LASER INTERFEROMETER GRAVITATIONAL WAVE OBSERVATORY
- LIGO -
CALIFORNIA INSTITUTE OF TECHNOLOGY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

Technical Note **LIGO-T1700410–v2** 2017/09/22

Computational costs of using the Bayes factor as a gravitational-wave detection statistic

SURF 2017 Final Report
Zhanpei Fang
Mentor: Rory Smith

California Institute of Technology
LIGO Project, MS 18-34
Pasadena, CA 91125
Phone (626) 395-2129
Fax (626) 304-9834
E-mail: info@ligo.caltech.edu

Massachusetts Institute of Technology
LIGO Project, Room NW22-295
Cambridge, MA 02139
Phone (617) 253-4824
Fax (617) 253-7014
E-mail: info@ligo.mit.edu

LIGO Hanford Observatory
Route 10, Mile Marker 2
Richland, WA 99352
Phone (509) 372-8106
Fax (509) 372-8137
E-mail: info@ligo.caltech.edu

LIGO Livingston Observatory
19100 LIGO Lane
Livingston, LA 70754
Phone (225) 686-3100
Fax (225) 686-7189
E-mail: info@ligo.caltech.edu

http://www.ligo.caltech.edu/

Abstract

The aim of this project is to determine the computational cost of applying the Bayesian methods of source parameter estimation to gravitational-wave searches for LIGO. By implementing parallel-tempered Markov-chain Monte Carlo (PTMCMC) with the Message Passing Interface (MPI) protocol, which distributes computational tasks across a cluster of parallel CPUs, one might in theory be able to complete data analysis in near real-time. The ability to complete parameter estimation analyses in near real-time would have important implications for searches, because parallel tempering also directly computes the evidence term and thus the Bayes factor, which allows us to select between models describing the data as either noise of noise plus signal. The Bayes factor is proposed as an alternative detection statistic that may be more robust than the signal-to-noise ratio (SNR), as using Bayesian evidences may result in fewer false dismissals of real signals. The SNR is cheap to compute, while the computational cost of calculating the Bayes factor is currently unclear. This project quantifies the cost of a search algorithm that uses the Bayesian methods of parameter estimation, with implications for the ways in which LIGO conducts data analysis.

1 Introduction

General relativity predicts that binary systems of compact objects - such as binary black holes, binary neutron stars, and neutron star-black hole systems - lose energy through the emission of gravitational radiation as they merge. As such, compact binary coalescence (CBC) events are among the most promising gravitational wave (GW) sources for ground-based interferometric devices such as LIGO. The first detection of such an event, GW150914, occurred on 14 September 2015, when LIGO observed a binary black hole coalescence waveform [3].

The shapes of the signals that such merging systems emit carry important information about the systems [2], and about the underlying astrophysics of GW sources [1]. The fifteen astrophysical properties used to describe a compact binary coalescence are:

1, 2) $m_{1,2}$, the masses of the two bodies[1];

3-8) $S_{1,2}$, the associated spin vectors, which have three components each[2];

9) d_L, the luminosity distance;

10) ι, the inclination angle between the observer's line of sight and the orbital angular momentum $\mathbf{L}$[3];

11, 12) the two sky angles, right ascension and declination;

[1] We also parameterize the masses using the symmetric mass ratio $\eta = \frac{m_1 m_2}{m_1+m_2}$, which ranges from 0 to $\frac{1}{4}$; and the chirp mass $\mathcal{M}_c = (m_1 m_2)^{3/5}/(m_1+m_2)^{1/5}$.

[2] We also define $\chi_{1,2}$, the components of the spins aligned with the orbital angular momentum $\mathbf{L}$; and χ_p, the in-plane spin magnitude of the more massive binary component.

[3] In the waveform generation model, we use θ_J, which is the angle between the line of sight and the total angular momentum $\mathbf{J}$.

13) t_c, time at coalescence;

14) ϕ_c, the phase at coalescence;

15) ψ, the polarization phase [4].

The next section describes the methods used to determine these properties from interferometric measurements.

2 Parameter estimation

Suppose we receive data from a gravitational-wave search, which we can write as:

$$d = h(\vec{\theta}) + n \tag{1}$$

where d is the data, h is the gravitational waveform as a function of the parameters $\vec{\theta}$, and n is the noise. We want to extract parameters given the observed data. The conditional probability of $\vec{\theta}$ given d is known as the posterior $p(\vec{\theta}|d)$, and the goal of parameter estimation is to map the full posterior density function. To determine the posterior from the data, we rely upon two assumptions:

1. We have a mathematical model for the signal $h(\vec{\theta})$, which can be calculated using the `LALSimulation` library.
2. We know that the noise at each frequency bin is Gaussian distributed, with mean zero and variance given by the power spectral density (psd), where we sum over all frequencies:

$$p(n_1, n_2, \ldots, n_N) \propto \exp\left[-\frac{1}{2}\sum_i^N \frac{n_i^* n_i}{\sigma_i^2}\right] \tag{2}$$

Bayes' theorem states that the posterior is related to the likelihood function:

$$p(\vec{\theta}|d) = \frac{p(d|\vec{\theta})p(\vec{\theta})}{p(d)} \tag{3}$$

where

- $p(\vec{\theta}|d)$ is the posterior, as defined above;
- $p(d|\vec{\theta})$ is the likelihood, or the probability of the observed data as a function of the parameters;
- $p(\vec{\theta})$ is the prior distribution that expresses any prior beliefs about the parameters;
- and $p(d) = \int d\vec{\theta} p(d|\vec{\theta})p(\vec{\theta})$ is the evidence term, or marginal likelihood.

If the prior and evidence term are known for all parameter choices, then mapping the likelihood is equivalent to mapping the posterior, as desired. In order to evaluate the likelihood, we assume two hypotheses:

1. We use the fact that the noise is Gaussian to construct a "null hypothesis" $d = n$, where $p(n)$ is given by equation 2.

2. The "signal hypothesis" is that $n = d - h(\vec{\theta})$ is Gaussian-distributed about zero. We use the signal hypothesis to write down the likelihood function:

$$p(d|\vec{\theta}) \propto \exp\left[-\frac{1}{2}\sum_{i}^{N}\frac{(d_i - h_i(\vec{\theta}))^*(d_i - h_i(\vec{\theta}))}{\sigma_i^2}\right] \tag{4}$$

Thus, the goal of parameter estimation is to find the parameters such that when we subtract the model signal from the observed data, the residual $d - h$ is consistent with the Gaussian distribution.

2.1 Outputs of parameter estimation

2.1.1 Posterior distributions

The main product of parameter estimation is the computation of the posterior density functions (PDFs) on compact binary parameters. Figure 1 shows selected posterior density functions (PDFs) for GW150914.

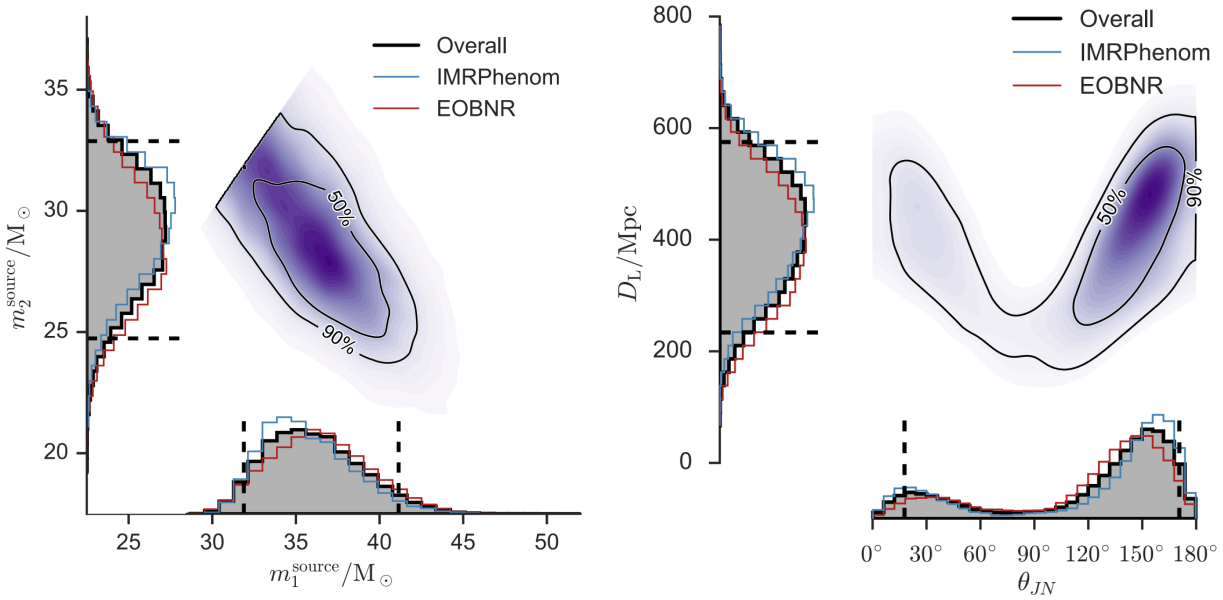

Figure 1: Selected posterior density functions (PDFs) for the merger of two black holes on 14 September 2015, from [2].

2.1.2 Evidence term and Bayes factor

Another product of parameter estimation analyses is the computation of the evidence term $p(d|\mathcal{H})$, which is the likelihood that the hypothesis $\mathcal{H}$, parameterized by $\vec{\theta}$, would have produced the data d. It provides a relative measure of how well one hypothesis is supported by the data over another through the Bayes factor, which is the ratio between evidence terms of two competing hypotheses:

$$B_{i,j} = \frac{p(d|\mathcal{H}_i)}{p(d|\mathcal{H}_j)} \tag{5}$$

3 Markov-chain Monte Carlo

How are posterior distributions generated? One method is to conduct a brute-force evaluation of the likelihood over a range of all possible parameters. However, uniform sampling of high-dimensional parameter space is costly and inefficient; due to the "curse of dimensionality", adding extra parameters to the space leads to an exponential increase in volume. An alternative is to conduct a random walk through parameter space, in order to identify the regions of highest posterior probability. Markov-chain Monte Carlo (MCMC) methods are a class of algorithms that randomly sample from the posterior probability distributions of the source parameters [7,8,11]. The MCMC Hammer (`emcee`) library in Python implements an affine-invariant MCMC [6]. The Metropolis-Hastings algorithm is used as a transition kernel; this is used to generate a proposed new value for the Markov chain, which is then accepted as the new state at random with some probability given by the acceptance ratio.

3.1 Parallel tempering

Though MCMC explores parameter space more efficiently than manually computing the likelihood for all possible parameter values, it still requires the calculation of multi-dimensional integrals. Moreover, for the multimodal densities which are common for compact binary parameters, a Metropolis-Hastings algorithm initialized in one of the modes will find difficulty in jumping to the other mode.

Parallel-tempered MCMC (PTMCMC), also known as replica exchange MCMC, is a method to improve the efficiency of MCMC methods [12]. Parallel tempering exploits the "flattening" of distributions with increasing temperature to create an ensemble of tempered chains. Higher-temperature chains can more easily explore parameter space because the likelihood is flatter and broader, so that they are more likely to accept jumps to lower posterior values. Regions of higher posterior value found by high-temperature chains are passed down through the temperature ensemble by swapping chains at adjacent temperatures, based on the Metropolis criterion (Figure 2); the cooler chains can then explore the peaks of the likelihood. In this way, the PTMCMC algorithm efficiently explores the PDFs of the parameters and establishes which regions of parameter space contain the highest posterior weight.

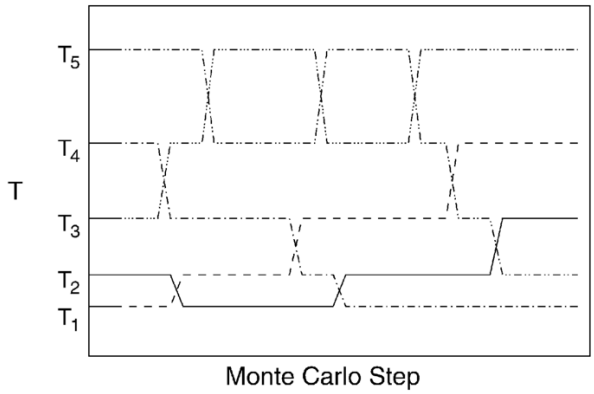

Figure 2: This schematic, from [5], illustrates swaps between adjacent Markov chains in parallel-tempered MCMC.

3.2 Message Passing Interface (MPI)

Though PTMCMC is effective, it is computationally intensive on its own. The simulation of M chains, rather than one, requires on the order of M times as much computational effort. However, parallel tempering can make efficient use of large computer clusters, where different chains can be run simultaneously. The Message Passing Interface (MPI) protocol distributes computational tasks across a cluster of parallel CPUs, where each processor accesses its own memory and processes a job.

Thus, parallel tempering greatly improves the efficiency of the algorithm when distributed across a large computing cluster. With enough CPUs, one might in theory be able to process LIGO data in close to real time.

3.3 Implications for searches; project rationale

The ability to complete parameter estimation analyses in near real-time would have important implications for searches, because parallel tempering directly computes the evidence term using thermodynamic integration, a technique for estimating the evidence integral using information from the chains at different temperatures. If we define the evidence as a function of inverse temperature,

$$Z(\beta) \equiv \int dx l^{\beta}(x) p(x) \tag{6}$$

then

$$\ln Z(\beta = 1) = \int_0^1 d\beta \, \langle \ln l \rangle_{\beta} \tag{7}$$

The evidence integral can be estimated from a PTMCMC by computing the average of the log-likelihood $\ln l$ within each chain and applying a quadrature formula [6].

The evidences are computed for the competing hypotheses: instrument noise, or instrument noise plus a coherent signal. The "coherent vs. incoherent" Bayes factor can then be used to distinguish between noise and signal, which are treated as competing hypotheses. Current

tests [9, 10, 13] indicate that using Bayes factors may result in fewer false dismissals of real signals, compared to using the signal-to-noise ratio (SNR) as a detection statistic. Searches have thus far been hampered by the inability to distinguish between non-Gaussian glitches and burst signals; research indicates that Bayesian hypothesis testing may be more robust to non-stationary non-Gaussian noise (Figure 3).

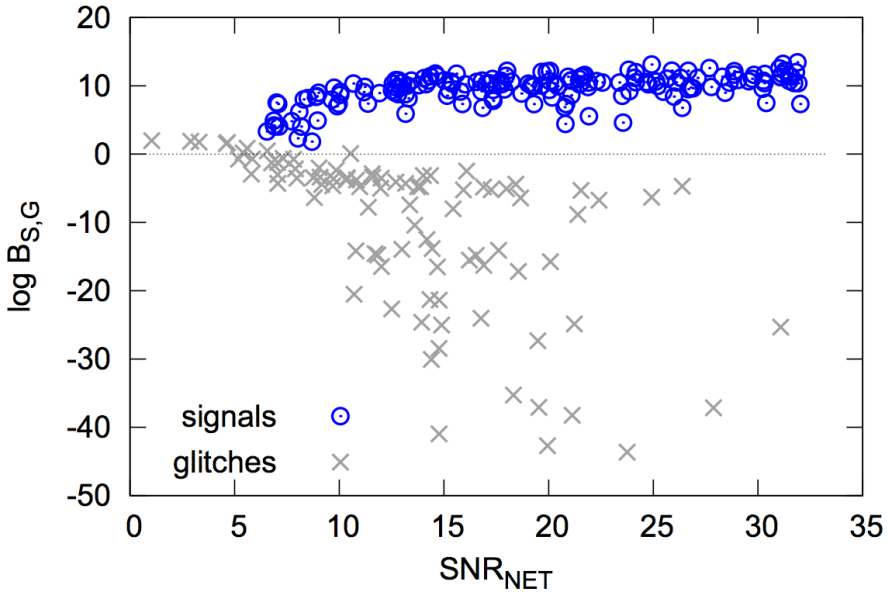

Figure 3: In this figure, from [10], the log Bayes factor shows a clear separation between the simulated signals and glitches.

The SNR is cheap to compute, while the computational cost of calculating the Bayes factor is currently unclear. The goal of this project is then quantify the cost of a search algorithm that uses the Bayesian methods of parameter estimation. Can we apply parameter estimation algorithms to the search pipeline?

4 Methodology

We ran parameter estimation analyses using PTMCMC and MPI on the ARCCA cluster, located at Cardiff University, with access to a maximum number of 24 CPUs. The below tests were run on waveform data from the GW170104 event.

5 Results of efficiency tests

5.1 Number of MCMC steps versus Bayes factor, $d \ln Z$

As we increase the number of steps taken by the MCMC walkers, how does the convergence of the algorithm improve? Figures 4 and 5 show the relationship between the number of steps, the Bayes factor and the error in the estimate of the log evidence. It is notable that

a relatively small number of samples are needed to compute the converged Bayes factor. In addition, the error in the log evidence reaches the typical minimum threshold < 0.1 for a relatively small number of samples; this is a commonly used `LALInference` termination criterion.

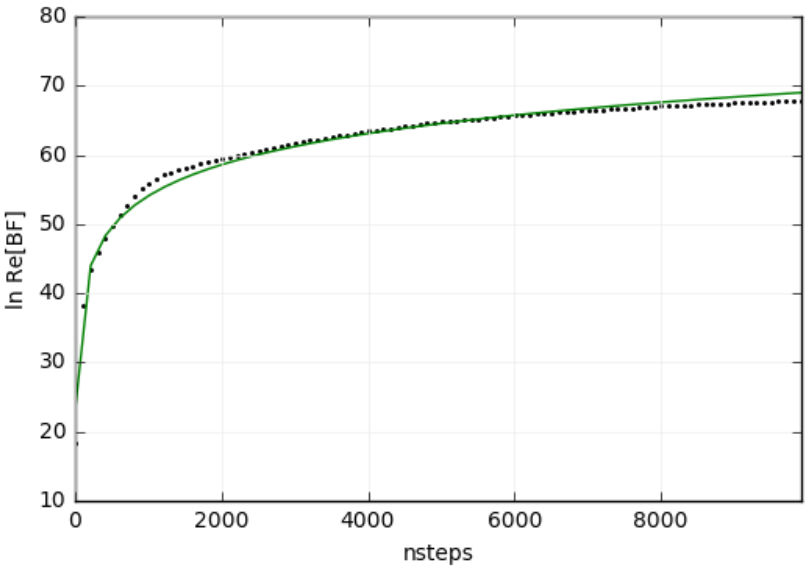

Figure 4: The relationship between the number of MCMC steps and the Bayes factor, when the number of CPUS is fixed to NCPUS = 24. The Bayes factor appears to converge after a relatively small number of steps. The relationship may be logarithmic, and so a log function was fit to the data, in green.

Analyses were also run on data from a "typical" day of data from the second observing run (O2) on 29 June 2017. Figures 6 and 7 confirm that the evidence and Bayes factor still converge in the absence of a signal.

5.2 Number of CPUs versus runtime

We are interested in knowing the number of CPUs necessary to efficiently complete the analysis. As we increase the number of CPUs while fixing the number of MCMC steps, how does the runtime scale? Figure 8 shows the results of the analysis; as expected, the runtime decreases as more CPUs are assigned to the task. It appears that the relationship may fit an exponential function, though an exponential fit to the data proved inconclusive.

6 Conclusions

The results of these efficiency tests indicate that with parallel tempering and MPI, parallel tempering could be conducted quickly enough that the Bayes factor could be used as a detection statistic for the search pipeline. As an order-of-magnitude estimate, it would take about $570,000 \times 24$ CPU hours to process all the data from the O2 observing run, from 30

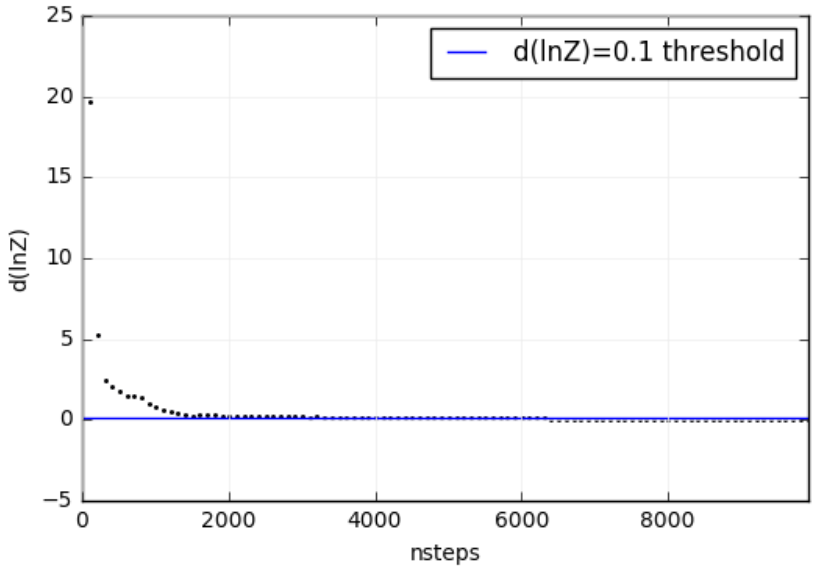

Figure 5: The relationship between number of MCMC steps and the error in the estimate of the log evidence $d \ln Z$. The typical threshold $d \ln Z = 0.1$ has been plotted.

November 2016 to 25 August 2017. This may be feasible, given enough CPUs. In future work, we would like to continue analysis of other datasets, such as those comprised of just glitches and noise, to further confirm the robustness of these novel parameter estimation algorithms.

References

[1] Benjamin P Abbott, Richard Abbott, TD Abbott, MR Abernathy, F Acernese, K Ackley, C Adams, T Adams, P Addesso, RX Adhikari, et al. Astrophysical implications of the binary black hole merger GW150914. *The Astrophysical Journal Letters*, 818(2):L22, 2016.

[2] Benjamin P Abbott, Richard Abbott, TD Abbott, MR Abernathy, F Acernese, K Ackley, C Adams, T Adams, P Addesso, RX Adhikari, et al. Properties of the binary black hole merger GW150914. *Physical Review Letters*, 116(24):241102, 2016.

[3] Benjamin P Abbott, Richard Abbott, TD Abbott, MR Abernathy, Fausto Acernese, Kendall Ackley, Carl Adams, Thomas Adams, Paolo Addesso, RX Adhikari, et al. Observation of gravitational waves from a binary black hole merger. *Physical Review Letters*, 116(6):061102, 2016.

[4] Theocharis A Apostolatos, Curt Cutler, Gerald J Sussman, and Kip S Thorne. Spin-induced orbital precession and its modulation of the gravitational waveforms from merging binaries. *Physical Review D*, 49(12):6274, 1994.

[5] David J Earl and Michael W Deem. Parallel tempering: Theory, applications, and new perspectives. *Physical Chemistry Chemical Physics*, 7(23):3910–3916, 2005.

[6] Daniel Foreman-Mackey, David W Hogg, Dustin Lang, and Jonathan Goodman. `emcee`: The MCMC hammer. *Publications of the Astronomical Society of the Pacific*, 125(925):306, 2013.

[7] Walter R Gilks, Sylvia Richardson, and David Spiegelhalter. *Markov Chain Monte Carlo in Practice*. CRC Press, 1995.

[8] W Keith Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970.

[9] Tyson B Littenberg and Neil J Cornish. Bayesian approach to the detection problem in gravitational wave astronomy. *Physical Review D*, 80(6):063007, 2009.

[10] Tyson B Littenberg, Jonah B Kanner, Neil J Cornish, and Margaret Millhouse. Enabling high confidence detections of gravitational-wave bursts. *Physical Review D*, 94(4):044050, 2016.

[11] Nicholas Metropolis, Arianna W Rosenbluth, Marshall N Rosenbluth, Augusta H Teller, and Edward Teller. Equation of state calculations by fast computing machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 1953.

[12] Marc van der Sluys, Vivien Raymond, Ilya Mandel, Christian Röver, Nelson Christensen, Vicky Kalogera, Renate Meyer, and Alberto Vecchio. Parameter estimation of spinning binary inspirals using Markov chain Monte Carlo. *Classical and Quantum Gravity*, 25(18):184011, 2008.

[13] John Veitch and Alberto Vecchio. Bayesian coherent analysis of in-spiral gravitational wave signals with a detector network. *Physical Review D*, 81(6):062003, 2010.

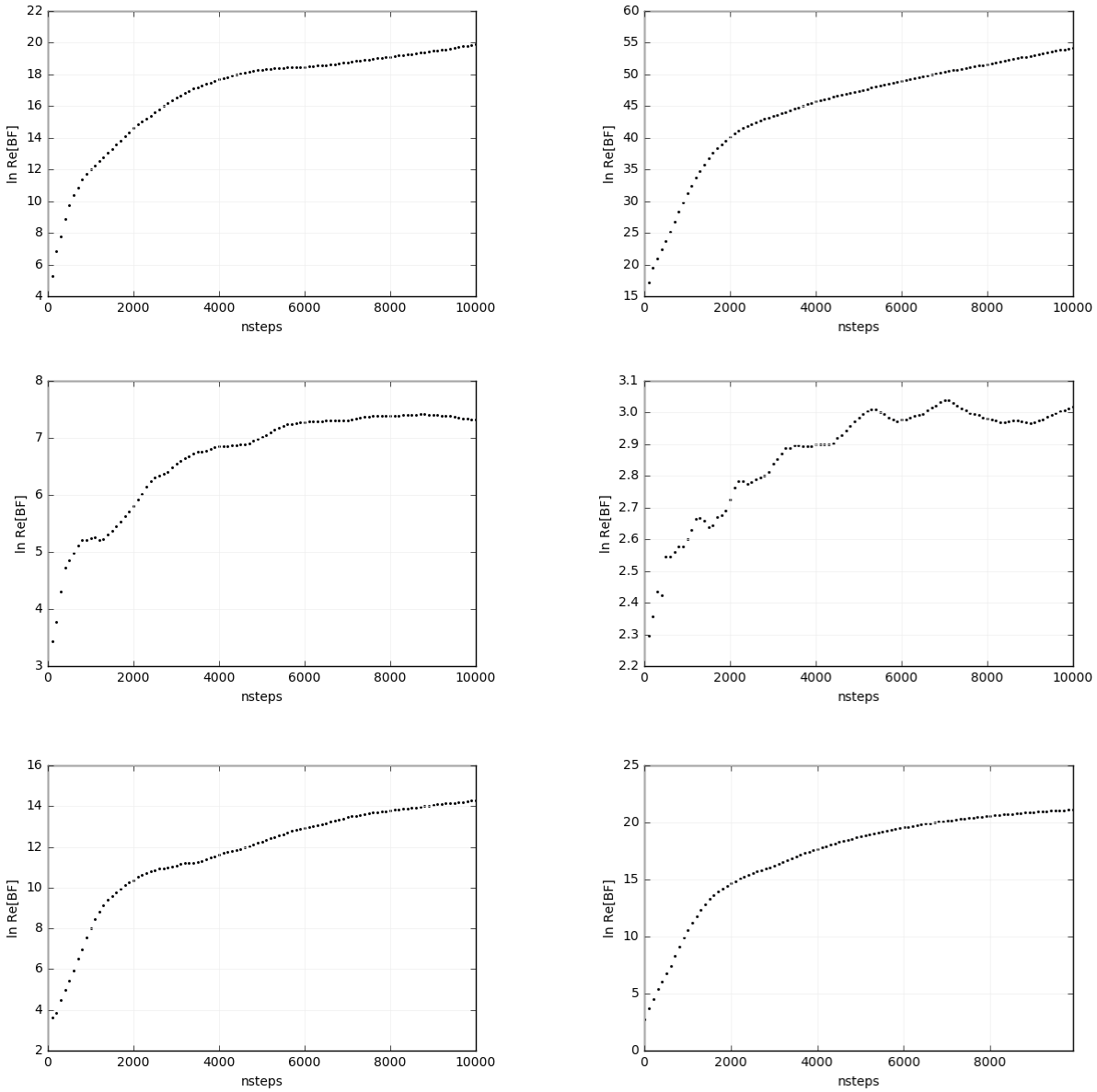

Figure 6: Examples of the convergence of the Bayes factor when the parameter estimation code is applied to 4-second slices of O2 background data from 29 June 2017.

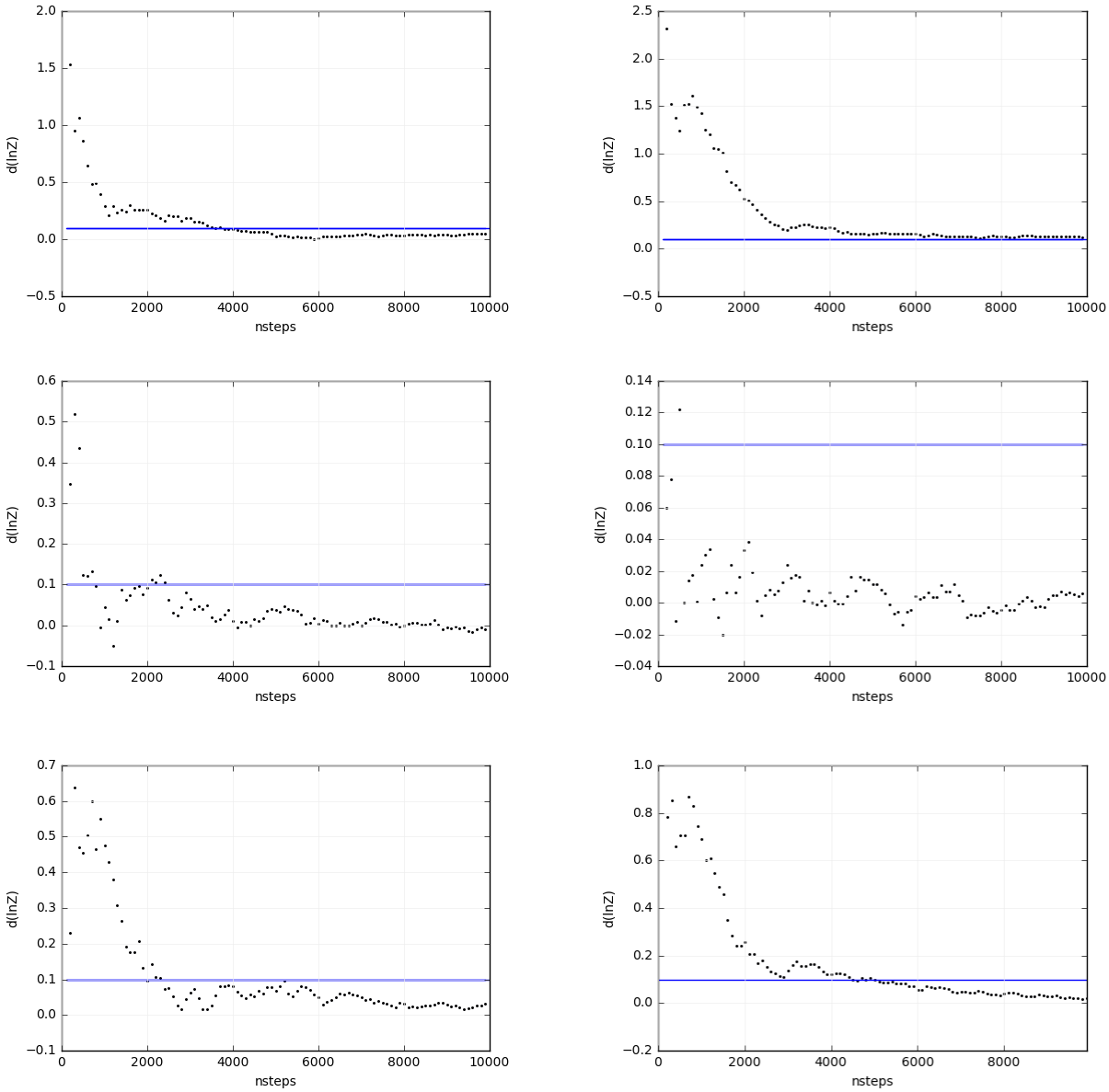

Figure 7: Examples of how the change in the log evidence $d \ln Z$ converges when the parameter estimation code is applied to 4-second slices of O2 background data from 29 June 2017.

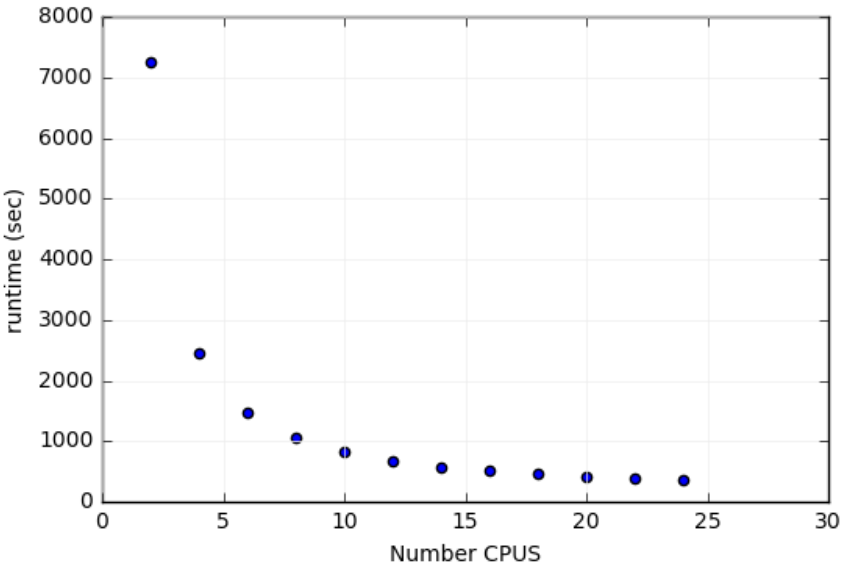

Figure 8: Runtime of MCMC algorithm as a function of number of CPUs in the MPI pool, when the number of MCMC steps is fixed to nsteps = 1000.