Black-box density function estimation using recursive partitioning Supplementary material

Erik Bodin [1] **Zhenwen Dai** [2] **Neill D. F. Campbell** [3] **Carl Henrik Ek** [4]

1. Down-stream tasks and applications

- Analytical expectations of functions with respect to the approximation, $\mathbb{E}_{\hat{\mathcal{P}}(\boldsymbol{\theta})}[g(\boldsymbol{\theta})]$, where $\hat{\mathcal{P}}(\boldsymbol{\theta}) \propto \hat{f}(\boldsymbol{\theta})$, and g is an arbitrary function of $\boldsymbol{\theta}$.
- Using $\hat{f}(\boldsymbol{\theta})$ as a proxy, allowing density queries without resorting to f, and constant-time re-sampling.
- Conditional approximations $\hat{\mathcal{P}}(\boldsymbol{\theta}_a|\boldsymbol{\theta}_b)$, where $\boldsymbol{\theta} = \{\boldsymbol{\theta}_a, \boldsymbol{\theta}_b\}$. without requiring more evaluations of f,
- Marginal approximations $\hat{\mathcal{P}}(\boldsymbol{\theta}_a) = \int \mathcal{P}(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b) d\boldsymbol{\theta}_b$, where $\mathcal{P}(\boldsymbol{\theta}_a)$ may optionally be substituted with $\hat{\mathcal{P}}(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b)$. Note that this constitute an integral density estimation problem for every $\boldsymbol{\theta}_a \in \Omega_a$, and where DEFER chooses a finite collection of points in a decision loop and constructs a tree akin to the other problems. For the inner loop of estimating each integral $\int \mathcal{P}(\boldsymbol{\theta}_a = \boldsymbol{\theta}_a, \boldsymbol{\theta}_b) d\boldsymbol{\theta}_b$ we may use an arbitrary integration method, including DEFER.
- Marginalisation also through arbitrary conditionals, such as $\int \mathcal{P}(\boldsymbol{\theta}_a|c(\boldsymbol{\theta}_b))\mathcal{P}(\boldsymbol{\theta}_b) d\boldsymbol{\theta}_b$ where c is a boolean-valued function. For an example, see Figure 8 in the paper.
- Composites of use-cases above, like estimating mutual information $I(\boldsymbol{\theta}_a; \boldsymbol{\theta}_b) = \mathbb{E}_{\mathcal{P}(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b)}[\log \frac{\mathcal{P}(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b)}{\mathcal{P}(\boldsymbol{\theta}_a)\mathcal{P}(\boldsymbol{\theta}_b)}]$, where the joint and both marginals may be approximated first, followed by the analytical expectation of density ratio term with respect to $\hat{\mathcal{P}}(\boldsymbol{\theta}_a, \boldsymbol{\theta}_b)$, querying the marginal approximation densities using tree search.
- Any use-case above in a (axis-aligned, hyper-rectangular) subregion of Ω, including mass integration or sampling, by forming a tree from a query region of the tree.
- Divergence estimation between different distributions over the same domain (or an overlap via the above).

We provide code and examples at https://github.com/bodin-e/defer.

2. Algorithm

Keeping track of $\hat{Z}$ and the highest mass partitions At every step of the algorithm, we keep track of the current total mass estimate $\hat{Z}$, and the top M mass partitions, as required for checking CR2 and CR3.

We implement both of these through aggregators which are updated at each *include* and *exclude* of a partition in the *leaf set*. The leaf set is the partitions that cover the domain in a non-overlapping fashion at the currently finest resolution, constituting the current Riemann sum. 'Include' refers to the creation of a new child node (and partition), and 'exclude' to the removal of the previous (parent) node from this set. To easily keep track of all the updates a partition division should lead to, we wrap the 'divide function' in another function that after division (the forming of child partitions), makes all the related updates, including the exclusion of the divided parent node.

The aggregator for the total mass estimate keeps track of the accumulate sum, where inclusion is addition and exclusion is subtraction. The aggregator for the top M partitions keeps track of such a current set of size up to M, where set updates are handled within the inclusion and exclusion functions.

Note on implementation Densities of typical density functions can have exceedingly small values, risking underflow using default float precisions. In the implementation, we use a 128-bit float to present the total mass aggregate and the density evaluation stored in each respective partition node object. An alternative is to represent the logarithm of the total mass and the densities, but these values would need to either be transformed to higher precision before being exponentiated and used to avoid underflow, or all checks and operations would need to be performed on the logarithm, which is more complicated.

[1]University of Bristol, United Kingdom [2]Spotify, United Kingdom [3]University of Bath, United Kingdom [4]University of Cambridge, United Kingdom. Correspondence to: Erik Bodin <mail@erikbodin.com>.

Proceedings of the 38^{th} International Conference on Machine Learning, PMLR 139, 2021.

2.1. Representer points

To check criterion CR2 and CR3 there are representer points to be chosen. The representer points are used to efficiently (but approximately) check for overlaps between respective partitions and the two spaces $\mathbf{\Phi}_{\text{linear}}$ and $\mathbf{\Phi}_{\text{balls}}$, associated with CR2 and CR3 respectively. Note that this is an approximate method for checking overlaps, and although there may be no false positives, there may be false negatives.

For each representer point, a tree search is performed to find the associated partition, which will be divided. Note that if a partition fulfils multiple individually sufficient criteria (CR1 to CR3), or is 'hit' by multiple representer points, it will still only be divided once.

CR2 We begin with the determination of $\mathbf{\Phi}_{\text{linear}}$, followed by $\boldsymbol{R}_{\text{linear}}$. To form the linear subspace $\mathbf{\Phi}_{\text{linear}}$ we carry out the following steps. We first obtain the corresponding centroids Θ_H of the H partitions as stacked vectors, which is then used to form a basis for the $(H-1)$-dimensional hyperplane being $\mathbf{\Phi}_{\text{linear}}$. Specifically, we let $\boldsymbol{\theta}_j$ be the centroid (column vector) among them closest to the centre of the unit cube, and $E = \Theta_{H,\neq j} - \boldsymbol{\theta}_j$ be the stacked basis vectors of the hyperplane. Then we obtain an orthogonal basis A using QR factorization of E. Let $\tilde{E}$ be the re-scaled version of E having unit-norm. Now we can map a vector u from a unit-cube onto a given hyper-plane as $\boldsymbol{\theta}_j + uA\tilde{E}$, as specified by the given centroids.

From $\mathbf{\Phi}_{\text{linear}}$ a discrete set of points $\boldsymbol{R}_{\text{linear}}$ is now to be chosen. We will concentrate the points on all lower-dimensional hyper-planes formed by all combinations of the H high mass partitions, which in line with the assumed heuristic that mass tends to concentrate on linear subspaces formed by these partitions. In practice this is implemented by ahead of time determining all $\sum_{s=1}^{S} \binom{M}{s+1}$ combinations of indices up to M where s is the dimensionality of a lower-dimensional hyperplane. These index combinations are then iterated through at runtime at step t, each one creating a subset H_* of high mass partitions forming a linear subspace. If a given set of centroids is colinear, it is skipped, as they would then only form a hyperplane of dimension *less than* s, and thus already be taken care of by another hyperplane. For each H_* we add a representer point at their average location in Ω (the weighted centre of the simplex they form). In addition, to spread points also throughout the linear subspace (and the constituent lower-dimensional linear subspaces), we add l representer points per H_* by sampling uniformly in a unit cube and map the points onto each lower dimensional linear subspace, in addition to the (highest dimensional) subspace $\mathbf{\Phi}_{\text{linear}}$, using the procedure described above (we use $l = 1$ in all experiments).

CR3 We now address the choice of the discrete set of points $\boldsymbol{R}_{\text{balls}}$ from $\mathbf{\Phi}_{\text{balls}}$. For each partition in H, we sample b points uniformly within the corresponding D-ball using the Muller method (Harman & Lacko, 2010). In practice we set $b = D$ in all experiments. The representer points are used as specified in the paper.

2.2. Algorithmic complexity analysis

We will now address the complexity analysis of the resulting algorithm. As discussed in the paper, the algorithm's time complexity must allow short decision times and scalability to a large number of partitions. In practice, we aim to maintain sub-millisecond decision times even after millions of density function evaluations. To simplify the analysis, we will assume that the number of iterations of the algorithm (Algorithm 1) is proportional to N_T. It will always be true that the number of iterations *is less than* N_T, as multiple partitions will be divided at each iteration, and also, each partition division will yield multiple function evaluations.

CR1 Updating the hash map of heaps for a new partition has average time complexity $\mathcal{O}(\log \frac{N_t}{U})$, where U is the number of unique abscissas (see Section 4). This is because the heap, found in constant time using the hash map, provides logarithmic time updates, where N_t/U is the average number of elements in a heap. As $N_t > U$ this simplifies as the worst-case $\mathcal{O}(\log \frac{N_t}{U}) = \mathcal{O}(\log N_t - \log U) = \mathcal{O}(\log N_t)$. Note that this requires the tree to be balanced in the sense that the number of levels of the tree grows logarithmically with respect to the number of nodes. Maximum tree imbalance would happen if, at each iteration, the node that is the highest number of levels away from the root is chosen to be divided. However, note that such a node (or partition) would have an associated volume that exponentially tends to zero, with a denominator in the base at least *three*, $(\frac{1}{3})^t$, which as a result would be very quickly dominated by other partitions. This is an important fact, as all partition division criteria are either based on upper bound *mass*, proportional to volume, or volume together with spatial vicinity. As a result, all criteria encourage the balancing of the tree. In practice, we will later see empirically that a logarithmic tree lookup time is maintained, evidenced by that runtime time cost of iterations of the whole algorithm is no worse than logarithmic.

Retrieving all top ordinate partitions from this structure at a given iteration t has time complexity $\mathcal{O}(U)$ as the hash table with U entries is traversed linearly, and the root of each heap is available in constant time. Using the U obtained coordinates, the convex hull is computed in $\mathcal{O}(U \log U)$ using Graham's scan (Graham, 1972). Following this, the hull's upper-right quadrant is obtained in $\mathcal{O}(U)$.

In summary we obtain the worst-case time complexity

$\mathcal{O}(\log N_t + U \log U)$ for checking this criterion for all partitions at step t. The space complexity of the hash map of heaps data structure is linear with respect to N_t.

CR2 The checking of this criterion entails computation associated with $\sum_{s=1}^{S} \binom{M}{s+1}$ linear subspaces, where M is the *maximum* number of high mass partitions (H). The number of partitions that will be divided as a result grows proportionally to this constant, so we simplify the analysis to treating one such linear subspace and an associated constant set of representer points. The two largest terms come from the QR factorization, with time complexity $\mathcal{O}(M^2 D)$, and the tree-search for the partition of a representer point has time complexity $\mathcal{O}(\log N_t)$. The space complexity is $\mathcal{O}(M)$ of keeping track of the high mass partitions. As D is constant with respect to N_t and M is upper bounded by D, we summarise this as time complexity $\mathcal{O}(\log N_t)$ and the space complexity as constant.

CR3 For the check of this criterion, all up to M high mass partitions are sampled in $\mathcal{O}(MD)$ time, and the tree-search for a partition is $\mathcal{O}(\log N_t)$. Analogous to CR2, we summarise this as a time complexity $\mathcal{O}(\log N_t)$ and the space complexity as constant.

Summary The combined time complexity of a step t is $\mathcal{O}(\log N_t + U \log U)$, where U is the number of *unique* abscissas (see paper). For an average number of steps proportional to N_T this results in a total average time complexity of the algorithm as $\mathcal{O}(N_T(\log N_T + \bar{U} \log \bar{U}))$. The space complexity (including storage of partitions) of the algorithm is linear, i.e. remains proportional to the number of observations. In Section 3.1 we show empirically that $\bar{U}$ is sufficiently small, and close to constant with respect to N_T, leading to a fast and scalable algorithm.

3. Experiments

3.1. Runtime experiments

The DNS and DEFER were run (single-threaded) on a 2.6 GHz Intel Core i7, and PTMCMC used multiple cores due to its implementation. In Figure 3 we note both that DEFER has a similar algorithmic cost to the other methods, and that the DEFER has a near constant cost per function evaluation with respect to N_T. With algorithmic cost we refer to the cost per 'decision' of where to evaluate the density function, which is computed from the total (wall-clock) time of inference minus the total function evaluation time, divided by the number of function evaluations made. We also illustrate the common situation where the function evaluation time dominates the decision time, making sample-efficiency a critical concern for total efficiency in these cases.

As discussed in Section 2.2, the time complexity of the algorithm is $\mathcal{O}(N_T(\log N_T + \bar{U} \log \bar{U}))$. To achieve the near linear scalability empirically illustrated, it is clear that $\bar{U}$ must be close to constant with respect to N_T. In Figure 4 we confirm this explicitly.

3.2. Ablation study of criteria

For a small visual ablation study of the partition division criterion (see Section 4 of the paper), see Figure 5 and Figure 6.

3.3. Baseline setups

We used the following implementations: DNS (Speagle, 2020), slice sampling (Abadi et al., 2016), and PTMCMC (Ellis & van Haasteren, 2017).

Slice sampling We use an initial step size corresponding to 0.05 of the unit domain sides, with max_doublings set to 5. The used burn-in ratio is 25%.

PTMCMC We follow the default settings and sample p_0 uniformly within the domain and set the initial covariance matrix to be diagonal with variance 0.01. We use covUpdate=500, 25% burn-in and all parameters set to the default.

DNS We use the default (as all other settings) of nlive_init=500, in practice nlive_init=min(500, 2 + int(N_T / 10)) as the max number of function evaluations N_T in a few experiments are small. For 'Posterior mode' pfrac = 1.0, and for 'Evidence mode' pfrac = 0.0.

3.4. Real-world density surfaces

3.4.1. Gravitational-wave physics

Problem background Motivation for use cases of this algorithm can be found in natural science research areas such as gravitational-wave physics (Collaboration et al., 2020; Abbott et al., 2019; Mandel & Fragos, 2020). In GW research, scientific knowledge is often expressed in the form of physically motivated likelihood functions and priors (Kalaghatgi et al., 2020). An increasing sophistication has brought challenges from an inference standpoint. Tractable gradients are often missing, the functions may exhibit undefined (or zero density) regions, and the induced density surfaces tend to be multi-modal, discontinuous, have mass concentrated in tiny regions, and exhibit complicated correlations. As a consequence, inference can be prohibitively slow even for problems of around ten dimensions. Simple techniques such as standard rejection sampling are typically infeasible due to small typical sets, requiring a prohibitively large number of function evaluations to obtain representative samples. Markov Chain Monte Carlo methods, generally popular for their asymptotic guar-

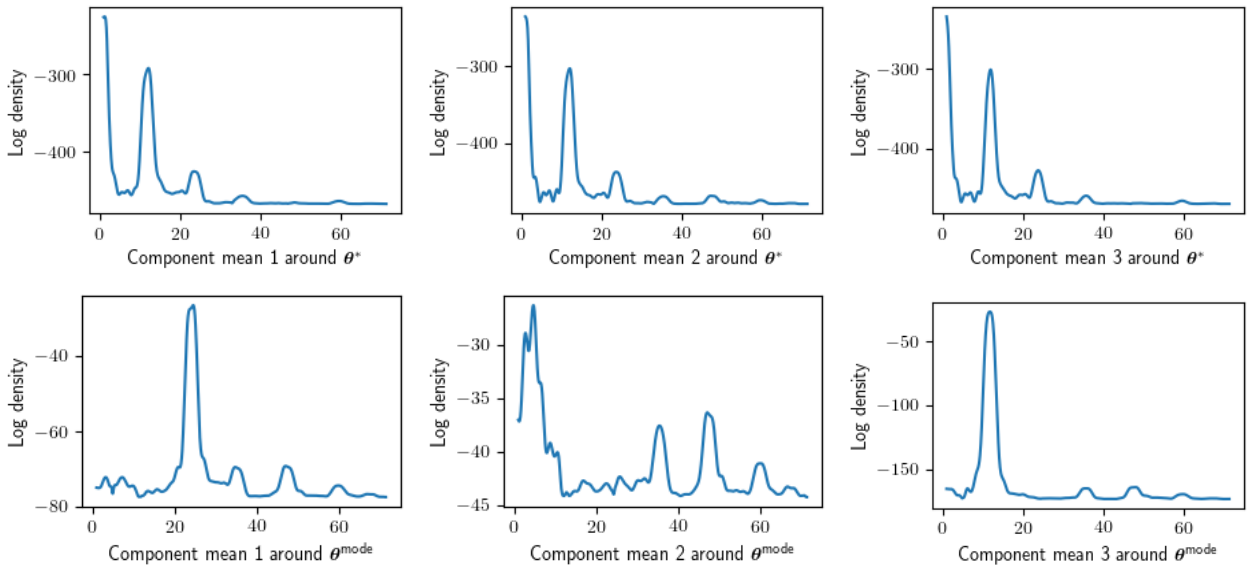

Figure 1. Multi-modality in the posterior of the parameters of the Spectral Mixture kernel. Shown is how the log density of the model changes with respect to each component mean within the Gaussian mixture representing the spectral density of the kernel. In the upper row the density changes with respect to each mean are shown at an uniformly sampled position in the (10D) domain. In the lower row these shown at the (estimated) mode of the parameter posterior. The estimation of the posterior mode was done using DEFER.

antees (Geyer, 1992) and strengths in high dimension (Neal et al., 2011), struggle in handling the multi-modality present in these problems. Furthermore, MCMC does not provide evidence estimation, which often is of crucial importance in scientific applications and elsewhere.

A popular family of methods to deploy instead is Nested Sampling (Higson et al., 2019; Collaboration et al., 2020), known for performing well on multi-modal and degenerate posteriors, as well as additionally providing evidence estimation. Still, the computation required to run an experiment can often be impractical, such as weeks on a cluster. Resampling, density re-weighting and local density integration have been identified as important tasks to save computation, when considering different priors (Mandel & Fragos, 2020) or estimating equations of state (Vivanco et al., 2019). Currently these tasks, as well as parameter and evidence estimation, require a portfolio of different algorithms with various tuning parameters, requiring significant expertise and effort to obtain reliable results. DEFER outputs a density function approximation with support for these tasks; the latter via making use of the domain-indexed search tree over partitions.

We apply DEFER to a simulated signal example from (Ashton et al., 2019) similar to (Collaboration et al., 2020). Shown in the corresponding figure in the paper are all the 2D marginals of a six-dimensional problem using the 'IMRPhenomPv2' waveform approximant. Inferred parameters are, for example, the luminosity distance, and the spin magnitudes of binary black-holes. We note the complicated interactions between parameters, showing the importance of handling multi-modality and strong correlations. Importantly, DEFER is able to handle the surface well without any tuning parameters.

To see results of the different baselines after varying budgets, see Figure 7 and Figure 8, where the latter figure is a modified example where the masses of the two black holes instead of the luminosity distance to the source and the orbital phase are being inferred, keeping the other four parameters.

Details The problem addressed concerns inferring model parameters to explain an event involving two black holes in a binary system. This 4D example (link) was turned into a 6D example with the injection of all parameters except the following six parameters, which are inferred:

- The luminosity distance to the source dL.
- Phase: The orbital phase of the binary at the reference time.
- Theta JN: The inclination of the system's total angular momentum with respect to the line of sight.
- Psi: The polarisation angle describes the orientation of the projection of the binary's orbital momentum vector onto the plane on the sky.
- Dimensionless spin magnitude a_1.

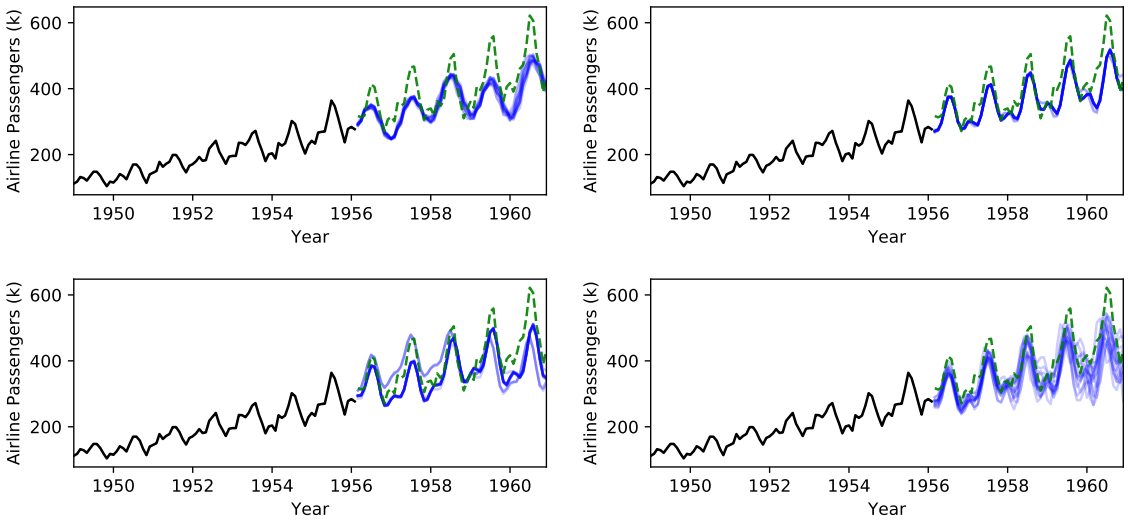

Figure 2. Time-series forecasting using Gaussian Processes with the Spectral Mixture (Wilson & Adams, 2013) kernel. The data is shown in black, ground truth in green and posterior GP samples in blue. Shown from the top left is slice sampling, followed by PTMCMC on its right, then DNS and lastly DEFER in the bottom right.

- Dimensionless spin magnitude a_2.

For more details, see (Veitch et al., 2015). To produce a deterministic surface the seed of the pseudo-random generator must be fixed. To set the global seed used by the waveform approximant, while not fixing the seed for DEFER or the baseline stochastic methods, we maintain the state of the pseudo-random generator used within and outside the function, respectively.

To see results of the different baselines after varying budgets, see Figure 7 and Figure 8, where the latter figure is a modified example where the masses of the two black holes instead of the luminosity distance to the source and the orbital phase are being inferred, keeping the other four parameters.

3.4.2. GP REGRESSION WITH SPECTRAL MIXTURE KERNEL

Problem background Apart from inferring a complex posterior surface, we also compare the quality of the posterior samples in terms of prediction accuracy. We consider a time-series forecasting problem because the parameter posterior of such problems are often complex and multi-modal, generally making approximate inference more difficult. We use a Gaussian process (GP) time-series regression model with a spectral mixture kernel (SMK) (Wilson & Adams, 2013). SMK can approximate any stationary kernel including periodic ones by learning the parameters of a Gaussian mixture to represent the kernel spectral density (Bochner, 1959). However, inference of the parameters of the mixture can be difficult as the induced density surface is multi-modal. In Figure 1 we confirm the presence of multi-modality, which may cause local optimizers and step-wise sampling techniques like MCMC to get stuck in poor solutions.

We use the airline passengers data (Wilson & Adams, 2013) and use the first 60% of months for training and the rest for test. The GP model has three mixture components plus a linear slope, which translates into a 10D parameter inference task. With a budget of 50k function evaluations, the negative log-likelihoods on the test data are 377.66, 365.97, 236.89 and **205.50**, for slice sampling, PTMCMC, DNS and DEFER, respectively. For predictions, see Figure 2. DEFER performs better than other methods, which evidenced by the figure, may be explained by better capturing multi-modality and thus a larger breadth of possible solutions. In contrast, the MCMC techniques capture only a single mode each.

Details PyTorch Spectral Mixture kernel GP example. This example was adapted for the Airline Passengers dataset used in (Wilson & Adams, 2013).

3.5. Synthetic density functions

The synthetic density functions used in the quantitative experiments are the following. All domains are scaled to be a unit hybercube.

Student's t The scale parameter was 0.01. The mean parameter was uniformly sampled $\mathcal{U}(0.2, 0.8)$ per dimension for each of the 20 runs. The degrees of freedom was set to $2.5 + (D/2)$.

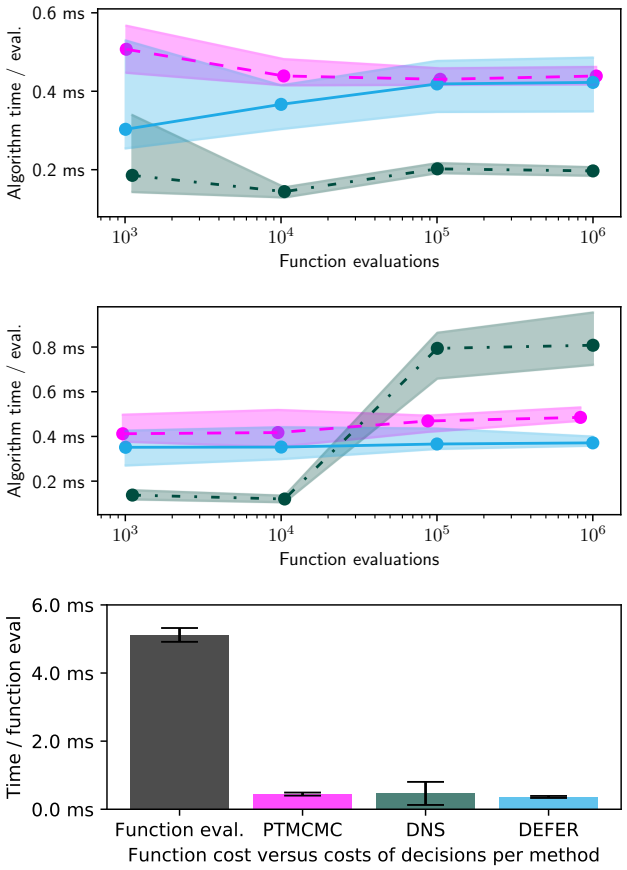

Figure 3. Algorithmic time. Shown in the two plots from the left is the associated runtime cost of each algorithm to decide where to evaluate the density function, for the Student's t 10D density function and gravitational-wave example, respectively (for legend see the bar plot on the right). Shown on the right is the time spent on evaluating the function versus the algorithm cost of respective method, with mean and standard deviation across 20 runs using the various settings of function evaluation budgets N_T.

Canoe

$$f(\boldsymbol{\theta}) = \max(v(\boldsymbol{\theta}), 0), \tag{1}$$

$$v(\boldsymbol{\theta}) = 2 + 5\mathcal{N}\boldsymbol{\theta}|(\boldsymbol{\mu}, \boldsymbol{\Sigma}_{\text{inner}}) - 10\mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma}_{\text{outer}}), \tag{2}$$

where $\boldsymbol{\mu} = \mathbf{0.5}$, $\boldsymbol{\Sigma}_{\text{outer}} = a$, $\boldsymbol{\Sigma}_{\text{inner}} = 0.01(0.95J_D + 0.05I_D)$. and $\boldsymbol{\Sigma}_{\text{outer}} = 0.02(0.60J_D + 0.40I_D)$. J_D is a $D \times D$ matrix of all ones, and I_D is the identity matrix.

Mixture of Gaussians

$$f(\boldsymbol{\theta}) = 2.5\mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_{\text{a}}, \boldsymbol{\Sigma}_{\text{a}}) + \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}_{\text{b}}, \boldsymbol{\Sigma}_{\text{b}}), \tag{3}$$

where $\boldsymbol{\mu}_a = [0.6326, 0.7401, 0.7232, 0.2471]$, $\boldsymbol{\mu}_b = [0.5139, 0.4667, 0.3777, 0.7995]$ (sampled from $\mathcal{U}(0.2, 0.8)$), and

$$\boldsymbol{\Sigma}_{\text{a}} = 0.01^2 \begin{bmatrix} 2.25 & -1.0 & 0 & 0 \\ -1.0 & 2.25 & 0 & 0 \\ 0 & 0 & 2.25 & 0 \\ 0 & 0 & 0 & 2.25 \end{bmatrix}, \tag{4}$$

$$\boldsymbol{\Sigma}_{\text{b}} = 0.01^2 \begin{bmatrix} 2.25^2 & -2.25 & 1.0 & -1.0 \\ -2.25 & 2.25^2 & 0 & 0 \\ 1.0 & 0 & 2.25^2 & 0 \\ -1.0 & 0 & 0 & 2.25^2 \end{bmatrix}. \tag{5}$$

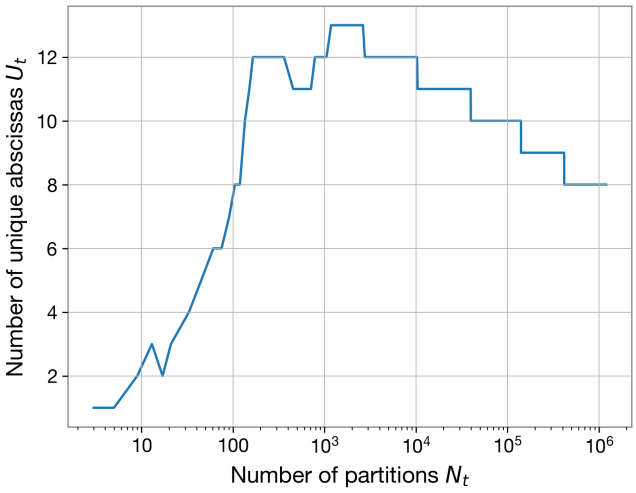

Figure 4. Shown in the plot is the number of unique abscissas U_t per number of partitions N_t, as the algorithm executes. Note that U_t does not continue growing with N_t after a while, here after about 100 partitions. Used for the illustration is the Student's t 10D density function.

Cigar

$$f(\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{\theta}|\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{6}$$

where $\boldsymbol{\mu} = \mathbf{0.5}$ and $\boldsymbol{\Sigma} = 0.01(0.99J_D + 0.01I_D)$. J_D is a $D \times D$ matrix of all ones, and I_D is the identity matrix.

References

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., et al. Tensorflow: A system for large-scale machine learning. In *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)*, pp. 265–283, 2016.

Abbott, B., Abbott, R., Abbott, T., Abraham, S., Acernese, F., Ackley, K., Adams, C., Adhikari, R., Adya, V., Affeldt, C., et al. Gwtc-1: A gravitational-wave transient catalog of compact binary mergers observed by ligo and virgo during the first and second observing runs. *Physical Review X*, 9(3):031040, 2019.

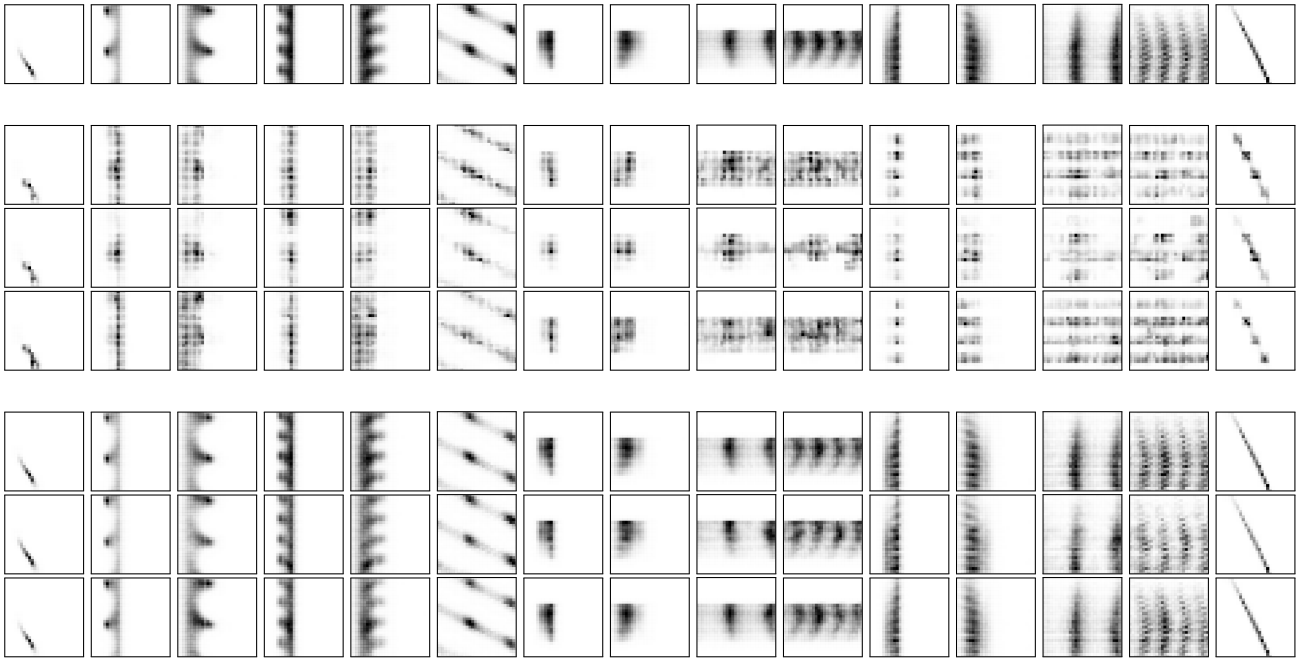

Figure 5. Visual ablation study of criteria on the gravitional-physics example. Shown are histograms of the two-dimensional marginals of the 6D problem (see Section 3.4.1), with samples produced after 200k density functions evaluations in the middle three rows, and after 5M evaluations in the bottom three rows. Shown in order, per group of three, is DEFER using all criteria (CR1-3), excluding CR2 (CR1,3), and excluding CR3 (CR1,2). The top row shows DEFER (using all criteria) after 10M evaluations for reference. Note that CR1 cannot be excluded as CR2 and CR3 would not explore the sample space at all on their own. We see that CR2, which complements the search along affine subspaces of high mass partitions, helps to 'fill in gaps' in various directions earlier than otherwise. CR3 has little effect in this example, but has seemingly no negative impact. For another example, see Figure 6.

Ashton, G., Hübner, M., Lasky, P. D., Talbot, C., Ackley, K., Biscoveanu, S., Chu, Q., Divakarla, A., Easter, P. J., Goncharov, B., et al. Bilby: A user-friendly bayesian inference library for gravitational-wave astronomy. *The Astrophysical Journal Supplement Series*, 241(2):27, 2019.

Bochner, S. *Lectures on Fourier integrals*, volume 42. Princeton University Press, 1959.

Collaboration, L. S., Collaboration, V., et al. Gw190412: Observation of a binary-black-hole coalescence with asymmetric masses. *arXiv preprint arXiv:2004.08342*, 2020.

Ellis, J. and van Haasteren, R. jellis18/ptmcmcsampler: Official release, October 2017. URL https://doi.org/10.5281/zenodo.1037579.

Geyer, C. J. Practical markov chain monte carlo. *Statistical science*, pp. 473–483, 1992.

Graham, R. L. An efficient algorithm for determining the convex hull of a finite planar set. *Info. Pro. Lett.*, 1: 132–133, 1972.

Harman, R. and Lacko, V. On decompositional algorithms for uniform sampling from n-spheres and n-balls. *Journal of Multivariate Analysis*, 101(10):2297–2304, 2010.

Higson, E., Handley, W., Hobson, M., and Lasenby, A. Dynamic nested sampling: an improved algorithm for parameter estimation and evidence calculation. *Statistics and Computing*, 29(5):891–913, 2019.

Kalaghatgi, C., Hannam, M., and Raymond, V. Parameter estimation with a spinning multimode waveform model. *Phys. Rev. D*, 101:103004, May 2020. doi: 10.1103/PhysRevD.101.103004. URL https://link.aps.org/doi/10.1103/PhysRevD.101.103004.

Mandel, I. and Fragos, T. An alternative interpretation of gw190412 as a binary black hole merger with a rapidly spinning secondary. *The Astrophysical Journal Letters*, 895(2):L28, 2020.

Neal, R. M. et al. Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2, 2011.

Speagle, J. S. dynesty: a dynamic nested sampling package for estimating bayesian posteriors and evidences. *Monthly Notices of the Royal Astronomical Society*, 493(3):3132–3158, 2020.

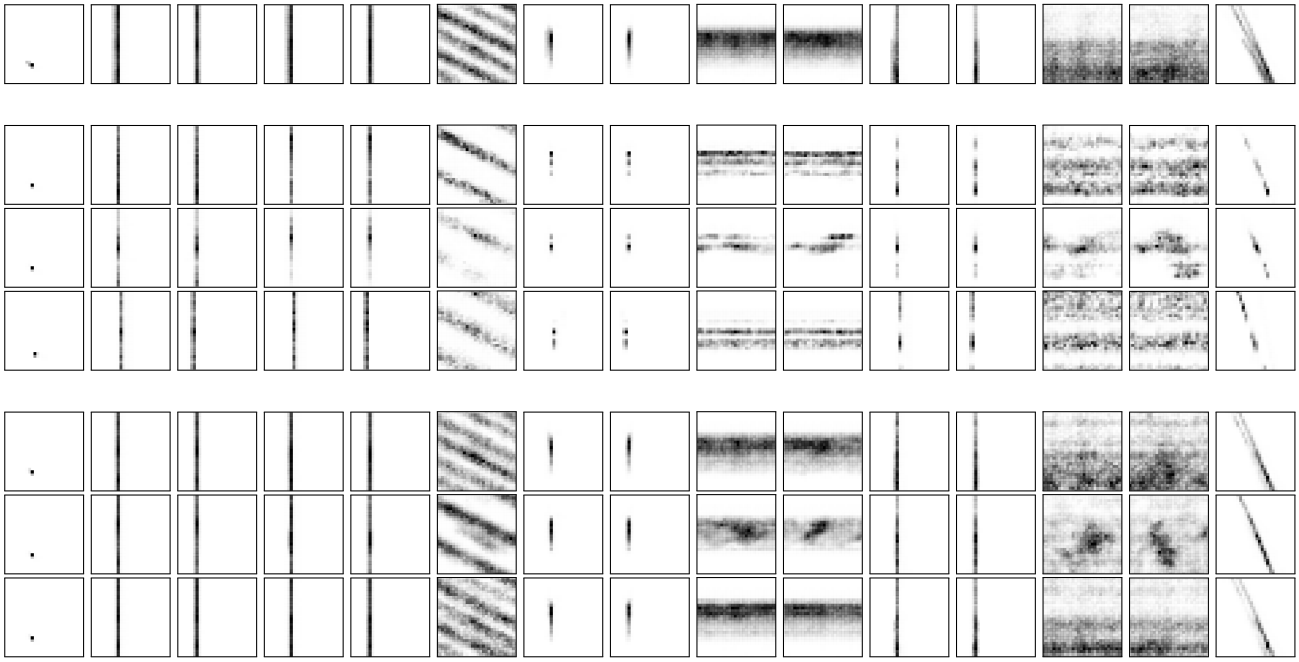

Figure 6. Visual ablation study of criteria on another gravitional-physics example, this time inferring the masses of the two black holes instead of the luminosity distance to the source and the orbital phase, together with the four other parameters (see Section 3.4.1). Shown are histograms of the two-dimensional marginals, with samples produced after 500k density functions evaluations in the middle three rows, and after 5M evaluations in the bottom three rows. Shown in order, per group of three, is DEFER using all criteria (CR1-3), excluding CR2 (CR1,3), and excluding CR3 (CR1,2). The top row shows DEFER (using all criteria) after 10M evaluations for reference. Note that CR1 cannot be excluded as CR2 and CR3 would not explore the sample space at all on their own. Similar to in Figure 5, we see that CR2 helps to fill in details earlier than otherwise, and the same is true for CR3 although to a lesser extent.

Veitch, J., Raymond, V., Farr, B., Farr, W., Graff, P., Vitale, S., Aylott, B., Blackburn, K., Christensen, N., Coughlin, M., et al. Parameter estimation for compact binaries with ground-based gravitational-wave observations using the lalinference software library. *Physical Review D*, 91(4): 042003, 2015.

Vivanco, F. H., Smith, R., Thrane, E., Lasky, P. D., Talbot, C., and Raymond, V. Measuring the neutron star equation of state with gravitational waves: The first forty binary neutron star merger observations. *Physical Review D*, 100(10):103009, 2019.

Wilson, A. and Adams, R. Gaussian process kernels for pattern discovery and extrapolation. In *International conference on machine learning*, pp. 1067–1075, 2013.

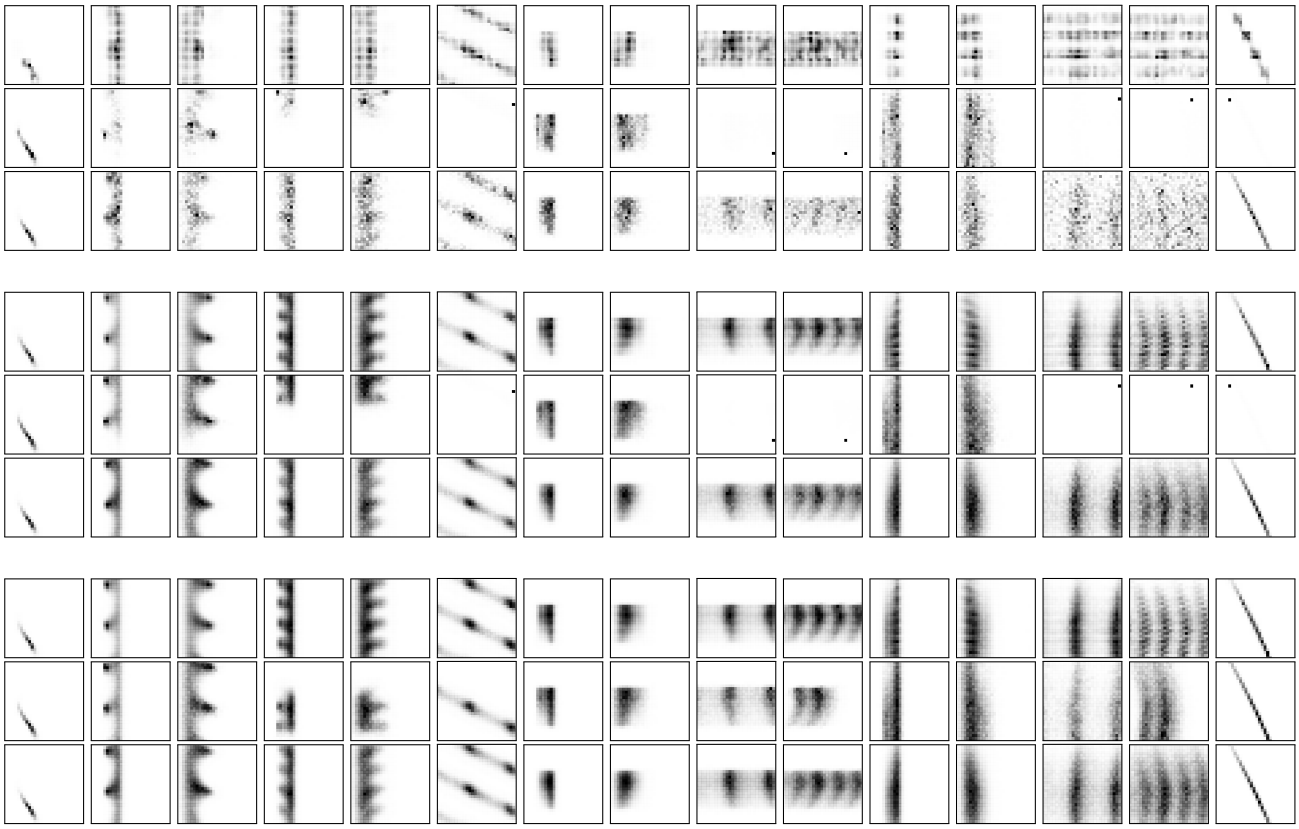

Figure 7. Gravitional-physics parameter inference using the different methods. Shown are histograms of the two-dimensional marginals of the 6D problem (see Section 3.4.1), with samples produced after 200k density functions evaluations in the upper three rows, after 5M evaluations in the middle three rows, and efter 10M evaluations in the bottom three rows. Shown in order, per group of three, is DEFER, PTMCMC and DNS. DEFER is able to capture the surface well. PTMCMC captures areas around some modes well, but fails to capture all modes. Note that we perform independent runs for each budget. For another gravitional-physics example, see Figure 8.

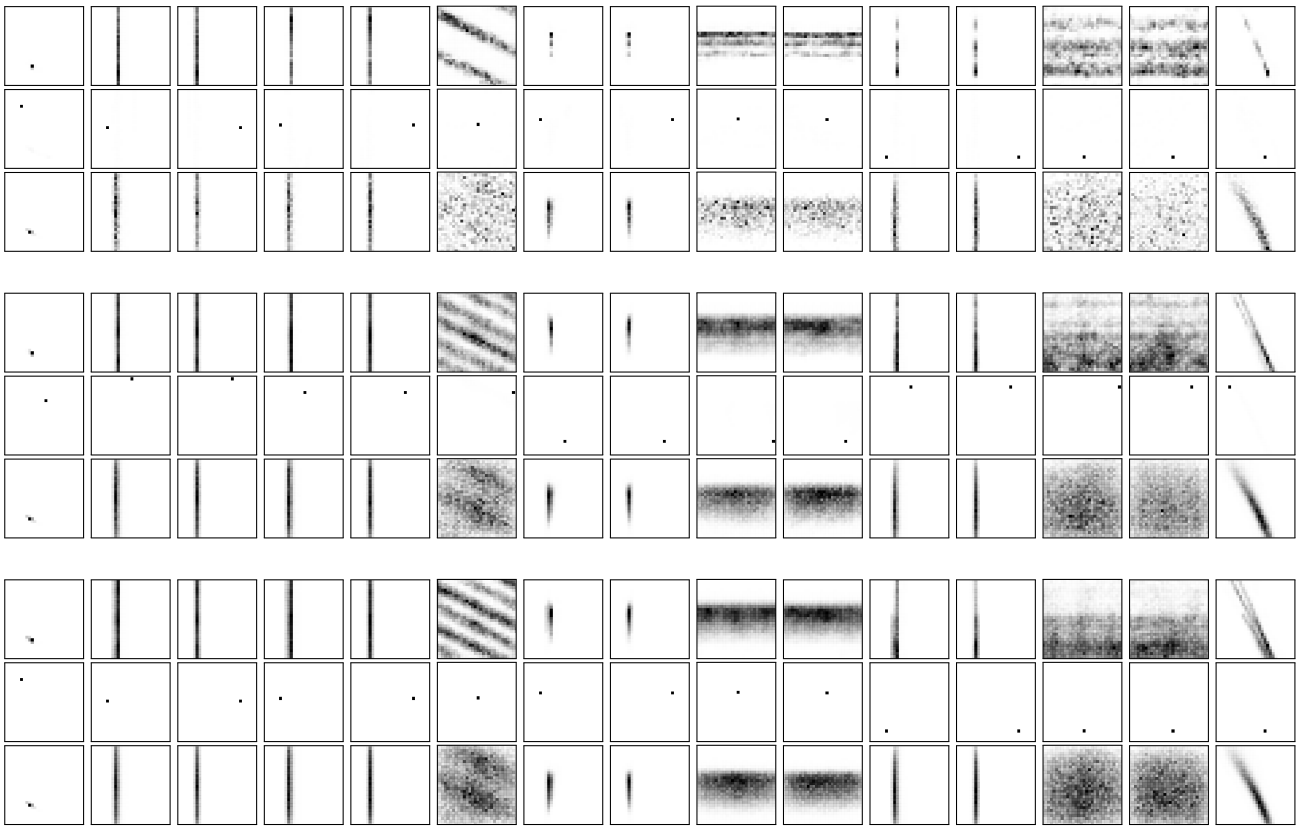

Figure 8. Gravitional-physics parameter inference using the different methods. Shown are histograms of the two-dimensional marginals of another 6D problem, this time inferring the masses of the two black holes instead of the luminosity distance to the source and the orbital phase, together with the four other parameters (see Section 3.4.1). The samples are produced after 500k density functions evaluations in the upper three rows, after 5M evaluations in the middle three rows, and efter 10M evaluations in the bottom three rows. Shown in order, per group of three, is DEFER, PTMCMC and DNS. DEFER is able to capture the surface in detail. PTMCMC got stuck on an insignificant mode within the available budget, and DNS seemingly captures a lower fidelity surface. Note that we perform independent runs for each budget. Dynamic Nested Sampling (DNS) is designed to automatically adapt the number of live points, which controls the fidelty of the nested sampling algorithm. A possible explanation for the lack of fidelity of DNS on this example could be a failure of the algorithm to detect a need for additional live points. Note that, on the other hand, a too high number of live points for a given problem lowers the overall sample-efficiency of the nested sampling algorithm.