Density-clustering of continuous gravitational wave candidates from large surveys

B. Steltner,[1,2,*] T. Menne,[1,2] M. A. Papa,[1,2,3] and H.-B. Eggenstein[1,2]

[1]*Max Planck Institute for Gravitational Physics (Albert Einstein Institute), Callinstrasse 38, 30167 Hannover, Germany*

[2]*Leibniz Universität Hannover, D-30167 Hannover, Germany*

[3]*University of Wisconsin Milwaukee, 3135 N Maryland Avenue, Milwaukee, Wisconsin 53211, USA*

(Received 2 August 2022; accepted 4 November 2022; published 30 November 2022)

Searches for continuous gravitational waves target nearly monochromatic gravitational wave emission from, e.g., nonaxisymmetric fast-spinning neutron stars. Broad surveys often require us to explicitly search for a very large number of different waveforms, easily exceeding $\sim 10^{17}$ templates. In such cases, for practical reasons, only the top, say $\sim 10^{10}$, results are saved and followed up through a hierarchy of stages. Most of these candidates are not completely independent of neighboring ones, but arise due to some common cause: a fluctuation, a signal, or a disturbance. By judiciously clustering together candidates stemming from the same root cause, the subsequent follow-ups become more effective. A number of clustering algorithms have been employed in past searches based on iteratively finding symmetric and compact overdensities around candidates with high detection statistic values. The new clustering method presented in this paper is a significant improvement over previous methods: it is agnostic about the shape of the overdensities, is very efficient and it is effective: at a very high detection efficiency, it has a noise rejection of 99.99%, is capable of clustering two orders of magnitude more candidates than attainable before and, at fixed sensitivity it enables more than a factor of 30 faster follow-ups. We also demonstrate how to optimally choose the clustering parameters.

DOI: 10.1103/PhysRevD.106.104063

I. INTRODUCTION

Continuous gravitational waves are long-lasting signals that may come from fast-spinning nonaxisymmetric neutron stars, unstable r-modes [1,2], the fast inspiral of dark-matter objects [3,4] or emission from clouds of axionlike particles around black holes [5,6]. Unlike the short-lived signals stemming from the mergers of compact binary objects [7–13], continuous gravitational wave have thus far eluded any detection, due to their strength being orders of magnitudes smaller than that of binary merger signals. The detection of continuous gravitational waves will open a new field of gravitational wave astronomy, may probe the fundamental nature of gravity [14,15] and unlock unprecedented information on neutron star interiors [16–18]. For these reasons researchers tirelessly search for continuous gravitational wave signals [19]. Broad surveys using months of data pose phenomenal challenges. We present here a new efficient method to identify the most promising candidates from broad parameter-space continuous waves surveys.

Independently of the emission mechanism, continuous gravitational waves are expected to be nearly monochromatic signals at the source, that due to the relative motion with respect to the source, appear to us on Earth to be frequency- and amplitude- modulated. Searches employ template parametrized by signal frequency, frequency derivatives, and source position, with $\sim 10^{17}$ template waveforms for observations lasting months. For template banks that are this big, typically only the top results are saved—say the $\sim 10^{10}$ results with the highest detection statistic values. Even though at this stage most of the results are not statistically significant, they are referred to as "candidates."

The candidates are followed up with a series of searches at increasing sensitivity. The signal-to-noise ratio of a signal increases from one stage to the next in a well-defined way, whereas noise does not, and this allows us to weed out noise candidates in the follow-ups [20–22]. Each follow-up search considers not only the candidates' parameters but a parameter-space region around each candidate. So if every candidate were to be followed up independently, the points in parameter space around nearby candidates

*benjamin.steltner@aei.mpg.de

would be searched more than once, resulting in a waste of computing resources and aggravating an already challenging problem. The core idea of clustering is to avoid this by identifying candidates likely due to the same root cause, bundling (*clustering*) them and considering them as a single entity in follow-up studies. Clustering is hence an important step in the postprocessing of the results because it organizes and reduces the $\sim 10^{10}$ candidates to a more useful and manageable set of $\approx$ independent $\sim 10^{6}$ candidates.

Each cluster is represented by the parameters of the so-called *seed* candidate and by a *containment region*. The latter measures how far from the seed associated with a signal, the true signal parameters are. In follow-up studies the entire containment region around each seed is surveyed. The containment region is the same for all seeds and it is determined statistically, such that it holds for a very large fraction ($> 99\%$) of signals, across the parameter space.

It has also been observed that a threshold on the minimum number of candidates in a cluster is effective at discarding noise-clusters. With a fixed computing budget for follow-ups, fewer candidates means that freed-up computational capacity can be used on additional, lower significance candidates which translates in deeper and more sensitive searches.

The most compute-intensive continuous waves searches have been carried out since the mid 2000s using idle cycles donated by the general public, through the volunteer distributed computing project Einstein@Home[1] [23–25]. The massive computational power that we can harvest today amounts to several Pflops, sustained 24×7, and enables us to investigate over 10^{19} waveforms.

For this reason clustering procedures have been in use for a long time: One of the first nontrivial clustering procedures is box-clustering [26,27], which dates back to nearly a decade ago. More recently a more flexible adaptive clustering technique has been used [28] which however does not converge fast enough when used on many data points. This is a significant drawback, as we want to set lower thresholds, which means considering more candidates in the follow-ups. Attempts to use machine-learning for clustering have been successful for directed searches, but not for all-sky searches [29,30].

We present here the new *density clustering* algorithm, able to process orders of magnitude more candidates than previous clustering strategies at comparable, if not lower, computing cost. We show how to choose the clustering parameters, and demonstrate its performance on real data. We concentrate on clustering results from very large template banks—with over 10^{16} points—and hence refer to the Einstein@Home results, but this method can also be employed in less challenging environments.

The paper is organized as follows: In Sec. II we describe the input data; in Sec. III the method itself; in Sec. IV the choice of the clustering parameters; in Sec. V the implementation; in Sec. VI the method is compared with adaptive clustering under realistic conditions, i.e., by applying it to the data of the Stage 0 results of the Einstein@Home all-sky search for continuous gravitational waves in Advanced LIGO data of the second observation run (O2) [22,31].

[1] www.einsteinathome.org/.

II. INPUT DATA TO CLUSTERING

Clustering works on a set of candidates, i.e., selected results from a search. A candidate is described by the values of the template that produced the detection statistic result, and the detection statistic result. For an all-sky search including up to second-order spin-down parameters, a generic candidate i is of the form

$$(f_i, \dot{f}_i, \ddot{f}_i, \alpha_i, \delta_i, \chi_i), \tag{1}$$

where f indicates the signal-template frequency, α, δ the source sky position and χ the value of the detection statistic used for the original candidate ranking.

We illustrate clustering for these 5 dimensions; fewer or more dimensions are treated analogously.

Since continuous waves are modulated by the Earth's rotation and orbit around the Sun, the sky grids are set up in sky coordinates projected on the ecliptic plane, x_{ecl}, y_{ecl}. Therefore for clustering we convert for the candidates $(\alpha_i, \delta_i) \to (x_{\mathrm{ecl}i}, y_{\mathrm{ecl}i})$—see Eqs. (14) and (15) in [28] for the conversion between $(\alpha, \delta) \to (x_{\mathrm{ecl}}, y_{\mathrm{ecl}})$.

The sky grids are approximately uniform hexagonal grids on the ecliptic plane and are defined by the hexagon edge length d:

$$d(m_{\mathrm{sky}}) = \frac{1}{f}\frac{\sqrt{m_{\mathrm{sky}}}}{\pi\tau_E}, \tag{2}$$

with $\tau_E \simeq 0.021$ s being half of the light travel-time across the Earth and m_{sky} a constant which controls the resolution of the sky grid [22]. From Eq. (2) it is clear that the sky-grid density increases with frequency f.

III. DENSITY CLUSTERING

We bin the parameter space in equally spaced cells of size

$$\delta b = (\delta f, \delta\dot{f}, \delta\ddot{f}, \delta x_{\mathrm{ecl}}, \delta y_{\mathrm{ecl}}) \tag{3}$$

in each dimension. The $\delta f, \delta\dot{f}, \delta\ddot{f}$ are each an integer multiple of the search grid spacing. The sky grid has a hexagonal tiling, so the square tiling of the bins above does not match it. The bins are usually chosen to be large enough that this does not matter and the square covering greatly simplifies the binning and the identification of neighboring bins. The bin size is always a multiple of the hexagon side, so the bins shrink with increasing frequency as the sky-grid pixels, keeping the average number of candidates per bin the same.

We only consider candidates with detection statistic values above a threshold Γ_L. In each bin j we count the number of candidates $N_{occ,j}$ with parameters in that bin. Bins with $N_{occ,j} \leq N_{occ,\min}$ are discarded. $N_{occ,\min}$ is one of the clustering parameters and its optimal value depends on the search setup and on the bin size.

Among the surviving bins, we cluster together nearby ones, to create a cluster. The basic notion of vicinity is controlled by two parameters: N^j and N_c. A bin b_a is a neighbor of bin b_c if the distances k^j in integer bin spacings

$$b_a - b_c = (k^1\delta f, k^2\delta\dot{f}, k^3\delta\ddot{f}, k^4\delta x_{\rm ecl}, k^5\delta y_{\rm ecl}) \quad (4)$$

satisfy the following conditions:

$$\begin{cases} k^j \leq N^j \text{ with } j = 1, \ldots, M \\ \sum_{j=1}^{M} k^j \leq N_c, \end{cases} \quad (5)$$

where M is the number of dimensions. The first condition sets the maximum distance in every dimension, whereas the second condition sets an overall maximum distance. With $M = 3$, $N_c = 1$ means that the two nearby bins have to share a face, $N_c = 2$ that they have to share an edge and $N_c = 3$ that they have to share a vertex. Default values are $N^j = 1$, equal for all j, and $N_c = M$.

Among the clusters from the previous step, we remove the ones with too few bins: $N_{\rm bins} \leq N_{\rm bins,min}$.

For each remaining cluster a representative candidate becomes the seed. The seed is by default the candidate with the highest detection statistic value (the loudest) of all candidates in the cluster. In noisier data it may make sense to look at the loudest candidate in the bin with the most candidates (densest bin) or the loudest candidate in the bin with the highest average over all detection statistic values of the candidates within that bin (loudest bin).

Finally all clusters with a seed with detection statistic value smaller than Γ_S are discarded. The process is illustrated in Fig. 1 for two dimensional, higher dimensions follow analogously.

An additional parameter can be used to mitigate binning effects: an overdensity of candidates may not be perfectly contained within one bin, but may extend across bin boundaries. For faint signals with just enough candidates to surpass the occupancy threshold $N_{occ,\min}$, this effect can make the difference between recovering a signal or not. Boundary effects can be partly mitigated by smoothing over bins, e.g., adding bin counts over neighboring bins or adding bin counts weighted with a Gaussian kernel. The overall impact of using smoothing procedures should be evaluated within the general framework of choosing the optimal clustering parameters, as described in the next section, but we will not explicitly consider it here.

IV. CHOOSING THE PARAMETERS OF THE CLUSTERING PROCEDURE

A number of parameters define the density clustering algorithm, and they are summarized in Table I. We choose the parameter values such that at fixed computational cost for the follow-up of the resulting seeds, the sensitivity of the clustering procedure is maximized. Below we describe how this optimization, yielding the values of the clustering parameters of Table I, is carried out.

The sensitivity of the clustering procedure is measured by the gravitational wave signal amplitude $h_0^{90\%}$ at which the detection efficiency ϵ of the clustering procedure is 90%, for signals with parameters in the search range. $h_0^{90\%}$ depends on the signal frequency like the amplitude spectral density of the noise $\sqrt{S_h(f)}$, so we maximize the quantity $\mathcal{D}^{90\%} = \sqrt{S_h(f)}/h_0^{90\%}(f)$, instead, that does not depend on frequency. $\mathcal{D}$ is also known as the sensitivity depth [27].

Since there is no way to predict the detection efficiency of the clustering procedure, we measure it with a Monte Carlo. We add fake signals from our target population to the real data, with amplitudes corresponding to a given value of $\mathcal{D}$. For each signal we perform the same search as the actual search, we cluster the results and produce seeds. If one of the seeds comes from the added signal, we consider the signal detected by the clustering procedure. The fraction of detected signals to total signals gives the detection efficiency at that sensitivity depth: $\epsilon(\mathcal{D})$. $\mathcal{D}^{90\%}$ is then

$$\epsilon(\mathcal{D}^{90\%}) = 90\%. \quad (6)$$

For each clustering setup we estimate

(i) $\mathcal{D}^{90\%}$

(ii) the containment region (see Sec. I).

TABLE I. Parameters of density clustering in the order that they are employed.

Parameter	Function
Input threshold $\Gamma_{\rm L}$	Discards candidates with detection statistic $\leq \Gamma_{\rm L}$. Filters input candidates
Bin sizes δb	Binning
Smoothing	Smooth histogram or not
Occupancy threshold $N_{occ,\min}$	Discard bins with $N_{occ} \leq N_{occ,\min}$ candidates
Neighbor criterion, N^j and N_c	Defines what a neighbor is
Cluster-size threshold $N_{\rm bins}$	Discard clusters with $N_{\rm bins} \leq N_{\rm bins,min}$ bins
Seed criterion	Loudest candidate in cluster, loudest in most-populated bin or in bin with highest average detection statistic
Output threshold $\Gamma_{\rm S}$	Discards cluster whose seed has detection statistic $\leq \Gamma_{\rm S}$. Reduces false alarms

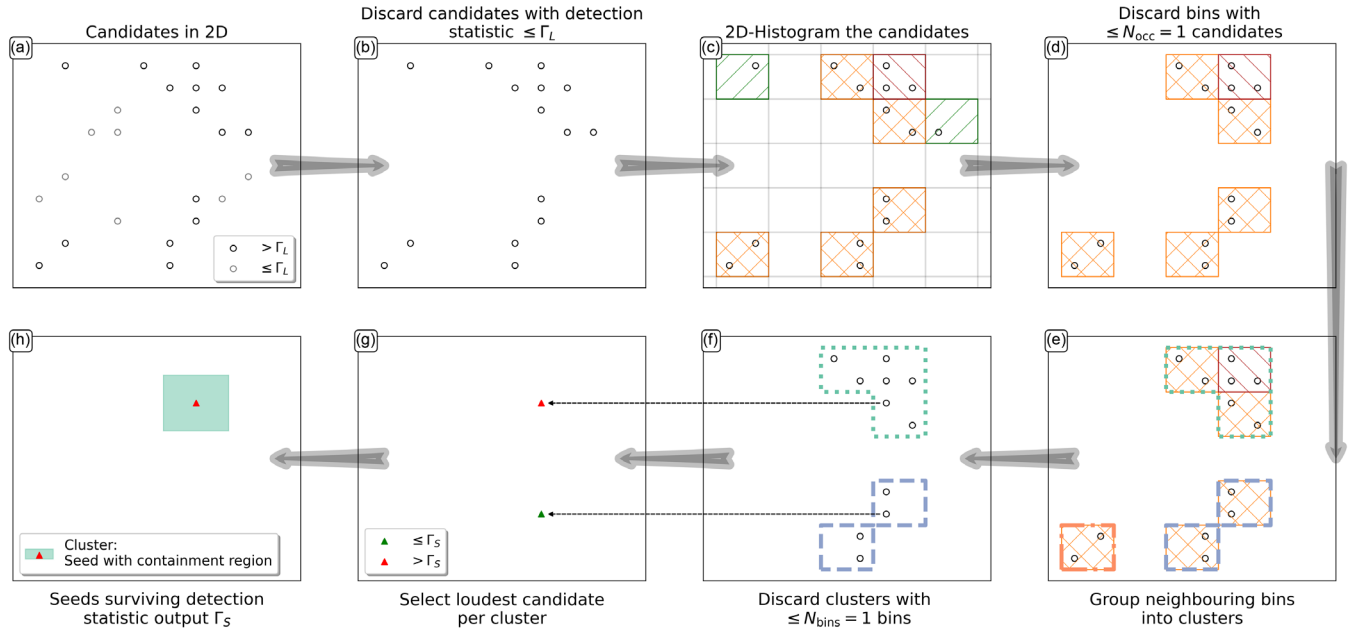

FIG. 1. Schematic illustration of the main steps of density clustering.

(iii) the false alarm rate. This is done by running the clustering on a subset on the search results, at different frequencies. Since we operate in the regime of very rare signals, we take this as a measure of the false alarm.

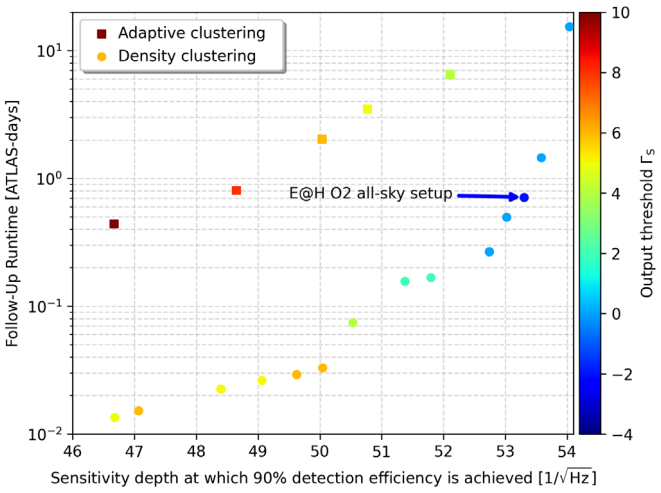

FIG. 2. Performance comparison between the previous clustering method, adaptive clustering, and density clustering. Each point represents a different clustering setup, used on the results of the Einstein@Home search [22]. To avoid excessive clutter we do not show all considered setups, but rather only those with runtime close to the smallest runtime at each $\mathcal{D}^{90\%}$. The color encodes the Γ_S threshold parameter value which illustrates the need to consider more candidates to achieve better sensitivities. The arrow indicates the density clustering setup chosen for the follow-up analysis reported in [22], which was the optimal for the clustering procedure at the time, under the constraint of maximum 1 ATLAS-day computing time for the first-stage follow-up.

For each clustering setup, from the number of expected seeds, the containment region and the timing model of our software [32], we estimate the computing cost of the first follow-up stage. This is illustrated in Fig. 2 for the results of the Stage-0 Einstein@Home search [22]. We can now identify the clustering setup that yields the highest $\mathcal{D}^{90\%}$, within the computing budget.

In principle one could optimize the follow-up search setup for each clustering setup. This would, however, be extremely expensive, and experience has shown that a setup choice guided by the sensitivity gain with respect to the previous stage, at accessible computing cost, lands a choice not significantly far from optimum. So we assume here that the follow-up setup is fixed.

V. IMPLEMENTATION

In the previous section we have described how the optimal combination of clustering parameters is identified. As we have seen, this requires a Monte Carlo in order to measure the false alarm and 90% detection-efficiency signal-amplitude $\mathcal{D}^{90\%}$, for every clustering setup.

For each setup we cluster $\gtrsim$2000 result files corresponding to data with different fake signals—this is to determine $\mathcal{D}^{90\%}$. We cluster $\gtrsim$500 search result files with no fake signals, in order to estimate the false alarm. These operations can be quite time-consuming, so we describe here how to reduce the computing cost of this step.

Einstein@Home search results typically come in files that cover a 50 mHz range of template frequencies, with size varying between a few MB to few GB, due to the different sky resolutions in the range 50–600 Hz. Each clustering instance uses as input one of these 50 mHz

results files. Since the time to cluster is ⋘ the time it takes to load such a file, it is faster to load a results file, keep it in memory, and test different clustering setups.

Further savings are obtained by reusing intermediate results:

(i) we compute a histogram for a choice of Γ_L and δb, and reuse it to produce the bin counts for different values of $N_{occ,\min}$
(ii) similarly, for a choice of Γ_L, δb and $N_{occ,\min}$ from the bin counts we produce different clusters for different values of $N_{\text{bins,min}}$
(iii) for each cluster different seeds are produced, based on different seed-selection criteria, e.g., the loudest cluster candidate, the loudest in the densest bin, the loudest in the bin with the highest average detection statistic (we call this the "loudest bin").
(iv) finally, each seed, and with it the whole cluster, may be discarded depending on the value of Γ_S.

With this scheme, testing a single clustering setup costs (on average, over many setups) just under a second, with more than half the time spent on the initial histogram and clustering. In order to reduce memory usage, the candidates are internally addressed only by an id. For thresholding on Γ_S and for computing the containment region, the actual seed parameters must be retrieved. This operation accounts for another 20% of the computing time. The remaining time is due to fluctuations in these estimates due to varying number of seeds and the initial overhead.

Given the computing-load profile described above, we parallelize the work among different independent processors, with each processor working only with a single results file and several $(\Gamma_L, \delta b)$-combinations. Say we have 2500 result files, 1000 $(\Gamma_L, \delta b)$-combinations and 1000 combinations of the remaining parameters, each processor analyzes 100 $(\Gamma_L, \delta b)$-combinations, exhausting all 1000 combinations of the remaining parameters. Hence, with 10 processes per result file, 25000 processes are spawned in total.

Using the large-capacity and fast-loading hdf5 and FITS file formats, and a HDD-raid configuration results file server, testing a single $(\Gamma_L, \delta b)$-combination and all 1000 combinations of the remaining parameters, takes ≈0.26 h. Thus one processor exhausting 100 $(\Gamma_L, \delta b)$-combinations takes ≈ a day. On the ATLAS cluster[2] using 25000 parallel processes the full testing of 1000×1000 setups is carried out in a day.

VI. PERFORMANCE ON E@H O2 ALL-SKY

We compare our density clustering with the adaptive clustering [28] on the results of the Stage-0 Einstein@Home O2 all-sky search [22].

[2]ATLAS is the super-computer cluster at the MPI for Gravitational Physics in Hannover: https://www.atlas.aei.uni-hannover.de/.

We characterize the detection efficiency on a set of ∼2900 fake signals from the target source population of the search: signals with spin-frequencies uniformly distributed; spin-downs log-uniform distributed and all other parameters distributed uniformly: orientation $\cos\iota \in [-1, 1)$, polarization angle $\psi \leq |\pi/4|$, sky position $0 \leq \alpha \leq 2\pi$ and $-1 \leq \sin\delta \leq 1$. The signal amplitude h_0 ranges from loud to faint signals with ∼1000 signals too faint to be detectable by either method.

The results of the procedure described in the previous section in order to identify the optimal density clustering parameters, are shown in Fig. 2. We compare with the results for the optimal parameter choice for adaptive clustering.

The density clustering setup chosen in [22] with a first-stage follow-up runtime-cost of ≤ 1 ATLAS-day is ≈10% more sensitive than the adaptive clustering setup at the same computing cost. In continuous gravitational wave searches a 10% improvement, solely due to a better search method, is a big gain.

Perhaps more immediately impressive is the fact that at fixed sensitivity, density clustering enables follow-ups that are a factor of ≳30 faster than previous methods.

This gain can be reinvested in deeper follow-ups by using a lower Γ_S, albeit the gain in practice is limited by the steep increase in computing cost for $\Gamma_S \lesssim 4$. With a threshold $\Gamma_S = -3.7$, density clustering is able to process two orders of magnitude more candidates than with a threshold $\Gamma_S = 4$, whereas adaptive clustering could not be used at all.

The performance of the adaptive clustering was characterized in [28] by the detection efficiency and the noise rejection NR defined as

$$\text{NR} := 1 - \frac{N_{\text{out}}}{N_{\text{in}}}, \tag{7}$$

where N_{in} is the number of candidates above the threshold Γ_S and N_{out} is the number of seeds produced by the clustering procedure.

With a threshold of $\Gamma_S \geq 4$ adaptive clustering and density clustering achieve similar performance with NR ≥ 99% and detection efficiencies above 98%. At lower thresholds, adaptive clustering does not converge in weeks of runtime, indicating that the method struggles to identify over densities due to faint signals. Density clustering, instead, can probe threshold values as low as −3.7, still achieving NR ≥ 99.99% and attaining a very respectable detection efficiency (now at the 85% level) on a set that includes very faint signals with detection statistic values $\in [-3.7, 4]$, which are much harder to find.

The density clustering setup chosen in [22] has a binning of 300 and 290 search bins in frequency and spindown respectively, and the square lattice on the projected ecliptic plane with edge lengths of $5 \cdot d(m_{\text{sky}} = 0.008)$. A cluster is

formed if there is at least one bin with more than three candidates. Of all candidates in the cluster the loudest becomes the seed.

VII. CONCLUSION AND OUTLOOK

We have presented a new, fast, and efficient clustering method—density clustering—for continuous gravitational wave search postprocessing.

Density clustering works by identifying overdensities of candidates in parameter space: clusters are purely build on candidates' closeness to each other and the detection statistic value is nearly irrelevant. This result may be somewhat surprising because the detection statistic ranks results based on the likeliness of they originating from a signal. However, some of our faintest—but still recoverable—signals show detection statistic values at which there are thousands to millions of louder candidates purely from noise. Our results show that in this regime overdensities are a better detection criterion than the significance given by the detection statistic value alone, even in Gaussian noise. This is probably due to the fact that the search is a semicoherent search.

The overdensities are uncovered by binning the parameter space and this is performed in one pass instead of the previously employed slower iterative procedures. The clustering step is thus largely independent of the number of input candidates, and this allows to process orders of magnitude more candidates with comparable computing resources, probing deeper into the noise.

Until now Einstein@Home searches have returned about $\mathcal{O}(10^4)$ candidates per work unit (e.g., [22]), which was more than adequate for what previous clustering algorithms could process. Density clustering can cluster orders of magnitude more candidates, which means that more results can be inspected, allowing to recover fainter signals in upcoming searches.

The previous clustering method, adaptive clustering, assumes compact overdensities, whereas signals typically present X- or Y-shaped overdensities which are hard to capture (and practically impossible to predict). Density clustering is agnostic about the shape of the overdensities and for this reason it is significantly more effective at identifying even very weak signals.

A different approach of using machine learning for clustering was developed and applied to the Einstein@Home O2 all-sky dataset in [29,30]. They cluster in $f, \dot{f}$ and achieve better sensitivity depths at fixed false alarms, but lack in sky localization to the point of clustering together candidates from "seemingly unrelated sky positions" [30]. This means that a follow-up would entail searching over the whole sky, whereas density clustering restricts the sky position to a patch of ∼9% to 0.01% of the full sky, depending on the frequency, between ∼20 Hz to 600 Hz, respectively. Even with the smaller uncertainties in $f, \dot{f}$ and only half the false alarms [30], the computational cost of their approach is higher by one order of magnitude compared to density clustering. They propose to generalize to include sky, and the results will be interesting to see.

Clustering is not a problem unique to gravitational wave astronomy, and a number of generic clustering methods exist. For example k-means [33] is a clustering method widely used in a variety of applications including signal-, image- and text-processing, health, cybersecurity, machine learning and big data [34]. It works based on minimizing the cluster-occupants' distance to the cluster center. Limitations of k-means are that the number of clusters must be known *a priori* and clusters are assumed to be roughly spherical and similar size. Density-based clustering applications exist: for example DBSCAN [35,36] and its many generalizations, like, e.g., OPTICS [37] or HDBSCAN [38], identify overdensities generated by a minimum number of points within a given volume. They are, however, not suitable for the large number of points in our results, and they are not as efficient as density clustering on our data.

A major advantage of our approach is the versatility of the method. Density clustering can cluster in any combination of dimensions, so it is easily extendable to, e.g., third/higher order spindowns $\ddot{f}, \ldots$ or to the 5 additional orbital parameters for searches for neutron stars in binary systems. In these searches signal-template offsets in orbital parameters can be to some extent compensated by offsets in frequency- and derivative(s). This translates into correlations between different templates and results in more candidates due to the same root cause [39], making clustering all the more important. All-sky binary searches are computationally extremely expensive and so are the follow-ups. A first test of density clustering on the results-data from [40] showed promising results within a few hours of clustering in 6 dimensions $(f, \alpha, \delta, \tau_{\mathrm{asc}}, P_b, a)$, showcasing the flexibility and ease of use of the method presented here.

ACKNOWLEDGMENTS

This work has utilized the ATLAS cluster computing at MPI for Gravitational Physics Hannover. We especially thank Carsten Aulbert for his help monitoring the work-load on the cluster for finding the optimal clustering-parameter combination. We use results from the Einstein@Home search [22] on LIGO data obtained for that search from the Gravitational Wave Open Science Center [41]. We thank again the LIGO-Virgo-KAGRA Collaboration for this service, and LIGO for producing that data.

[1] P. D. Lasky, Gravitational waves from neutron stars: A review, Pub. Astron. Soc. Aust. **32**, e034 (2015).

[2] B. J. Owen, L. Lindblom, C. Cutler, B. F. Schutz, A. Vecchio, and N. Andersson, Gravitational waves from hot young rapidly rotating neutron stars, Phys. Rev. D **58**, 084020 (1998).

[3] C. J. Horowitz and S. Reddy, Gravitational Waves from Compact Dark Objects in Neutron Stars, Phys. Rev. Lett. **122**, 071102 (2019).

[4] C. Horowitz, M. Papa, and S. Reddy, Search for compact dark matter objects in the solar system with LIGO data, Phys. Lett. B **800**, 135072 (2020).

[5] A. Arvanitaki, M. Baryakhtar, and X. Huang, Discovering the QCD axion with black holes and gravitational waves, Phys. Rev. D **91**, 084011 (2015).

[6] S. J. Zhu, M. Baryakhtar, M. A. Papa, D. Tsuna, N. Kawanaka, and H.-B. Eggenstein, Characterizing the continuous gravitational-wave signal from boson clouds around Galactic isolated black holes, Phys. Rev. D **102**, 063020 (2020).

[7] B. P. Abbott *et al.* (LIGO Scientific and Virgo Collaborations), GWTC-1: A Gravitational-Wave Transient Catalog of Compact Binary Mergers Observed by LIGO and Virgo during the First and Second Observing Runs, Phys. Rev. X **9**, 031040 (2019).

[8] R. Abbott *et al.* (LIGO Scientific and Virgo Collaborations), GWTC-2: Compact Binary Coalescences Observed by LIGO and Virgo During the First Half of the Third Observing Run, Phys. Rev. X **11**, 021053 (2021).

[9] R. Abbott *et al.* (LIGO Scientific, VIRGO, and KAGRA Collaborations), GWTC-3: Compact binary coalescences observed by LIGO and Virgo during the second part of the third observing run, arXiv:2111.03606.

[10] A. H. Nitz, C. Capano, A. B. Nielsen, S. Reyes, R. White, D. A. Brown, and B. Krishnan, 1-OGC: The first open gravitational-wave catalog of binary mergers from analysis of public Advanced LIGO data, Astrophys. J. **872**, 195 (2019).

[11] A. H. Nitz, T. Dent, G. S. Davies, S. Kumar, C. D. Capano, I. Harry, S. Mozzon, L. Nuttall, A. Lundgren, and M. Tápai, 2-OGC: Open gravitational-wave catalog of binary mergers from analysis of public Advanced LIGO and Virgo data, Astrophys. J. **891**, 123 (2020).

[12] A. H. Nitz, C. D. Capano, S. Kumar, Y.-F. Wang, S. Kastha, M. Schäfer, R. Dhurkunde, and M. Cabero, 3-OGC: Catalog of gravitational waves from compact-binary mergers, Astrophys. J. **922**, 76 (2021).

[13] A. H. Nitz, S. Kumar, Y.-F. Wang, S. Kastha, S. Wu, M. Schäfer, R. Dhurkunde, and C. D. Capano, 4-OGC: Catalog of gravitational waves from compact-binary mergers, arXiv:2112.06878.

[14] M. Isi, A. J. Weinstein, C. Mead, and M. Pitkin, Detecting beyond-Einstein polarizations of continuous gravitational waves, Phys. Rev. D **91**, 082002 (2015).

[15] M. Isi, M. Pitkin, and A. J. Weinstein, Probing dynamical gravity with the polarization of continuous gravitational waves, Phys. Rev. D **96**, 042001 (2017).

[16] F. Gittins and N. Andersson, Modelling neutron star mountains in relativity, Mon. Not. R. Astron. Soc. **507**, 116 (2021).

[17] F. Gittins, N. Andersson, and D. I. Jones, Modelling neutron star mountains, Mon. Not. R. Astron. Soc. **500**, 5570 (2020).

[18] J. A. Morales and C. J. Horowitz, Neutron star crust can support a large ellipticity, arXiv:2209.03222.

[19] K. Riles, Searches for continuous-wave gravitational radiation, arXiv:2206.06447.

[20] M. A. Papa *et al.*, Hierarchical follow-up of subthreshold candidates of an all-sky Einstein@Home search for continuous gravitational waves on LIGO sixth science run data, Phys. Rev. D **94**, 122006 (2016).

[21] B. P. Abbott *et al.* (LIGO Scientific and Virgo Collaborations), First low-frequency Einstein@Home all-sky search for continuous gravitational waves in Advanced LIGO data, Phys. Rev. D **96**, 122004 (2017).

[22] B. Steltner, M. A. Papa, H. B. Eggenstein, B. Allen, V. Dergachev, R. Prix, B. Machenschalk, S. Walsh, S. J. Zhu, and S. Kwang, Einstein@Home all-sky search for continuous gravitational waves in LIGO O2 public data, Astrophys. J. **909**, 79 (2021).

[23] BOINC, http://boinc.berkeley.edu/ (2020).

[24] D. P. Anderson, BOINC: A system for public-resource computing and storage, in *Proceedings of the Fifth IEEE/ACM International Workshop on Grid Computing (GRID04)* (IEEE Computer Society, Washington, DC, 2004), pp. 4–10.

[25] D. P. Anderson, C. Christensen, and B. Allen, Designing a runtime system for volunteer computing, in *Proceedings of the 2006 ACM/IEEE conference on Supercomputing* (Association for Computing Machinery, New York, NY, 2006), pp. 126–136.

[26] J. Aasi *et al.* (LIGO Scientific and VIRGO Collaborations), Directed search for continuous gravitational waves from the Galactic center, Phys. Rev. D **88**, 102002 (2013).

[27] B. Behnke, M. A. Papa, and R. Prix, Postprocessing methods used in the search for continuous gravitational-wave signals from the Galactic Center, Phys. Rev. D **91**, 064007 (2015).

[28] A. Singh, M. A. Papa, H.-B. Eggenstein, and S. Walsh, Adaptive clustering procedure for continuous gravitational wave searches, Phys. Rev. D **96**, 082003 (2017).

[29] B. Beheshtipour and M. A. Papa, Deep learning for clustering of continuous gravitational wave candidates, Phys. Rev. D **101**, 064009 (2020).

[30] B. Beheshtipour and M. A. Papa, Deep learning for clustering of continuous gravitational wave candidates II: Identification of low-SNR candidates, Phys. Rev. D **103**, 064027 (2021).

[31] M. Vallisneri, J. Kanner, R. Williams, A. Weinstein, and B. Stephens, The LIGO open science center, J. Phys. Conf. Ser. **610**, 012021 (2015).

[32] J. Ming, M. A. Papa, B. Krishnan, R. Prix, C. Beer, S. J. Zhu, H.-B. Eggenstein, O. Bock, and B. Machenschalk, Optimally setting up directed searches for continuous gravitational waves in Advanced LIGO O1 data, Phys. Rev. D **97**, 024051 (2018).

[33] S. Lloyd, Least squares quantization in PCM, IEEE Trans. Inf. Theory **28**, 129 (1982).

[34] M. Ahmed, R. Seraj, and S. M. S. Islam, The k-means algorithm: A comprehensive survey and performance evaluation, Electronics **9**, 1295 (2020).

[35] M. Ester, H.-P. Kriegel, J. Sander, and X. Xu, A density-based algorithm for discovering clusters in large spatial databases with noise, in *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96 (AAAI Press, Portland, USA, 1996), p. 226–231.

[36] E. Schubert, J. Sander, M. Ester, H. P. Kriegel, and X. Xu, Dbscan revisited, revisited: Why and how you should (still) use dbscan, ACM Transactions on Database Systems **42**, 1 (2017).

[37] M. Ankerst, M. M. Breunig, H.-P. Kriegel, and J. Sander, Optics: Ordering points to identify the clustering structure, SIGMOD Record **28**, 49 (1999).

[38] R. J. G. B. Campello, D. Moulavi, and J. Sander, Density-based clustering based on hierarchical density estimates, in *Advances in Knowledge Discovery and Data Mining*, edited by J. Pei, V. S. Tseng, L. Cao, H. Motoda, and G. Xu (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013), pp. 160–172.

[39] A. Singh, M. A. Papa, and V. Dergachev, Characterizing the sensitivity of isolated continuous gravitational wave searches to binary orbits, Phys. Rev. D **100**, 024058 (2019).

[40] P. B. Covas, M. A. Papa, R. Prix, and B. J. Owen, Constraints on r-modes and mountains on millisecond neutron stars in binary systems, Astrophys. J. Lett. **929**, L19 (2022).

[41] https://www.gw-openscience.org/