Monthly Notices of the ROYAL ASTRONOMICAL SOCIETY

MNRAS **482,** 2878–2892 (2019) doi:10.1093/mnras/sty2788

HDMClouds: a hierarchical decomposition of molecular clouds based on Gaussian mixtures

Martín Villanueva [1★] Mauricio Araya,[1,2★] Claudio E. Torres[1,2★] and Pía Amigo[3]

[1]*Departamento de Informática, Universidad Técnica Federico Santa María, Casilla 110-V, Valparaíso, Chile*
[2]*Centro Científico Tecnológico de Valparaíso, Universidad Técnica Federico Santa María, Valparaíso, Chile*
[3]*Instituto de Física, Pontificia Universidad Católica de Valparaíso, Casilla 4059, Valparaíso, Chile*

Accepted 2018 October 4. Received 2018 September 4; in original form 2017 September 21

ABSTRACT

The identification and characterization of independent entities within molecular clouds is a key challenge for astronomical data analysis. The ever-increasing volume, resolution and sensitivity of observations demand automatic routines to identify and deblend candidate entities to be analysed. Additionally, the intrinsically hierarchical nature of molecular gas distributions demands an automatic identification of the nesting relations between these entities. We propose a novel approach for decomposing molecular clouds in two steps: first we fit the data to a Gaussian mixture with many components, then reconstruct the cloud using a hierarchical model using a Gaussian-mixture reduction algorithm. We use a continuous-space representation, because it is well suited for disentangling coupled entities of emission compared with pixel-based ones, and build a tree structure to represent the hierarchical connections between mixture components. This allows us to select different groups of components in the tree without additional computational effort, including overlapping substructures. We assess our proposal quantitatively and qualitatively using data from the Atacama Large Millimeter Array (ALMA) science verification archive, as well as synthetic data. We also compare the results from some state-of-the-art clump identification algorithms. The experiments and comparisons show that our approach is an effective way to inspect and represent the hierarchical structure of molecular clouds.

Key words: methods: numerical – techniques: image processing – techniques: spectroscopic – ISM: structure.

1 INTRODUCTION

Understanding how molecular clouds are structured is a key issue in understanding their evolution and the formation of stars (Shadmehri & Elmegreen 2011). The identification of coherent regions within molecular clouds has been used for studying the relationship of the interstellar medium (ISM) with the stellar initial mass function (IMF) (Williams, Blitz & McKee 2000; Shadmehri & Elmegreen 2011). The structure of molecular clouds reflects the conditions under which they form, acts as an indicator of their evolution and (at sufficiently high densities) is related to the mass scale of stars and the slope of the IMF (Blitz & Williams 1999). Nevertheless, fundamental aspects of molecular clouds, like lifetimes, formation mechanisms and star formation rates, still remain unclear (Shu, Adams & Lizano 1987; Blitz & Williams 1999; Pringle, Allen & Lubow 2001; Heyer & Dame 2015). What has become clearer, especially with the increase of angular resolution and sensitivity, is the fact that the structure of molecular clouds is clumpy and hierarchical on different scales (Rosolowsky et al. 2008; Leroy et al. 2013; Heyer & Dame 2015): small-scale compact clumps (Williams, De Geus & Blitz 1994) are always contained in larger-scale and less compact gas envelopes. Additionally, molecular clouds and clumps at low intensities may be blended together (Blitz & Williams 1999).

Throughout this article, *cloud entities* will refer to the coherent regions inside the image or data cube observations of molecular clouds, which might be considered as independent *molecular gas clusters* in the ISM. According to Blitz & Williams (1999)'s seminal work, the objects in the hierarchical cloud structure can be categorized into clouds, clumps and cores, going from large to small scale respectively. In the terminology of this work, a *cloud entity* may range from a diffuse molecular cloud to a self-gravitating compact clump with its molecular gas envelope.

Characterizing these cloud entities becomes a challenging problem when the volume, resolution and sensitivity of observations increase. This is mainly due to the appearance of blended emission and multi-scale hierarchical dependences in the molecular cloud.

★ E-mail: martin.villanueva@usm.cl (MV); maray@inf.utfsm.cl (MA); ctorres@inf.utfsm.cl (CET)

Most clump identification algorithms consist of decomposing the data into a set of discrete clumps that might not overlap. This leads directly to an underestimation of flux and mass function, since the blended part of the clumps is not taken into account. An exception is Stutzki & Guesten (1990)'s GAUSSCLUMPS, which does handle blended emission. However, this algorithm assumes an ellipsoidal Gaussian shape of clumps, which is not a realistic assumption, as molecular clouds and clumps are known to be fairly irregularly shaped (Myers et al. 1991). An even more critical problem is that most of the clump identification algorithms (Stutzki & Guesten 1990; Williams et al. 1994; Bertin & Arnouts 1996; Berry 2015) do not take into account the hierarchical structure of molecular clouds. These algorithms perform a segmentation of the data into discrete groups of pixels at a fixed scale. The only alternative to decomposing a cloud entity that might contain denser objects is re-running these algorithms with a different configuration of parameters. Notable evidence of this problem is presented in Pineda, Rosolowsky & Goodman (2009). They showed that, for the same region, CLUMPFIND (Williams et al. 1994) identifies a different number of clumps and their derived properties are variable when using different spatial resolution. Further, they concluded that it is not possible to derive a single mass function describing the entities of a molecular cloud when using a non-hierarchical decomposition. The approach to tackle this problem is using hierarchical representations, where a tree-like data structure is navigated for analysing molecular clouds at different scales and in different groups. That is the case for Houlahan & Scalo (1992)'s *structure tree* and Rosolowsky et al. (2008)'s *dendrograms*. However, these are structure description techniques and do not address the problem of identification of cloud entities.

We propose a method that takes into account the blended and hierarchical nature of molecular clouds, allowing the identification of cloud entities at different scale sizes. This method is formulated as a two-step process. First, we map the data from the discrete pixel domain (i.e. image or cube) to a continuous domain, where Gaussian mixtures can be properly fitted to represent the data. The model used is similar to that of Stutzki & Guesten (1990), but we do not assume a one-to-one correspondence between clumps and Gaussian components of the mixture. To provide an accurate fit, we minimize a functional following a variational approach (Logan 2013) that has been successful in image denoising and image restoration problems (Wang, Serpedin & Qaraqe 2014). In a second step, we use Gaussian mixture reduction (GMR) algorithms to generate a hierarchical binary tree similar in purpose to dendrograms (Rosolowsky et al. 2008), but containing Gaussian parameters rather than pixels. Then, any branch of the tree can be isolated by reconstructing the signal produced by the Gaussian mixture components on the leaves. The result is a flexible data structure that allows us to trace the hierarchical structure of the molecular cloud, but also to identify the cloud entities at multiple scale sizes in the hierarchy. Concretely, the contribution of this article is twofold: (1) we present a variational approach for performing a thorough fitting of the data to a Gaussian mixture that handles blended emission and (2) we propose a novel method to perform structural analysis by organizing the Gaussian components into a tree data structure that characterizes the hierarchical cloud structure.

In Section 2, we briefly discuss the algorithms and representations used in astronomy for molecular cloud analysis. Section 3 introduces the model and methods to obtain a Gaussian mixture representation of the data, while in Section 4 we explain the procedure for finding independent molecular cloud entities with this representation. In Section 5 we present the results of testing the capabilities of our proposal by using data from the Atacama Large Millimeter Array (ALMA) Science Verification (SV) database and, finally, Section 6 presents the conclusions of this work.

2 RELATED WORK

There are several approaches for detecting compact molecular cloud entities like clumps automatically and each method can differ significantly from the others, depending on the representation used and the assumptions. When each clump is represented by a set of contiguous pixels, we say that the algorithm that produced that set uses a *discrete representation*. In contrast, when clumps are characterized by a set of continuous-function parameters, we say that it uses a *continuous representation*. In addition, if the representation takes into account the nesting relations between clumps for multi-scale analysis, we say that it uses a *hierarchical representation*. If this is not the case, we say that the algorithm uses a *plain representation*.

2.1 Discrete representations

Since images and cubes are usually available in discrete pixel-based formats (e.g. Flexible Image Transport System images), the most common approach is to separate clumps directly in the pixel domain.

SExtractor (Source Extractor: Bertin & Arnouts 1996) is a very popular and mature algorithm that identifies sources within a discrete representation based on background subtraction and thresholding. Even though this algorithm has a deblending stage, each pixel is assigned to only one source, depending on the probability of belonging to each of the blended sources. While this method can be used for molecular clouds (Watson et al. 2005), it is designed and optimized to work with optical and near-infrared images in extragalactic astronomy.

Another classical algorithm in this category is CLUMPFIND (Williams et al. 1994), which seeks for clumps by contouring the data at multiple levels and connecting them when they overlap. The output is a discrete pixel representation called the *clump assignment array* (CAA), where each pixel is assigned exclusively to one clump identifier.

The GETSOURCES algorithm (Men'Shchikov et al. 2012) is a multi-scale and multi-wavelength source extraction method. It analyses fine spatial decompositions of original images across a wide range of scales and across all wavebands.

A more recent algorithm is FELLWALKER (Berry 2015). This is a gradient-tracing scheme that assigns a pixel to a clump if it is on the path to the local maximum (peak of the clump). The output is also a CAA, where each pixel belongs to a unique clump.

A compelling analysis and assessment of the clumping algorithms implemented in the CUPID software package (GAUSSCLUMPS, CLUMPFIND, REINHOLD, FELLWALKER: Berry et al. 2007) can be found in Watson (2010).

2.2 Continuous representations

The algorithms presented in the previous section associate each pixel with a single clump, segmenting the image into disjoint sets of pixels (Williams et al. 1994; Bertin & Arnouts 1996; Berry 2015). Unfortunately, assigning each pixel to a clump neglects the blended nature of the molecular cloud structure.

An alternative to pixel-level analysis is to represent the data as a linear combination of continuous functions, as proposed by the GAUSSCLUMPS algorithm (Stutzki & Guesten 1990; Kramer et al.

Table 1. Summary of related works. A partition into four groups is made, distinguishing between discrete/continuous representations and plain/hierarchical representations.

	Discrete	Continuous
Plain	SExtractor (Bertin & Arnouts 1996) clumpfind (Williams et al. 1994) getsources (Men'Shchikov et al. 2012) fellwalker (Berry 2015)	gaussclumps (Stutzki & Guesten 1990) bubbleclumps (Araya et al. 2017)
Hierarchical	SWT (Alves et al. 2007) SWT + cupid (Gregorio et al. 2014) 3D-SWT + cupid (Villanueva & Araya 2017) Dendrograms (Rosolowsky et al. 2008) SCIMES (Colombo et al. 2015)	*HDMC – our proposal*

1998). As its name states, we get a set of Gaussians with different parameters, which, summed plus the background noise, reconstructs the original signal. This allows a representation of overlapping clumps using only a few parameters per clump. However, this early approach has many limitations, like several free parameters, and strong assumptions, such as clumps being strictly *Gaussian* (i.e. symmetric and unimodal).

Recently, Araya et al. (2017) also proposed a continuous representation of the data by using *Gaussian homogeneous representations*, which exploit its homogeneity to transform the data into an independent and identically distributed sampling space to be analysed.

2.3 Hierarchical representations

Molecular clouds can be composed of several independent entities at different scales. Even though plain representation algorithms might identify them by using different parameters for each scale, the nesting relations between clumps at different scales are lost. This information is important not only for structural analysis, but also for organizing and navigating complex regions. Fortunately, this type of structural analysis can be conducted by using hierarchical representations.

Alves, Lombardi & Lada (2007) studied the origin of the stellar initial mass function of molecular clouds, through the use of multiresolution analysis based on *stationary wavelet transforms* (SWT). Gregorio et al. (2014) followed a similar methodology, but combined the SWT analysis with cupid routines (Berry et al. 2007). Then Villanueva & Araya (2017) extended this idea to work with spectroscopic cubes using 3D discrete wavelet transforms and the explicit construction of the hierarchical tree as the output result.

A different type of hierarchical description of molecular clouds is the *dendrogram* (Rosolowsky et al. 2008), which tracks the tree-like structure of the components explicitly over a wide range of scales. The authors explicitly present this technique as a statistical and hierarchical representation of the molecular ISM. Recently, Colombo et al. (2015) proposed the Spectral Clustering for Interstellar Molecular Emission Segmentation (SCIMES) method, which they describe as a more *physically oriented* approach than previous algorithms, in the sense that it clusters discrete regions with similar emission properties in giant molecular clouds (GMCs). SCIMES interprets dendrograms of GMCs in the broader framework of graph theory and uses spectral clustering techniques for the segmentation of discrete objects.

It is important to mention that, despite the hierarchical focus of the algorithms presented above, they all remain at the discrete representation level. This implies that, no matter the output these algorithms produce, they still associate each pixel with a single independent entity.

Table 1 summarizes the related work presented in this section. We grouped the methods depending on their output: whether it is discrete or continuous and whether a plain or hierarchical approach is used. Most algorithms remain in the pixel-level representation, while our proposal (bottom-right group) is the only algorithm that uses a continuous and hierarchical representation.

3 GAUSSIAN MIXTURE REPRESENTATION

The first step of the proposed method is to compute a *Gaussian mixture representation* of the data, i.e. a representation of the data as a linear combination of Gaussian functions. This section focuses on obtaining an accurate representation, while Section 4 addresses how to use this representation for analysis.

The use of Gaussian mixtures for clump detection is commonly associated with the early gaussclumps algorithm (Stutzki & Guesten 1990). Here, we propose a different approach, which fits the Gaussian mixture in a global and coupled way, without making *a priori* assumptions about the shape of the clumps.

To help in following the explanation of these steps, an outline of the procedure is shown in Fig. 1(a)–(d) for a unidimensional signal case. First, the coordinates and intensity values of the observed signal are mapped to the [0,1] interval and the background level is estimated in a region without signal (see Fig. 1a). Then, the background level is subtracted from the observed data to find the *significant emission* region (e.g. emission between the vertical dotted lines in Fig. 1b). This is done by discarding pixels with negative values and those belonging to small pixel groups. Then an initial guess is estimated for the Gaussian representation within the significant emission region (see Fig. 1c). Finally, the Gaussian mixture estimate is improved iteratively through a non-linear optimization process (see Fig. 1d).

The rest of this section describes the details of each of the aforementioned steps required to produce an accurate Gaussian mixture representation of the data.

3.1 Data pre-processing

Since each input data point has a different coordinate scale, we standardize them by mapping each coordinate axis to the [0,1] range. To do this, we assume that the data are in $\mathbb{R}^D$, with $D = 2$ in the case of images and $D = 3$ in the case of spectroscopic cubes. The intensity values are also mapped to the [0,1] range. See Appendix A for further explanation.

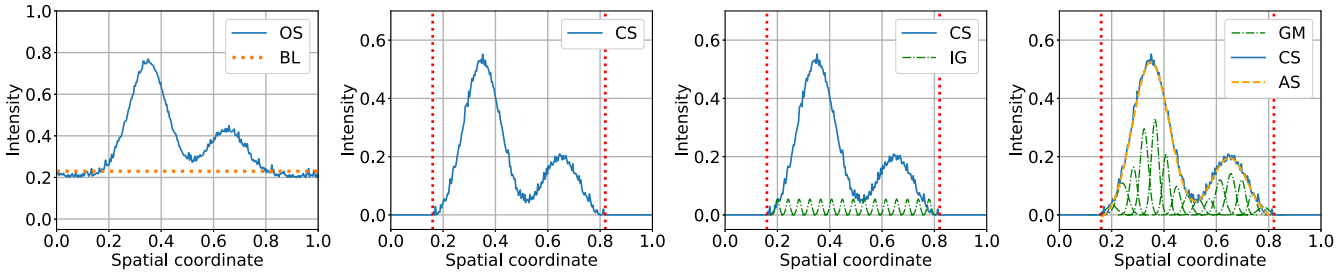


Figure 1. Unidimensional illustrative outline of the Gaussian mixture representation step. (a) The observed signal (OS) is shown with a solid line (bimodal), while an estimation of the background level (BL) is shown in the horizontal dotted line. (b) The background-subtracted and cleaned signal (CS) is shown with a solid line and the region with significant emission is bounded by the dotted vertical lines. (c) The initial-guess estimation (IG) of the Gaussian mixture representation is shown with a dash-dotted line. (d) The final approximated solution (AS) is obtained and shown with a dashed line and the individual Gaussian mixture components (GM) are shown with a dash-dotted line.

Let us define $f_0\colon\ [0,1]^D \to [0,1]$ as the function that describes the standardized observed data and $f\colon\ [0,1]^D \to [0,1]$ as the function that describes the background-free data. Then,

$$f_0(\boldsymbol{x}) = f(\boldsymbol{x}) + \epsilon,$$

where $\boldsymbol{x} \in [0,1]^D$ is a *position–position* ($D = 2$) or *position–position–velocity* ($D = 3$) coordinate and ε is the additive background noise. Having the data as a continuous function is required by our proposal, since we need to evaluate f_0 anywhere in the domain $[0,1]^D$. The function f_0 is built as a bilinear interpolator in the case of 2D data and as a trilinear interpolator for 3D data. It is worthwhile mentioning that this kind of interpolator might not be the optimal for interpolating radioastronomy data, but it has shown a good compromise between accuracy and computational cost.

From all the available coordinates, we are only interested in the subset of *significant emission*: $\Omega \subseteq [0,1]^D$. Each coordinate in Ω is what we call a *significant pixel*. To identify these regions, we estimate the background level considering a sourceless region. All pixels with values below this level are discarded from Ω and the background level is subtracted from all the other pixels. This creates segmented groups of pixels, from which only those larger than a given size are considered significant. In practice, this selection is made using erosion and dilation morphological operations, which is comparable to how Berry (2015)'s FELLWALKER algorithm performs the same operation using cellular automatons.

After this pre-processing stage, we obtain an estimation of the background-free data $\tilde{f} : [0,1]^D \to [0,1]$. Please note that our estimation considers that the background emission ε is uniformly distributed, which is a very optimistic assumption, as it might not be the case in many datasets. We direct the reader to the work by Men'shchikov (2017), where a more sophisticated background-subtraction method is proposed.

3.2 Solution structure

Now, the main goal is to find a function $u\colon\ \Omega \subseteq [0,1]^D \to [0,1]$ as similar as possible to f. We constrain u to be a Gaussian mixture of N components, where each Gaussian component is defined by its weight, mean (or centre location) and covariance matrix: $(c_i, \boldsymbol{\mu}_i, \Sigma_i)$, respectively. The general form of such Gaussian mixtures is as follows:

$$u(\boldsymbol{x}) = \sum_{i=1}^{N} c_i \exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_i)^{\mathrm{T}}\Sigma_i^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_i)\right). \tag{1}$$

We define[1] $\boldsymbol{\eta} = [c_1, \ldots, c_N, \boldsymbol{u}_1^{\mathrm{T}}, \ldots, \boldsymbol{u}_N^{\mathrm{T}}, \mathrm{vec}(\Sigma_1)^{\mathrm{T}}, \ldots, \mathrm{vec}(\Sigma_N)^{\mathrm{T}}]^{\mathrm{T}}$ as the vector of all the parameters that define u. Additionally, we impose further restrictions on the solution space in order to get a simple and computationally tractable method:

(i) the weights c_i should always be positive;
(ii) the Gaussian centres $\boldsymbol{\mu}_i$ are restricted to be in the region of significant emission Ω;
(iii) the Gaussians are restricted to be spherically symmetric.

A discussion about these restrictions and how they are implemented is presented in Appendix B1. The problem can then be formulated as optimizing the parameters $\boldsymbol{\eta}$ of u under these restrictions, in order to find an accurate approximation of f. The method to obtain such a function u is presented in the next section.

3.3 Variational formulation

A problem that may arise when trying to approximate $\tilde{f}$ with a Gaussian mixture u is that u might have larger values than $\tilde{f}$ in certain areas of the domain Ω. This results in an undesired overestimation of flux in u with respect to $\tilde{f}$, a phenomenon that we call *flux addition*.

In order to obtain an accurate representation of $\tilde{f}$ and also reduce the possible flux overestimation in u, we formulate the problem as a minimization of the following functional:

$$\begin{aligned} u^* &= \arg\min_u J(u) \\ &= \arg\min_u \int_\Omega L(\boldsymbol{x}, u, u_x, u_y, \cdots)\,\mathrm{d}\Omega \\ &= \arg\min_u \int_\Omega \underbrace{(\tilde{f}-u)^2}_{\text{Similarity}} + \underbrace{\alpha\Psi(u-\tilde{f})}_{\text{Flux Control}}\,\mathrm{d}\Omega, \end{aligned} \tag{2}$$

where $u*$ is the minimizing function, $\alpha \in \mathbb{R}_0^+$ is a weight parameter and $\Psi \geq 0$ is a function described in detail in Appendix C. The Lagrangian function $L : \mathbb{R} \to \mathbb{R}$ determines the properties of the solution we are looking for. The proposed Lagrangian has two main terms, described as follows.

[1]Here, vec() refers to the vectorization operator, which converts the input matrix into a column vector by stacking the columns of the matrix.

(i) *Similarity:* this is the standard term that penalizes the L_2-norm of the difference between observed data $\tilde{f}$ and the Gaussian mixture approximation u.

(ii) *Flux control:* this penalizes the overestimation of flux of u over f; it penalizes u when it adds extra (non-existent) flux as part of the solution, i.e. u should be upper bounded by f_0. This is explained in detail in Appendix C.

The minimum $u*$ of the functional (2) can be obtained by solving the *Euler–Lagrange* equation (Gupta 1996; Logan 2013):

$$\frac{\partial L}{\partial u} - \sum_{k=1}^{D} \frac{\mathrm{d}}{\mathrm{d}x_k} \frac{\partial L}{\partial u_{x_k}} = 0, \quad \text{with} \quad u(\partial\Omega) = f(\partial\Omega), \tag{3}$$

where Dirichlet boundary conditions are imposed on $\partial\Omega$. If we replace the Lagrangian in (2) in the Euler–Lagrange equation (3), we obtain the equation

$$2(\tilde{f} - u) + \alpha\Psi'(u - \tilde{f}) = 0. \tag{4}$$

Since partial derivatives of u are not present in the Lagrangian L, equation (4) is algebraic and non-linear in u. It is important to point out that one could solve equation (4) by setting $u = \tilde{f}$. However, this does not satisfy our assumption of representing the data as a mixture of Gaussians with the constraints mentioned in Section 3.2.

3.4 Gaussian mixture parameter optimization

As we described in Section 3.2, we have parametrized the structure of u as a constrained mixture of Gaussians. Now we want to find a function u with this predefined structure that solves equation (4).

We will use $u(\boldsymbol{x}, \boldsymbol{\eta})$ to denote the constrained Gaussian mixture parametrized by $\boldsymbol{\eta}$. Then, to complete the first step of the method, we need to find the *parameter vector* $\boldsymbol{\eta}$ to approximate the function $u*$. In order to do this, we define $E(\boldsymbol{x}, \boldsymbol{\eta})$ as the expression obtained when we evaluate $u(\mathbf{x}; \boldsymbol{\eta})$ on the left side of the Euler–Lagrange equation (4) at $\boldsymbol{x}$ and for a given $\boldsymbol{\eta}$:

$$E(\boldsymbol{x}, \boldsymbol{\eta}) = 2[f(\boldsymbol{x}) - u(\boldsymbol{x}; \boldsymbol{\eta})] + \alpha\Psi'[u(\boldsymbol{x}; \boldsymbol{\eta}) - \tilde{f}(\boldsymbol{x})] \tag{5}$$

and we also define $\mathrm{BC}(\boldsymbol{x}, \boldsymbol{\eta}) = u(\boldsymbol{x}; \boldsymbol{\eta}) - f(\boldsymbol{x})$. Then, the problem can be reformulated as solving

$$E(\boldsymbol{x}, \boldsymbol{\eta}) = 0 \text{ with boundary condition } \mathrm{BC}(\boldsymbol{x}, \boldsymbol{\eta}) = 0.$$

This non-linear system of equations arises by evaluating $E(\boldsymbol{x}, \boldsymbol{\eta})$ at collocation points $\mathcal{Q}$ and evaluating $\mathrm{BC}(\boldsymbol{x}, \boldsymbol{\eta})$ at boundary points $\mathcal{B}$. The discussion and method for generation of these points is described in Appendix D. If $\boldsymbol{x}_{\mathrm{col}}^{(i)} \in \mathcal{Q}$ are the selected collocation points and $\boldsymbol{x}_{\mathrm{bound}}^{(i)} \in \mathcal{B}$ are the selected boundary points, then the following non-linear system of equations is obtained:

$$r(\boldsymbol{\eta}) = \begin{pmatrix} E\left(\boldsymbol{x}_{\mathrm{col}}^{(1)}, \boldsymbol{\eta}\right) \\ \vdots \\ E\left(\boldsymbol{x}_{\mathrm{col}}^{(|\mathcal{Q}|)}, \boldsymbol{\eta}\right) \\ \mathrm{BC}\left(\boldsymbol{x}_{\mathrm{bound}}^{(1)}, \boldsymbol{\eta}\right) \\ \vdots \\ \mathrm{BC}\left(\boldsymbol{x}_{\mathrm{bound}}^{(|\mathcal{B}|)}, \boldsymbol{\eta}\right) \end{pmatrix} = \mathbf{0}. \tag{6}$$

In many cases, we need a fine-grained evaluation of $E(\boldsymbol{x}; \boldsymbol{\eta})$ in Ω to get an accurate approximation, obtaining an over-determined non-linear system of equations.[2] Additionally, we observed a good empirical behaviour in the non-linear least-squares solvers for this problem. For these reasons, we decided to solve the system (6) through a least-squares approach, by minimizing the residual function $R(\boldsymbol{\eta}) = \frac{1}{2} r(\boldsymbol{\eta})^{\mathrm{T}} r(\boldsymbol{\eta})$ with the Levenberg–Marquardt algorithm (LM). We chose LM because of its stability properties (local convergence ensured) and its computational efficiency (Moré 1978).

4 GAUSSIAN MIXTURE REDUCTION

4.1 Gaussian agglomeration

The second part of this method is to use the Gaussian mixture representation to identify and quantify independent cloud entities and their hierarchical dependences. For this purpose, we perform a Gaussian mixture reduction (GMR) algorithm on the Gaussian mixture. This was the main reason why we used Gaussians instead of any other particular basis function.

A *Gaussian mixture* can be formally written as a linear combination of normal probability density functions (PDFs):

$$h_N(\boldsymbol{x}) = \sum_{i=1}^{N} w_i \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \Sigma_i) \quad \text{with}$$

$$\mathcal{N}(x; \boldsymbol{\mu}_i, \Sigma_i) = \frac{1}{\sqrt{(2\pi)^D |\Sigma_i|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^{\mathrm{T}} \Sigma_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i)\right), \tag{7}$$

also denoted as $\tilde{M} := \{(w_1, \boldsymbol{\mu}_1, \Sigma_1), \ldots, (w_N, \boldsymbol{\mu}_N, \Sigma_N)\}$, where w_i are the weights, $\boldsymbol{\mu}_i$ the mean vectors and Σ_i the covariance matrices. The GMR algorithms consist of reducing an N-component Gaussian mixture $h_N(\boldsymbol{x})$ to an M-component Gaussian mixture $h_M(\boldsymbol{x})$ (with $M < N$), such that h_M and h_N are as little dissimilar as possible, in a sense to be defined next in this section.

Since all the results in the literature of GMR algorithms apply only to Gaussian mixtures as presented in (7), we need to perform an algebraic arrangement on the $u(\boldsymbol{x})$ function:

$$\begin{aligned} u(\boldsymbol{x}) &= \sum_{i=1}^{N} c_i \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^{\mathrm{T}} \Sigma_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i)\right) \\ &= \sum_{i=1}^{N} c_i \frac{\sqrt{(2\pi)^D |\Sigma_i|}}{\sqrt{(2\pi)^D |\Sigma_i|}} \exp\left(-\frac{1}{2}(\boldsymbol{x} - \boldsymbol{\mu}_i)^{\mathrm{T}} \Sigma_i^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_i)\right) \\ &= \sum_{i=1}^{N} w_i \mathcal{N}(\boldsymbol{x}; \boldsymbol{\mu}_i, \Sigma_i), \end{aligned} \tag{8}$$

with $w_i = c_i \sqrt{(2\pi)^D |\Sigma_i|}$. We need to point out that (8) is the same $u(\boldsymbol{x})$, but rewritten and interpreted in a convenient way for GMR purposes. Each component in this mixture represents exactly the same component in the original linear combination of Gaussians.

There are many algorithms to perform GMR and a complete survey of them can be found in Crouse et al. (2011). Among these algorithms, we chose the Runnalls (2007) approach for the following reasons. (1) It is computationally efficient. (2) It is the most robust among simple GMR algorithms: advanced algorithms like Gaussian

[2]This over-determined non-linear system is obtained by increasing the size of the collocation point set ($\mathcal{Q}$), such that $|\mathcal{Q}| + |\mathcal{B}| > |\eta|$, i.e. the number of equations is greater than the number of unknown parameters.

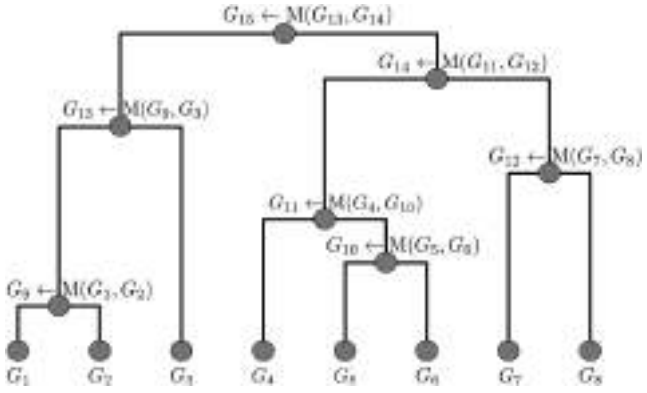


Figure 2. Example tree describing the hierarchical agglomeration process. Here G_i represents a single Gaussian and $M()$ is the merging function, which in this case computes the *moment-preserving merge*.

Mixture Reduction via Clustering (GMRC: Schieferdecker & Huber 2009) and Constraint Optimized Weight Adaptation (COWA: Chen, Chang & Smith 2010) use its output as an initial guess. (3) It is a *hierarchical clustering algorithm*, i.e. it allows the construction of a hierarchical binary tree describing the relations between the reduced mixture components.

In order to understand how the Runnalls (2007) GMR algorithm works, we need to introduce the following two concepts.

(i) **Moment-preserving merge**. The moment-preserving merge of two Gaussians $\{(w_i, \boldsymbol{\mu}_i, \Sigma_i), (w_j, \boldsymbol{\mu}_j, \Sigma_j)\}$ corresponds to a single Gaussian $(w_m, \boldsymbol{\mu}_m, \Sigma_m)$ that preserves the zeroth, first- and second-order moments of the two-component mixture. Its parameters are obtained as follows:

$$\begin{aligned} w_m &= w_i + w_j, \\ \boldsymbol{\mu}_m &= \frac{w_i}{w_m}\boldsymbol{\mu}_i + \frac{w_j}{w_m}\boldsymbol{\mu}_j, \\ \Sigma_m &= \frac{w_i}{w_m}\Sigma_i + \frac{w_j}{w_m}\Sigma_j + \frac{w_i w_j}{w_m^2}(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)^{\mathrm{T}}. \end{aligned} \tag{9}$$

(ii) **Kullback–Liebler divergence** (KLd). The Runnalls (2007) approach compares each two-component mixture $\{(w_i, \boldsymbol{\mu}_i, \Sigma_i), (w_j, \boldsymbol{\mu}_j, \Sigma_j)\}$ with its moment-preserving merge $\{(w_m, \boldsymbol{\mu}_m, \Sigma_m)\}$ using a Kullback–Liebler divergence (KLd) upper bound. In Williams & Maybeck (2003), the KLd is described as the 'ideal cost function' for Gaussian mixture reduction, but it has the disadvantage of lacking any *closed form* in the case of Gaussian mixtures. However, Runnalls (2007) obtained the following upper bound:

$$d_{\mathrm{KL}}(i, j) = \frac{1}{2}\left((w_i + w_j)\log|\Sigma_m| - w_i \log|\Sigma_i| - w_j \log|\Sigma_j|\right), \tag{10}$$

which exhibits very good empirical results.

The agglomeration process then works as follows: *while more than one component remains in the Gaussian mixture $\tilde{M}$, find the two components that are the least dissimilar in the sense of (10) and replace them in $\tilde{M}$ with its moment-preserving merge.*

Through this agglomeration process, we build a binary tree $\mathcal{T}$, which contains the hierarchical dependences between the mixture components and the cloud entities these components represent. A visualization of the creation process of $\mathcal{T}$ is shown in Fig. 2. This tree is built from the bottom to the top and each node represents a single Gaussian. Every time we merge two components, we replace the branch by its moment-preserving Gaussian.

4.2 Gaussian hierarchical decomposition

In this step, we decompose the tree $\mathcal{T}$ hierarchically (see Fig. 2). This process consists of traversing $\mathcal{T}$ from top to bottom, decomposing it into a set of disjoint subtrees, each one representing an independent cloud entity. In particular, we are interested in obtaining a set of disjoint subtrees of $\mathcal{T}$: $\{\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_K\}$, where each one is *linked* to a single cloud entity. This data structure has the advantage that it allows itself to be navigated through, thus making it possible to dive into the structure of the molecular cloud. This method is very convenient, because it allows identification of the cloud entities at multiple scale-levels across the hierarchical structure of the molecular cloud, but one can also obtain an accurate Gaussian mixture representation for each of the substructures.

An example of the decomposition of the hierarchical tree is illustrated in Fig. 3, where the first two steps of the decomposition procedure are shown. In Fig. 3(a), two cloud entities are represented with the subtrees $\mathcal{T}_1$ and $\mathcal{T}_2$, while in Fig. 3(b) the subtree $\mathcal{T}_2$ is decomposed into its corresponding subtrees $\mathcal{T}_2'$ and $\mathcal{T}_3$. In simple terms, each of the subtrees in Fig. 3 is *linked* to a cloud entity. In this case, the cloud entities represented by trees $\mathcal{T}_2'$ and $\mathcal{T}_3$ are substructures of the cloud entity represented by tree $\mathcal{T}_2$. In this way, we are able to track the hierarchical relationships between the different independent cloud entities identified.

A key feature of our method is that we keep the mixture of leaf Gaussians as the flux representation, rather than using the moment-preserving merge Gaussians at the root of each subtree. Therefore, there is no strict Gaussianity assumption (or any other shape) of the clumps. Also, we can reconstruct accurate images of the identified clumps from the leaf Gaussians.

5 EXPERIMENTS

In this section we assess our method quantitatively and qualitatively[3]. We study its parameter dependence for N and α in Sections 5.1 and 5.2, respectively.

Afterwards, we perform several experiments on a set of astronomical images in Section 5.3 and on spectroscopic cubes in Section 5.4. For these purposes, we take advantage of the publicly available ALMA SV data set[4] (Hills 2011). We collected calibrated continuum maps of Orion KL, NGC 3256 and NGC 4038, part of the Antennae galaxies and the spectroscopic line cube of Orion KL. Table 2 shows a summary of some characteristics of the selected data. These observations were selected because they display a clumpy and hierarchical structure, perfect for our purposes.

To evaluate the performance of our algorithm, the following quantities are measured.

(i) **RMS of residual**: $\left(\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}(\tilde{f}_i - u_i)^2\right)^{1/2}$, which is the root-mean-square (RMS) value of the residual over $\mathcal{I}$.

(ii) **Flux addition**: $\sum_{i\in\mathcal{I}}(u_i - \tilde{f}_i)^{+5}$, which is the amount of flux added by u (over $\tilde{f}$).

[3]This implementation is based on PYTHON and is supported by standard libraries: NUMPY, SCIPY, NUMBA, MATPLOTLIB, ASTROPY, among others. All this code is publicly available in the repository for reproducibility of the experiments.

[4]https://almascience.nrao.edu/alma-data/science-verification

[5]Here, $()^+$ denotes the *positive part function*, defined as $(x)^+ := \max(0, x)$.

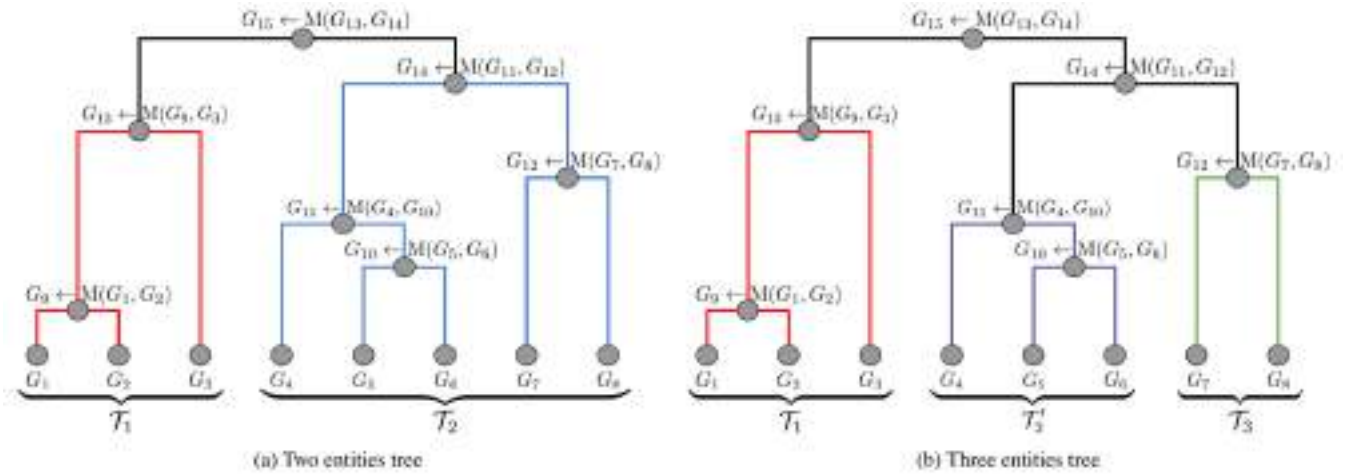


Figure 3. First two steps of the decomposition process.

Table 2. FITS data used from the ALMA Science Verification archive. The first three entries correspond to continuum maps and the remaining two are data cubes. Here, NPIX-RA denotes the number of pixels in RA coordinates, NPIX-Dec denotes the number of pixels in Dec. coordinates, NPIX-Freq denotes the number of pixels in frequency coordinates, Frequency corresponds to the central frequency of the observation, ARes denotes angular resolution, B_{Min} and B_{Maj} denote the minor and major axes of the beam and SRes denotes spectral resolution.

Name	NPIX-RA	NPIX-Dec	NPIX-Freq	Frequency [MHz]	ARes [″]	B_{Min} [″]	B_{Maj} [″]	SRes [MHz]
Orion_KL	100	100	1	230.949	0.40	1.368	1.856	–
NGC 3256-CO1_0	100	100	1	107.366	1.00	4.609	6.770	–
NGC 4038-CO2_1	500	500	1	344.031	0.13	0.658	1.206	–
Orion_KL-CH3OH	100	100	41	229.753	0.40	1.375	1.894	0.49

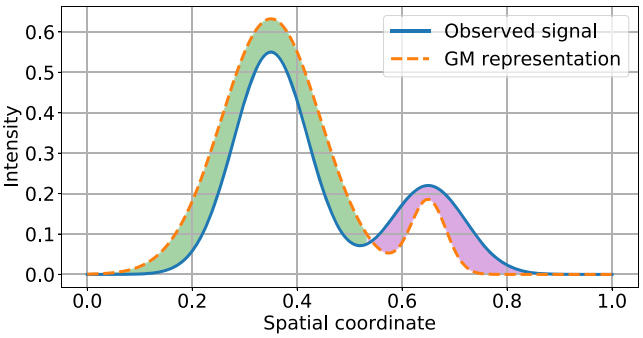


Figure 4. An illustrative sketch for flux addition (left, green area) and flux loss (right, pink area) measures between the observed signal and its Gaussian mixture representation.

(iii) **Flux lost:** $\sum_{i\in\mathcal{I}}(\tilde{f}_i - u_i)^+$, which is the amount of flux not represented by u (under $\tilde{f}$).

Here, $\mathcal{I}$ is defined as the set of the indexes of *significant pixels* and $\tilde{f}_i$ and u_i are the values of $\tilde{f}$ and u evaluated at the ith pixel.

Fig. 4 shows an illustrative sketch for better understanding of these measures. Notice that the RMS of the residual evaluates how similar u is to $\tilde{f}$, i.e. it evaluates the impact of the first term of the functional (2). The remaining measures assess whether flux is being lost or added by the Gaussian mixture representation. These evaluate the impact of the second term in the functional (2).

5.1 Number of Gaussians

The first set of experiments consists of the study of the *quality* of the obtained solution u and the number of Gaussians required for it. In Fig. 5 (left column), we observe the results for the 2D images in Table 2 as a function of N. Note that there is a decreasing behaviour for all quantities as we increase N. The reduction of the RMS of the residual, flux addition and flux lost indicate that we are achieving a better approximation of $u*$.

As expected, adding components to the mixture representation allows us to obtain a more accurate representation. However, it is important to note that the computational complexity is quadratic with the number of centres, $\mathcal{O}(N)^2$ (See Fig. 6).

5.2 Flux addition

The second set of experiments consists of the study of the importance of the *free parameter* α. As presented in Appendix C, α controls the relative importance between the terms in the functional (2). This means that higher values of α give a greater relevance to the *flux control* term with regard to the *similarity* term.

In Fig. 5 (right column), the results for the 2D images in Table 2 are presented. We observe that there is an evident decrease in the amount of flux addition, which is what we expected when increasing the value of α. As a direct consequence, there is an increase in the flux lost and the RMS of the residual. This happens because the relative importance of the *similarity* term in (2) decreases.

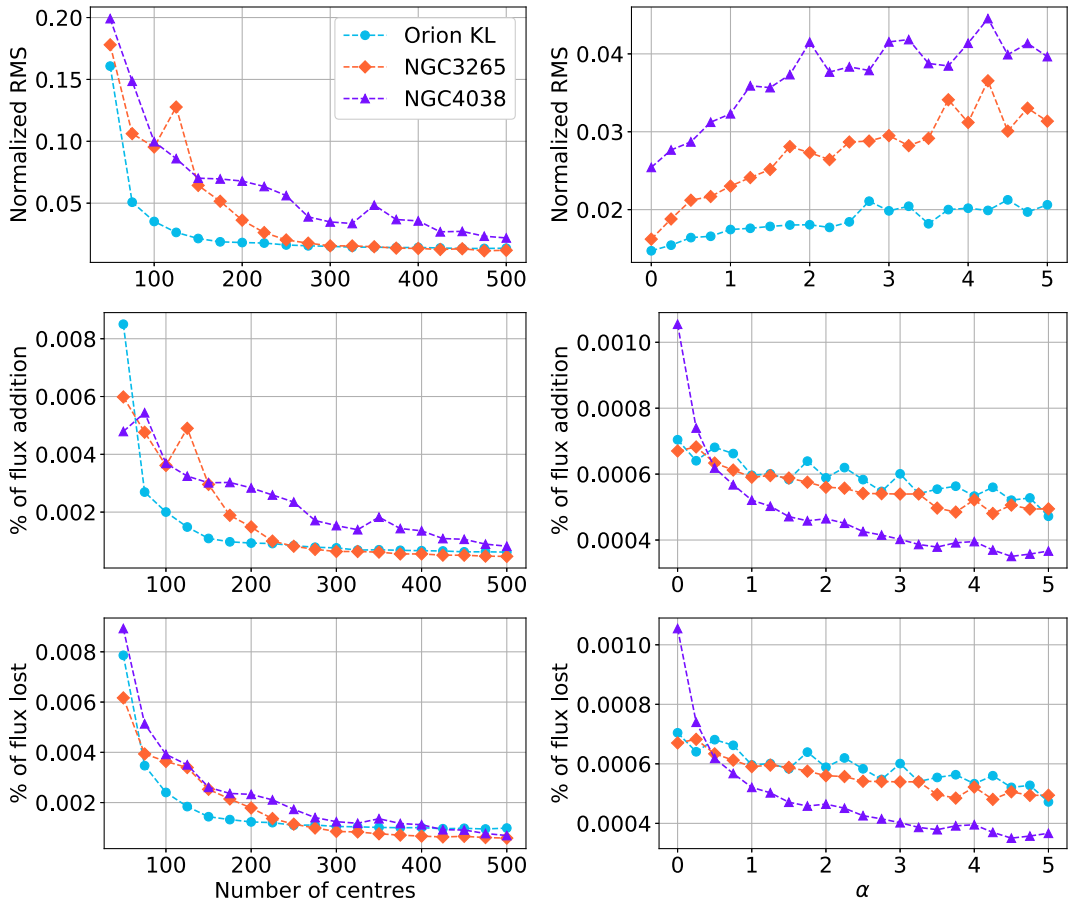


Figure 5. Normalized RMS, per cent flux addition and per cent flux loss of the residual (RMS was normalized by the RMS of $\tilde{f}_0$). In the *left column* are the results for the *number of centres* experiment (with $\alpha = 0$ fixed). In the *right column* are the results for the *alpha variation* experiments (with Number of centres $= 300$ fixed).

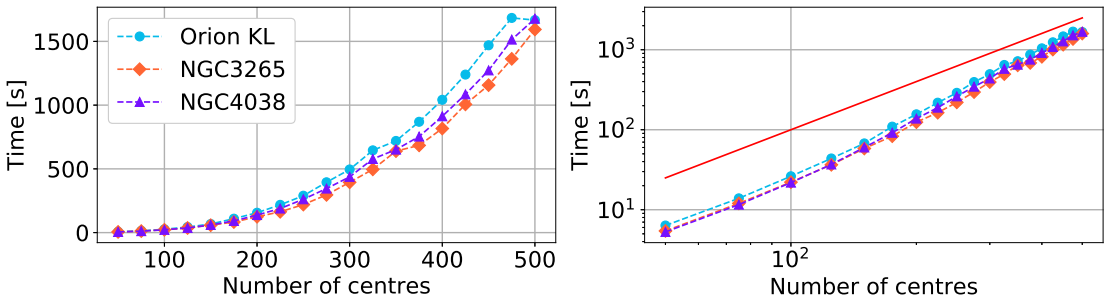


Figure 6. Computation times versus the number of centres used. The red line in the right figure is a quadratic function of the number of centres, plotted to help us visualize the computational complexity.

5.3 Structural analysis of images

In this section, we show the capabilities of our proposal for the hierarchical identification and description of independent cloud entities, by applying it to real and synthetic data, and we also show how the structural analysis is performed.

For assessment purposes, we use the synthetic generated image shown in Fig. 7. This image was generated as the sum of eight blended Gaussians. Additive background noise modelled with the half-normal distribution (Byers 2005): $\varepsilon \sim |X|$, with $X \sim \mathcal{N}(0, \sigma^2)$ and $\sigma = 0.01$, was generated and added to the image. This allows us to know exactly which part of the emission corresponds to the signal and therefore we also know the real flux distribution of the clumps (the number of clumps and the shape of each one).

To test the capability to identify entities within the cloud further, we use the continuum map of the Orion KL complex from ALMA SV. This region is one of the archetypal massive star-forming regions in the Galaxy (Crockett et al. 2014) and is of particular interest due to its close distance (in fact, it is the nearest star-forming region, at a distance of 414 ± 7 pc: Menten et al. 2007) and its rich chemical inventory (Friedel & Looney 2017). Thus, it has been described in detail (e.g. in Crockett et al. 2014; Feng et al. 2015), chemically and structurally. Regarding the latter, it is known that emission in Orion KL is due to several distinct spatial/velocity components and

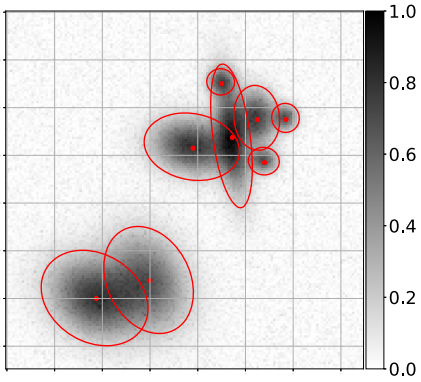


Figure 7. Synthetic generated data of 150 × 150 pixels; the red dots and ellipses indicate the positions of each Gaussian emission.

several clumps and sources have been identified (e.g. see table 1 in Friedel & Looney 2017 for a summary overview of these components). The most prominent are the hot core, compact ridge, plateau and extended ridge (Crockett et al. 2014). In the continuum map of Orion KL from SV data, only the hot core and the compact ridge appear, as shown in fig. 8 of Crockett et al. (2014).

In Fig. 8, we present four steps of the hierarchical decomposition process over the Orion KL image, showing very good results. For a good visualization, we display each detected entity with a contour line at the background level. In this case, up to seven cloud entities are clearly identified. We can decompose each identified entity even further (a feature of the method), but this may lead to over-decomposing them. It is clear from Fig. 8 that our algorithm is capable of recognizing the known entities in the Orion KL complex.

As a comparison, in Fig. 9 we show the results obtained by the CLUMPFIND and FELLWALKER algorithms. Here we display the boundary of the detected clumps. Similar entities are detected for all algorithms. However, with our method, we were able to capture emission of blended entities and identify substructures that the other algorithms are not able to find.

It is important to mention that our method does not determine the number of clumps, as CLUMPFIND or FELLWALKER optimistically try to do. However, we offer the possibility of producing a decomposition for an arbitrary number of clumps, allowing us to dig deeper into the hierarchical structure of molecular gas by being able to identify cloud entities at different scale sizes across this hierarchy. Moreover, this is done with no extra significant computational cost, since all the data needed are stored in the hierarchical tree data structure. This is not possible with classical clump identification algorithms, which have to be re-executed with a different set of parameters to get a different number of clumps identified.

In real data, it is not possible to separate the contribution of different entities in the flux value for each pixel, thus there exists some blending in the distribution of flux per pixel among the entities. This issue can be addressed by using synthetic data, where we know how much flux belongs to each entity. In this way, our knowledge of the structure of the cloud is accurate. We perform the same structural analysis as with Orion KL on the synthetic image in Fig. 7.

Fig. 10 shows the decomposition results obtained. At each step, we split the cloud entity, which exposes a hierarchical dependence, with denser objects contained inside that cloud, attempting to match

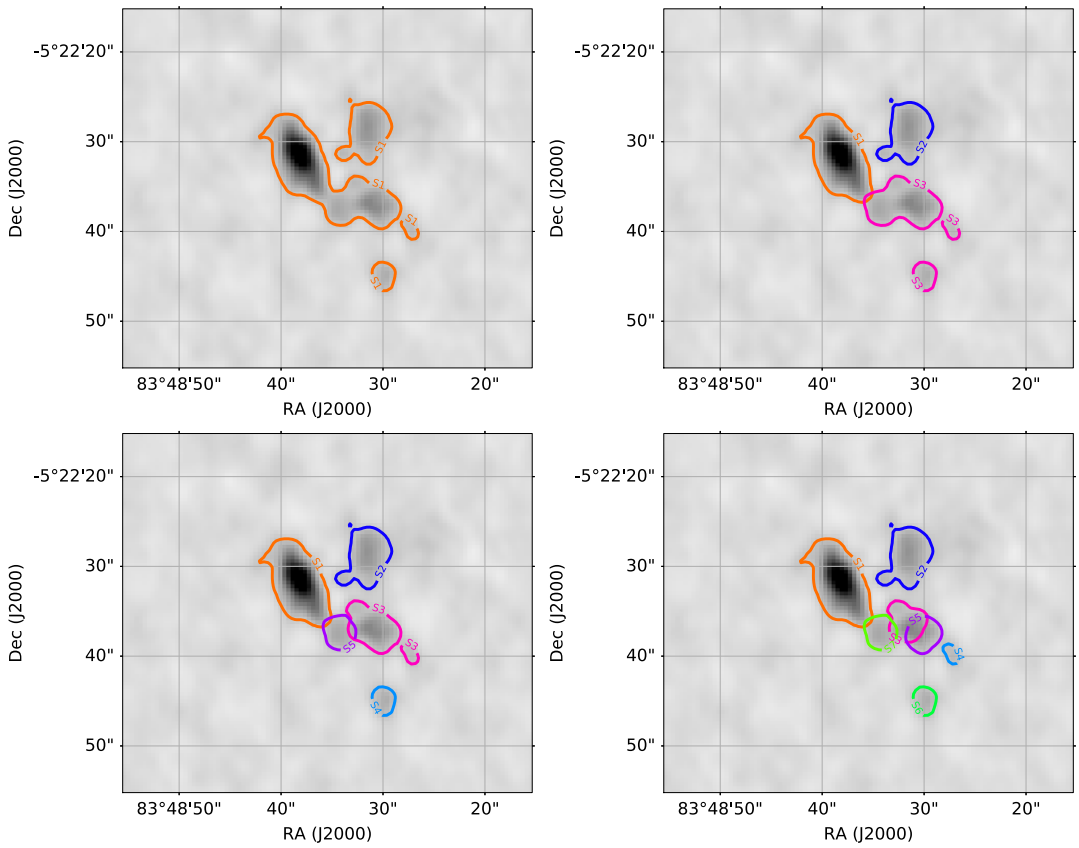
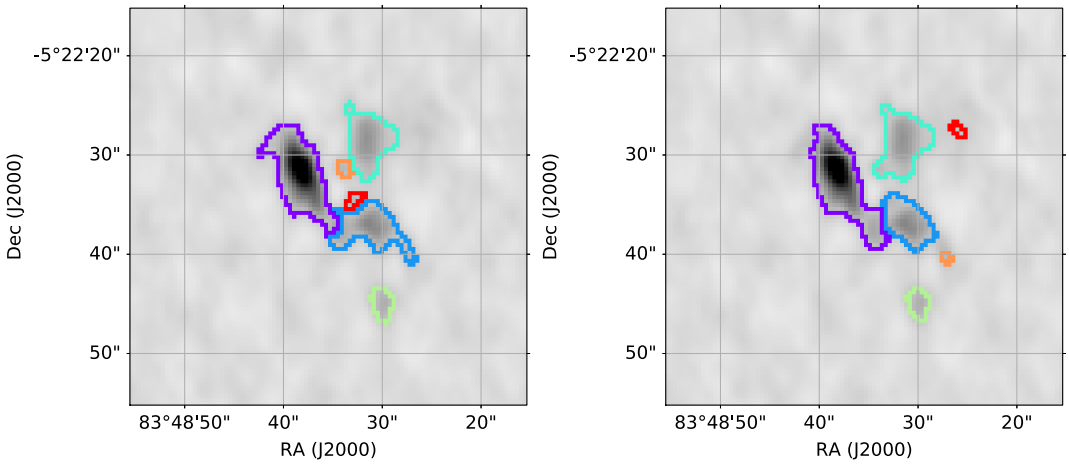


Figure 8. Gaussian decomposition process over Orion KL. Solution with 1, 3, 5 and 7 entities detected is shown, from left to right and top to bottom.



(a) *clumpfind* (b) *fellwalker*

Figure 9. Segmentation results of classic clumping algorithms over Orion KL.

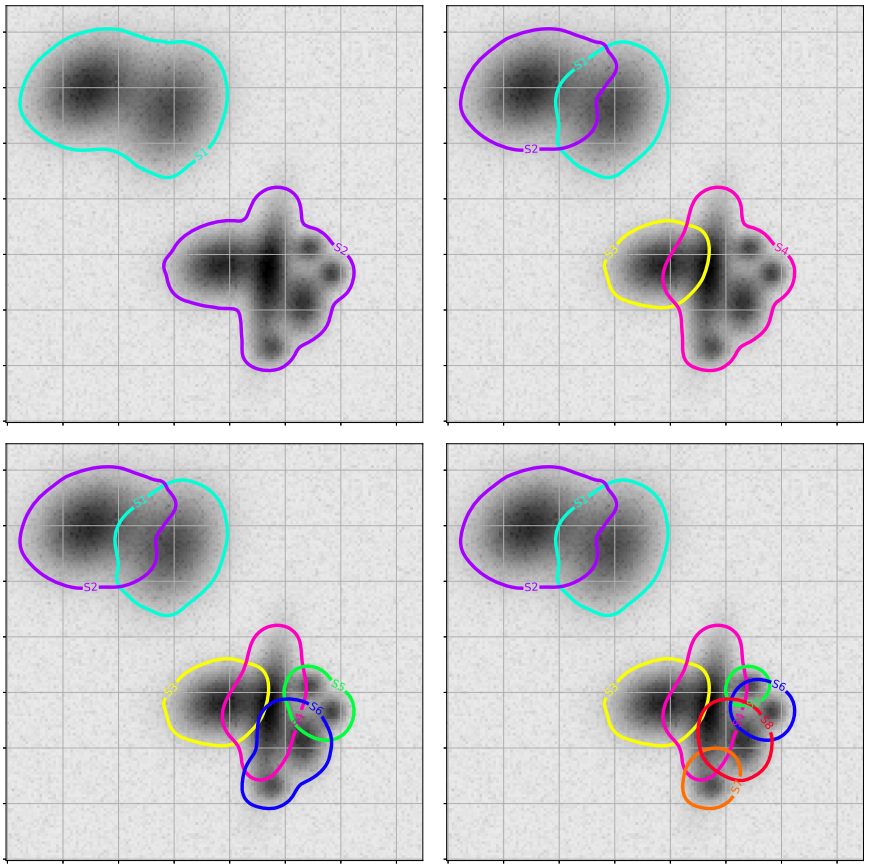
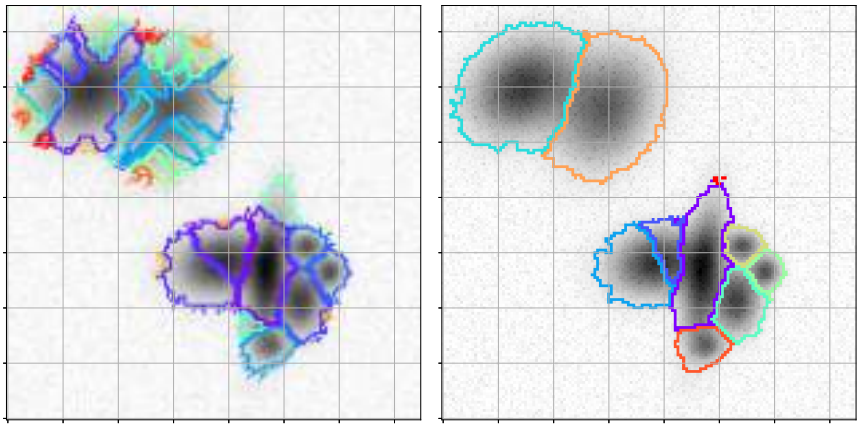


Figure 10. Gaussian decomposition process over synthetic data. Decomposition with 2, 4, 6 and 8 detected entities is shown, from left to right and top to bottom.

the flux distribution of the clumps of the synthetic data. All but one of the clumps was correctly detected and represented. The exception happens because that particular clump has low intensity and small extent, making it difficult for any algorithm to identify.

For comparison, we show in Fig. 11 the boundary of the clumps detected by the CLUMPFIND and FELLWALKER algorithms in the synthetic image. We observe that CLUMPFIND has serious problems with noise, producing as a result a large number of clumps as output. On the other hand, FELLWALKER does a very good job of detecting the clumps. However, it is unable to capture the blended emission of coupled entities and thus the representation of such blended clumps is not accurate.



(a) *clumpfind* solution. (b) *fellwalker* solution.

Figure 11. Segmentation results of classic clumping algorithms over synthetic data.

5.4 Structural analysis on spectroscopic cubes

Finally, we analyse spectroscopic data cubes. Fig. 12 shows the final results of the hierarchical decomposition process. For good visualization and interpretation, we show the three 2D projections (one for each stacked axis) of the cloud entities identified by our method. Here, each detected entity is represented with a contour surface at the background level. This means of visualization is useful for our decomposition procedure, because if we see a cloud entity that exhibits a hierarchical dependence by containment of substructures inside it in any of these projections, then we can choose to decompose it.

We observe in Fig. 12 a decomposition of the integrated map obtained from the line of methanol (CH_3OH) in Orion KL. This line traces the hot core and the compact ridge (Crockett et al. 2014). Again, our results show that we can identify the entities in at least one of the projections.

We observe in Fig. 12 a decomposition of the integrated map obtained from the line of methanol CH_3OH (rest frequency = 229.759 GHz) in Orion KL. This lines traces the entities we observed in the continuum map, namely the hot core (HC) and the compact ridge (CR). The right panel from Fig. 12 shows the decomposition obtained by our algorithm in the complex. Structures S2 and S5 correspond approximately to the HC and CR, respectively. The peak frequency corresponds to a value of $v_{\rm lsr} \approx 8\,{\rm km\,s^{-1}}$ for S2 (HC) and $v_{\rm lsr} \approx 6.8\,{\rm km\,s^{-1}}$ for S5 (CR), in good agreement with previous works (Friedel & Looney 2017).

It is important to mention that the computational cost for processing 3D spectroscopic cubes with our method increases considerably. This is due to the higher number of Gaussians needed to obtain an accurate representation.

6 CONCLUSIONS

In this work we have introduced a new and novel approach for hierarchical analysis of two- and three-dimensional observations of molecular clouds. We presented a convenient alternative to state-of-the-art algorithms (Alves et al. 2007; Rosolowsky et al. 2008; Men'Shchikov et al. 2012; Gregorio et al. 2014) to perform structural analysis, which was able to trace the nested relations between multi-scale cloud entities. The novelty of our proposal is that not only is it a descriptor of the hierarchical structure of the molecular cloud, but it also builds a tree data structure that allows us to identify molecular cloud entities at different scales. Furthermore, each identified cloud entity is represented by a mixture of Gaussians. We have also shown that this representation is accurate and handles emission of blended cloud entities successfully.

The proposed algorithm consists of two main steps. The first step produces a continuous representation of the image as a Gaussian mixture and the second produces a binary tree data structure through a Gaussian mixture reduction algorithm.

A set of experiments was performed to show the capabilities and limits of our proposal. We demonstrated that the quality of the solution obtained, i.e. the accuracy of the representation, improves as the number of Gaussians increases (within a reasonable extra computation time). We also validated that the increment of the parameter α was able to reduce the overestimation of flux of the Gaussian mixture for the original data. This quality test was performed using continuum maps of molecular gas from ALMA SV with different levels of clumpiness and resolution.

We also presented the output obtained with our algorithm applied to the continuum map of OrionKL from the ALMA SV database. Our algorithm successfully recovers the well-known cloud entities identified in the literature. When compared with other state-of-the-art algorithms such as CLUMPFIND and FELLWALKER, our proposal deals properly with the emission of blended entities in molecular clouds and thus is more sensitive for identifying substructures in the cloud. Moreover, with our method, we were able to identify and produce an accurate representation of multiple-scale entities transversely in the molecular cloud hierarchy. This is possible because our method combines features from *continuous representations* as well as *hierarchical representations* of previous approaches to clump and cloud identification. This analysis included 2D images and 3D spectroscopic cubes.

Some features of HDMC require further analysis and may extend and improve the results presented here considerably. The most remarkable are the following.

(i) In order to simplify the model, we decided to use spherically symmetrical Gaussians. However, with data with very different resolution scales, this would require the use of more Gaussians in the

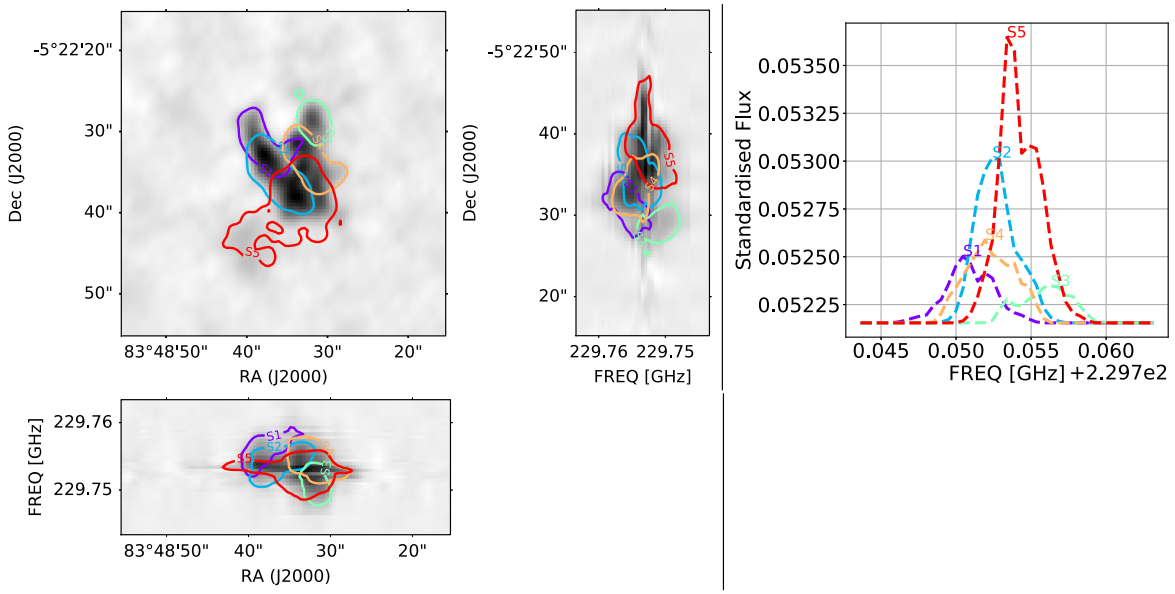


Figure 12. Structural analysis over Orion_KL-CH3OH with a decomposition of five entities. *Left*: visualization of the three 2D projections of the detected entities. *Right*: frequency spectrum of the detected entities, standardized by the total sum of flux.

mixture representation, which would imply a higher computational cost.

(ii) The current implementation requires long computation times for spectroscopic cubes. This is mainly due to the differences in the resolution mentioned above.

(iii) In this article, we restricted our analysis to molecular cloud observations only. However, the metric we used to define the agglomeration criteria is a fully data-driven approach (see Section 4.1), therefore our method can be used to perform structural analysis on other kinds of observation: galaxy formation and protoplanetary discs, among others. The agglomeration process can be strengthened by combining both data and astronomy knowledge.

This method provides the astronomical community with a new tool with features unseen before. Among these new features, we highlight a dynamical hierarchical decomposition, i.e. it is possible to decide how many cloud entities should be provided or which molecular entity needs to be navigated in depth. Also, it provides a Gaussian mixture representation for each of these. Therefore, we have proposed, analysed, compared and validated a new and novel algorithm for identification and hierarchical decomposition of molecular cloud entities in blended astronomical data.

REPRODUCIBLE RESEARCH

We release the implementation of our method for research purposes under the GitHub platform. The code can be obtained at: https://github.com/mavillan/HDMC.git. In the repository, we include the implementation of HDMClouds with scripts and JUPYTER NOTEBOOKS that allow the reproduction of our results. The project is licensed using GNU GENERAL PUBLIC LICENSE version 3 (GPLv3) terms of use.

ACKNOWLEDGEMENTS

This work has been partially funded by CCTVal, CONICYT PIA/Basal FB0821, CONICYT PIA/Basal FB0008, FONDEF IT 15I10041, FONDECYT 11160744 and FONDECYT 1150810.

This paper makes use of the following ALMA data: ADS/JAO.ALMA#2011.0.00002.SV, ADS/JAO.ALMA#2011.0.00003.SV, ADS/JAO.ALMA#2011.0.00009.SV, ADS/JAO.ALMA#2011.0.000010.SV; ALMA is a partnership of ESO (representing its member states), NSF (USA) and NINS (Japan), together with NRC (Canada), MOST and ASIAA (Taiwan) and KASI (Republic of Korea), in cooperation with the Republic of Chile. The Joint ALMA Observatory is operated by ESO, AUI/NRAO and NAOJ.

We also thank Dr Rodrigo Herrera for his helpful discussion, which helped to improve this article.

REFERENCES

Alves J., Lombardi M., Lada C., 2007, A&A, 462, L17
Araya M., Mendoza M., Solar M., Mardones M., Bayo A., 2018, A&C, 24, 25
Berry D., 2015, Astronomy and Computing, 10, 22
Berry D., Reinhold K., Jenness T., Economou F., 2007, in Shaw R. A., Hill F., Bell D. J., eds, ASP Conf. Ser.Vol. 376, Astronomical Data Analysis Software and Systems XVI. Astron. Soc. Pac., San Francisco, p. 425
Bertin E., Arnouts S., 1996, A&AS, 117, 393
Blitz L., Williams J. P., 1999, in Lada C. J., Kylafis N. D., eds, The Origin of Stars and Planetary Systems. Springer-Verlag, Netherlands, p. 3
Byers R., 2005, Half-Normal Distribution∗. Wiley StatsRef, Wiley StatsRef: Statistics Reference Online
Chen H., Chang K., Smith C., 2010, Proc. SPIE, 7697, 76970N
Colombo D., Rosolowsky E., Ginsburg A., Duarte-Cabral A., Hughes A., 2015, MNRAS, 454, 2067
Crockett N. R. et al., 2014, ApJ, 787, 112

Crouse D. F., Willett P., Pattipati K., Svensson L., 2011, Proceedings of the International Conference on Information Fusion, A Look At Gaussian Mixture Reduction Algorithms. IEEE, Chicago, IL,p. 1
Faure H., Lemieux C., 2009, ACM Transactions on Modeling and Computer Simulation (TOMACS), 19, 15
Feng S., Beuther H., Henning T., Semenov D., Palau A., Mills E. A. C., 2015, A&A, 581, A71
Friedel D. N., Looney L. W., 2017, AJ, 154, 152
Gregorio R., Solar M., Mardones D., Pichara K., Parada V., Contreras R., 2014, Proc. SPIE, 9152, 91522R
Gupta A., 1996, Calculus of Variations with Applications. PHI learning, Delhi, India
Heyer M., Dame T. M., 2015, ARA&A, 53, 583
Hills R., 2011, ALMA Science Verification, ALMA Newsletter
Houlahan P., Scalo J., 1992, ApJ, 393, 172
Kramer C., Stutzki J., Rohrig R., Corneliussen U., 1998, A&A, 329, 249
Leroy A. K. et al., 2013, AJ, 146, 19
Logan J. D., 2013, Applied Mathematics, fourth edn. John Wiley & Sons, p. 221
Marquardt D. W., 1980, Journal of the American Statistical Association, 75, 87
Men'shchikov A., 2017, A&A, 607, A64
Men'Shchikov A., André P., Didelon P., Motte F., Hennemann M., Schneider N., 2012, A&A, 542, A81
Menten K. M., Reid M. J., Forbrich J., Brunthaler A., 2007, A&A, 474, 515
Moré J. J., 1978, in Watson G. A., eds, Conference on Numerical Analysis. Argonne National Lab., IL, USA,p. 105
Myers P., Fuller G., Goodman A., Benson P., 1991, ApJ, 376, 561
Pineda J. E., Rosolowsky E. W., Goodman A. A., 2009, ApJ, 699, L134
Pringle J., Allen R. J., Lubow S., 2001, MNRAS, 327, 663
Rosolowsky E., Pineda J., Kauffmann J., Goodman A., 2008, ApJ, 679, 1338
Runnalls A. R., 2007, Proc. IEEE, 43, 989
Schieferdecker D., Huber M. F., 2009, Information Fusion, 2009. FUSION'09. 12th International Conference, Gaussian Mixture Reduction via Clustering. IEEE, Seattle, p. 1536
Shadmehri M., Elmegreen B. G., 2011, MNRAS, 410, 788
Shu F. H., Adams F. C., Lizano S., 1987, ARA&A, 25, 23
Stutzki J., Guesten R., 1990, ApJ, 356, 513
Villanueva M., Araya M., 2017, 8th International Conference of Pattern Recognition Systems (ICPRS 2017), Multiresolution and Hierarchical Analysis of Astronomical Spectroscopic Cubes using 3D Discrete Wavelet Transform. IET, Madrid, Spain
Wang X., Serpedin E., Qaraqe K., 2014, Advances in Mathematical Models and Production Systems in Engineering. WSEAS, Brasov, Romania, p. 11
Watson R. A., Rebolo R., Rubiño-Martín J., Hildebrandt S., Gutiérrez C., Fernández-Cerezo S., Hoyland R., Battistelli E., 2005, ApJ, 624, L89
Watson M. E., 2010, Master's thesis, University of Hertfordshire
Williams J. L., Maybeck P. S., 2003, Proceedings of the Sixth International Conference of Information Fusion. IEEE, Piscatawat, NJ, p. 1047
Williams J. P., De Geus E. J., Blitz L., 1994, ApJ, 428, 693
Williams J. P., Blitz L., McKee C. F., 2000, in Mannings V., Boss A. P., Russell S. S., eds, Protostars and Planets IV. Univ. Arizona Press, Tucson, p. 97

APPENDIX A: SCALING AT PRE-PROCESSING

The data used in this article consider spatial coordinates, frequencies and intensity values, each of them with their own scale. This diversity of scales is a great challenge for numerical computing when the algorithms used need to couple the variables. A traditional and yet useful strategy to manage this is by means of normalization of the variables used (Marquardt 1980).

In the general case of a data cube, we have two spatial coordinates, with N_x and N_y pixels in each dimension, respectively, and a spectral coordinate with N_v elements. If we let $N_{\max} = \max(N_x, N_y, N_v)$, then the coordinate mapping from the pixel indices to a subset of the $[0, 1]^D$ domain is the following:

$$(i, j, k) \xrightarrow{\text{mapping}} (x_i, y_i, v_k) \xrightarrow{\text{mapping}} \left(\frac{i+1/2}{N_{\max}}, \frac{j+1/2}{N_{\max}}, \frac{k+1/2}{N_{\max}}\right),$$

where $i \in \{0, 1, \ldots, N_x - 1\}, j \in \{0, 1, \ldots, N_y - 1\}, k \in \{0, 1, \ldots, N_v - 1\}$ and the 1/2 factor is added so it is located at the centre of each pixel. Notice that the dimension with a larger number of pixels gets mapped to [0,1] and the other two are mapped to a subset of [0,1].

APPENDIX B: SOLUTION PARAMETRIZATION

B1 Parameter constraints

In order to achieve a simple and computationally tractable method, the following restrictions are imposed over the parameters of u.

(i) The weights c_i should always be positive: $c_i \in \mathbb{R}_0^+$. This is for two reasons: (1) since Gaussians in u represent *pieces* of a cloud structure, they should be positive in order to ensure physical interpretation and (2) the Gaussian agglomeration step presented in Section 4.1 interprets each Gaussian function as if they were *probability density functions* (PDF), hence they should be positive. This is done by defining $c_i = \hat{c}_i^2$ with $\hat{c}_i \in \mathbb{R}$.

(ii) The Gaussian centres $\boldsymbol{\mu}_i$ are restricted to be in the region of significant emission Ω. This is because it is meaningless that u approximates $\tilde{f}$ in the regions where there is no significant emission. For such reasons, we introduce the *reference centres* $\hat{\boldsymbol{\mu}}_i$, the neighbourhood length $\boldsymbol{\delta}_i \in \mathbb{R}^D$ and a control parameter $\boldsymbol{\theta}_i \in \mathbb{R}^D$. Then the *real centres* for each Gaussian are computed as[6]

$$\boldsymbol{\mu}_i = \hat{\boldsymbol{\mu}}_i + \boldsymbol{\delta}_i \odot \sin(\boldsymbol{\theta}_i).$$

(iii) The Gaussians are spherically symmetric, i.e. Σ_i is a diagonal matrix with the same value in the diagonal. Additionally, we introduce the *minimal and maximal width* ($\sigma_{\min}$ and $\sigma_{\max}$, respectively) parameters, which are the minimal and maximal standard deviation allowed for each Gaussian. The structure of the covariance matrix is

$$\Sigma_i = \begin{pmatrix} \sigma_i^2 & 0 & 0 \\ 0 & \sigma_i^2 & 0 \\ 0 & 0 & \sigma_i^2 \end{pmatrix}, \quad \text{with } \sigma_i^2 = (\sigma_{\max}^2 - \sigma_{\min}^2)\tanh(\hat{\sigma}_i^2) + \sigma_{\min}^2, \tag{B1}$$

with $\hat{\sigma}_i \in \mathbb{R}$. The reasons for bounding the standard deviation of the Gaussians are basically two: (1) it allows us to bound the size of the Gaussians, rendering the optimization process numerically more stable, and (2) it contributes to obtaining a more homogeneous mixture of Gaussians, which is a good property for the agglomeration step described in Section 4.1.

It is important to distinguish between the *real model* parameters $\boldsymbol{\eta}$ as presented in Section 3.2 and the *internal model* parameters presented in this section that allow us to bound the real ones. These *internal model* parameters correspond to $\{\hat{c}_i, \boldsymbol{\theta}_i^{\mathrm{T}}, \hat{\sigma}_i\}_{i=1}^N$. The other parameters such as $\hat{\boldsymbol{\mu}}_i$, $\boldsymbol{\delta}_i$, $\sigma_{\min}$ and $\sigma_{\max}$ are set heuristically, as

[6]Here, sin () is applied elementwise and $\odot$ denote elementwise multiplication.

explained in Appendix B2. Then the *internal model* parameter vector is defined as $\tilde{\boldsymbol{\eta}} = [c_1, \ldots, c_N, \boldsymbol{\theta}_1^{\mathrm{T}}, \ldots, \boldsymbol{\theta}_N^{\mathrm{T}}, \hat{\sigma}_1, \ldots, \hat{\sigma}_N]$ and is mapped directly to $\boldsymbol{\eta}$.

B2 Heuristically set parameters

(i) *Minimal and maximal width.* These two ($\sigma_{\min}$, $\sigma_{\max}$) are estimated as

$$\sigma_{\min} = \kappa \frac{1}{6D} \sum_{d=1}^{D} \mathrm{pl}_d,$$
$$\sigma_{\max} = K\sigma_{\min} \text{ with } K = \frac{N_{\text{significant pixels}}}{N_{\text{Gaussians}}}, \tag{B2}$$

where pl_d is the pixel length in each dimension and κ is a constant value. The reason we set $\sigma_{\min}$ in this way is that the smallest Gaussian fits in κ pixels. For a single pixel ($\kappa = 1$), the coverage of a Gaussian (we consider 3σ) is set to be half the size of a pixel, where the pixel size is estimated as the mean of the pixel lengths in each dimension pl_d. This gives us equation (B2). The value κ is estimated as BSize/ARes, the quotient between the *Beam Size* and *Angular Resolution* of the input data. The value of $\sigma_{\max}$ is just K times $\sigma_{\min}$, where K is an estimation of the number of pixels each Gaussian must cover. It considers Gaussian centres uniformly distributed across the significant pixels.

(ii) *Reference centres.* The vectors $\hat{\boldsymbol{\mu}}_i$ are determined with the method presented next in Section D1.

(iii) *Neighbourhood of reference centres.* The maximum distance the actual centres can be The length of neighbourhood around maximum distance the, namely $\boldsymbol{\delta}_i$, is set equal for all Gaussians. We set this parameter as $\boldsymbol{\delta}_i = [\sigma_{\min}\ \sigma_{\min}]^{\mathrm{T}}$ or $\boldsymbol{\delta}_i = [\sigma_{\min}\ \sigma_{\min}\ \sigma_{\min}]^{\mathrm{T}}$ (depending if $D = 2$ or $D = 3$), i.e. centres can move a distance equal to the minimal width.

APPENDIX C: FLUX CONTROL

The main reason for the variational formulation in Section 3.3 was to include the flux-control term in (2). This term is introduced to reduce the overestimation of flux in our solution u over $\tilde{f}$ and has the form $\Psi(u - \tilde{f})$. Therefore, the expected response of $\Psi(x)$ is not to penalize negative inputs ($x < 0$), since in that case there is no additional flux being added. For positive inputs ($x \geq 0$), a high and rapidly increasing penalization is expected. Then the proposal is to use $\Psi(x) = \widehat{\psi}(\lambda\, x)$, with ψ being defined as

$$\widehat{\psi}(x) = \begin{cases} 0, & x < 0, \\ 10x^3 - 15x^4 + 6x^5, & x \in [0, 1], \\ 1, & x > 1, \end{cases} \tag{C1}$$

where ψ is a fifth-order spline with domain in [0,1], built to be continuous and smooth in its entire domain. Because of its polynomial structure, evaluating the function and its derivatives is computationally fast, as they are lower order polynomials.

In this way, the parameter α in (2) controls the *maximum amount of penalization*, whereas λ controls the *speed of penalty*. To set λ, we notice that the maximum amount of penalization is reached at $x = 1/\lambda$. Since typically $|u - f| \ll 1$ during the optimization process, then $\lambda \geq 1$, with typical values in the range [1, 10], as shown in Fig. C1.

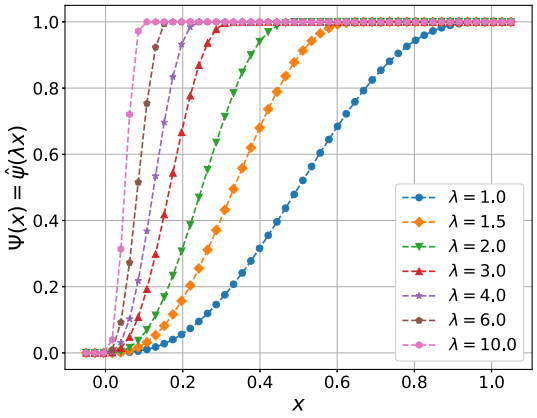


Figure C1. Fifth-order spline penalizing function and its behaviour as we move λ in the range [1., 10.]. To achieve a faster penalization in the [0., 1.] domain, higher λ values should be used.

APPENDIX D: POINTS GENERATION

It is necessary to introduce three sets of points that we use: the centre points $\mathcal{P}$ corresponding to the centres (means) of Gaussians, the collocation points $\mathcal{Q}$, which are the points where equation (4) is evaluated, and the boundary points $\mathcal{B}$, which are the points at the boundaries of the structures to be represented. In what follows, we present the methods we use to generate each of these points.

D1 Centre points

As introduced in Section B1, there are two kinds of centre points (Gaussian centres): the *reference centre* points $\hat{\mathcal{P}}$ are the fixed points around which the *real centre* points $\mathcal{P}$ can move. A good spatial distribution of these points gives us the capacity to obtain an accurate representation of the data over the region of significant pixels.

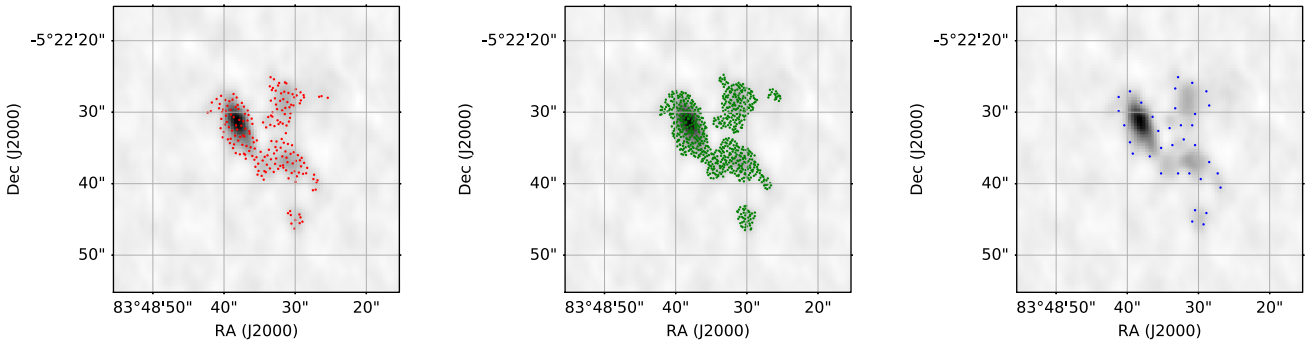
For that purpose, we make use of generalized Halton sequences (Faure & Lemieux 2009), which is a method for generation of a low-discrepancy sequence of N-dimensional points. It is a *quasi-random* sequence generator that maximizes the distance between the generated points, thereby producing an approximately uniform set of points.

Then the *reference centre* points $\hat{\mathcal{P}}$ are generated as a sequence of Halton points over the regions of significant pixels. Since the regions of significant pixels are very irregular (can have any shape) and Halton sequences fill square domains ($[0, 1]^D$), we generate Halton points until our irregular domain is filled with the desired number of points. An example of how these points are generated is shown in Fig. D1(a).

D2 Collocation points

These correspond to the points in the region of significant emission Ω where equation (4) is evaluated. In order to obtain a consistent non-linear system of equations (6), these points must satisfy$|\mathcal{Q}| \geq (D + 2)|\mathcal{P}| - |\mathcal{B}|$, where D is the dimensionality of the input data. In other words, the points in $\mathcal{Q}$ are much more than those of $\mathcal{P}$.

These points are also generated as Halton sequences over Ω. Moreover, collocation points include the *reference centre* points



(a) Centre points. (b) Collocation points. (c) Boundary Points.

Figure D1. Point-generation example over the Orion KL image from ALMA Science Verification archive. (a) 50 *reference* and *real* centre points generated as generalized Halton sequences, (b) 200 collocation points also generated as a generalized Halton sequence and (c) 50 boundary points.

previously generated: $\hat{\mathcal{P}} \subset \mathcal{Q}$. An example of how these points are generated is shown in Fig. D1(b).

The reason for $\hat{\mathcal{P}} \subset \mathcal{Q}$ is that every Gaussian has a collocation point near its centre. Having a collocation point close to each Gaussian centre is necessary to reach a good solution, otherwise it has been observed empirically that a single Gaussian could grow too much, deteriorating the overall solution.

D3 Boundary points

In order to evaluate the boundary condition $u(\partial\Omega) = f(\partial\Omega)$, a finite set of points in the boundary are selected. These points are generated as a random sample in the set of boundary pixels. The random sampling is performed assuming a uniform probability for the boundary pixels. We sample the boundary pixels iteratively: every time a pixel is selected, this and its neighbour pixels are set at probability 0. This is done with the aim of obtaining a *spatially distributed* sample. An example of how these points are generated is shown in Fig. D1(c).

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