Monthly Notices
of the
ROYAL ASTRONOMICAL SOCIETY

MNRAS **483,** 2044–2056 (2019)
Advance Access publication 2018 November 15

doi:10.1093/mnras/sty3090

NESTCHECK: diagnostic tests for nested sampling calculations

Edward Higson [1,2]★ Will Handley,[1,2] Michael Hobson[1] and Anthony Lasenby[1,2]

[1]*Astrophysics Group, Battcock Centre, Cavendish Laboratory, JJ Thomson Avenue, Cambridge CB3 0HE, UK*
[2]*Kavli Institute for Cosmology, Madingley Road, Cambridge, CB3 0HA, UK*

Accepted 2018 October 18. Received 2018 October 6; in original form 2018 May 19

ABSTRACT

Nested sampling is an increasingly popular technique for Bayesian computation, in particular for multimodal, degenerate problems of moderate to high dimensionality. Without appropriate settings, however, nested sampling software may fail to explore such posteriors correctly, for example producing correlated samples or missing important modes. This paper introduces new diagnostic tests to assess the reliability of both parameter estimation and evidence calculation using nested sampling software, and demonstrates them empirically. We present two new diagnostic plots for nested sampling and give practical advice for nested sampling software users in astronomy and beyond. Our diagnostic tests and diagrams are implemented in NESTCHECK, a publicly available[1] PYTHON package for analysing nested sampling calculations, which is compatible with output from MULTINEST, POLYCHORD, and DYPOLYCHORD.

Key words: methods: data analysis – methods: numerical – methods: statistical.

1 INTRODUCTION

Nested sampling (Skilling 2006) is a method for Bayesian analysis that simultaneously provides Bayesian evidences and posterior samples. The popular MULTINEST (Feroz & Hobson 2008; Feroz et al. 2008, 2013) and POLYCHORD (Handley et al. 2015b,a) implementations are now used extensively in many areas of science, and in particular in astronomy (see e.g. Samushia et al. 2014; Desvignes et al. 2016; Joudaki et al. 2016; Planck Collaboration XX 2016b; Chua et al. 2018; DES Collaboration 2018). Though originally designed for evidence calculation, nested sampling is now widely employed for parameter estimation and performs well compared to Markov chain Monte Carlo (MCMC)-based alternatives for multimodal and degenerate posteriors due to having no thermal transition property. In addition the POLYCHORD implementation is designed to handle higher dimensional problems.

Methods for numerically estimating the uncertainty in nested sampling results due to the stochasticity of the nested sampling algorithm are now available for both evidence calculation (see Skilling 2006; Keeton 2011) and parameter estimation (see Higson et al. 2018). However, all of these techniques assume that the nested sampling algorithm was executed perfectly – which requires sampling randomly from the prior within a hard likelihood constraint. This can only be done exactly in special cases, such as for spherically symmetric calculations using PERFECTNS (Higson 2018c). Nested sampling software used for practical problems can only perform such sampling approximately and as a result may produce additional errors – for example due to correlations between samples or due to sampling from only part of the prior volume contained within a likelihood constraint. We term these additional errors *implementation-specific effects* to distinguish them from the intrinsic stochasticity of the nested sampling algorithm.

Diagnosing whether significant implementation-specific effects are present is of great practical importance for researchers as they can cause large uncertainty in results and lead to potentially incorrect conclusions – for example if the calculation misses a significant mode[2] in a multimodal posterior. Conversely, if implementation-specific effects are shown to be negligible, users can simply increase the number of live points for more accurate results and can confidently use standard techniques to estimate numerical uncertainty from the nested sampling algorithm.

Typically a software has a setting that the user can adjust to reduce implementation-specific effects at the cost of increased computation, such as POLYCHORD's `num_repeats` and MULTINEST's `efr` (see Section 7 for more details). Assessing whether the software is able to explore the posterior reliably is therefore particularly useful when taking significantly more samples is computationally costly, as is often the case for high-dimensional problems. In the authors' experience, software users typically try to check their results by running a calculation several times and qualitatively assessing whether the posterior distributions look similar in each case. However, this is not very reliable and does not differentiate between

★ E-mail: e.higson@mrao.cam.ac.uk
[1]Available at https://github.com/ejhigson/nestcheck.

[2]Here we refer to cases where the software does not detect the mode and, as a result, samples are not drawn from the entire prior volume within specified likelihood constraints. Another less common problem is that, if the number of live points is very low, a given run might not contain a single sample within a particular mode even when the nested sampling algorithm is performed perfectly; this is not an implementation-specific effect according to our definition.

implementation-specific effects and the expected variation from the inherent stochasticity of the nested sampling algorithm.

We are not aware of any diagnostic tests in the literature for checking calculation results for practical problems for implementation-specific effects, although Buchner (2016) proposes a diagnostic for evidence calculations that uses analytically solvable test problems. In contrast, MCMC-based methods, which do not require sampling within a hard likelihood constraint, have an extensive literature on diagnostics for practical problems (see e.g. Cowles & Carlin 1996; Hogg & Foreman-Mackey 2018).

This paper introduces new heuristic tests and diagrams to check the reliability of nested sampling results for practical problems, and to determine whether the software settings should be changed. It is also intended to serve as a practical guide for nested sampling practitioners based on the authors' experience using nested sampling software. We begin with a brief overview of the nested sampling algorithm and its associated errors in Section 2 and discuss the challenges of detecting implementation-specific effects in Section 3. We then introduce our new diagnostic tests:

(i) Section 4 discusses diagnostic plots and presents two new diagrams for nested sampling ;
(ii) Section 5 describes how the implementation-specific effects can be measured from a number of nested sampling runs;
(iii) Section 6 introduces diagnostic tests that can be applied to pairs of nested sampling runs and are useful when few runs are available.

We empirically test the effects of changing nested sampling software settings and the dimension of the problem on both implementation-specific effects and total calculation errors in Section 7; the tests use PolyChord, although the discussion and conclusions are relevant for other software. Our practical advice for software users is summarized in Section 7.5. Finally in Section 8 we apply our methods to astronomical data from the *Planck* survey. Our diagnostic tests and diagrams are implemented in nestcheck (Higson 2018a), an open source python package for analysing nested sampling calculations. nestcheck is compatible with output from a variety of nested sampling software packages, including MultiNest, PolyChord, and dyPolyChord (Higson 2018b).

2 BACKGROUND: NESTED SAMPLING AND SAMPLING ERRORS

This section provides a brief overview of the nested sampling algorithm and the sampling errors involved in the process – for more details see Higson et al. (2018). A comparison of nested sampling with other sampling methods is beyond of the scope of this paper; for this we refer the reader to Allison & Dunkley (2014) and Murray (2007).

Nested sampling (Skilling 2006) performs Bayesian computations by maintaining a set of samples from the prior $\pi(\boldsymbol{\theta})$, called *live points*, and repeatedly replacing the point with the lowest likelihood $\mathcal{L}(\boldsymbol{\theta})$ with another sample from the region of the prior with a higher likelihood. The samples that have been removed, termed *dead points*, are then used for evidence calculations and posterior inferences (the live points remaining when the algorithm terminates can also be included). The fraction of the prior volume remaining after each point i with likelihood $\mathcal{L}_i$, which is defined as

$$X(\mathcal{L}_i) \equiv \int_{\mathcal{L}(\boldsymbol{\theta})>\mathcal{L}_i} \pi(\boldsymbol{\theta})\,\mathrm{d}\boldsymbol{\theta}, \tag{1}$$

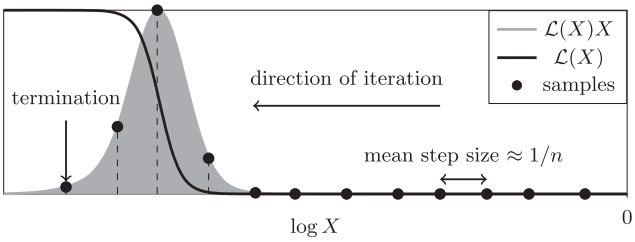


Figure 1. Illustration of nested sampling with a constant number of live points n (reproduced from Higson et al. 2018). The algorithm samples an exponentially shrinking fraction of the prior X as it moves towards increasing likelihoods. The relative posterior mass contained at each $\log X$ value is proportional to $\mathcal{L}(X)X$, where $\mathcal{L}(X) \equiv X^{-1}(\mathcal{L})$.

shrinks exponentially; this process is illustrated schematically in Fig. 1. The shrinkage at each step is unknown but is estimated statistically and used to weight the samples produced.

The sampling errors from this process can be estimated by dividing a completed nested sampling run with some number of live points into many valid nested sampling runs with only one live point. These single live point runs, termed *threads*, can then be resampled using standard techniques such as the bootstrap as described in section 4 of Higson et al. (2018). The resampling is valid as the $\log X$ values of the dead points of a nested sampling run with n live points are a Poisson process with rate n, so the $\log X$ values for the dead points in each of its constituent threads form a Poisson process of rate 1. Here and in the remainder of this paper log denotes the natural logarithm.

3 MEASURING IMPLEMENTATION-SPECIFIC EFFECTS

This paper is concerned with developing practical diagnostics for assessing whether nested sampling calculation results contain implementation-specific effects due to imperfect execution of the nested sampling algorithm. It is important to emphasize that diagnosing such effects without additional information about the likelihood and prior is very challenging, and it is impossible to conclude a priori with certainty that they are not present. For example, one cannot eliminate the possibility of missing an extremely narrow mode for a general posterior without an exhaustive search of the parameter space (Wolpert & Macready 1997). Hogg & Foreman-Mackey (2018, section 5) provide an interesting and analogous discussion of the similarly heuristic nature of MCMC convergence tests. In addition, nested sampling's iteration towards successively higher likelihoods means it never reaches a steady state. As a result heuristics based on autocorrelation of samples like those used in testing for MCMC convergence cannot be applied.

The main idea behind the diagnostic tests we present is to assess whether the variation of the results of different nested sampling runs is consistent with the statistical properties expected of nested sampling without implementation-specific effects. Consequently, these diagnostics require multiple nested sampling runs. A limitation of this approach is that a systematic bias in the calculation results will lead to the implementation-specific effects being underestimated, although they are still likely to be detectable. Such cases have been observed in the literature for evidence calculations with challenging posteriors (see e.g. Beaujean & Caldwell 2013); we discuss systematic bias in detail in Section 7.3. Furthermore our diagnostics are unable to detect implementation-specific effects that do not change the variation of the runs, although we have not come across such a

case in practice. A theoretical example would be if every run available missed a significant mode while exploring all the rest of the parameter space correctly.

3.1 Test problems

We now introduce two test problems, which we will use to demonstrate the diagnostic tests presented in the following sections.

As an example of a simple likelihood, we consider a d-dimensional Gaussian with $\sigma = 1$ centred on the origin

$$\mathcal{L}(\boldsymbol{\theta}) = (2\pi)^{-d/2} \mathrm{e}^{-|\boldsymbol{\theta}|^2/2}. \tag{2}$$

We also use the challenging LogGamma–Gaussian mixture model likelihood introduced by Beaujean & Caldwell (2013), which was designed to represent a particle physics problem involving heavy-tailed distributions and several distinct modes. In this case $\mathcal{L}(\boldsymbol{\theta}) = \prod_{i=1}^{d} \mathcal{L}(\theta_i)$ with

$$\mathcal{L}(\theta_{\hat{1}}) = \frac{1}{2}\mathrm{LogGamma}(\theta_{\hat{1}} - 10|1, 1) + \frac{1}{2}\mathrm{LogGamma}(\theta_{\hat{1}} + 10|1, 1),$$

$$\mathcal{L}(\theta_{\hat{2}}) = \frac{1}{2}\mathrm{Normal}(\theta_{\hat{2}} - 10|0, 1) + \frac{1}{2}\mathrm{LogGamma}(\theta_{\hat{2}} + 10|0, 1),$$

and, if $d > 2$,

$$\mathcal{L}(\theta_{\hat{i}}) = \begin{cases} \mathrm{LogGamma}(\theta_{\hat{i}}|1, 1) & \text{for } 3 \leq i \leq \frac{d+2}{2}, \\ \mathrm{Normal}(\theta_{\hat{i}}|0, 1) & \text{for } \frac{d+2}{2} \leq i \leq d. \end{cases} \tag{3}$$

Here the number of dimensions d is even and the LogGamma distribution is

$$\mathrm{LogGamma}(x|\alpha, \beta) = \frac{\mathrm{e}^{\beta x}\mathrm{e}^{-\mathrm{e}^x/\alpha}}{\alpha^\beta \Gamma(\beta)}, \tag{4}$$

where Γ denotes the gamma function.

Our numerical tests all use uniform priors $\in [-30, 30]$ for each parameter. As equations (3) and (2) are both normalized to 1 and there is negligible posterior mass outside the prior, in both cases the evidence is almost exactly equal to the normalization constant on the uniform prior – i.e.

$$\mathcal{Z}_{\mathrm{true}} = 60^{-d}. \tag{5}$$

4 DIAGNOSTIC PLOTS

Before discussing quantitative diagnostics in Sections 5 and 6, we first introduce some diagnostic plots which illustrate nested sampling and its associated errors. It is good practice for users of sampling software to represent their results visually, in order to assess whether they are reasonable given background knowledge about the problem. Many software packages exist for plotting one - and two-dimensional marginalized distributions from weighted samples using kernel density estimation. As an example, Fig. 2 shows posterior distributions for the LogGamma mixture likelihood (equation 3); this was made using `getdist` (Lewis 2015) with a zero-centred Gaussian kernel and the default settings.

While plots like Fig. 2 are useful, it is unclear to what extent the differences between the two nested sampling runs are due to implementation-specific effects or merely what is expected from the stochasticity of the nested sampling algorithm. Furthermore, these plots do not illustrate the distinctive manner in which nested sampling iterates towards higher likelihoods. We therefore propose two additional diagnostic plots in Section 4.1 and 4.2, which can be calculated from nested sampling runs to show this extra information. These are focused on distributions of parameters and so do not directly assess evidence calculations, but any significant inconsistencies in sample allocations observed between runs may also impact evidence estimates.

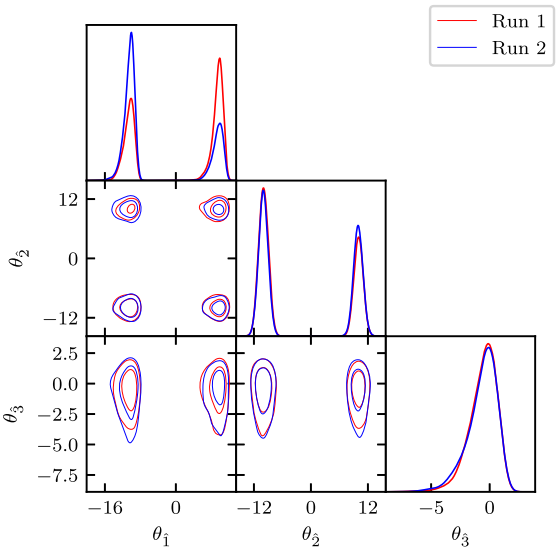


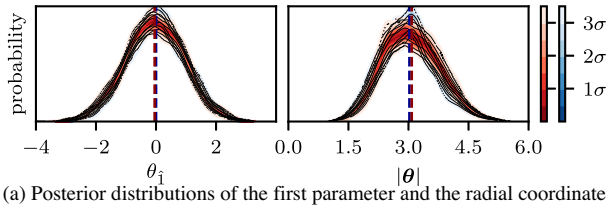
Figure 2. Triangle plot of the posterior distributions for two nested sampling runs (red and blue lines), calculated using the ten-dimensional LogGamma mixture likelihood (equation 3) and a uniform prior. The on-diagonal plots show one-dimensional marginalized posterior distributions on the first three parameters, and the remaining plots show calculated two-dimensional 68 per cent and 95 per cent credible intervals on the joint posterior distribution. The results for the two runs differ due to errors from both the intrinsic stochasticity of the nested sampling algorithm and implementation-specific effects. Each nested sampling run has 250 live points, and uses the PolyChord setting `num_repeats` = 20 – this low setting is deliberately chosen to illustrate large implementation-specific effects.

4.1 Plotting the uncertainty on posterior distributions

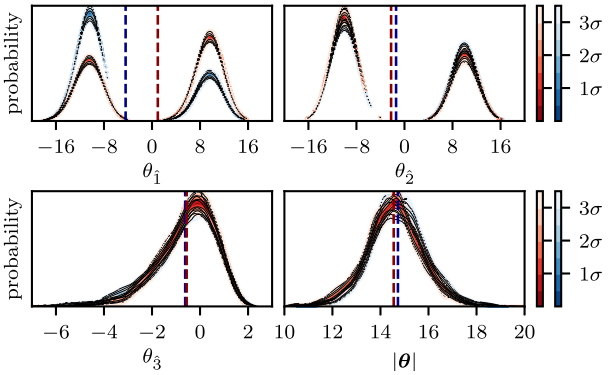
The uncertainty on the posterior distributions due to nested sampling stochasticity can be estimated from a run by creating bootstrap resamples of the run using the procedure described in Higson et al. (2018; section 4). This uncertainty can be visually represented by plotting the distribution of the posteriors obtained from each resample (which is a nested sampling run) to give an *uncertainty distribution on the posterior distribution*. Such plots can be used for assessing whether the calculation error is sufficiently small for the given use case, and are illustrated in Fig. 3. If they are of interest, the posterior distributions of functions of parameters can also be plotted; Figs 3a and b both show the radial coordinate $|\boldsymbol{\theta}| = (\sum_i \theta_i^2)^{1/2}$. The coloured contours are plotted using the fgivenx package (Handley 2018). [3]

Plotting results from multiple runs on the same axis allows visual assessment of whether implementation-specific effects are present. If posterior distributions differ by more than would be expected from

[3]When calculating plots like those in Fig. 3, the posterior distribution for each bootstrap replication must be calculated from the weighted samples without reducing them to evenly weighted samples in a stochastic manner – such as by including each sample with probability proportional to its weight – as this adds extra variation. nestcheck contains an implementation of one-dimensional kernel density estimation that takes sample weights as an argument, and does not require conversion to evenly weighted samples.



(a) Posterior distributions of the first parameter and the radial coordinate $|\boldsymbol{\theta}|$ for a 10-dimensional Gaussian likelihood (2).



(b) Posterior distributions of the first 3 parameters and $|\boldsymbol{\theta}|$ for a 10-dimensional LogGamma mixture likelihood (3). The nested sampling runs are the same ones used in Figure 2 with the corresponding colours.

Figure 3. Diagrams of posterior distributions for two nested sampling runs (red and blue), showing the uncertainty due to the stochasticity of the nested sampling algorithm. Each run uses 250 live points, and has `num_repeats` = 20 deliberately set to a low value to illustrate implementation-specific effects. The coloured contours show iso-probability credible intervals on the marginalized posterior probability density function at each parameter value. The dashed dark blue and dark red lines show the estimated posterior means of each parameter for the blue and red runs, respectively.

their bootstrap sampling error distribution, then implementation-specific effects are likely to be the cause. For example, the top left-hand panel of Fig. 3b, in which the coloured distributions are clearly separated, suggests large implementation-specific effects are present in this case with the settings used. Fig. 3 deliberately uses low values for the PolyChord `num_repeats` and number of live points settings to illustrate implementation-specific effects; these effect can be reduced with a more appropriate choice of settings (discussed in Section 7).

4.2 Plotting distributions of samples in $\log X$

We now propose a diagram to illustrate the distinctive manner in which a nested sampling run progresses by sampling from the prior with successively higher likelihood constraints, based on the discussion in Higson et al. (2018; section 3.1). This involves plotting sample parameters and weights against the fraction of the prior volume remaining, X, which is defined in equation (1). A log scale is used as the shrinkage in X at each step is exponential.

In each plot the top right-hand panel shows the relative posterior mass $\mathcal{L}(X)X$ (i.e. the weight assigned to samples in that $\log X$ region) on a relative scale; this is similar to Fig. 1. The $\log X$ coordinates of the samples are estimated statistically, with their uncertainty distribution displayed using coloured contours. Each subsequent row represents a parameter or function of parameters, with the right-hand panel showing the parameter value of each sample on the same $\log X$ scale. [4] The left-hand panel is the same as the plots in the previous section (Figs 3a and b), and shows the posterior distribution on the parameter values on a shared scale with the left plot (including the uncertainty due to the stochasticity of the nested sampling algorithm).

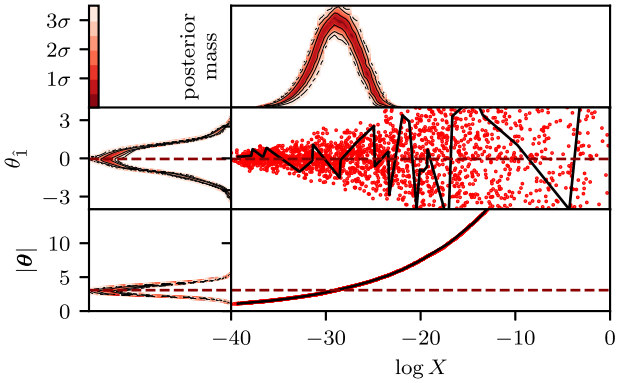


Figure 4. Diagram of samples' distributions in $\log X$ for a single run with a ten-dimensional Gaussian likelihood (2). The top right-hand panel shows the relative posterior mass (total weight assigned to all samples in that region) as a function of $\log X$. The next two rows show the first parameter and the radial coordinate $|\boldsymbol{\theta}|$; for each the right-hand panel plots its sampled values against $\log X$ and the left-hand panel shows its posterior distribution in the same way as Figs 3a and b. The coloured contours show iso-probability credible intervals on the marginalized posterior probability density function at each parameter or $\log X$ value. The nested sampling run shown uses 250 live points and `num_repeats` = 20. The solid black line shows the evolution of an individual thread (chosen at random). The estimated mean value of the posterior distribution for each row is marked with a dashed line.

Our proposed diagram is illustrated in Figs 4 and 5. The lower limit of the $\log X$ axis is chosen to include all points with non-negligible posterior mass, and the upper limit is set to 0 (the start of the nested sampling run). The y-axis limits of the plots in the right column are simply chosen to include all samples with non-negligible posterior weight, or which are otherwise of interest.

In addition, the evolution of individual threads can be traced by drawing lines linking their constituent points. [5] This shares similarities with MCMC trace plots but, unlike for a converged MCMC chain, the distribution of parameters changes as the algorithm iterates over different $\log X$ values. Furthermore, as the algorithm progresses towards lower values of $\log X$ it moves from right to left in the diagram; in MCMC trace plots, chains typically move from left to right.

Figs 4 and 5 are useful for visualizing the nested sampling process and parts of the posterior such as degeneracies and modes with which nested sampling software may struggle. Furthermore, if additional information about the posteriors is available, such as that they should have certain symmetries or be unimodal, this type of diagram can be useful in working out where the sampler is not

[4] The scatter plots in the right-hand column of Figs 4 and 5 can be replaced with a colour plot of the estimated distribution of values at each $\log X$ using kernel density estimation (similar to the colour distributions shown in fig. 3 of Higson et al. 2018). However, doing this accurately is computationally challenging and requires a lot of samples, so simple scatter plots are typically more convenient for checking calculation results.

[5] Plots that trace individual threads in $\log X$ are also produced by the DyNesty dynamic nested sampling package. See https://github.com/joshspeagle/dynesty for more information.

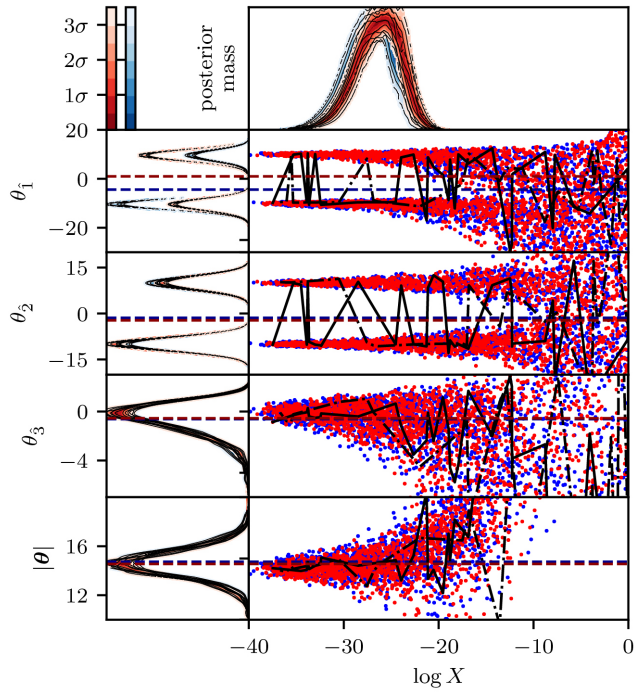


Figure 5. Diagram of samples' distributions in $\log X$ for two nested sampling runs from a ten-dimensional LogGamma mixture likelihood (equation 3). The two runs (shown in red and blue) are the same ones used for Figs 2 and 3b; each uses 250 live points and `num_repeats` = 20. The top right-hand panel shows the relative posterior mass (total weight assigned to all samples in that region) as a function of $\log X$. The next four rows show the first three parameters and the radial coordinate $|\boldsymbol{\theta}|$; for each the right-hand panel plots its sampled values against $\log X$ and the left-hand panel shows its posterior distribution in the same way as Figs 3a and b. The coloured contours show iso-probability credible intervals on the marginalized posterior probability density function at each parameter or $\log X$ value. In each row, the estimated posterior means for the blue and red runs are shown with dashed dark blue and dark red lines. The solid and dot–dashed black lines show the evolution of an individual thread chosen at random from the red and blue runs, respectively.

behaving as expected. For example, Fig. 5 clearly shows the multimodality of the LogGamma mixture likelihood, as well as giving an indication of when in the nested sampling process the modes separate. In addition the bottom right-hand panel of Fig. 4 shows that the radial coordinate $|\boldsymbol{\theta}|$ has negligible spread at any given $\log X$ value in this case; this is due to the likelihood and prior's spherical symmetry.

Furthermore, multiple nested sampling runs can be added to the same axis – as shown in Fig. 5. This allows comparison of where runs differ; for example, one may be able to see on the plot that one of the runs had missed a mode that the other run found (although in Fig. 5 the samples from the two runs overlap). One can also see from Fig. 5 that the two runs agree closely on the relative weights assigned at different $\log X$ values (top panel), meaning that the difference between the posterior distributions (left-hand panels) is due to the parameter values sampled in each $\log X$ region rather than the distribution of posterior mass.[6]

[6]It is common for the parameter values sampled to be the main difference between parameter estimation calculations using different runs, as only the relative weights of points affect the calculation (see Higson et al. 2018 for more details).

5 ESTIMATING IMPLEMENTATION-SPECIFIC EFFECTS

Following the diagnostics plots of the previous section, the remainder of this paper discusses quantitatively measuring implementation-specific effects. The total error on nested sampling calculations can be estimated by measuring the variation of results when a calculation is repeated multiple times, as this includes both implementation-specific effects and the intrinsic stochasticity of the algorithm. This provides a lower bound on the total error, but will underestimate it in the case that implementation-specific effects cause calculation results to be systematically biased.

While the nature of implementation-specific effects depends on the specific software used, they are very likely to be uncorrelated with the errors from the stochasticity of the nested sampling algorithm, which can be calculated using the bootstrap resampling approach. Assuming that they are indeed uncorrelated, the variance in posterior inferences (such as the calculated values of parameter means or the Bayesian evidence) due to implementation-specific effects σ^2_{imp} is related to the variance estimated from bootstrap resampling σ^2_{bs} and the sample variance of calculation results σ^2_{values} by the standard relation for the sum of the variances of uncorrelated random variables (the Bienaymé formula)

$$\sigma^2_{\text{values}} = \sigma^2_{\text{bs}} + \sigma^2_{\text{imp}}. \tag{6}$$

Using this result, we propose calculating the standard deviation of the uncertainty distribution due to implementation-specific effects σ_{imp} as

$$\sigma_{\text{imp}} = \begin{cases} \sqrt{\sigma^2_{\text{values}} - \sigma^2_{\text{bs}}} & \text{if } \sigma^2_{\text{values}} > \sigma^2_{\text{bs}}, \\ 0 & \text{otherwise.} \end{cases} \tag{7}$$

To summarize, here σ_{values} is the observed sample standard deviation of results, σ_{bs} represents the standard deviation we would expect if the nested sampling algorithm was performed perfectly, and σ_{imp} represents the implementation-specific effects causing the difference.

If a number of nested sampling runs are available, the implementation-specific effects on calculations of scalar quantities such as the mean and median of parameters can be calculated directly from equation (7) and compared to the variation of results. One can also estimate the fraction of the observed variation that is due to implementation-specific effects $\sigma_{\text{imp}}/\sigma_{\text{values}}$ – when implementation-specific effects are large this is easy to measure accurately as the variation of results is much greater than the bootstrap error estimates and

$$\frac{\sigma_{\text{imp}}}{\sigma_{\text{values}}} = \frac{\sqrt{\sigma^2_{\text{values}} - \sigma^2_{\text{bs}}}}{\sigma_{\text{values}}} = 1 - \frac{\sigma_{\text{bs}}}{2\sigma_{\text{values}}} + \mathcal{O}\left(\frac{\sigma^2_{\text{bs}}}{\sigma^2_{\text{values}}}\right). \tag{8}$$

The number of runs required to estimate σ_{imp} is primarily determined by the accuracy of the sample standard deviation σ_{values}. Ahn & Fessler (2003) give a formula for the fractional uncertainty of the sample standard deviation as a function of the number of data points; for computationally expensive problems in our research, we typically use ∼10 runs to estimate σ_{imp}. In practice σ_{bs} makes a negligible contribution to the uncertainty on σ_{imp}; it can be estimated accurately from a single run, and the accuracy can be further improved by averaging estimates from all the runs available.

Fig. 6 shows the ratio of the inferred implementation error to the total variation of results for 100 nested sampling runs using ten-dimensional Gaussian (equation 2) and LogGaussian mixture (equation 3) likelihoods. As for Figs 2, 3, 4, and 5 we use the PolyChord setting `num_repeats` = 20, which is deliberately chosen to be

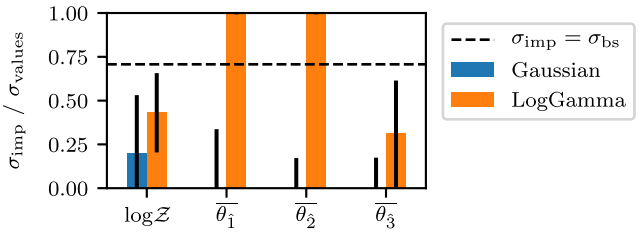


Figure 6. Ratios of estimated implementation-specific effects (equation 7) to variation of results for ten-dimensional Gaussian (equation 2) and LogGamma mixture (equation 3) likelihoods. The dashed horizontal line at $\sigma_{\rm imp}/\sigma_{\rm values} = \frac{1}{\sqrt{2}}$ shows the level where implementation-specific effects and the stochasticity of the nested sampling algorithm make equal contributions to the total error; ratios above this value imply the majority of the error is due to implementation-specific effects. Each bar is calculated using 100 PolyChord runs, each with 250 live points and `num_repeats` = 50. Results are shown for the log-evidence, the mean of the two parameters, the mean radial coordinate, and the second moment of $\theta_{\hat{1}}$. The numerical results plotted in this figure are given in Tables B1 and B2 in Appendix B.

low in order to illustrate implementation-specific effects. The numerical results plotted in Fig. 6 are given in Tables B1 and B2 in Appendix B, along with the absolute values of the variation of results, root-mean-squared-errors, and implementation error estimates. With these PolyChord settings, implementation-specific effects are the dominant source of parameter estimation errors for the LogGamma mixture likelihood. However, the implementation fraction of the error for the log-evidence calculations is significantly lower than for parameter estimation; this is because errors from the stochasticity of the nested sampling algorithm are much larger for evidence calculation than for parameter estimation.

The mean calculated value of $\log \mathcal{Z}$ for the LogGamma mixture likelihood (equation 3), shown in Table B2, differs by 0.10 ± 0.03 from the true value from (equation 5) of $\log \mathcal{Z}_{\rm true} = -d \log(60)$. This systematic bias is due to PolyChord failing to consistently explore the posterior in this challenging case with the deliberately low setting used – it can be reduced by increasing `num_repeats`. However, despite the bias, our approach successfully detected implementation-specific effects in this case. Furthermore, using the true value, we can calculate implementation-specific effects by using the rms error (RMSE) in equation (7):

$$\sigma_{\rm imp,RMSE} = \begin{cases} \sqrt{\mathrm{RMSE}^2 - \sigma_{\rm bs}^2} & \text{if } \mathrm{RMSE}^2 > \sigma_{\rm bs}^2, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

In this case the estimated $\sigma_{\rm imp}/\sigma_{\rm values}$ ratio of 0.43 ± 0.23 shown in Fig. 6 is only a small underestimate compared to $\sigma_{\rm imp,RMSE}/\mathrm{RMSE} = 0.50 \pm 0.14$. Assessing results for systematic bias when the true value of the quantity is not available is discussed in Section 7.3.

Skilling (2006) recommends that inferences from multiple nested sampling runs are made by combining them into a single run rather than simply averaging the results from each run, as this allows more accurate estimation of sample weights. If implementation-specific effects are negligible, then uncertainty estimates can be calculated from the combined run using standard techniques, but this will be inaccurate if implementation-specific effects are the dominant source of error. In the latter case, the approximate error on the combined inference $\sigma_{\rm combined}$ from N nested sampling runs with the same settings can be roughly estimated as

$$\sigma_{\rm combined} = \sigma_{\rm values}/\sqrt{N}. \tag{10}$$

This may be an overestimate as it does not including the benefits of combining the runs, but in practice this effect is likely to be small compared to the uncertainty in the sample standard deviation of the separate runs $\sigma_{\rm values}$ unless N is very large.

6 DIAGNOSTIC TESTS FOR WHEN FEW RUNS ARE AVAILABLE

For computationally expensive problems there may not be enough nested sampling runs available to calculate the implementation-specific effects directly using the method described in the previous section. In Section 6.1 and 6.2 we therefore consider diagnostics that assess whether two nested sampling runs have consistently explored a parameter space while accounting for the stochastic nature of the nested sampling algorithm. Due to the relatively small amount of information available in this case, it is useful to also consider qualitative comparisons using diagnostic plots of the types shown in Section 4 as well as any problem-specific knowledge of what the results should be. If $N > 2$ runs are available, then $\binom{N}{2}$ pairwise tests can be computed and their results combined for greater accuracy.

6.1 Testing for correlations between threads

We now introduce a test to assess whether a nested sampling software is consistently exploring a posterior by comparing the statistical properties of the set of constituent threads (single live point runs) of two nested sampling runs. Each thread represents a valid nested sampling run and can be used to make posterior inferences about quantities such as the evidence and the mean and median of parameters. The actual values calculated from each thread will have large errors due their small number of samples, but this does not matter for testing if the distributions of values obtained from each run's threads are consistent.

We propose applying the two-sample Kolmogorov–Smirnov (KS) test (Massey 1951) to different runs' constituent threads by using each thread to calculate an estimate of a scalar quantity of interest (such as parameter means or the Bayesian evidence $\mathcal{Z}$) with the following procedure:

(i) divide the first nested sampling run into its n_1 constituent threads, and calculate an estimate of the quantity from each;
(ii) divide the second nested sampling run into its n_2 constituent threads, and calculate an estimate of the quantity from each;
(iii) apply the two-sample KS test to the n_1 and n_2 values calculated from the first and second runs, respectively.

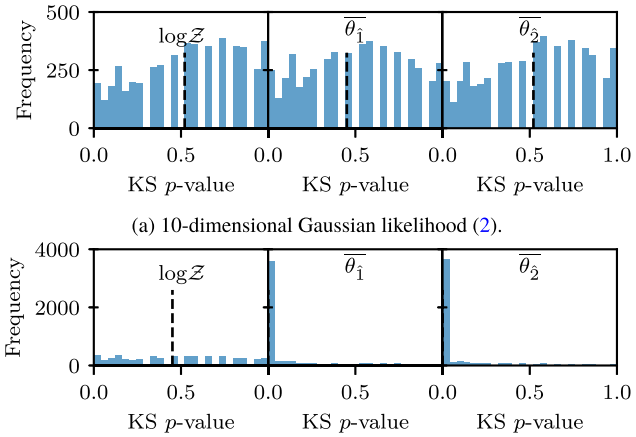
As a test statistic for distributions $p(x)$ and $q(x)$, the KS test uses the maximum distance between their cumulative distributions $F_p(x)$ and $F_q(x)$

$$D_{p,q} = \sup_x |F_p(x) - F_q(x)|, \tag{11}$$

where sup is the supremum. If n_1 and n_2 samples from $p(x)$ and $q(x)$ respectively are used, the corresponding p-values are

$$\alpha = 2 \exp\left(-\frac{2n_1 n_2}{n_1 + n_2} D_{p,q}^2\right). \tag{12}$$

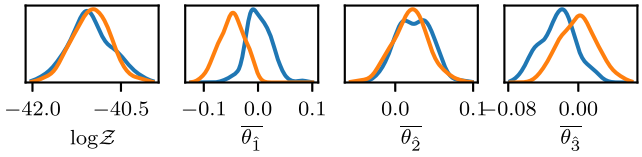
In this case the p-value produced represents the probability of observing a KS statistic $D_{p,q}$ of this size or greater if the threads in the two runs were drawn from the same distribution. A p-value close to zero implies that the values obtained from the threads in the two runs are statistically inconsistent, and hence that implementation-specific effects are likely to be present. This procedure can also



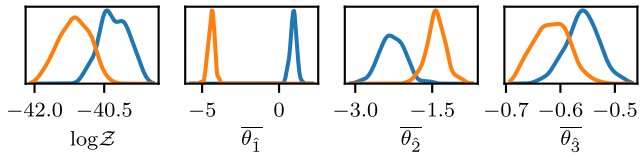
(a) 10-dimensional Gaussian likelihood (2).

(b) 10-dimensional LogGamma mixture likelihood (3).

Figure 7. Distributions of KS p-values from pairwise comparison of different runs' constituent threads, using $\log \mathcal{Z}$ and the first two parameters. A p-value of 0 means the quantities calculated from threads in the two runs are from different distributions, implying the threads within each run are correlated with each other and implementation-specific effects are present. The black dashed line shows the median p-value for each plot. The nested sampling runs are the same ones that were used for Fig. 6 – the 100 runs allow $\binom{100}{2} = 4950$ pairwise statistics to be computed.



(a) 10-dimensional Gaussian likelihood (2).



(b) 10-dimensional LogGamma mixture likelihood (3).

Figure 8. Plots of the sampling error distribution calculated from bootstrap resampling threads for different quantities. Each plot shows two nested sampling runs (represented by different line colours), each with 250 live points and `num_repeats` $= 20$. The kernel density estimation of the posterior distributions uses a Gaussian kernel with the bandwidth selected using Scott's rule (Scott 2015). These plots are designed for use when the true values are not available (although in this case the true values for the distributions shown can be found in Tables B1 and B2).

be used with other distribution-free tests such as the two-sample Anderson-Darling test (Scholz & Stephens 1987) as an alternative to the KS test.

Fig. 7 shows distributions of the p-values computed by applying this procedure to different pairs of nested sampling runs. For the LogGamma mixture likelihood the median p-values for $\overline{\theta_{\hat{1}}}$ and $\overline{\theta_{\hat{2}}}$ are 2×10^{-4} and 5×10^{-5}, respectively, strongly suggesting that implementation-specific effects are present (in agreement with Fig. 6). However, the approach is not able to detect significant evidence of implementation-specific effects in $\log \mathcal{Z}$ calculations, as implementation-specific effects comprise only a fraction of the total variation of results in this case and so the pairs of runs do not provide enough information.

In addition there are many quantities that can be tested – for example the Bayesian evidence and the mean, median, higher moments, and credible intervals of each parameter.[7] Considering a number of quantities allows sensitive testing for implementation-specific errors from only two runs, even if the implementation-specific effects are smaller than in the LogGamma mixture case. One could also test multiple quantities together using a multidimensional KS test, although this is challenging as there is no unique order for quantity values in more than one dimension (see Fasano & Franceschini 1987 for a more detailed discussion). An alternative is to use multiple hypothesis testing with p-value corrections, for example with the Holm–Bonferroni method (Holm 1979).

For MULTINEST runs using the setting `mmodal = True`, when a new mode is recognized, the run is split and live points assigned to the mode remain in that mode and evolve independently from the remainder of the run. As a result, even when there are no implementation-specific effects, the threads within such a run are not independently drawn from the same distribution and the KS test will not give correct p-values. The test is valid for POLYCHORD runs and MULTINEST runs with `mmodal = False` as in these cases threads move between modes; this can be seen in Fig. 5.

It is important to note that the KS p-value only determines whether implementation-specific effects are present and does not provide information about the size of implementation error, which must be assessed to determine whether they are problematic for a given use case. [8] This can be done with the help of bootstrap resamples, as discussed in the next section.

6.2 Testing the consistency of sampling error distributions

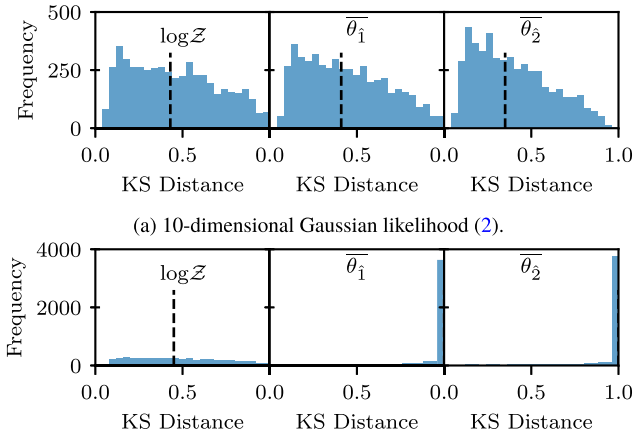
Our second diagnostic assesses whether calculations of scalar quantities from the two different runs differ by more than would be expected given the estimated uncertainties from the intrinsic stochasticity of the nested sampling algorithm. These uncertainty distributions on posterior point estimates can be calculated from bootstrap resamples using the method described in Higson et al. (2018), and are illustrated in Figs 8a and b. This has some similarities with Figs 3a and b but considers only errors on single numbers (such as the means of parameters shown by dashed vertical lines in those figures) rather than on whole posterior distributions. As a result this approach can also be applied to the Bayesian evidence $\mathcal{Z}$, which is a number rather than a distribution.

Bootstrapped point estimates can be qualitatively compared across runs using plots like Fig. 8, or the statistical distance between the distributions can be quantified. As with the comparisons of threads in Section 6.1 it may be hard to draw conclusions from any one quantity, but the two runs can be compared using many different posterior estimates. Quantification may be more convenient than plotting graphs when comparing many different quantities or pairs of runs.

We use the KS statistic (equation 11) as a statistical distance measure; this constitutes a metric as it is non-negative, zero if and only if the distributions are equal, symmetric, and satisfies the triangle inequality. Its numerical values are also easy to interpret, with a value

[7] Tests on functions of the same parameter will not be independent.

[8] In particular with enough data (threads) one can get very low p-values even if the implementation-specific effects are relatively small and/or not important for the practical problem being examined.



(a) 10-dimensional Gaussian likelihood (2).

(b) 10-dimensional LogGamma mixture likelihood (3).

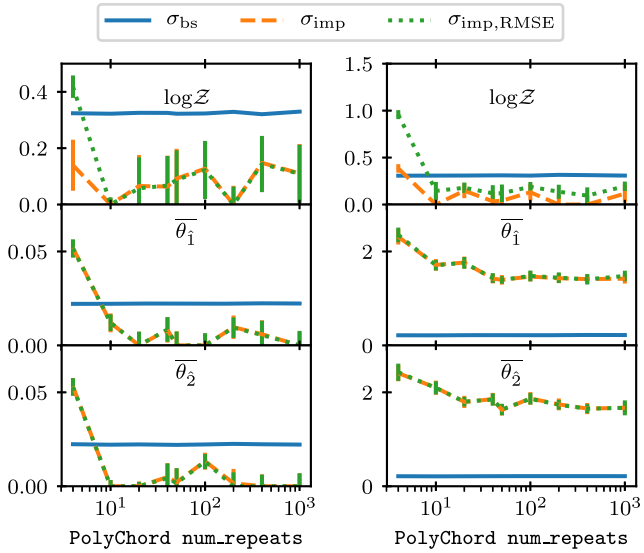
Figure 9. Distributions of KS statistical distances (equation 11) between bootstrap uncertainty distributions on point estimates of the type shown in Fig. 8. For each likelihood, the three columns show results for $\log \mathcal{Z}$ calculations and for the mean of the parameters $\theta_{\hat{1}}$ and $\theta_{\hat{2}}$. The nested sampling runs are the same ones that were used for Fig. 6; the 100 runs are compared pairwise to give $\binom{100}{2} = 4950$ KS statistical distances for each quantity. A KS statistic of close to 1 means there is little overlap between the distributions, implying that the differences in the runs' values cannot be explained by the intrinsic stochasticity of the nested sampling algorithm and that implementation-specific effects are present. The black dashed line shows the median KS distance for each plot.

of 0 meaning the distributions are the same and a value of 1 meaning they do not overlap. KS statistical distances between bootstrapped posterior point estimates from different pairs of nested sampling runs are shown in Fig. 9. These distributions show strong evidence for implementation-specific effects in parameter estimation for the LogGamma mixture case, with calculations of $\overline{\theta_{\hat{1}}}$ and $\overline{\theta_{\hat{2}}}$ having 65.7 per cent and 67.9 per cent of their pairwise statistical distances equalling 1, respectively. These estimates are particularly sensitive to changes in the relative weighting of different modes in the posterior. However, as for the diagnostic introduced in Section 6.1, two runs do not provide enough information to detect the relatively weaker implementation-specific effects in the LogGamma mixture $\log \mathcal{Z}$ estimates.

The KS statistical distances are more difficult to interpret than the p-values in Section 6.1, but they have the advantage that together with plots like Fig. 8 they contain information about the size of any implementation-specific effects. In this context, the KS statistic values are simply used as a distance measure and cannot be interpreted as p-values. This is because, even without implementation-specific effects, nested sampling runs will differ due to the stochasticity of the algorithm, and these differences mean bootstrap resamples of different runs are drawn from different distributions.

7 IMPLEMENTATION-SPECIFIC EFFECTS IN PRACTICE

Having introduced our diagnostic tests, we now empirically test how different software settings and problem dimension affect the size of implementation-specific effects. As an example we use PolyChord, but we intend this section to be informative for users of other software packages such as MultiNest and dyPolyChord. The section finishes with practical advice for software users.



(a) 10-dimensional Gaussian likelihood (2) with a uniform prior.

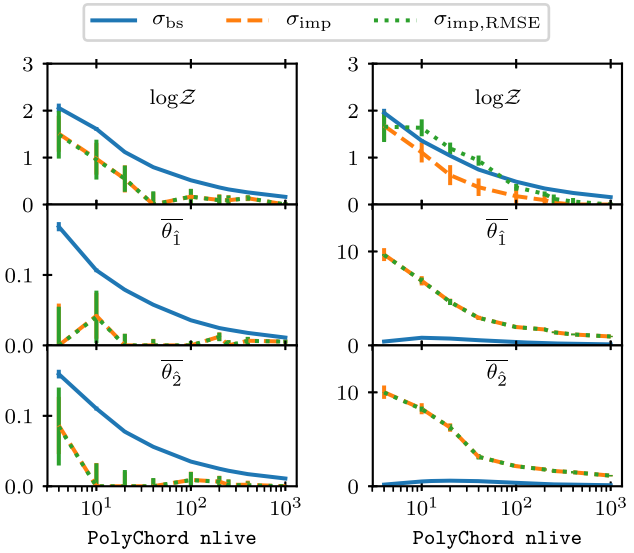
(b) 10-dimensional LogGamma mixture (3) with a uniform prior.

Figure 10. The effect of PolyChord's `num_repeats` setting on results' errors; each subfigure shows calculations of the log-evidence and the mean of the first two parameters. Results for every `num_repeats` value were calculated using 100 nested sampling runs, each with 250 live points. Blue solid lines show the mean bootstrap error estimate and orange dashed lines show implementation-specific effect estimates from equation (7). Green dotted lines show the implementation-specific effects calculated using the rms error (9); where the green dotted and orange dashed lines are equal, there is no systematic bias in the results. Error bars show the uncertainty on results for each `num_repeats` value considered.

7.1 Effect of sampling efficiency settings

Nested sampling software packages typically have settings controlling the process of sampling within a hard likelihood constraint which can reduce implementation-specific effects at the cost of increased computation. PolyChord and dyPolyChord both have a `num_repeats` setting, which controls the number of slice samples taken before sampling each new live point – increasing this value reduces correlation between points and increases the accuracy with which they perform the nested sampling algorithm. Other examples of similar parameters include MultiNest's `efr`, which controls the efficiency of its rejection sampling algorithm by determining the size of the ellipsoid within which MultiNest samples. If `efr` is lowered, samples are drawn from a larger ellipsoid, increasing the rejection rate whilst consequently decreasing the chance of missing part of the parameter space within the iso-likelihood contour. Hence, in contrast with `num_repeats`, implementation-specific effects are made smaller by *reducing* `efr`.

Fig. 10 shows the effect on calculation errors of PolyChord's `num_repeats` setting. As expected, we see that as `num_repeats` is increased the implementation-specific effects are reduced – showing PolyChord is performing the nested sampling algorithm with increasing accuracy. However, the `num_repeats` value required for implementation-specific effects to be a small fraction of the total error is highly problem dependent, even for the same number of dimensions. For the ten-dimensional Gaussian likelihood `num_repeats` $= 10$ is easily sufficient, but for the challenging ten-dimensional LogGamma likelihood `num_repeats` $> 10^3$ is needed. `num_repeats` can be tuned by, for example, doubling it until results show small implementation errors. In principle a



(a) 10-dimensional Gaussian likelihood (2) with a uniform prior.

(b) 10-dimensional LogGamma mixture (3) with a uniform prior.

Figure 11. The effect of the number of live points on errors in PolyChord calculations; the two subfigures both show calculations of the log-evidence and the mean of the first two parameters. Results for each number of live points considered were calculated using 100 nested sampling runs with `num_repeats` = 10. Blue solid lines show the mean bootstrap error estimate and orange dashed lines show implementation-specific effect estimates from equation (7). Green dotted lines show the implementation-specific effects calculated using the rms error (9); where the green dotted and orange dashed lines are equal, there is no systematic bias in the results. Error bars show 1σ uncertainties on results for each number of live points considered.

sufficiently high `num_repeats` value can make such errors negligible even for challenging likelihoods, but this will become impractically computationally expensive and gives diminishing returns in cases like the LogGamma mixture shown in Fig. 10b. Once `num_repeats` is high enough that the calculations are not systematically biased, simply repeating the calculation many times is more efficient at improving accuracy. One can check for such a bias by assessing whether the mean value of results changes when `num_repeats` is increased (if a bias is present, increasing `num_repeats` should reduce it).

7.2 Effect of the number of live points

In addition to software-specific settings, the main choice a nested sampling user must make is the number of live points, which controls the resolution of sampling and is proportional to the expected number of samples produced. For simplicity we consider only runs with a constant number of live points n, although our conclusions also apply to dynamic nested sampling (Higson et al. 2017) – in which the number of live points varies to increase calculation accuracy. Furthermore, NestCheck is compatible with the output of several dynamic nested sampling software packages including dyPolyChord, dynesty,[9] and PerfectNS.

The changes in calculation errors with changes in the number of live points used are shown in Fig. 11. As expected, increasing the number of live points reduces the implementation-specific effects, as well as the errors from the stochasticity of the nested sampling algorithm (measured by bootstrap resampling), which are approximately proportional to $1/\sqrt{n}$. The fraction of the total error made up by implementation-specific effects does not necessarily decrease with increased n – this depends on how the implementation-specific effects scale with n. For the Gaussian likelihood, implementation-specific effects cause only a small part of the total variation of results, whereas for the more challenging LogGamma mixture likelihood they are the main source of errors.

Given that increasing n reduces both implementation-specific effects and errors from the stochasticity of the nested sampling algorithm, this is often a better way to reduce total errors for the same computational cost than increasing `num_repeats`. However, it may not reduce the fraction of errors caused by implementation-specific effects. Consequently, techniques for estimating nested sampling errors that do not account for implementation-specific effects may still underestimate the total uncertainties.

7.3 Calculation results with a systematic bias

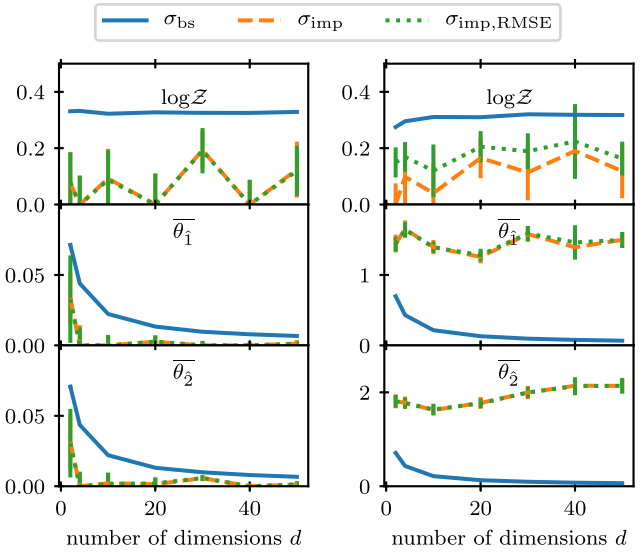
Figs 10 and 11 show that for $\log \mathcal{Z}$ calculations, if `nlive` and `num_repeats` are set too low, estimates of the implementation-specific effects using the standard deviation of results and the rms error can start to differ. This is due to the algorithm failing to fully explore the posterior and iterating inwards too quickly, which leads to a systematic bias in $\log \mathcal{Z}$ (this is discussed in detail in Buchner 2016). The `nlive` and `num_repeats` settings required to remove the bias depend on the posterior, with challenging multimodal or degenerate posteriors needing more samples (as for implementation-specific effects). The challenging LogGamma mixture likelihood shows a bias with the PolyChord settings used (as shown in Table B2 in Appendix B), but this is small compared to the standard deviation of calculation results and can be reduced by increasing `num_repeats` or the number of live points. Systematic biases in a parameter estimation calculation are also possible with inappropriate settings, but in the authors' experience this is much rarer.

The failure to fully explore the posterior that causes a systematic bias typically also results in differences between runs that are not explained by the stochasticity of the nested sampling algorithm – these implementation-specific effects can be detected the diagnostic tests presented in this paper. However, the bias causes these diagnostics to underestimate the size of the implementation-specific effects. If significant implementation-specific effects are detected in runs and the results of $\log \mathcal{Z}$ calculations are of interest, one can check for bias by repeating the calculation with higher `nlive` and `num_repeats` settings and checking if the mean calculated result changes.

7.4 Effect of dimensionality

Fig. 12 shows implementation errors for the Gaussian and LogGamma mixture likelihoods for different numbers of dimensions d. Each calculation uses $25 \times d$ live points and `num_repeats` $= 5 \times d$ (the default settings in PolyChord's python interface). These are proportional to d in order to give approximately constant errors in $\log Z$ (Handley et al. 2015a), with the additional samples produced for higher d leading to lower parameter estimation errors. With these settings, as d increases, our plot shows no strong upwards or downwards trend in the implementation error. Furthermore, the small bias in the $\log \mathcal{Z}$ calculation results for the LogGamma mixture likelihood (shown by the difference between the green dotted and orange dashed lines in the top panel

[9] See https://github.com/joshspeagle/dynesty for more information.



(a) 10-dimensional Gaussian likelihood (2) with a uniform prior.

(b) 10-dimensional LogGamma mixture (3) with a uniform prior.

Figure 12. The effect of increasing the dimension d on errors in PolyChord calculations: Each subfigure shows calculations of the log-evidence and the mean of the first two parameters. Results for every dimension d use $25 \times d$ live points and the PolyChord setting `num_repeats` $= 5 \times d$. Blue solid lines show the mean bootstrap error estimate and orange dashed lines show implementation-specific effect estimates from equation (7). Green dotted lines show the implementation-specific effects calculated using the rms error (equation 9); where the green dotted and orange dashed lines are equal, there is no systematic bias in the results. Error bars show 1σ uncertainties on results for different numbers of dimensions.

of Fig. 12b) remains much smaller than the standard deviation of the result values $\sigma_{\rm values} = \sqrt{\sigma_{\rm bs}^2 + \sigma_{\rm imp}^2}$.

7.5 Practical advice for software users

We finish by giving a summary of the authors' approach to checking nested sampling calculations for challenging likelihoods where implementation errors may be present, based on our experience using nested sampling software.

We advise performing multiple nested sampling runs and plotting the results to first assess their variation by eye as described in Section 4. One can then perform a rough check for implementation-specific effects using the techniques described in Section 5 and/or Section 6, depending on how many runs are available. If implementation-specific errors are negligible,

(i) Accuracy can be increased by simply calculating more runs and/or increasing the number of live points.

(ii) The computational cost of future runs can be reduced by reducing the computational effort spent decorrelating samples (e.g. halving PolyChord's `num_repeats`, doubling MultiNest's `efr` or changing the equivalent setting in the software package used). After large changes to the settings, the new results should be checked for implementation-specific effects.

(iii) Uncertainties on the results can be calculated using standard nested sampling methods such as the bootstrap resampling of threads, which will be accurate in this case.

In contrast, if implementation-specific effects are significant or are the dominant source of error,

(i) Results should be recalculated with more live points and/or using more computational effort decorrelating samples (e.g. doubling PolyChord's `num_repeats`, halving MultiNest's `efr`, or changing the equivalent setting in the software used). If the calculation is already very computationally costly, increasing the number of live points is typically the best option as this will also reduce errors from the stochasticity of the nested sampling algorithm.

(ii) There may be an additional systematic bias present in the results of evidence calculations. The mean calculated value for results using the new settings should be checked to see if it is significantly different to the mean result produced with the previous settings.

(iii) The uncertainty on the combined results from the nested sampling runs can be roughly estimated from equation (10).

8 APPLICATION TO PLANCK SURVEY DATA

We now apply the tests introduced in this paper to astronomical data from the *Planck* survey, which measures anisotropies in the cosmic microwave background (CMB). A detailed description of the associated cosmology and the lambda cold dark matter (ΛCDM)

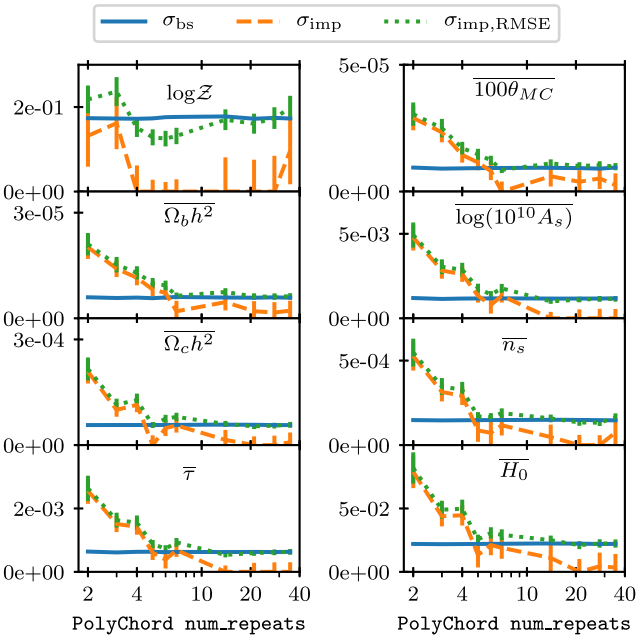


Figure 13. Implementation-specific effects in calculations using *Planck* data for different PolyChord `num_repeats` settings. The left-hand column shows results for the evidence $\log \mathcal{Z}$ and the mean of the present-day Baryon density $\Omega_{\rm b}h^2$, present-day cold matter density $\Omega_{\rm c}h^2$, and Thompson scattering optical depth of the CMB τ. The right-hand column shows results for calculations of the mean of the ratio of the sound horizon to angular distance (scaled by 100) $100\theta_{\rm MC}$, the log power of the primordial curvature perturbations $\log(10^{10}A_{\rm s})$, the spectral index of the scalar primordial power spectrum $n_{\rm s}$, and the present-day Hubble constant (derived from the other parameters) H_0. Results for every `num_repeats` value were calculated using 25 runs, each with 500 live points. Blue solid lines show the mean bootstrap error estimate and orange dashed lines show implementation-specific effect estimates from equation (7). Green dotted lines show the implementation-specific effects calculated using the rms error (equation 9); where the green dotted and orange dashed lines are equal, there is no systematic bias in the results. Error bars show the 1σ uncertainty on results for each `num_repeats` value considered.

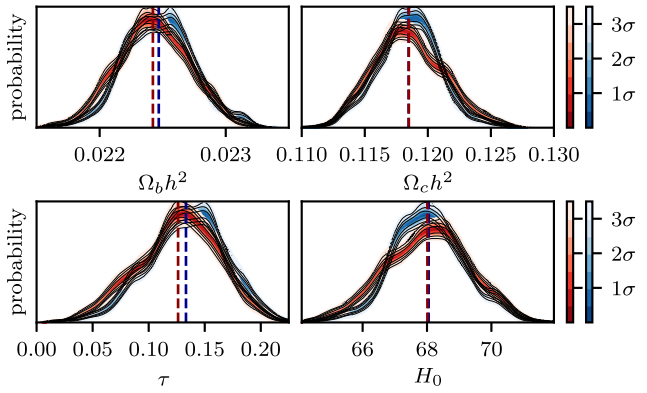


Figure 14. As for Fig. 3 but using the *Planck* survey likelihood. The first row shows the present-day Baryon density $\Omega_b h^2$ and the present-day cold matter density $\Omega_c h^2$; the second row shows the optical depth of the CMB τ and the present-day Hubble constant H_0. Each run uses 500 live points, and has `num_repeats` $= 1$ – the low value is chosen to illustrate implementation-specific effects. The coloured contours show iso-probability credible intervals on the marginalized posterior probability density function at each parameter value due to the stochasticity of the nested sampling algorithm. The dashed dark-blue and dark-red lines show the estimated posterior means of each parameter for the blue and red runs, respectively.

concordance model is beyond the current scope; for this we refer the reader to Planck Collaboration (2013).

Given the ΛCDM concordance model, we can describe the Universe's cosmology using only six parameters. Four of these are 'late-time' parameters, governing the physics of the Universe during and after reionization: the present-day values of the Hubble constant H_0, the baryonic and cold dark matter fractions Ω_b and Ω_c, and the optical depth of the CMB τ. The remaining two parameters delineate the primordial Universe through the amplitude A_s and tilt $n_s - 1$ of the power spectrum of comoving curvature perturbations. To aid with MCMC sampling techniques, `cosmomc` (Lewis & Bridle 2002) reparametrizes the matter fractions as $\Omega_b h^2$ and $\Omega_c h^2$ in terms of the reduced Hubble constant h, defined by $H_0 = 100h\,\mathrm{km\,s^{-1}\,Mpc^{-1}}$, and in place of the Hubble constant uses $100\theta_{MC}$ ($100\times$ the ratio of the approximate sound horizon to the angular diameter distance). For more details about the parameters, see the first *Planck* parameters paper (Planck Collaboration 2013).

Given a set of cosmological parameters, using a Boltzmann code such as `camb` (Lewis et al. 2000), one may compute theoretical CMB power spectra, which are then provided as inputs to cosmological likelihoods derived from CMB observations. We use the `Plik_lite` TT likelihood detailed by Planck Collaboration XI (2016a) and the default `CosmoChord` priors (see Handley et al. 2015b for more information); these were used in Planck Collaboration XX (2016b). The likelihood introduces a single additional nuisance parameter for measurement calibration, increasing the dimensionality of the parameter space to seven.

Fig. 13 shows estimates of implementation-specific effects for calculations using the *Planck* likelihoods and priors. Each calculation uses 500 live points. As expected, there is a clear trend showing increasing `num_repeats` reduces implementation-specific effects. Furthermore in this case the PolyChord setting `num_repeats` $= 35$ (5 times the number of dimensions) is sufficient to make such effects small for all the calculations shown.

However, as in the test cases in previous sections, significant implementation-specifics are present in the calculations if `num_repeats` is set too low. This is illustrated in Fig. 14 for `num_repeats` $= 1$; with this setting the two runs (in red and blue) differ by more than the uncertainty expected from the stochasticity of the nested sampling algorithm shown by the coloured distributions. Such implementation-specific effects can also be detected with the diagnostic tests described in Section 6 (we do not show these for brevity). In addition, Fig. C1 in Appendix C1 shows a plot of the type described in Section 4.2 for the two runs in Fig. 14.

It should be noted that in cosmology one traditionally uses likelihoods with many more nuisance parameters than in this analysis. One of the innovations that PolyChord provided to the *Planck* collaboration was its ability to exploit a fast–slow hierarchy of parameter speeds (Lewis 2013). In this context, nuisance parameters that do not require recomputation of expensive parts of the likelihood may be varied at negligible cost in comparison with the slower cosmological parameters. Increasing the number of steps in nuisance parameter directions greatly aids mixing and the reduction of implementation-specific errors. However, a full analysis of this specific case is beyond the scope of this paper.

Table 1. Summary of the diagnostic tests and plots introduced in this paper.

Diagnostic	Introduced	Summary
Posterior distribution uncertainty plots	Section 4.1	Illustrates uncertainty on posterior distributions due to the stochasticity of the nested sampling algorithm. Useful for comparing two or more runs to visually assess whether their variation imples implementation-specific effects are present. Examples are shown in Figs 3 and 14.
log X plots	Section 4.2	Shows the distribution of samples through the nested sampling process. Can be used to understand and visualize posteriors and the manner in which the software explores them, as well as to assess whether two runs are consistent. Examples are shown in Figs 4, 5, and C1.
Calculating errors due to implementation-specific effects	Section 5	Quantitatively estimates errors due to implementation-specific effects. This diagnostic provides the most information about the size implementation-specific effects, but it requires enough nested sampling runs to be able to estimate the standard deviation of their results.
Testing correlations between threads	Section 6.1	Checks whether point estimates using threads from two runs are drawn from the same distribution. Can detect implementation-specific effects when only two runs are available, but does not give insight about their size.
Testing sampling error distributions	Section 6.2	Checks whether point estimates from different runs are consistent with each other given the stochasticity of the nested sampling algorithm. This can be done qualitatively with plots or quantitatively using statistical distances, and can be used when only two runs are available.

9 SUMMARY

In this paper we introduced diagnostic tests for nested sampling software, which uses numerical techniques to generate approximately uncorrelated samples within hard likelihood constraints. As a result additional errors may be produced that would not be present if the nested sampling algorithm was performed perfectly; we term these implementation-specific effects. Detecting the presence of significant implementation-specific effects is of great importance for software users as it determines whether results and estimates of uncertainties can be relied upon, and if the settings should be changed.

We suggested two new diagnostic diagrams for visualizing nested sampling results and uncertainties, and comparing runs; these are shown in Figs 3, 4, 5, 14, and C1. Section 5 introduced a quantitative measure of implementation-specific effects, which can be used to estimate them directly if enough runs are available to estimate the standard deviation of results. In addition, Section 6 provided two diagnostic tests that can be applied with only two runs. The diagnostic tests and plots introduced in this paper are summarized in Table 1. We find that due to the larger errors from the stochasticity of the nested sampling algorithm in evidence calculations, implementation-specific errors form a smaller fraction of the total error in this case – and are consequently less important and harder to detect than in parameter estimation.

In Section 7 we empirically tested the effects of software settings and the number of dimensions on implementation-specific effects and discussed dealing with cases where nested sampling results are systematically biased. The authors' practical advice for nested sampling software users based on our experience is summarized in Section 7.5. Finally, Section 8 demonstrated the application of our diagnostics to an astronomical problem using data from the *Planck* survey.

We have written a publicly available software package NESTCHECK (Higson 2018a), which performs diagnostics on input nested sampling runs and produces plots like Figs 3, 4, 5, 14, and C1; it can be downloaded at https://github.com/ejhigson/nestcheck.

ACKNOWLEDGEMENTS

We thank the anonymous reviewer for their detailed comments and suggestions.

REFERENCES

Ahn S., Fessler J., 2003, EECS Department. University of Michigan, p. 1
Allison R., Dunkley J., 2014, MNRAS, 437, 3918
Beaujean F., Caldwell A., 2013, preprint (arXiv:1304.7808)
Buchner J., 2016, Stat. Comput., 26, 383
Chua A. J. K., Hee S., Handley W. J., Higson E., Moore C. J., Gair J. R., Hobson M. P., Lasenby A. N., 2018, MNRAS, 478, 28
Cowles M. K., Carlin B. P., 1996, J. Am. Stat. Assoc., 91, 883
DES Collaboration, 2018, Physical Review D, 98, 043526
Desvignes G. et al., 2016, MNRAS, 458, 3341
Fasano G., Franceschini A., 1987, MNRAS, 225, 155
Feroz F., Hobson M. P., 2008, MNRAS, 384, 449
Feroz F., Hobson M. P., Bridges M., 2008, MNRAS, 398, 1601
Feroz F., Hobson M. P., Cameron E., Pettitt A. N., 2013, preprint (arXiv:1306.2144)
Handley W., 2018, JOSS , 3, 849
Handley W., Hobson M., Lasenby A., 2015a, MNRAS, 15, 1
Handley W., Hobson M., Lasenby A., 2015b, MNRASL , 450, L61
Higson E., 2018a, JOSS , 3, 916
Higson E., 2018b, JOSS , 3, 965
Higson E., 2018c, JOSS , 3, 985
Higson E., Handley W., Hobson M., Lasenby A., 2017, preprint (arXiv:1704.03459)
Higson E., Handley W., Hobson M., Lasenby A., 2018, Bayesian Analysis, 13, 873
Hogg D. W., Foreman-Mackey D., 2018, ApJS , 236, 11
Holm S., 1979, Scand. J. Stat., 6, 65
Joudaki S. et al., 2016, MNRAS, 2052, 2033
Keeton C. R., 2011, MNRAS, 414, 1418
Lewis A., 2013, Phys. Rev. D, 87, 103529
Lewis A., 2015, GetDist: Kernel Density Estimation, Available from download at http://cosmologist.info/notes/GetDist.pdf
Lewis A., Bridle S., 2002, Phys. Rev. D, 66, 103511
Lewis A., Challinor A., Lasenby A., 2000, ApJ, 538, 473
Massey F. J., 1951, J. Am. Stat. Assoc., 46, 68
Murray I., 2007, PhD thesis, University College London
Planck Collaboration, 2013, A&A , 571, 1
Planck Collaboration XI, 2016a, A&A , 594, A11
Planck Collaboration XX, 2016b, A&A, 594, A20
Samushia L. et al., 2014, MNRAS, 439, 3504
Scholz F. W., Stephens M. A., 1987, J. Am. Stat. Assoc., 82, 918
Scott D. W., 2015, Multivariate Density Estimation: Theory, Practice, and Visualization. John Wiley & Sons, New York
Skilling J., 2006, Bayesian Analysis, 1, 833
Wolpert D. H., Macready W. G., 1997, IEEE T. Evolut. Comput., 1, 67

APPENDIX A: CODE

The code used to perform the numerical tests and generate the results in this paper can be downloaded at https://github.com/ejhigson/diagnostic; this provides examples of NESTCHECK's use.

APPENDIX B: NUMERICAL RESULT TABLES

Tables B1 and B2 given numerical results for the nested sampling runs plotted in Fig. 6.

Table B1. Calculation error results for the 100 nested sampling runs with a Gaussian likelihood shown in Fig. 6. The first two rows shows the true value for each estimator and the mean calculation result. The next three rows show the bootstrap error estimate, implementation error estimate (equation 7), and ratio of the implementation estimate to the standard deviation of results. The final three rows show the rms error, the implementation-specific effects estimate from equation (9), and the ratio of the two. Columns show results for the log-evidence and the mean of the first three parameters. Numbers in parentheses show the 1σ numerical uncertainty on the final digit.

	$\log \mathcal{Z}$	$\overline{\theta_1}$	$\overline{\theta_2}$	$\overline{\theta_3}$
True value	−40.9434	0.0000	0.0000	0.0000
Mean result	−40.93(3)	0.002(2)	0.000(2)	0.000(2)
σ_{values}	0.33(2)	0.022(2)	0.019(1)	0.019(1)
σ_{bs}	0.326(3)	0.0223(2)	0.0223(2)	0.0221(2)
σ_{imp}	0.07(11)	0.000(7)	0.000(3)	0.000(3)
$\sigma_{\text{imp}}/\sigma_{\text{values}}$	0.20(33)	0.00(34)	0.00(17)	0.00(17)
Values RMSE	0.33(2)	0.022(2)	0.019(1)	0.019(1)
$\sigma_{\text{imp, RMSE}}$	0.06(11)	0.000(7)	0.000(2)	0.000(3)
$\sigma_{\text{imp, RMSE}}$/RMSE	0.17(33)	0.00(34)	0.00(17)	0.00(19)

Table B2. As in Table B1 but for calculations using the LogGamma mix likelihood (equation 3).

	$\log \mathcal{Z}$	$\overline{\theta_1}$	$\overline{\theta_2}$	$\overline{\theta_3}$
True value	−40.9434	−0.5772	0.0000	−0.5772
Mean result	−40.84(3)	−0.49(18)	−0.22(18)	−0.572(3)
σ_{values}	0.34(2)	1.78(13)	1.81(13)	0.032(2)
Values RMSE	0.36(2)	1.77(12)	1.81(10)	0.032(2)
σ_{bs}	0.309(3)	0.217(2)	0.215(2)	0.0300(3)
σ_{imp}	0.15(8)	1.76(13)	1.80(13)	0.01(1)
$\sigma_{\text{imp}}/\sigma_{\text{values}}$	0.43(23)	0.993(1)	0.993(1)	0.31(30)
$\sigma_{\text{imp, RMSE}}$	0.18(6)	1.76(13)	1.80(10)	0.011(9)
$\sigma_{\text{imp, RMSE}}$/RMSE	0.50(14)	0.992(1)	0.9930(8)	0.33(28)

APPENDIX C: PLANCK SURVEY DATA LOG X PLOT

Fig. C1 shows a plot of samples' distributions in $\log X$ (of the type described in Section 4.2) using the same runs as Fig. 14. In this case as the posterior is relatively simple and unimodal, and the samples overlap closely.

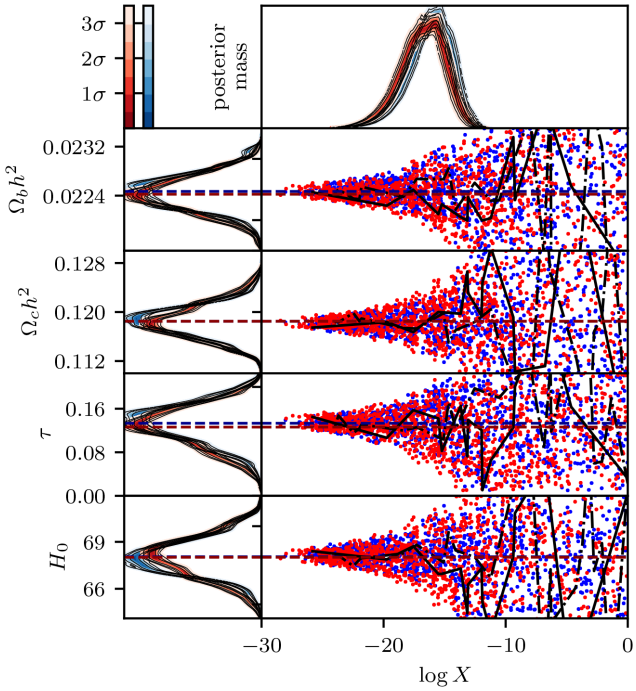


Figure C1. As for Section 6 but using the *Planck* survey likelihood. The two runs (shown in red and blue) are the same ones used for Fig. 14. The top right-hand panel shows the relative posterior mass (total weight assigned to all samples in that region) as a function of $\log X$. The final four rows show the present-day Baryon density $\Omega_b h^2$, the present-day cold matter density $\Omega_c h^2$, the optical depth of the CMB τ, and the present-day Hubble constant H_0. The coloured contours show iso-probability credible intervals on the marginalized posterior probability density function at each parameter or $\log X$ value. In each row, the estimated posterior means for the blue and red runs are shown with dashed dark-blue and dark-red lines. The solid and dot–dashed black lines show the evolution of an individual thread chosen at random from the red and blue runs, respectively.

This paper has been typeset from a TEX/LATEX file prepared by the author.