prior distributions for mass, spin, and luminosity distance is left to future work.

4 RESULTS

In all of the below studies, the `bilby` analyses were executed on a 16-core CPU instance, while the `blackjax ns` analyses were performed on a single NVIDIA L4 GPU.

4.1 Simulated signals

4.1.1 4-second simulated signal

We begin by analysing a 4-second simulated signal from a binary black hole (BBH) merger. The injection parameters for this signal are detailed in Table 2. To ensure a direct, like-for-like comparison, the signal was injected into simulated detector noise using the `bilby` library, and then loaded into both sets of analyses. The analysis uses a three-detector network, assuming the design sensitivity for the fourth LIGO-Virgo-KAGRA observing run (O4). The frequency range of data analysed is from 20 Hz to 1024 Hz.

Both the CPU and GPU-based analyses were configured with 1000

Table 2. Injection parameters for the 4s signal.

Parameter	Value
M_c	35.0 $M_\odot$
q	0.9
$s_{1,z}$	0.4
$s_{2,z}$	-0.3
d_L	1000 Mpc
$\theta_{\rm JN}$	0.4
ψ	2.659
ϕ_c	1.3
t_c	0.0 s
α	1.375
δ	-1.211

live points and a termination condition of dlogZ < 0.1, the default in bilby. The prior ranges for the chirp mass and mass ratio were set to $[25.0, 50.0]$ $M_\odot$ and $[0.25, 1.0]$, respectively, with uniform priors on the aligned spin components over the range $[-1, 1]$.

The recovered posterior distributions, shown in Figure 2, demonstrate excellent statistical agreement between the two frameworks. This result validates that our custom 'acceptance-walk' kernel within the vectorized blackjax ns framework is functionally equivalent to the trusted sequential implementation in bilby. The computed log-evidence values are also in strong agreement, as shown in Figure 1, confirming that our implementation provides a robust unified framework for both parameter estimation and model selection.

The CPU-based bilby run completed in 2.99 hours on 16 cores, for a total of 47.8 CPU-hours. In contrast, the GPU-based blackjax ns run completed in 0.75 hours. This corresponds to a wall-time speedup factor of 63×. In this case, the batch-adaptive nature of the GPU sampler only led to a slightly higher number of likelihood evaluations (64.8 million) compared to the sequential CPU sampler (62.9 million).

Beyond wall-time, we also consider the relative financial cost. Based on on-demand rates from Google Cloud, the rental cost for the 16-core CPU instance and the L4 GPU instance were approximately equivalent at the time of this work. This equivalence in hourly cost implies a direct cost-saving factor of approximately 4× for the GPU-based analysis.

4.2 Injection Study

Give details of exact injections and comparison with bilby+dynesty. Write here about pp-plots?

4.3 Real Data

Give real data examples.

5 DISCUSSION

Compare to bilby in terms of runtimes!

6 CONCLUSIONS

Talk about putting more waveforms on GPU. Current things we have not implemented that are available in bilby. The compatibility of using NFs with this, as we can evaluate many samples at once, meaning approaches like in nessai or PR become more efficient.

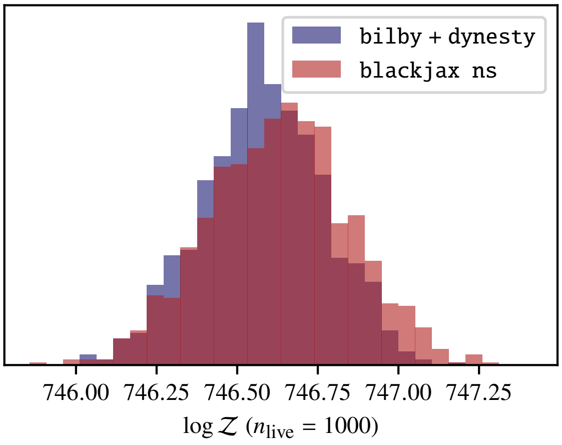

Figure 1. Comparison of the log evidence for the 4s signal. The results are in excellent agreement, demonstrating the robustness of the blackjax ns implementation in recovering the same posteriors and evidence as the bilby implementation. This unifies parameter estimation and evidence evaluation into a single GPU-accelerated framework.

ACKNOWLEDGEMENTS

The Acknowledgements section is not numbered. Here you can thank helpful colleagues, acknowledge funding agencies, telescopes and facilities used etc. Try to keep it short.

DATA AVAILABILITY

The inclusion of a Data Availability Statement is a requirement for articles published in MNRAS. Data Availability Statements provide a standardised format for readers to understand the availability of data underlying the research results described in the article. The statement may refer to original data generated in the course of the study or to third-party data analysed in the article. The statement should describe and provide means of access, where possible, by linking to the data or providing the required accession numbers for the relevant databases or DOIs.

REFERENCES

Abbott B. P., et al., 2016, Phys. Rev. Lett., 116, 061102
Abbott B. P., et al., 2017a, Phys. Rev. Lett., 119, 161101
Abbott B. P., et al., 2017b, Nature, 551, 85
Abbott B. P., et al., 2019, Phys. Rev. X, 9, 031040
Abbott B. P., et al., 2020a, Living Rev. Rel., 23, 3
Abbott B. P., et al., 2020b, Classical and Quantum Gravity, 37, 055002
Abbott R., et al., 2021, Phys. Rev. D, 103, 122002
Abbott R., et al., 2023a, Phys. Rev. X, 13, 011048
Abbott R., et al., 2023b, Phys. Rev. X, 13, 041039
Abbott R., et al., 2024, Phys. Rev. D, 109, 022001
Acernese F., et al., 2014, Classical and Quantum Gravity, 32, 024001
Ashton G., et al., 2019, Astrophys. J. Suppl., 241, 27
Ashton G., Bernstein N., Buchner J., et al. 2022, Nature Reviews Methods Primers, 2, 39

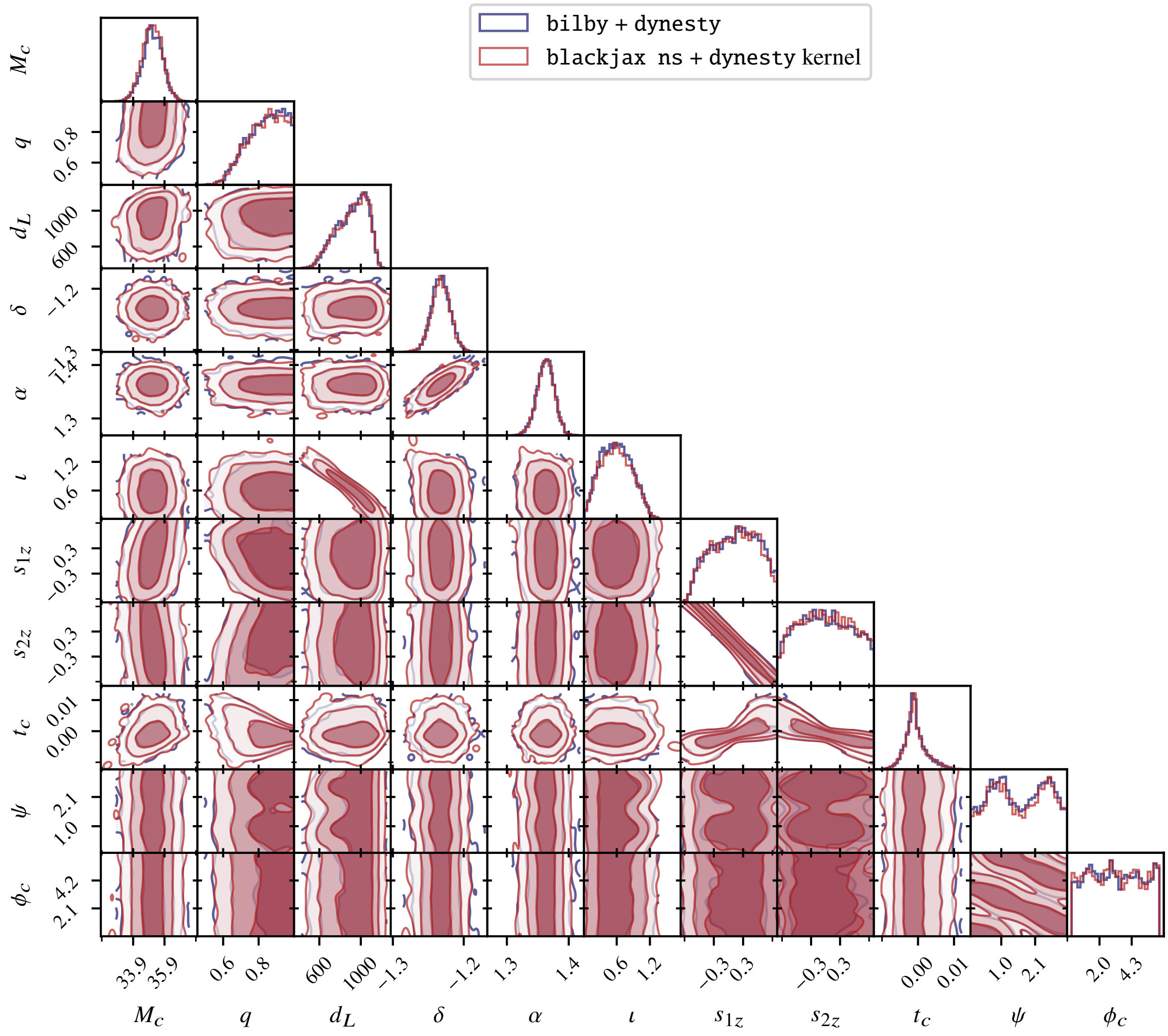

Figure 2. Recovered posteriors for the 4s signal. The posteriors are in agreement with each other, demonstrating that `blackjax ns` implementation with our custom kernel is functionally equivalent to the `bilby` + `dynesty` implementation.

Bayes T., 1763, Philosophical Transactions of the Royal Society of London, 53, 370

Branchesi M., et al., 2023, Journal of Cosmology and Astroparticle Physics, 2023, 068

Cabezas A., Corenflos A., Lao J., Louf R., 2024, BlackJAX: Composable Bayesian inference in JAX (`arXiv:2402.10797`)

Cornish N. J., 2013, Fast Fisher Matrices and Lazy Likelihoods (`arXiv:1007.4820`), `https://arxiv.org/abs/1007.4820`

Edwards T. D. P., Wong K. W. K., Lam K. K. H., Coogan A., Foreman-Mackey D., Isi M., Zimmerman A., 2024, Phys. Rev. D, 110, 064028

Hu Q., Veitch J., 2024, Costs of Bayesian Parameter Estimation in Third-Generation Gravitational Wave Detectors: a Review of Acceleration Methods (`arXiv:2412.02651`), `https://arxiv.org/abs/2412.02651`

Khan S., Husa S., Hannam M., Ohme F., Pürrer M., Jiménez Forteza X., Bohé A., 2016, Phys. Rev. D, 93, 044007

Krishna K., Vijaykumar A., Ganguly A., Talbot C., Biscoveanu S., George R. N., Williams N., Zimmerman A., 2023, Accelerated parameter estimation in Bilby with relative binning (`arXiv:2312.06009`), `https://arxiv.org/abs/2312.06009`

Leslie N., Dai L., Pratten G., 2021, Physical Review D, 104

Romero-Shaw I. M., et al., 2020, Monthly Notices of the Royal Astronomical Society, 499, 3295

Skilling J., 2006, Bayesian Analysis, 1, 833

Speagle J. S., 2020, Monthly Notices of the Royal Astronomical Society, 493, 3132–3158

Storn R., Price K., 1997, J. Global Optim., 11, 341

The LIGO Scientific Collaboration et al., 2015, Classical and Quantum Gravity, 32, 074001

Thrane E., Talbot C., 2019, Publications of the Astronomical Society of Australia, 36

Veitch J., et al., 2015, Physical Review D, 91

Wong K. W. K., Isi M., Edwards T. D. P., 2023a, Fast gravitational wave parameter estimation without compromises (`arXiv:2302.05333`), `https://arxiv.org/abs/2302.05333`

Wong K. W. k., Gabrié M., Foreman-Mackey D., 2023b, J. Open Source Softw., 8, 5021

Yallup D., Kroupa N., Handley W., 2025, in Frontiers in Probabilistic Inference: Learning meets Sampling. `https://openreview.net/forum?id=ekbkMSuPo4`

Zackay B., Dai L., Venumadhav T., 2018, Relative Binning and Fast Likelihood Evaluation for Gravitational Wave Parameter Estimation (`arXiv:1806.08792`), `https://arxiv.org/abs/1806.08792`

ter Braak C. J. F., 2006, Statistics and Computing, 16, 239

APPENDIX A: SOME EXTRA MATERIAL

If you want to present additional material which would interrupt the flow of the main paper, it can be placed in an Appendix which appears after the list of references.

This paper has been typeset from a TeX/LaTeX file prepared by the author.