Mean shift cluster recognition method implementation in the nested sampling algorithm

M. Trassinelli, Pierre Ciccodicola

Mean shift cluster recognition method implementation in the nested sampling algorithm

M. Trassinelli[1,*] and P. Ciccodicola[1]

[1]*Institut des NanoSciences de Paris, INSP, CNRS,*

Sorbonne Université, F-75005 Paris, France

(Dated: January 28, 2020)

Abstract

Nested sampling is an efficient algorithm for the calculation of the Bayesian evidence and posterior parameter probability distributions. It is based on the step-by-step exploration of the parameter space by Monte Carlo sampling with a series of values sets called live points that evolve towards the region of interest, i.e. where the likelihood function is maximal. In presence of several local likelihood maxima, the algorithm converges with difficulty. Some systematic errors can also be introduced by unexplored parameter volume regions. In order to avoid this, different methods are proposed in the literature for an efficient search of new live points, even in presence of local maxima. Here we present a new solution based on the mean shift cluster recognition method implemented in a random walk search algorithm. The clustering recognition is integrated within the Bayesian analysis program NestedFit. It is tested with the analysis of some difficult cases. Compared to the analysis results without cluster recognition, the computation time is considerably reduced. At the same time, the entire parameter space is efficiently explored, which translates into a smaller uncertainty of the extracted value of the Bayesian evidence.

* martino.trassinelli@insp.jussieu.fr

I. INTRODUCTION

At present, Bayesian methods are routinely used in many fields: astrophysics and cosmology [1–8], particle physics [9], plasma physics [10, 11], machine learning [12] and many others [13, 14]. In the past few years, they were also applied to nuclear [15, 16] and atomic physics [17–21]. On one hand, one of the reasons for this success is related to the possibility of assigning a probability value to models (hypotheses) from the analysis of the same set of data in a very defined framework. In opposite to this, classical statistical tests and criterions (e.g. chi-square and likelihood ratio, Aikaike information criterion [22], etc.) are completely powerless if any defined preference does not emerge. On the other hand, the implementation of Bayesian methods is only now widely possible thanks to the recent relatively cheap cost of computation power. A large computing capability is in fact required for the fine exploration of the probability distribution of the model parameters. Unlike standard methods, which are mostly reduced to minimization/maximization problems (of the likelihood function or chi-squares), Bayesian approaches have to deal with non-trivial integrations in multi-dimensional space. One of the key points of Bayesian model selection is in fact the calculation of the *Bayesian evidence*, also called *marginal likelihood*, defined by

$$E(\mathcal{M}) \equiv P(Data|\mathcal{M}, I) = \int P(Data|\boldsymbol{a}, \mathcal{M}, I) P(\boldsymbol{a}|\mathcal{M}, I) d^J\boldsymbol{a} = \int L^{\mathcal{M}}(\boldsymbol{a}) P(\boldsymbol{a}|\mathcal{M}, I) d^J\boldsymbol{a}. \tag{1}$$

It consists in the integral of the likelihood function $L^{\mathcal{M}}(\boldsymbol{a}) = P(Data|\boldsymbol{a}, \mathcal{M}, I)$ in the J-dimensional parameter space (with J the number of the parameters) weighted by the prior probability $P(\boldsymbol{a}|\mathcal{M}, I)$ of the parameters $\boldsymbol{a}$ of a defined model $\mathcal{M}$ and where I represents the background available information. From the evidence, the probability of the model $P(\mathcal{M}|Data, I)$ is simply evaluated by the formula

$$P(\mathcal{M}|Data, I) \propto E(\mathcal{M}) P(\mathcal{M}|I), \tag{2}$$

where $P(\mathcal{M}|I)$ is the prior probability of the model itself. The challenging part resides in the multi-dimensional integration of Eq. (1). For this matter, different approaches have been developed in the past, some of them are Markov chain Monte Carlo (MCMC) based techniques (see e.g. [14, 23]) for the integration of $L^{\mathcal{M}}(\boldsymbol{a}) P(\boldsymbol{a}|\mathcal{M}, I)$. As an alternative, the *nested sampling* method has been proposed by Skilling in 2004 [24–26]. With this method, the multi-dimensional integral in Eq. (1) is reduced to a one-dimensional integral and calculated. Because of its high-efficiency

and relatively moderate calculation power requirement compared to other approaches, the nested sampling method is actually implemented in several data analysis codes such *Multinest* [3, 27], *Diamonds* [28], *Polycord* [29], *DNest4* [30] and *Dynesty* [31] for the computation of the Bayesian evidence and posterior probability distributions. Because of its efficient sampling, nested sampling is also routinely used to study thermodynamic partition functions [32–35] and to explore potential energy landscapes of atomistic systems [36–38].

When several maxima of the likelihood function are present, nested sampling algorithm can however encounter problems with converging correctly. The parameter space exploration can become inefficient or exclude entire regions, which introduces systematic errors in the estimation of the evidence. In order to avoid such a problem, several solutions are proposed in the literature. Here we present an original approach based on cluster recognition with the *mean shift method*, one of the classic clustering algorithm widely used and included in the major machine learning libraries. This method is implemented in the program *NestedFit*, a code developed by one of the authors and described in details in Refs. [39, 40].

An introduction to nested sampling and NestedFit code is presented in Sec. II. The description of the mean shift algorithm, its implementation on NestedFit and the results of some tests are presented in Sec. III. The article will end with a conclusive section (Sec. IV).

II. NESTED SAMPLING AND NESTEDFIT

A. The nested sampling algorithm

Nested sampling is based on the reduction of the multi-dimensional integral in Eq. (1) for the evidence computation into a one-dimensional integral

$$E(\mathcal{M}) = \int_0^1 \mathcal{L}(X) dX. \tag{3}$$

X represents the normalized value of the volume, weighted by the prior probability $P(\boldsymbol{a}|I)$, of the portion of J-dimensional space of parameters where $L(\boldsymbol{a})$ is higher than a certain value $\mathcal{L}$:

$$X(\mathcal{L}) = \int_{L(\boldsymbol{a})>\mathcal{L}} P(\boldsymbol{a}|I) d^J\boldsymbol{a}. \tag{4}$$

Equation (3) is numerically calculated using the rectangle integration method subdividing the $[0, 1]$ interval in $M + 1$ segments with an ensemble $\{X_m\}$ of M ordered points $0 < X_M < ... < X_2 <$

$X_1 < X_0 = 1$:

$$E(\mathcal{M}) \approx \sum_m \mathcal{L}_m \Delta X_m, \tag{5}$$

where $\mathcal{L}_m = \mathcal{L}(X_m)$ isgiven by the invertible relation in Eq. (4) and ΔX_m is simply given by $X_m - X_{m+1}$ or by the more accurate trapezoid rule $\Delta X_m = 1/2(X_{m-1} - X_{m+1})$. Each ΔX_m represents a slice of parameter space of nested hypervolumes defined by Eq. (4), giving the algorithm its name.

The evaluation of $\mathcal{L}_m$ is obtained by a recursive step-by-step exploration of the likelihood function by a Monte Carlo sampling. A collection of K parameter values $\{\boldsymbol{a}_k\}$, called *live points*, corresponds to K random points $\{\xi_{1,k}\}$ in $[0, 1]$ interval. When the live point $\tilde{\boldsymbol{a}}_1 = \boldsymbol{a}_{1,k'}$ corresponding to the highest value of $\{\xi_{1,k}\}, \xi_{1,k'} = \max\{\xi_{1,k}\}$ (with $\mathcal{L}_1 = \min\{\mathcal{L}(\xi_{1,k})\} = \mathcal{L}(\xi_{1,k'}) \equiv L(\tilde{\boldsymbol{a}}_1)$ from Eq. (4)) is discarded, the mean value of the interval occupied by the remaining ξ_k points shrinks to

$$X_m = \max_{k \neq k'}\{\xi_k\} \approx \left(\frac{K}{K+1}\right)^m \approx e^{-m/K} \tag{6}$$

with, at this first step, $m = 1$.

If a new live point $\boldsymbol{a}_{new}$ is found with the condition $L(\boldsymbol{a}_{new}) > \mathcal{L}_{m=1}$, a new set of $\xi_{m=2,k}$ points is constructed and the next procedure iteration step starts. For each step, the discarded values $\tilde{\boldsymbol{a}}_m = \boldsymbol{a}_{m,k'}$ are stored together with their corresponding likelihood values $\mathcal{L}_m = L(\tilde{\boldsymbol{a}}_m)$. The X_m are obtained by their average expectation value from Eq. (6). Step by step, the nested volumes built with the condition $L(\boldsymbol{a}) > \mathcal{L}_m$ converge around the parameter space regions corresponding to high values of the likelihood function. When the algorithm converges, the evidence is evaluated from the different values $\mathcal{L}_m, \Delta X_m$ using Eq. (5). From the set of collected values of the discarded live points $\tilde{\boldsymbol{a}}_m$ and the associated weights $w_m = \mathcal{L}_m \Delta X_m$, the posterior probability $P(\boldsymbol{a}|Data, \mathcal{M}, I)$ can be determined. More details on the nested sampling algorithm and its implementation can be found in Refs. [3, 24–27, 41, 42].

B. Bottleneck of nested sampling and proposed solutions

The difficulty of this elegant method is to efficiently find a new live point at each step within the hypervolume contour defined by $L(\boldsymbol{a}) > \mathcal{L}_m$. Codes that use the nested sampling method generally encounter difficulties to find new live points $\boldsymbol{a}_{new}$ when several maxima of the likelihood function are present. In this case, the exploration of the parameter space becomes generally

inefficient or can consider only one local maximum while introducing systematic errors in the estimation of the evidence. In order to avoid these problems, different strategies have been proposed in the literature. These strategies can be divided into two categories: with a cluster recognition algorithms and without cluster recognition, but with other improvements of the search algorithm for new live points.

A first attempt to improve the search of new live points for multimodal problems via MCMC has been proposed by Veitch and collaborators in 2010 [42]. Here 10% of the steps of the random walk are determined by a combination of three past points and not only the previous point of the Markov chain. In this way, a more efficient sampling is obtained without need of cluster recognition.

Another improved random walk method for nested sampling algorithm is the *diffusive nested sampling*, developed by Brewer et al. in 2011 [43] and implemented in *DNest4* [30] program. Here, the passage between maxima is facilitate by blurring the condition $L(\boldsymbol{a}_i) > \mathcal{L}_m$ for the parameter values explored by the MCMC, allowing to momentarily pass in regions with lower values of the likelihood function.

Alternatively to random walks, the use of single- or multi-particle trajectories have been implemented for improving the search of new points in complex landscapes of the function to maximize or minimize. This is the principle of Galilean and Hamiltonian Monte Carlo exploration [34, 44]. In the first case, linear trajectories and reflection from hard boundaries, given by the minimal likelihood threshold value, are considered. In the second case, more complex trajectories are computed from the motion determined by the Hamiltonian function, like in molecular dynamics, assimilated here to the likelihood function.

In the case of the presence of several maxima, these methods significantly improves the search of new points but do not allow to pass from one maximal region to another, which limits their efficiency. A completely different approach has been proposed by Martiniani and collaborators in 2014 [45]. To take into account the presence of several maxima without recurring to cluster recognition, they suggest global optimization techniques to use the knowledge of identified local maxima and their statistical weight and then to perform parallel nested sampling in correspondence of each significant region.

A first solution with the use of a cluster recognition algorithm has been implemented in *Multinest* code already in 2008 [3, 27]. Here, new live points are randomly selected in an ellipsoid that is defined by the covariance matrix of the present live points. Cluster analysis is used for partition-

ing of the parameter spaces in a series of ellipsoids. This is obtained by implementing the k-means clustering algorithm, which is triggered when the estimated volume occupied by the live points is much smaller than the ellipsoid volume estimated from their covariance matrix. A partition in two cluster is initially performed ($k = 2$) and recursively repeated (always with $k = 2$) to obtain an efficient partition of the space with many ellipsoids.

In the more recent *Polycord* program [29], where the search of new live points is based on the slice sampling[46], the cluster recognition is obtained by the k-nearest neighbor algorithm. Once the different cluster are identified, for each of them a parallel exploration and analysis via slice sampling MCMC is independently performed.

In the recent and very complete nested sampling code *Dynesty* [31], different sampling methods are proposed: from random uniform selections in ellipsoids, like Multinest, to a series of MCMC (random walks, slice sampling,...). Difficult cases with several likelihood maxima are treated by decomposing the parameter space in several ellipsoids via a cluster analysis (using k-means algorithm like Multinest), or spheres or cubes (with same radius/side, one per each live point) with no need of any cluster recognition technique.

In the following sections, we present a new alternative method based on a MCMC and where the mean shift algorithm is used for the identification of clusters. It is implemented in the existing nested sampling code *NestedFit*, which is briefly introduced as well in the next paragraph.

C. The NestedFit program

NestedFit is a general-purpose code for the evaluation of Bayesian evidence and parameter probability distributions based on nested sampling algorithm. It is written in Fortran90 with some subroutines in Fortran77, and parallelized via OPEN- MPI. It is mainly developed and implemented for the analysis in the fields of atomic, nuclear and solid-state physics [16, 39, 40, 47–50]. It is accompanied by a Python function library for visualization of the results and for automatization of series of analyses. In this publication we present the version 3.2 that has the cluster analysis of the live points as substantial upgrade with respect to older versions (see Ref. [39] for v. 0.7 and Ref. [40] for v. 2.2). In addition, in this new version some new improvements in the search of live point are also implemented. The source code is freely available in the repository `https://github.com/martinit18/nested_fit`.

The code requires two main input files: the main input file (`nf_input.dat`) where the anal-

ysis parameters are selected, and the data file, in the format (channel, counts) or (channel, y value, y uncertainty). Dependent on the data format, a Poisson or Gaussian statistics likelihood function is used. The function name in the input file indicates the model to be used for the calculation of the likelihood function. Several functions are already defined in the function library for different model of spectral lines. Additional functions can be easily defined by the user in a dedicated routine. Non-analytical or simulated profile models can be considered as well. In this case, one or more additional files have to be provided by the user for interpolation by B-splines using FITPACK routines [51].

Several data sets can be analyzed at the same time. This is particularly important for the correct study of physically correlated spectra at the same time, e.g., background and signal-plus-background spectra. This is implemented by using a global user-defined function composed by different models to describe each spectra but with common parameters between the models.

The main analysis results are summarized in one output file (`nf_output_res.dat`). Here the details of the computation, number of trials, number of iteration, can be found as well as the final evidence value and its uncertainty $E \pm \delta E$, the parameter values corresponding to the maximum of the likelihood function, but also the mean, the median, the standard deviation and the confidence intervals one, two and three sigma (68%, 95% and 99%) of the posterior probability distribution of each parameter. δE, or more precisely $\delta(\ln E)$ is evaluated by running the nested sampling several time for different sets of starting live points. $\delta(\ln E)$ is obtained by the standard deviation of the different values of $\ln E$, the natural estimation to study the uncertainty of E [52, 53]. The information gain $\mathcal{H}$ and the Bayesian complexity are also provided in the output. Data for plots and for further analyses are provided in separated files. The step-by-step information of the nested sampling exploration can be found in the largest output file that contains the live points used during the parameter space exploration $\tilde{\boldsymbol{a}}_m$, their associated likelihood values $\mathcal{L}_m$ and weight $w_m = \mathcal{L}_m \Delta X$. From this file, the different parameter probability distributions and joint probabilities can be build from the marginalization of the unretained parameters.

Details of the NestedFit search algorithm are presented in the next section. Additional information can be found in Refs. [39, 40].

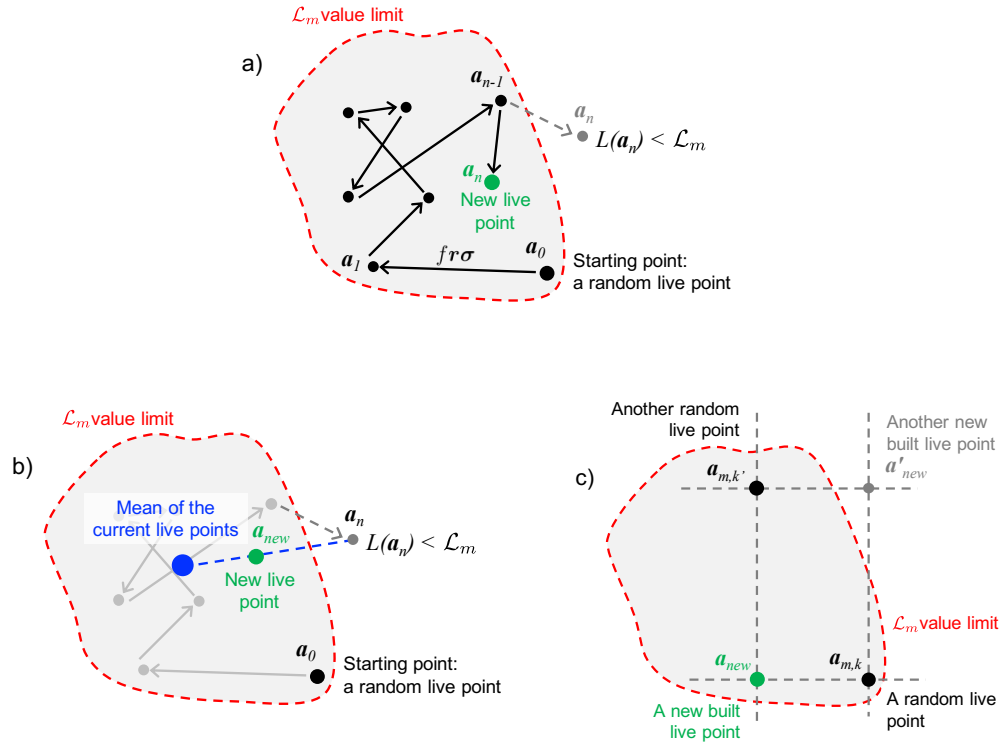

FIG. 1. Graphical presentation of the different search algorithms discussed in the text. a: Exploration of the parameter volume via the *lawn mower robot* for finding a new live point. b: Search of a new live point from the parameter set $\boldsymbol{a}_n$ outside the limit $L(\boldsymbol{a}) > \mathcal{L}_m$ and the barycenter of the current live points. c: Construction of the new live point from different coordinates of the current live points.

D. NenstedFit search algorithm

The search of new live points in NestedFit is based on a random walk called *lawn mower robot* [39, 40, 54], which is represented in Fig. 1a. It is composed by a sequence of N steps (or jumps, with N selected by the user) starting from a randomly chosen live point. Each step has an amplitude and direction given by the J-dimensional vector $f\boldsymbol{r\sigma}$ where each component $fr_j\sigma_j$ is determined by factor f, selected by the user, a random number r_j and the standard deviation of the current live points σ_j with respect to the jth parameter. For an efficient covering of the entire parameter space, f and N should be chosen with the criterion

$$f \times N \gtrsim 1 \tag{7}$$

to explore regions within a distance of the order of one standard deviation around the starting point. Each new step, which correspond to a new parameter set $\boldsymbol{a}_n$, is accepted if $L(\boldsymbol{a}_n) > \mathcal{L}_m$. If

$L(\boldsymbol{a}_n) < \mathcal{L}_m$, a new set of r_j is chosen. The number of total tries n_t is recorded. The choice of the values for f and N is very critical and it could vary from case to case. N has to be large enough to lose the memory of the starting live point position, but an increase of it produces a linear increase in computation time. A similar situation arises for f. If it is too small, a strong correlation between live points is artificially created. If it is too large, many failures can occur. From our experience, a reasonable range of values is $N = 20 - 40$ and $f = 0.1 - 0.2$. In any case we suggest a visual check of the explored live points for detecting possible correlations.

If the number of failures becomes too high ($n_t \gg N$), two different strategies are implemented for finding a new live point. In the first one, schematically represented in Fig. 1b, a new parameter set is determined by randomly choosing a point between the last failing chain point $\boldsymbol{a}_n$ with $L(\boldsymbol{a}_n) < \mathcal{L}_m$ and the barycenter of the current live points[55]. The second method, represented in Fig. 1c, consists on building a new synthetic live point $\boldsymbol{a}_{new}$ from the jth components from distinct live points: $(a_{new})_j = (a_{m,k})_j$ where k is randomly chosen between 1 and K (the total number of live points) for each j. If $\boldsymbol{a}_{new}$ $L(\boldsymbol{a}_{new}) > \mathcal{L}_m$, the new point is considered, otherwise another random live point is chosen as start of the random walk.

The two strategies are chosen randomly when $n_t = N_t$ (N_t chosen by the user in the configuration file) and n_t is reset to zero. As suggested by the schemes in Fig. 1, the first one favors a re-centering of the live points. In the opposite, the second can more easily explore peripheral regions. This second strategy was in fact the only present in the previous versions of NestedFit (where also N_t was a fixed parameter of the code), and it was developed to improve the search algorithm for multimodal cases facilitating jumps between maximal regions of the likelihood function [39, 40].

If the entire above procedure is repeated subsequently too many times (NN_t, selected by the user), the cluster analysis, described in the following sections, is triggered for improving the search of new live points.

III. MEAN SHIFT CLUSTERING ALGORITHM AND ITS IMPLEMENTATION

A. Preliminary tests and considerations on other cluster recognition algoritms

Before the implementation of one particular cluster recognition method in NestedFit, different algorithms from classical machine learning libraries (`https://scikit-learn.org` as

example) have been considered and some of them have been tested with simple Python scripts. For this purpose, we used different ensembles of live points issued from NestedFit runs on real data when convergence problems were encountered. We excluded a priori Density-Based Spatial Clustering of Applications with Noise (DBSCAN) method. This method is well adapted for detecting cluster with singular shapes (e.g. arc of circle) without necessary improving the implemented random walk algorithm that is based on the standard deviation of the recognized cluster. We then test the Gaussian mixture method with the determination of the number of clusters based on the expectation-maximization algorithm. The results were not convincing and requires external criterions for determining the number of clusters. For similar reasons, we excluded the k-means method that requires a preliminary choice of number of clusters and the x-means method that uses external criterions to determinate the best choice of k. We did not consider the recursive use of k-means with $k = 2$, like in the *Multinest* code, to keep a simple cluster recognition implementation. From these preliminary tests and considerations, the *mean shift* clustering algorithm [56, 57] emerges for its simplicity of implementation, its robustness and, more important, because it does not require a choice of the number of clusters before the analysis.

B. The mean shift algorithm for cluster recognition

Mean shift is a recursive algorithm based on the iterative calculation of the mean of points within a given region. Considering an ensemble $\{\boldsymbol{x}_i\}$, for each point the mean value m_i of the neighbor points $NH(\boldsymbol{x}_i)$ is calculated recursively via a kernel function $K(\boldsymbol{x}_i, \boldsymbol{x}_j)$ via

$$m_{s,i} = \frac{\sum_{\boldsymbol{x}_{s,j} \in NH(\boldsymbol{x}_{s,i})} K(\boldsymbol{x}_{s,i}, \boldsymbol{x}_{s,j}) \boldsymbol{x}_{s,j}}{\sum_{\boldsymbol{x}_{s,j} \in NH(\boldsymbol{x}_{s,i})} K(\boldsymbol{x}_{s,i}, \boldsymbol{x}_{s,j})}, \tag{8}$$

with $s = 1$ and $\boldsymbol{x}_{s=1,i} = \boldsymbol{x}_i$ for the first step. Then the procedure is repeated considering instead of the initial points $\boldsymbol{x}_i$, the mean values of the previous step, $\boldsymbol{x}_{s,i} = m_{s-1,i}$, until convergence or the maximum number of allowed steps is reached. Different points belonging to the same cluster are identified by the vicinity of the final $m_{s,i}$ values.

With the present implementation, via a Fortran module in NestedFit, the identification of the neighbor points NH is determined by the Euclidean distance $d(\boldsymbol{x}_i, \boldsymbol{x}_j) < D$, with D selected by the user. Two choices of K are available: a flat kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = 1$, and a Gaussian kernel $K(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp(-d(\boldsymbol{x}_i, \boldsymbol{x}_j)/\ell)$, with ℓ the bandwidth selected by the user. Before the implementation of the mean shift algorithm, the live points are normalized to their minima and maxima

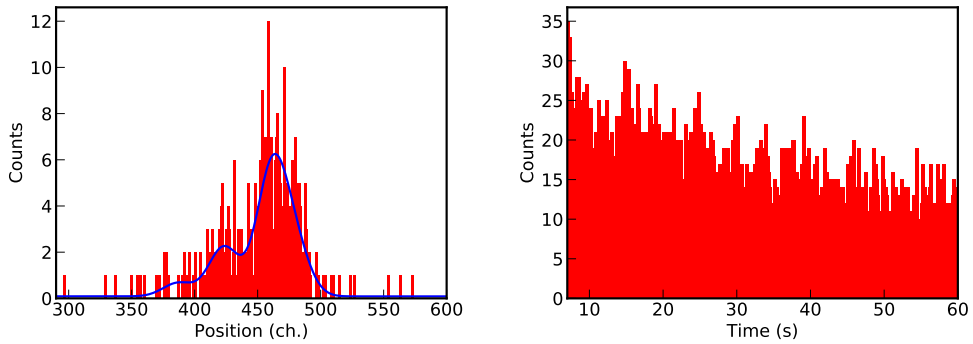

FIG. 2. Data corresponding to the high-resolution X-ray spectrum of the helium-like uranium $1s2p\ ^3P_2 \rightarrow 1s2s\ ^3S_1$ intrashell transition obtained by Bragg diffraction from a curved crystal [58] (left) and of the single decay of H-like $^{142}_{61}$Pm ions to the stable $^{142}_{60}$Nd bare nucleus via electron capture [16].

$(x_k)_j = [(a_{m,k})_j - \min\{(a_{m,k})_j\}]/[\max\{(a_{m,k})_j\} - \min\{(a_{m,k})_j\}]$ to have parameter D and ℓ dimensionless and in a fixed possible range $[0, 1]$.

At the end of the analysis, each live point has an additional flag indicating its belonging cluster that is used in the main NestedFit search algorithm.

C. Mean shift implementation in NestedFit

As written above, the cluster analysis is triggered when there are too many tries in the main search algorithm ($n_t = N_t \times NN_t$). Once the cluster analysis is performed, the algorithm restarts from a random live point but, instead of the standard deviation of whole ensemble of live points $\boldsymbol{\sigma}$, only the standard deviation of the belonging cth cluster $\boldsymbol{\sigma}_c^{cluster}$ is used for the random walk. Even if the cluster analysis is not perfect (e.g. too many or too few clusters are recognized), the generally smaller values of $\boldsymbol{\sigma}_c^{cluster}$ compared to $\boldsymbol{\sigma}$ significantly improves the efficiency of the nested sampling. When the algorithm becomes inefficient (n_t reach N_t), a new starting live point is chosen. When n_t is becoming too high again ($n_t = N_t \times NN_t$), a new cluster analysis is performed and the calculation continues until the end of the evidence calculation. Because of the random selection of the starting live point, small clusters have small probability to be chosen, and naturally disappear (or eventually grow) to the advantage (disadvantage) to clusters with higher (lower) likelihood values during the nested sampling.

To illustrate the cluster recognition at work in NestedFit, two practical examples are considered. In both cases, a Gaussian kernel has been used with a relatively large value of $D = 0.5 - 0.6$ in order to avoid having too many isolated clusters and $\ell = 0.1 - 0.2$, which ensures a good convergence of the algorithm. The cluster analysis is triggered after few failures, $NN_t = 2 - 3$, with a relatively low number of maximal tries n_t ($N_t = 100 - 200$) to change search strategy quite often when it becomes critical. With these criteria, the cluster analysis is triggered only about $2 - 10$ times for one entire nested sampling computation.

The first example consists of the analysis of a high-resolution X-ray spectrum corresponding to the helium-like $1s2p\ ^3P_2 \rightarrow 1s2s\ ^3S_1$ intrashell transition of uranium obtained by Bragg diffraction from a curved crystal [58]. For the analysis of the spectra, we assume the presence of four Gaussian peaks with the same width and a flat background. The second analysis is related to the measurement of the single decay of H-like $^{142}_{61}$Pm ions to the stable $^{142}_{60}$Nd bare nucleus via electron capture. Here, an exponential decay with a sinusoidal modulation is used as a model, considered parameters are the relative amplitude, pulsation and phase (see Ref. [16] for more details). Both data sets, presented in Fig. 2, are characterized by low statistic and the presence of many local maxima of the likelihood function, which makes them therefore difficult to analyze. In the first case, the possible permutations of the position of different peaks correspond to different maxima of the likelihood (4!=24 maxima for four peaks). In the second case, the multimodal behavior is caused by the different possible combinations of phase and pulsation values and corresponding beats.

To observe the evolution of the nested sampling algorithm with and without cluster analysis in the first case, we represent in Fig. 3 the evolution of one of the model parameters $(\tilde{\boldsymbol{a}}_m)_j$ relative to the position of one of the four Gaussian peaks as function of the step number m for ten different choices (tries) of starting live points. Different colors correspond to different values of the step weight $w_m = \mathcal{L}_m \Delta X_m$. Parameters with higher values or w_m have a higher influence on the final evidence and probability distributions $P(\boldsymbol{a}|Data, \mathcal{M}, I)$. When the cluster analysis is not implemented (Fig. 3 (top)), each try slowly converges to one likelihood maximum only, which corresponds to one of the four possible positions. The convergence in different maxima produces as consequence a spread of the values of the Bayesian evidence E.

In contrast, when the cluster analysis is turned on (Fig. 3 (bottom)), all four possible peak positions are considered at the same time and are equally explored for any try. The convergence improvement is directly observable in the smaller value of uncertainty of the evidence E. When

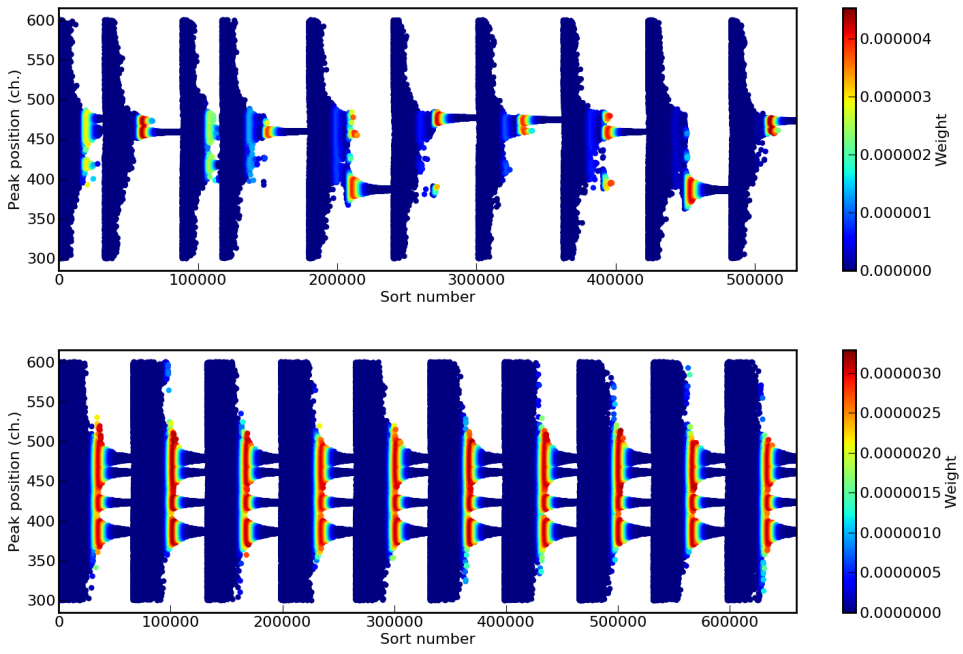

FIG. 3. Evolution of one of the components of the discarded live points $\tilde{\boldsymbol{a}}_m$ relative to the position of one of the four considered Gaussian peaks (see text) as function of the nested sampling step and for ten different choices starting live points. Results relative to the analysis without (top) and with the cluster analysis (bottom).

the cluster analysis is off, we have $\ln E = -320.52 \pm 1.71$ and $\ln E = -323.22 \pm 0.17$ when it is on. These results have been obtained with $f = 0.1$ for the analysis without clusters and $f = 0.2$ for the analysis with it, $N = 20$ and $K = 2000$ for both cases. The uncertainty of the previous values, and for all following evaluations, is obtained from the standard deviation of 16 different $\ln E$ values obtained running the analysis with 16 different sets of starting live points. The smaller value of f for the run without clusters has been chosen to reduce the computation time, which is still about 8 times longer than with the cluster analysis. It is interesting to note that, surprisingly, the two main values with and without cluster analysis are compatible (note: a difference of 0.9 in the $\ln E$ corresponds to about two sigmas [59]), without a systematic shift due to the exploration of a smaller parameter space. Only the associated uncertainty significantly changes.

To better visualize the cluster analysis process, a 3D presentation of the evolution of three components of $\tilde{\boldsymbol{a}}_m$, relative to the position of three peaks is presented in Fig. 4. Each image is obtained just after a cluster analysis, where different clusters are represented by different colors. To note, the analysis is triggered only a few times (four times for this selected example with

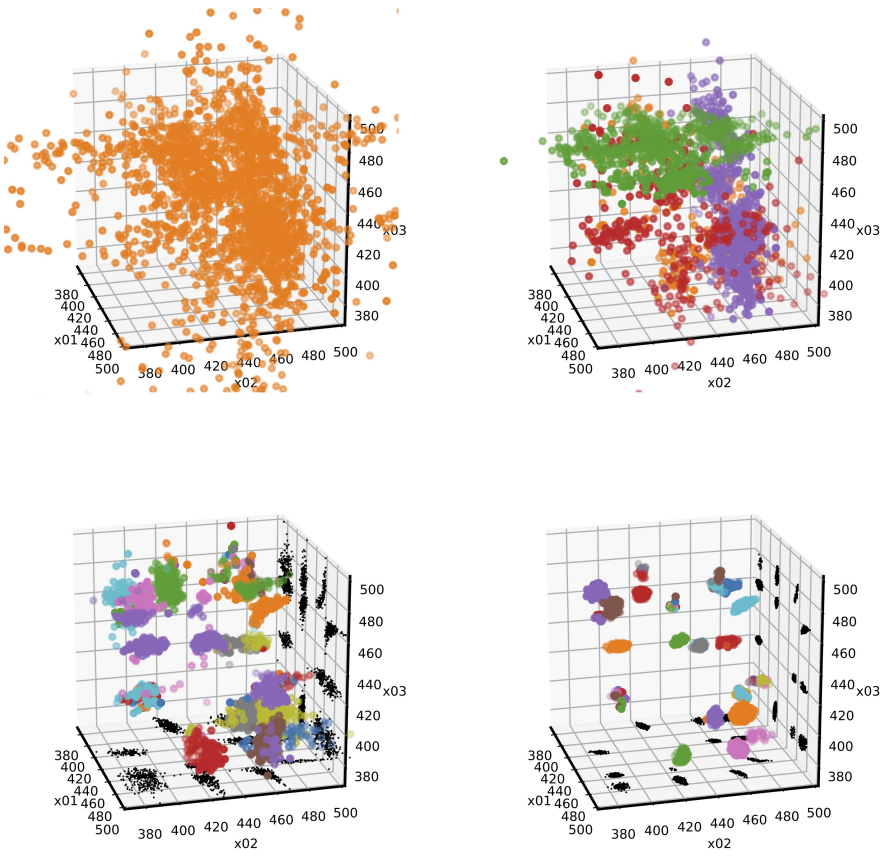

FIG. 4. Results of the cluster recognition corresponding to the analysis of four Gaussian peak. The position of three peaks are represented. Different colors represent different identified clusters. In black, the projection to some planes are represented. The 24 likelihood maxima (corresponding to the 4! permutation of the position of four peaks) are well visible.

$K = 2000, N_t = 200, NN_t = 2$ and with a Gaussian kernel with $D = 0.6$ and $\ell = 0.2$), showing the efficiency of the clustering recognition in the search of new live points (for about 60000 steps for each run). After the first run of the mean shift analysis, only a large cluster (and few isolated live points) are identified. In the following cluster analysis, all 24 different maxima likelihood regions are correctly identified.

The correct and simultaneous identification of all maxima translates to a more regular his-

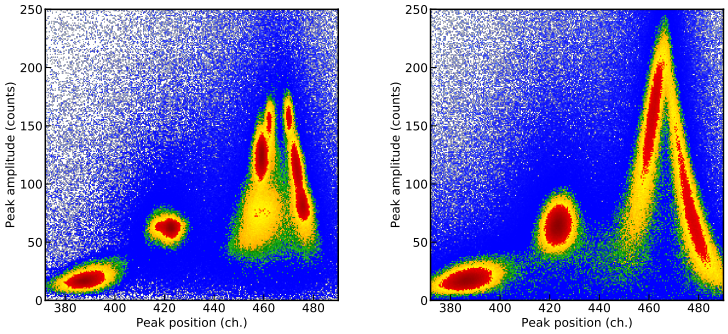

FIG. 5. Joint probability distribution of the position and amplitude of one of the four considered peaks obtained without (left) and with cluster analysis (right).

togram of probability distributions evaluated from the nested sampling outputs. This is shown in Fig. 5, where the 2D histogram relative to the joint probability of position and amplitude of one of the peaks is presented for the analysis with and without cluster recognition. When the cluster analysis is not implemented, the presence of very localized maxima of the probability distribution reflects the pathological behavior of the nested sampling convergence to only one of the likelihood maxima. On the contrary, a much smoother distribution of the joint probability is present when the cluster analysis is on.

A more quantitative measurement on the cluster analysis is obtained by varying the number of used live points K. As it can be observed in Fig. 6 (left, top), the final evidence does not change significantly with K. In opposite, the evaluated uncertainty (in blue) changes by several order of magnitudes and is systematically larger than its theoretical estimation (in black) $\delta(\log E) \approx \sqrt{\mathcal{H}/K}$ [25, 52], where $\mathcal{H}$ is the information gain. When the evaluated evidence uncertainty is plotted in logarithmic scale (Fig. 6 left, bottom), it can be observed that, for high values of K (≥ 500), $\delta(\log E)$ is proportional to $1/\sqrt{\mathcal{H}/K}$ as expected ($\delta(\log E) \propto K^c$ with $c = -0.52 \pm 0.02$), but is systematically higher by a factor of about 1.6 than the estimated accuracy (not shown in the bottom figure). When K is too low ($K < 500$ in the present case), even with the cluster analysis, the nested sampling algorithm cannot efficiently explore the 24 minima producing a systematic increases of $\delta(\log E)$.

As expected, the computation time (equivalent for one single CPU) per set of live points grows almost linearly with K. A simple fit gives an exponential dependency $\propto K^c$ with $c = 1.13 \pm 0.01$. A significant deviation is observed for $K = 10000$. In this case the cluster analysis, which number

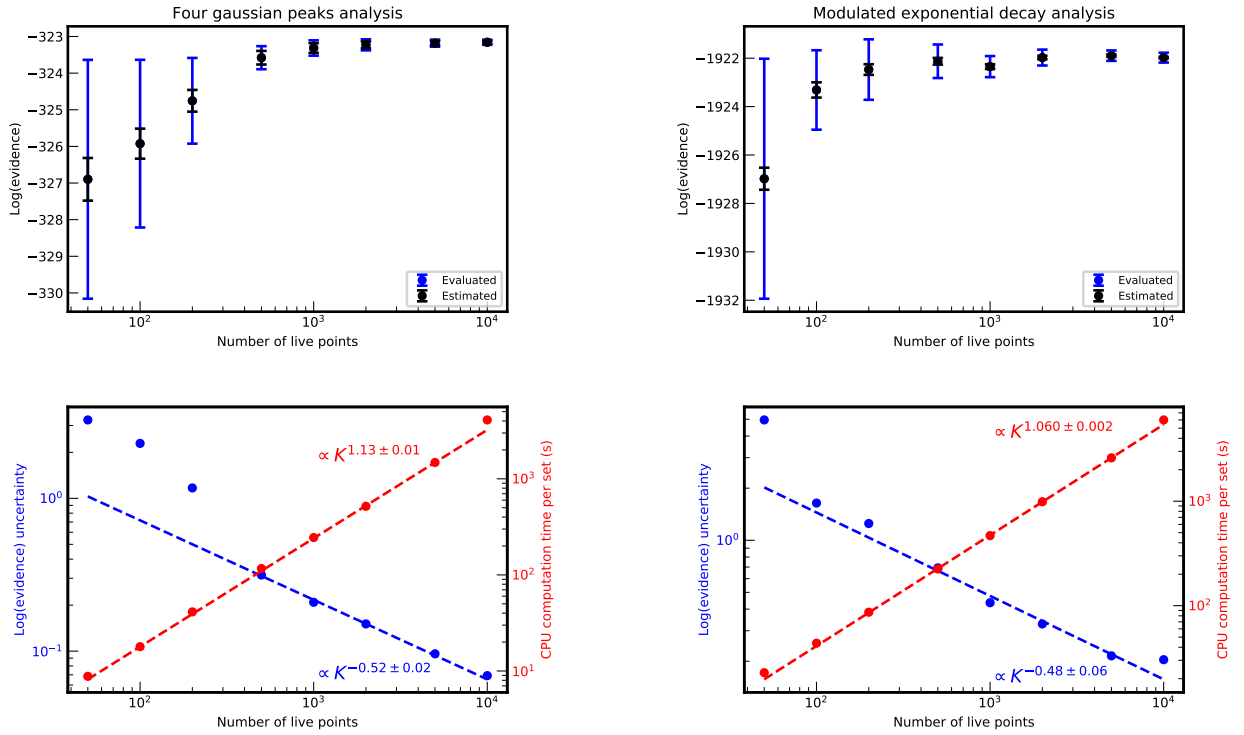

FIG. 6. Top: evaluation of the logarithm of the evidence for different number of live points K for the four gaussian peaks analysis (left) and the modulated exponential decay (right). In blue are indicated the uncertainty values evaluated by the results of 16 different run for each case. In black the theoretical uncertainty $\sqrt{\mathcal{H}/K}$ estimated from the information gain $\mathcal{H}$. Bottom: dependency of the evaluated uncertainty and CPU time with K. The dashed lines are the fits with power laws, which results are also shown. Data relative to $K < 500$ and $K > 5000$ are excluded for the fit of $\log E$ uncertainty and CPU time, respectively.

of operations is proportional to K^2, is significantly contributing to the total computation time.

Cluster analyses of above results have been obtained all with the same set of parameter: with a Gaussian kernel and with $D = 0.6, \ell = 0.2, N_t = 200$ and $NN_t = 2$. The exploration of the algorithm efficiency as dependence of these parameters is investigated and the corresponding results are resumed in Fig. 7, where the final evidence values and required computation time are presented for different parameter sets. Several cases are considered with flat and Gaussian kernel, indicated in the label by 'f' and 'g', respectively, and different values of D and ℓ, indicated in the label as well (only D for the flat kernel). As it can be noticed, for too small values of D and ℓ the final accuracy is poor. This is related to the identification of too many and too small clusters that finally induce an inaccurate, but fast, exploration of the parameter space. On the opposite, for too

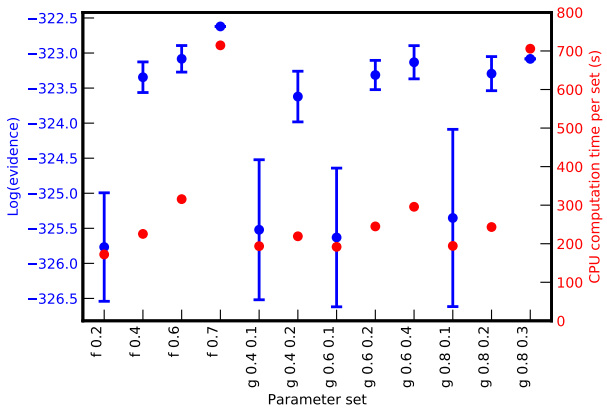

FIG. 7. Values of $\log E$ and CPU time for different choices of parameter values of the cluster recognition algorithm. Uncertainties of the evidence relative to the labels 'f 0.7' and 'g 0.8 0.3' are not evaluated because of the large computation time corresponding to these cases.

large values, one or very few clusters are identified. In these cases, the cluster algorithm is called very often without really improving the situation but increasing significantly the total computation time. Gaussian kernel results to be more robust and flexible than a flat kernel, probably due to the presence of the counter-reaction of the two parameters. The optimal parameter choice depends on the specific problem and the values of N_t and NN_t. It is generally observed that low values of N_t allow for changing starting live point often enough improve the efficiency of the algorithm. NN_t has to be adapted to trigger enough times the cluster analysis, but not too often.

The analysis of the other considered case is characterized by a completely different cluster evolution. In Fig. 8 we represent the amplitude, pulsation and phase of the modulation values after each cluster analysis. After the first run, several clusters are identified even if no clear structures are visible. In the following analysis, a very complex landscape is drawn, with many clusters and with very narrow values in omega. Even if characterized by very different sizes for the different parameters (even after their normalization), different clusters are well identified by the mean shift algorithm.

The complex dependency on the modulation pulsation ω is also presented in Fig. 9, where its evolution as function of the nested sampling step is represented for two different choices of starting live points. It can be observed that the rich landscape of the likelihood value as function of ω is well reproduced for each try, demonstrating again the efficiency of the cluster analysis

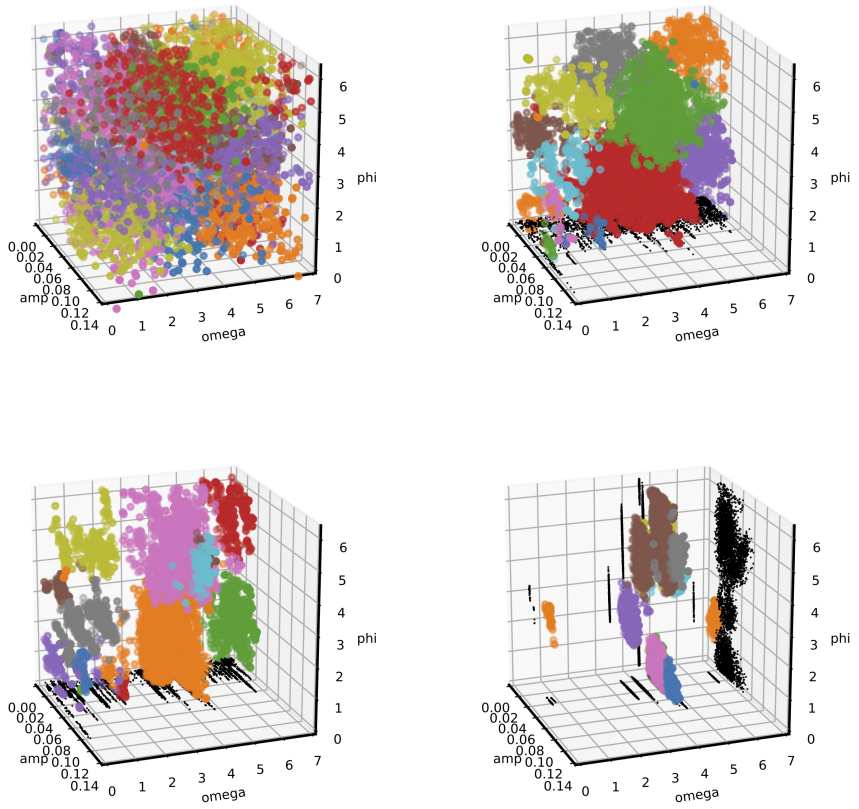

FIG. 8. Results of the cluster recognition corresponding to the analysis of the modulation of the exponential decay. The relative amplitude, pulsation and phase are represented. Different colors represent different identified clusters. In black, the projection to some planes are represented.

implementation once again.

As in the previous example, similar values of the Bayesian evidence are found: $\ln E = -1921.54 \pm 0.12$ without cluster analysis and -1922.04 ± 0.21 with. In contrast to the previous case, the uncertainty for the analysis without cluster analysis is very small. This is mainly caused by the choice of the value $f = 0.014$ (and $N = 40$, $K = 5000$), a very small value compared to the value set for the analysis with cluster analysis (with $f = 0.1$, $N = 20$, $K = 5000$). This small value of f contradicts in fact also the recommendation from Eq. (7) with the risk to

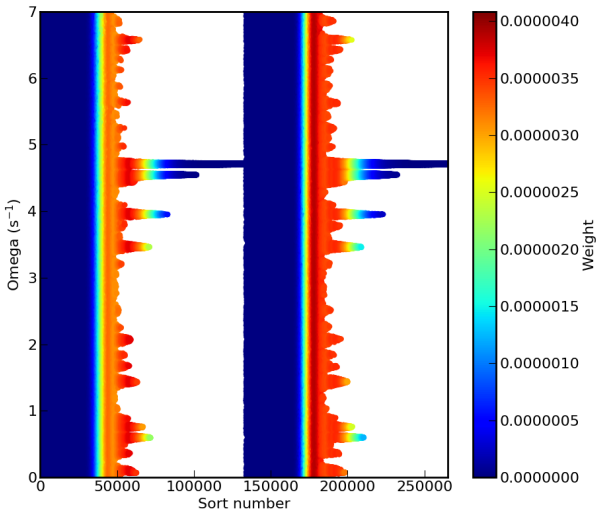

FIG. 9. Evolution of one of the components of the discarded live points $\tilde{\boldsymbol{a}}_m$ relative to the pulsation ω of the modulation of the single ion exponential decay (see text) as function of the nested sampling step with cluster analysis and for two different starting live points selections.

introduce some systematic errors in the computation. It was however required for insuring the convergence of the computation, which was otherwise impossible without cluster analysis. Like the previous example, the computation time without cluster analysis is in the best case about eight times longer than with the cluster analysis.

When the number of live points K is varied, keeping the other parameters fixed (Gaussian kernel with $D = 0.6, \ell = 0.2, N_t = 100$ and $NN_t = 3$), we can observe in Fig. 6 (right) a similar tendency for the results as in the previous case. The estimated evidence accuracy is found to be proportional, as expected, to $1/\sqrt{K}$ ($\delta(\log E \propto K^c$ with $c = -0.48 \pm 0.08$). Here too, $\delta(\ln E)$ is by a factor of 4.4–5.5 higher than the estimated accuracy. Because of the presence of less local minima than in the case of the four Gaussian peaks problem, the evaluated accuracy follows the proportionality to $1/\sqrt{K}$ down to $K = 100$. An almost linear dependency of the computation time on K is visible in this case too (CPU time $\propto K^c$ with $c = 1.13 \pm 0.01$), with a significant deviation for $K = 10000$ due to the high cluster analysis requirements for high K.

These two examples show the general behavior of the cluster algorithm and its dependency on the parameters choice. However, each case can be different and the user should vary the different parameters to reach the required accuracy. A general and simple suggestion is to use a large

number of live points to efficiently explore the whole parameter space. This is crucial when multiple local maxima of the likelihood function are present to avoid missing one of them. This is an important requirement even when a cluster analysis is available.

IV. CONCLUSIONS

We present a new application of cluster recognition to a nested sampling algorithm for the evaluation of the Bayesian evidence and posterior parameter probability distributions. For this matter, we selected the method of the mean shift, a robust and simple classical cluster recognition method widely used in the machine learning community. This clustering algorithm proved itself well adapted to critical data analysis when several likelihood maxima are present. It has been implemented in the program NestedFit and tested with two different benchmark cases, proving its efficiency in exploring the parameter space without excluding any region. As a consequence, the computation time is reduced by a factor at least eight. At the same time, a smaller value of the estimated evidence uncertainty is obtained. As a result from study the dependency on the different algorithm parameters, a sufficiently high number of live point should be always used, even when the cluster analysis is implemented, to efficiently explore all local likelihood maxima. Moreover for a good efficiency of the mean shift cluster recognition, its typical parametric distances (D and ℓ, the maximal neighbours distance and the bandwidth of the Gaussian kernel) should neither be too small or too large. In one case very low accuracy, but fast computation is obtained, in the other case the computation time increases too much.

In this article we explore only the implementation of the mean shift algorithm for cluster recognition. In the future, we plan to explore other methods like the k-nearest neighbours and the x-means method, successfully used in other nested sampling codes, and compare NestedFit performances with these codes in benchmark cases.

ACKNOWLEDGMENTS

M.T. thanks the Alexander von Humboldt Foundation that provide the financing to attend the MaxEnt2019 conference. M.T. would like to express once again his deep gratitude to Leopold M. Simons who introduced the author to the Bayesian data analysis and without whom this work could not have been started. We thank D. Schury and E.B. for the carful reading of the manuscript.

In addition, we would like to thank Lorenzo Paulatto for his help for the introduction of git and github for NestedFit future version development and sharing. The development of this program would not be possible without the close interactions and discussions with many collaborators that the author would like to thank as well R. Grisenti, A. Lévy, D. Gotta, Y. Litvinov, J. Machado, N. Paul and all members of the Pionic Hydrogen, FOCAL and GSI Oscillation collaborations and the ASUR group at INSP.

---

[1] A. Lewis and S. Bridle, Phys. Rev. D **66**, 103511 (2002).

[2] R. Trotta, Contemp. Phys. **49**, 71 (2008).

[3] F. Feroz, M. P. Hobson, and M. Bridges, Mon. Not. R. Astron. Soc. **398**, 1601 (2009).

[4] J. Xu, D. A. van Dyk, V. L. Kashyap, A. Siemiginowska, A. Connors, J. Drake, X.-L. Meng, P. Ratzlaff, and Y. Yu, Astrophys. J. **794**, 97 (2014).

[5] X. Yu, G. D. Zanna, D. C. Stenning, J. Cisewski-Kehe, V. L. Kashyap, N. Stein, D. A. van Dyk, H. P. Warren, and M. A. Weber, Astrophys. J. **866**, 146 (2018).

[6] M. N. Günther, F. J. Pozuelos, J. A. Dittmann, D. Dragomir, S. R. Kane, T. Daylan, A. D. Feinstein, C. X. Huang, T. D. Morton, A. Bonfanti, L. G. Bouma, J. Burt, K. A. Collins, J. J. Lissauer, E. Matthews, B. T. Montet, A. Vanderburg, S. Wang, J. G. Winters, G. R. Ricker, R. K. Vanderspek, D. W. Latham, S. Seager, J. N. Winn, J. M. Jenkins, J. D. Armstrong, K. Barkaoui, N. Batalha, J. L. Bean, D. A. Caldwell, D. R. Ciardi, K. I. Collins, I. Crossfield, M. Fausnaugh, G. Furesz, T. Gan, M. Gillon, N. Guerrero, K. Horne, S. B. Howell, M. Ireland, G. Isopi, E. Jehin, J. F. Kielkopf, S. Lepine, F. Mallia, R. A. Matson, G. Myers, E. Palle, S. N. Quinn, H. M. Relles, B. Rojas-Ayala, J. Schlieder, R. Sefako, A. Shporer, J. C. Suárez, T.-G. Tan, E. B. Ting, J. D. Twicken, and I. A. Waite, Nat. Astron. **3**, 1099 (2019).

[7] B. Abbott, R. Abbott, T. Abbott, F. Acernese, K. Ackley, C. Adams, T. Adams, P. Addesso, R. Adhikari, V. Adya, C. Affeldt, B. Agarwal, M. Agathos, K. Agatsuma, N. Aggarwal, O. Aguiar, L. Aiello, A. Ain, P. Ajith, B. Allen, G. Allen, A. Allocca, M. Aloy, P. Altin, A. Amato, A. Ananyeva, S. Anderson, W. Anderson, S. Angelova, S. Antier, S. Appert, K. Arai, M. Araya, J. Areeda, M. Arène, N. Arnaud, K. Arun, S. Ascenzi, G. Ashton, M. Ast, S. Aston, P. Astone, D. Atallah, F. Aubin, P. Aufmuth, C. Aulbert, K. AultOneal, C. Austin, A. Avila-Alvarez, S. Babak, P. Bacon, F. Badaracco, M. Bader, S. Bae, P. Baker, F. Baldaccini, G. Ballardin, S. Ballmer, S. Banagiri, J. Barayoga, S. Bar-

clay, B. Barish, D. Barker, K. Barkett, S. Barnum, F. Barone, B. Barr, L. Barsotti, M. Barsuglia, D. Barta, J. Bartlett, I. Bartos, R. Bassiri, A. Basti, J. Batch, M. Bawaj, J. Bayley, M. Bazzan, B. Bécsy, C. Beer, M. Bejger, I. Belahcene, A. Bell, D. Beniwal, M. Bensch, B. Berger, G. Bergmann, S. Bernuzzi, J. Bero, C. Berry, D. Bersanetti, A. Bertolini, J. Betzwieser, R. Bhandare, I. Bilenko, S. Bilgili, G. Billingsley, C. Billman, J. Birch, R. Birney, *et al.*, Phys. Rev. X **9**, 011001 (2019).

[8] B. Abbott, R. Abbott, T. Abbott, S. Abraham, F. Acernese, K. Ackley, C. Adams, R. Adhikari, V. Adya, and C. Affeldt, arXiv preprint arXiv:2001.01761 (2020).

[9] Particle Data Group, Phys. Rev. D **98**, 030001 (2018).

[10] A. Langenberg, J. Svensson, O. Marchuk, G. Fuchert, S. Bozhenkov, H. Damm, E. Pasch, A. Pavone, H. Thomsen, N. A. Pablant, R. Burhenn, and R. C. Wolf, Rev. Sci. Instrum. **90**, 063505 (2019).

[11] J. Milhone, K. Flanagan, M. D. Nornberg, M. Tabbutt, and C. B. Forest, Rev. Sci. Instrum. **90**, 063502 (2019).

[12] D. Barber, *Bayesian Reasoning and Machine Learning* (Cambridge University Press, 2012).

[13] U. von Toussaint, Rev. Mod. Phys. **83**, 943 (2011).

[14] W. von der Linden, V. Dose, and U. von Toussaint, *Bayesian Probability Theory: Applications in the Physical Sciences* (Cambridge University Press, 2014).

[15] G. King, A. Lovell, L. Neufcourt, and F. Nunes, Phys. Rev. Lett. **122**, 232502 (2019).

[16] F. C. Ozturk, B. Akkus, D. Atanasov, H. Beyer, F. Bosch, D. Boutin, C. Brandau, P. Bühler, R. B. Cakirli, R. J. Chen, W. D. Chen, X. C. Chen, I. Dillmann, C. Dimopoulou, W. Enders, H. G. Essel, T. Faestermann, O. Forstner, B. S. Gao, H. Geissel, R. Gernhäuser, R. E. Grisenti, A. Gumberidze, S. Hagmann, T. Heftrich, M. Heil, M. O. Herdrich, P. M. Hillenbrand, T. Izumikawa, P. Kienle, C. Klaushofer, C. Kleffner, C. Kozhuharov, R. K. Knöbel, O. Kovalenko, S. Kreim, T. Kühl, C. Lederer-Woods, M. Lestinsky, S. A. Litvinov, Y. A. Litvinov, Z. Liu, X. W. Ma, L. Maier, B. Mei, H. Miura, I. Mukha, A. Najafi, D. Nagae, T. Nishimura, C. Nociforo, F. Nolden, T. Ohtsubo, Y. Oktem, S. Omika, A. Ozawa, N. Petridis, J. Piotrowski, R. Reifarth, J. Rossbach, R. Sánchez, M. S. Sanjari, C. Scheidenberger, R. S. Sidhu, H. Simon, U. Spillmann, M. Steck, T. Stöhlker, B. H. Sun, L. A. Susam, F. Suzaki, T. Suzuki, S. Y. Torilov, C. Trageser, M. Trassinelli, S. Trotsenko, X. L. Tu, P. M. Walker, M. Wang, G. Weber, H. Weick, N. Winckler, D. F. A. Winters, P. J. Woods, T. Yamaguchi, X. D. Xu, X. L. Yan, J. C. Yang, Y. J. Yuan, Y. H. Zhang, and X. H. Zhou, Phys. Lett. B **797**, 134800 (2019).

[17] J. K. Stockton, X. Wu, and M. A. Kasevich, Phys. Rev. A **76**, 033613 (2007).

[18] D. Calonico, F. Levi, L. Lorini, and G. Mana, Metrologia **46**, 267 (2009).

[19] A. Mooser, H. Kracke, K. Blaum, S. A. Bräuninger, K. Franke, C. Leiteritz, W. Quint, C. C. Rodegheri, S. Ulmer, and J. Walz, Phys. Rev. Lett. **110**, 140405 (2013).

[20] D. S. Covita, D. F. Anagnostopoulos, H. Fuhrmann, H. Gorke, D. Gotta, A. Gruber, A. Hirtl, T. Ishiwatari, P. Indelicato, T. S. Jensen, E.-O. L. Bigot, V. E. Markushin, M. Nekipelov, V. N. Pomerantsev, V. P. Popov, J. M. F. dos Santos, P. Schmid, L. M. Simons, M. Theisen, M. Trassinelli, J. F. C. A. Veloso, and J. Zmeskal, Eur. Phys. J. D **72**, 72 (2018).

[21] P. Heim, M. Rumetshofer, S. Ranftl, B. Thaler, W. E. Ernst, M. Koch, and W. von der Linden, Entropy **21**, 93 (2019).

[22] H. Akaike, IEEE Trans. Automat. Contr. **19**, 716 (1974).

[23] A. Lawrence, *Probability in Physics* (Springer, 2019).

[24] J. Skilling, AIP Conf. Proc. **735**, 395 (2004).

[25] J. Skilling, Bayesian Anal. **1**, 833 (2006).

[26] D. S. Sivia and J. Skilling, *Data analysis: a Bayesian tutorial*, 2nd ed. (Oxford University Press, 2006).

[27] F. Feroz and M. P. Hobson, Mon. Not. R. Astron. Soc. **384**, 449 (2008).

[28] E. Corsaro and J. D. Ridder, Astron. Astrophys. **571**, A71 (2014).

[29] W. J. Handley, M. P. Hobson, and A. N. Lasenby, Mon. Not. R. Astron. Soc. **453**, 4384 (2015).

[30] B. J. Brewer and D. Foreman-Mackey, J. Stat. Softw. **86**, 33 (2018).

[31] J. S. Speagle, arXiv preprint arXiv:1904.02180 (2019).

[32] I. Murray, D. J. C. MacKay, Z. Ghahramani, and J. Skilling, *Nested sampling for Potts models*, Proceedings of the 18th International Conference on Neural Information Processing Systems (MIT Press, Vancouver, British Columbia, Canada, 2005) p. 947–954.

[33] S. O. Nielsen, J. Chem. Phys. **139**, 124104 (2013).

[34] R. J. N. Baldock, N. Bernstein, K. M. Salerno, L. B. Pártay, and G. Csányi, Phys. Rev. E **96**, 043311 (2017).

[35] P. G. Bolhuis and G. Csányi, Phys. Rev. Lett. **120**, 250601 (2018).

[36] L. B. Pártay, A. P. Bartók, and G. Csányi, J. Phys. Chem. B **114**, 10502 (2010).

[37] N. Burkoff, C. Várnai, S. Wells, and D. Wild, Biophys. J. **102**, 878 (2012).

[38] L. B. Pártay, A. P. Bartók, and G. Csányi, Phys. Rev. E **89**, 022302 (2014).

[39] M. Trassinelli, Nucl. Instrum. Methods B **408**, 301 (2017).

[40] M. Trassinelli, Proceedings **33**, 14 (2019).

[41] P. Mukherjee, D. Parkinson, and A. R. Liddle, Astrophys. J. Lett. **638**, L51 (2006).

[42] J. Veitch and A. Vecchio, Phys. Rev. D **81**, 062003 (2010).

[43] B. J. Brewer, L. B. Pártay, and G. Csányi, Stat. Comput. **21**, 649 (2011).

[44] J. Skilling, Proceedings **33**, 19 (2019).

[45] S. Martiniani, J. D. Stevenson, D. J. Wales, and D. Frenkel, Phys. Rev. X **4**, 031034 (2014).

[46] A MCMC that uses the live point covariant matrix to provide a probability distribution for the choice of the random walk direction.

[47] M. Trassinelli, D. F. Anagnostopoulos, G. Borchert, A. Dax, J. P. Egger, D. Gotta, M. Hennebach, P. Indelicato, Y. W. Liu, B. Manil, N. Nelms, L. M. Simons, and A. Wells, Phys. Lett. B **759**, 583 (2016).

[48] M. Trassinelli, D. Anagnostopoulos, G. Borchert, A. Dax, J.-P. Egger, D. Gotta, M. Hennebach, P. Indelicato, Y.-W. Liu, B. Manil, N. Nelms, L. Simons, and A. Wells, EPJ web conf. **130**, 01022 (2016).

[49] I. Papagiannouli, M. Patanen, V. Blanchet, J. D. Bozek, M. de Anda Villa, M. Huttula, E. Kokkonen, E. Lamour, E. Mevel, E. Pelimanni, A. Scalabre, M. Trassinelli, D. M. Bassani, A. Lévy, and J. Gaudin, J. Phys. Chem. A **122**, 14889 (2018).

[50] M. D. A. Villa, J. Gaudin, D. Amans, F. Boudjada, J. Bozek, R. E. Grisenti, E. Lamour, G. Laurens, S. Macé, C. Nicolas, I. Papagiannouli, M. Patanen, C. Prigent, E. Robert, S. Steydli, M. Trassinelli, D. Vernhet, and A. Lévy, Langmuir **35**, 11859 (2019).

[51] P. Dierckx, *Curve and surface fitting with splines* (Oxford University Press, 1995).

[52] J. Skilling, AIP Conf. Proc. **1193**, 277 (2009).

[53] N. Chopin and C. P. Robert, Biometrika **97**, 741 (2010).

[54] M. Theisen, *Analyse der Linienform von Röntgenübergängen nach der Bayesmethode*, Master thesis, Faculty of Mathematics, Computer Science and Natural Sciences, RWTH Aachen University (2013).

[55] As for the *lawn mower robot* method, also this algorithm is due to L. Simons [54] but it was not implemented in the past versions of NestedFit.

[56] K. Fukunaga and L. Hostetler, IEEE Trans. Inf. Theory **21**, 32 (1975).

[57] Y. Cheng, IEEE T Pattern. Anal. **17**, 790 (1995).

[58] M. Trassinelli, A. Kumar, H. F. Beyer, P. Indelicato, R. Märtin, R. Reuschl, Y. S. Kozhedub, C. Brandau, H. Bräuning, S. Geyer, A. Gumberidze, S. Hess, P. Jagodzinski, C. Kozhuharov, D. Liesen, U. Spillmann, S. Trotsenko, G. Weber, D. F. A. Winters, and T. Stöhlker, Eur. Phys. Lett. **87**, 63001

(2009).

[59] C. Gordon and R. Trotta, Mon. Not. R. Astron. Soc. **382**, 1859 (2007).