Master's thesis

A max-tree-based astronomical source finder

Christiaan Arnoldus

27th August 2015

Primary supervisor: Dr Michael H.F. Wilkinson
Secondary supervisor: Prof. Dr Jos B.T.M. Roerdink

This thesis project was performed in collaboration with promotion candidates Ugo Moschini (of the Johann Bernoulli Institute for Mathematics and Computer Science), as well as Nadine Giese and Davide Punzo (of the Kapteyn Astronomical Institute), who provided additional supervision and feedback.

Contents

Introduction

Astronomy has entered an era where acquisition and analysis of large data volumes acquired from telescopes form an important part of research and discoveries. These data include 2D optical surveys resolved in space and 3D radio emission surveys resolved in both space and velocity. To handle these increasing volumes of data, automated solutions are becoming more and more important. Several algorithms have already been developed or are in development. A discussion of a few of these methods is given by Popping et al. [2012], as well as in section 1.1. Presently, the most important purpose of the analysis of such data sets is pure research, an effort to map the universe's stars, galaxies, and other objects.

One algorithm for analysing radio volumes was introduced by Moschini et al. [2014], which uses a max-tree representation and statistical testing, as does the optical method of Teeninga et al. [2013] (called *MT objects*), on which it was based. However promising, this algorithm requires better validation, as optical measurements possess noise characteristics different from those of radio emissions.

The aim of this thesis is to identify problems of the existing method by Moschini et al. [2014] and propose improvements. Thus, the focus here is on radio volumes, although optical datasets are sometimes used as examples, as they are more easily visualised. Moschini et al. [2014]'s method was chosen in particular because it is based on the versatile max-tree data structure [Salembier et al., 1998], whose flexibility has a high potential for extension. Additionally, fast computational performance can be achieved using the max-tree construction algorithm of Moschini et al. [2015a].

Since algorithms for the detection of astronomical objects in radio volumes are often called *source finders* (as they search for sources of radio emission), the max-tree-based method first proposed by Moschini et al. [2014] and refined here is referred to as the *MT source finder*. This allows it to be distinguished from MT objects, which is the method developed by Teeninga et al. [2013] for optical analysis.

1 Research questions

More concretely, the research aims of this thesis project are as follows:

1. Many source finders employ smoothing as an initial step. Can smoothing be incorporated in the MT source finder and if so, what is the most flexible way to do this? Which smoothing techniques are useful for this purpose?

2. The classification model (based on statistical testing) used by Moschini et al. [2014] was inherited from a different type of dataset (namely optical instead of radio data). Is this classification model still applicable and if not, how can it be improved?

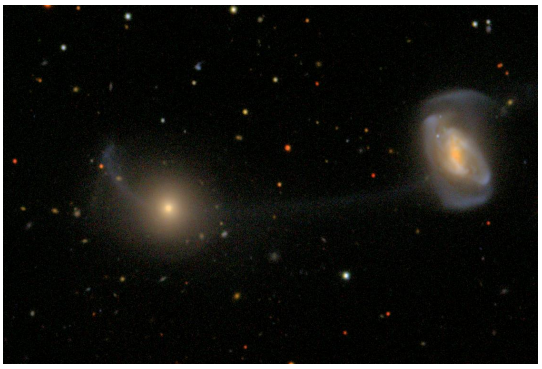

Figure 1: Example of an optical dataset. This is essentially an outer-space photograph.

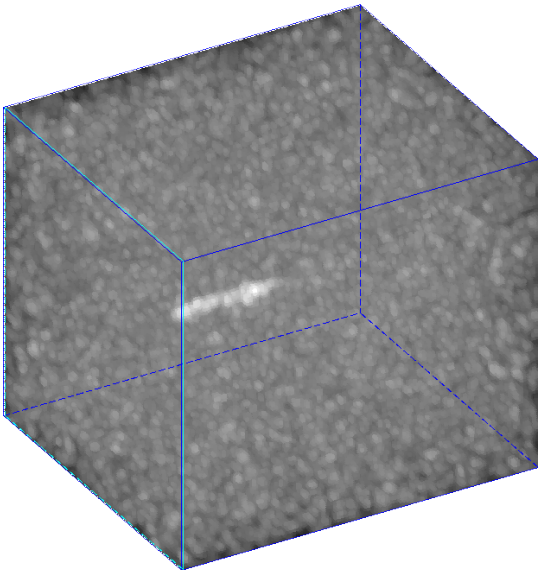

Figure 2: An example radio volume, visualised using a grey-scale map. A single faint object is embedded in noise. Segmentation of radio volumes is the main research subject of this thesis.

3. When dealing which large datasets, false detections are an inevitable problem. Many existing methods rely on techniques such as attribute thresholds to lower the number of false positives. Can similar techniques be employed here?

4. After any adaptations resulting from the above research questions, how does the performance (in terms of computing time spent and, more importantly, in terms of quality of the output) compare to the existing algorithms?

2 Thesis overview

Chapter 1 discusses some of the existing techniques for image segmentation in an astronomical context, one of which will later be used to assess the relative performance of any newly developed and refined MT source finder. Chapter 2 gives an overview of the MT source finder and introduces its several stages. An integral part of the method is the use of the max-tree data structure and related concepts from mathematical morphology, which are discussed in chapter 3. Chapters 4, 5 and 7 are the core of this thesis and attempt to answer the first three research questions posed above, which are accompanied by chapters 6 and 8 which assess the performance of the newly developed techniques (in terms of both the quality of the output and the computation time) and thus answer the last question. The thesis is concluded by chapter 9, which returns to the research questions and discusses possible future work.

3 Notes on terminology

Some terms in this thesis report are used interchangeably and do not convey any subtle differences in meaning. This thesis involves *segmentation* of images and, mainly, volumes in order to detect *objects*, which are defined as features of interest of the studied artefact. In the context of radio volumes, the objects are (radio) *sources* and *source finding* is considered a synonym for segmentation and object detection. Also, the terms *method* and *technique* are sometimes used as synonym for *algorithm*.

1 Related work

The aim of this chapter is to introduce some existing methods used in astronomy for segmentation of images and volumes. Some of these methods are particularly relevant because they form the basis for the MT source finder method extended here, while others will serve as comparison material when evaluating the performance of the various methods.

Here, a distinction between optical and radio data is made. Optical surveys give a 2D view of the sky and are essentially photographs. Radio volumes result from capturing a form of radio emission from remote galaxies. These volumes are usually 3D, where the first two dimensions are spatial (they represent the sky), while the third dimension (the spectral dimension) represents velocity magnitude (speed). Since radio waves emitted from more distant sources are measured at a higher velocity than those of closer sources, it is tempting to interpret the third dimension as a spatial one. However, this should be done with care, as this reasoning is only correct for the centroid of objects and their shape in the third dimension does not correspond to a spatial shape.

1.1 Image segmentation using optical data

Teeninga et al. [2013] propose an algorithm, named MT objects, for the extraction of astronomical objects from sky surveys, which are images taken by telescopes. They claim that this new algorithm outperforms the state-of-the-art method Source extractor, successfully extracting objects when the signal-to-noise ratio is up to eight times worse than Source extractor's limit. A qualitative comparison of Source extractor's and MT objects' results on the same input image, created by Teeninga et al. [2015b], is shown in figs. 1.1 and 1.2. In this example MT objects' result is clearly superior.

Source extractor [Bertin and Arnouts, 1996] extracts objects from images by first subtracting a background estimate and then thresholding the residual image. The background estimate is a local mode estimate. Teeninga et al. [2013] found this estimate to correlate strongly with objects and therefore they opted to use a constant, global estimate instead. In addition to object detection Source extractor sports some more features, such as splitting of inadvertently merged objects, object attribute computation, and classifying galaxies versus stars using neural networks.

The MT objects algorithm utilises the max-tree representation to segment the image in objects and background. Max-tree nodes are classified as objects when the power attribute fails the χ^2-test. Under the null hypothesis (the component is due to noise), the statistic power$(P)/\sigma^2$ follows a χ^2-distribution with area(P) degrees of freedom when the background noise is independent and Gaussian. The variance σ^2 is estimated as $f(\text{parent}(P))/\text{gain} + \sigma_B^2$, where σ_B^2 is the variance of the background. The gain is a property of the measurement apparatus and describes how the incoming signal affects the noise variance. The technique used to estimate the statistical properties of the background is described by Teeninga et al. [2015a]. The image is divided in square tiles. Tiles that contain a Gaussian signal are identified using D'Agostino–Pearson's K^2-test, while Student's

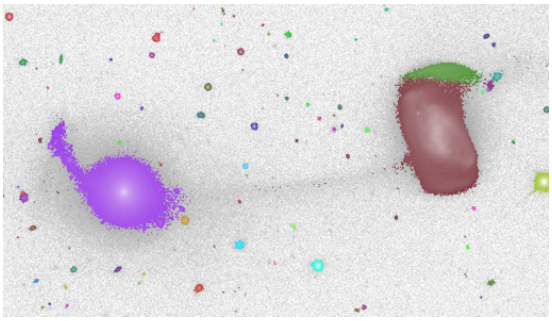

Figure 1.1: Segmentation result of Source extractor, computed by Teeninga et al. [2015b] (colours map identifiers). Note that the connection between the galaxies is not recovered.

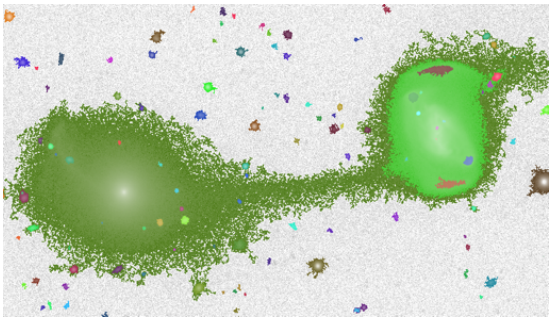

Figure 1.2: Segmentation result of MT objects, computed by Teeninga et al. [2015b] (colours map identifiers). The connection between the galaxies is recovered and more of the outer detail of the galaxies is retained.

t-test is used on different parts of a tile to reject tiles that have a linear slope. The remaining tiles are used to estimate the mean and variance σ_B^2 of the background.

Before the max-tree is built, the background mean is subtracted from the image and negative values are set to zero. Setting negative values to zero does not affect the applicability of the χ^2-test, as zero pixels solely belong to the root component, on which the χ^2-test is not applied. Nodes in the max-tree are marked as either insignificant, when the test succeeds and the peak component is due to noise, or significant, when the test fails and the peak component represents part of an object. An advantage of this method is that the significance level α is an intuitive parameter, as it is the probability of marking a peak component as significant when it is not.

The significant nodes of the max-tree are grouped into objects, merging significant nodes along the same branch of the tree in a bottom-up fashion. When multiple significant nodes share the same significant ancestor, the largest descendent (by area) is considered to be part of the same object as the ancestor, while the smaller descendant is considered a separate object. Some nodes at the base of a branch representing an object have their object labels removed, as in practice a good amount of noise around objects is marked as significant using this method. The range of nodes that has their object label removed is specified by λ, which is a factor of σ.

Teeninga et al. [2013], and Teeninga et al. [2015b] in particular, also consider additional statistical models, including ones that apply smoothing filters in advance to constructing the max-tree and those that compute the power relative to the closest *significant* ancestor, rather than the immediate parent. In these cases the χ^2-distribution no longer applies. Instead, the rejection boundaries are determined from histograms computed using Monte Carlo simulations.

Some further work was done by Moschini et al. [2015b], employing classification trees to separate interacting galaxies from galaxies that overlap in the image, but are not physically interacting. Additional techniques like bagging and boosting were used the improve the performance of the classification trees.

1.2 Radio volume segmentation

Popping et al. [2012] give an overview of several source finders that have been developed so far. One of these is the *Duchamp source finder* [Whiting, 2012], which detects object voxels by using a simple threshold. Then, the detected voxels are merged into objects and detections that are likely false based on certain criteria are rejected. Finally, some attributes of the detected objects are computed. Performance can be improved by applying a smoothing filter or wavelet reconstruction on the input.

Wavelet reconstruction is also used by the *2D–1D wavelet reconstruction source finder*, introduced by Flöer and Winkel [2012]. Wavelet reconstruction is used to decompose the data in detail representations, whose wavelet coefficients are thresholded in order to suppress noise. This process is executed several times, where each iteration considers the residual of the last result and the input data. The results of all iterations are aggregated and the dataset is reconstructed, which can then be thresholded to obtain an object mask. Since the radio data is not isotropic, the spatial and spectral dimensions are considered separately. First, slices along the spatial dimensions are considered, followed by threads along the spectral dimension.

The *smooth-and-clip source finder* independently applies a variety of smoothing filters, flags voxels by thresholding each smoothed volume separately, and takes the union to get the result. The resulting object mask is segmented in objects using 6-connectivity and objects that are too small are discarded. Serra et al. [2012] extend the S+C source finder with techniques for determining the reliability of detections, allowing false detections to be rejected. This is done by applying kernel density estimation to two 3D spaces, one containing positive detections and one containing negative detections. Then, the reliability of each detection with parameter vector $\vec{x}$ is computed as $R(\vec{x}) = (P(\vec{x}) - Q(\vec{x}))/P(\vec{x})$, where is $P(\vec{x})$ is the density of positive sources and $Q(\vec{x})$ is the density of negative sources. A *reliability filter* is then implemented by removing detections with an $R(\vec{x})$ below a certain threshold.

Other algorithms mentioned by Popping et al. [2012] are the *characterised noise HI (CNHI) source finder* [Jurek, 2012] and *GammaFinder*. Their completeness comparison for a data cube with extended sources is shown in fig. 1.3, where completeness is defined as the number of detected sources divided by the number of total sources. Each algorithm (except 2D–1D wavelet reconstruction) was executed twice, the difference being that, in the second run, close detections are merged. The percentages in the legend are the reliabilities of the source finders, where *reliability* is the number of true detections divided by the number of total detections. Popping et al. [2012] argue that S+C is the best source finder for extended sources, while GammaFinder performs worst.

The Duchamp, S+C, and wavelet reconstruction methods mentioned above employ some form of noise reduction (smoothing or wavelet reconstruction) and then use a global threshold. Using a max-tree has the advantage of allowing local attributes to be used with such a threshold, which is not possible when considering the volume as a whole. As mentioned in the introduction due to its relevance to this thesis, Moschini et al. [2014] investigate using MT objects on radio volumes (so it becomes the MT source finder), rather than the 2-dimensional images with an optical signal used before. One useful property the measurements have in the case of radio volumes is that the gain is infinite, so the noise variance can be assumed not to be dependent on the signal.

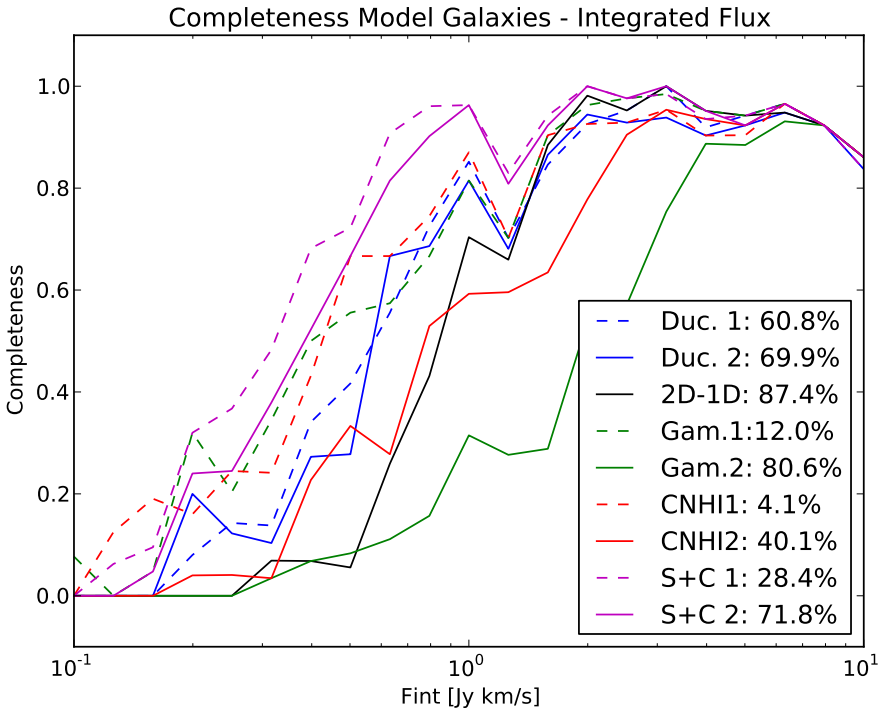

Figure 1.3: Popping et al. [2012]'s completeness comparison between several source finders (Duchamp, 2D–1D wavelet reconstruction, GammaFinder, CNHI, S+C), with completeness as a function of integrated flux, when they are applied to a dataset containing extended sources. The percentages in the legend indicate reliabilities.

2 Segmentation pipeline

In order to facilitate the segmentation of astronomical radio volumes, that is, detect objects, this thesis proposes the following segmentation pipeline for the MT source finder. The input is a raw radio volume. A radio volume is a cube with two spatial axes (the sky) and one spectral axis (the radiation velocity magnitude), whose floating-point intensities represent radio flux. The output is an object mask, which is an integer volume labelling voxels as belonging to objects (positive labels) or background (label zero). A diagram of the pipeline is shown in fig. 2.1.

2.1 Pre-processing

The first step is to improve the signal-to-noise ratio of the raw value by smoothing. Several proposals for smoothing techniques are discussed in chapter 4, which also discusses how to use the resulting smoothed volume for max-tree construction, using a method inspired by mask-based connectivity [Ouzounis and Wilkinson, 2007] termed mask-like connectivity. The smoothed volume, which drives the max-tree construction, is called the *max-tree mask*. The negative part of the max-tree mask is set to zero, to ensure that there will be no negative-total-flux nodes in the max-tree. The raw volume is not discarded, it is still useful to compute certain attributes.

2.2 Max-tree construction

After the max-tree mask has been created, a max-tree is built. The concept of a max-tree is briefly touched upon in section 1.1 and is discussed more extensively in chapter 3. The max-tree construction algorithm of Moschini et al. [2015a] is also described, as its potential for parallelisation makes it very well suited for the construction of large volumes. It is what was used for all experiments in this thesis (although there is no requirement to use it, any max-tree construction algorithm would do).

Part of the max-tree construction is the computation of attributes of max-tree nodes. Several attributes are mentioned in this thesis, which are computed from the max-tree structure, usually in a bottom-up manner.

2.3 Object detection

This is the core of the MT source finder. When the max-tree has been constructed, the next step is to identify nodes corresponding to objects and label those nodes as such. To achieve this, a technique based on the method of Teeninga et al. [2013] is used. This method relies on statistical testing, as explained in chapter 1, and chapter 5 suggests several modifications to make

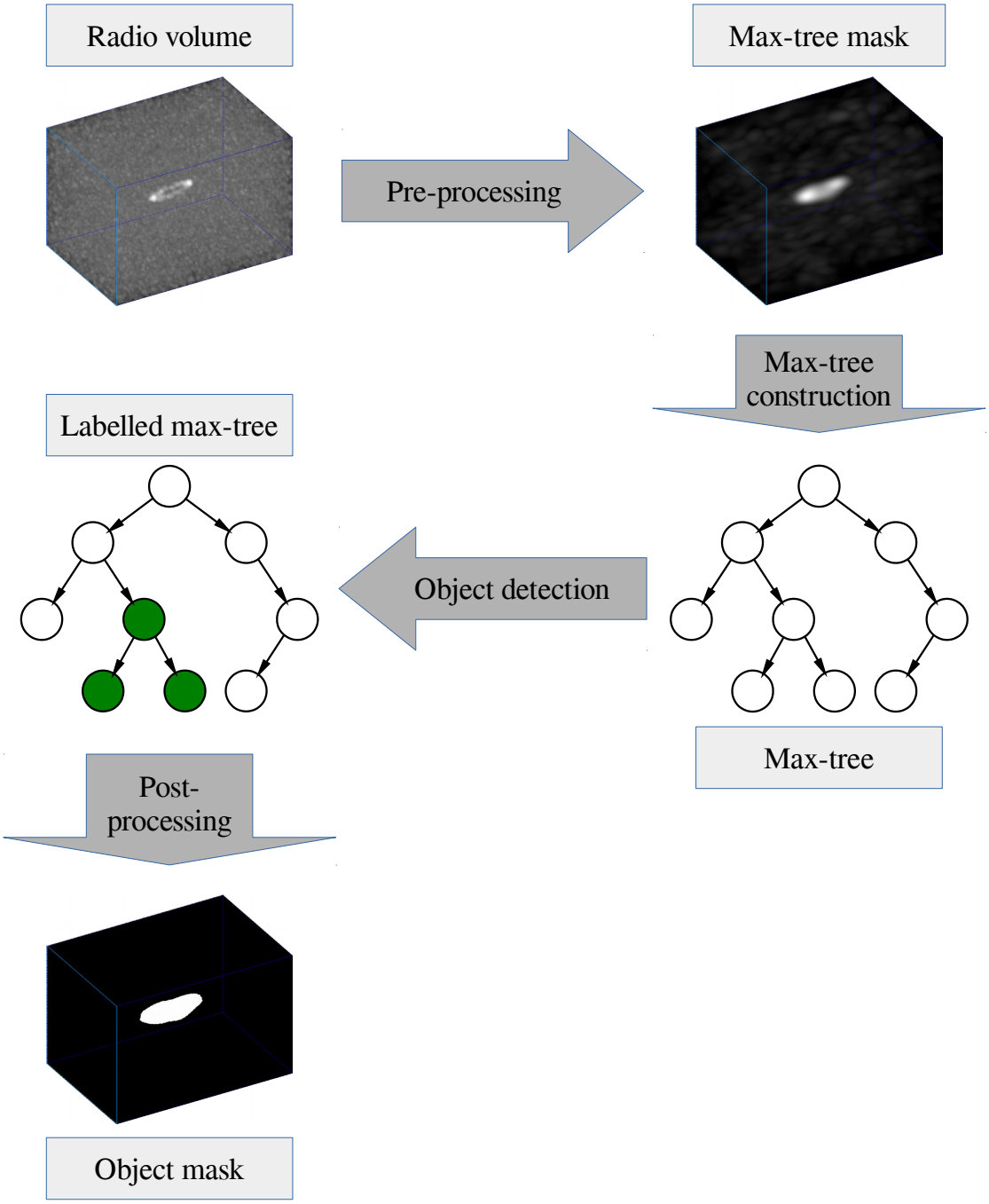

Figure 2.1: An overview of the MT source finder's pipeline. The input is a radio volume, which is then turned into the max-tree mask. A max-tree of the max-tree mask is built and object nodes are labelled as such. Post-processing optionally removes spurious detections and flattens the max-tree to an object mask.

the statistical model better fit the radio data. Overlapping objects are merged and noise attached to objects is removed. The result is a max-tree where some nodes are marked as objects.

2.4 Post-processing

The final step is to remove some of the false object detections. This is done by a combination of techniques. One of these techniques is a size filter, which removes objects that are too small to be reliable given certain properties of the measuring device. Another technique relies on rejecting certain objects based on their attributes, using the machine learning techniques discussed in chapter 7 (or a combination of the two). After this, object nodes are numbered and the max-tree is flattened to get the *object mask*.

3 Max-trees

The *max-tree* data structure is a tree-based image (or volume) representation, which stores the connected components of threshold sets in a hierarchical manner. Max-trees were introduced by Salembier et al. [1998], although many similar structures have been described before [Ouzounis and Wilkinson, 2007, sec. 5.1] (with names such as component tree or connectivity tree).

One important application of max-trees are connected operators. Connected operators are morphological operators that work on the connected components of an image, rather than on individual pixels. The advantage of this approach is that images can be filtered without distorting any edge information, as connected components are either removed or retained, but no new ones are created [Salembier et al., 1998, Ouzounis and Wilkinson, 2007]. Examples of connected operators include attribute filters, which accept or reject connected components by comparing an attribute of these components to a threshold [Ouzounis and Wilkinson, 2007, sec. 1].

In the case of binary images, connected operators are defined as all operators ψ for which the set difference $X \setminus \psi(X)$ consists solely of connected components of X or X^c [Salembier et al., 1998, sec. II]. Here, an N-dimensional binary image X is defined as a set of positions $\{x \mid x \in \mathbb{Z}^N\}$. Connected operators can be generalised to grey-scale images by the principle of *threshold superposition*. Given a grey-scale image f, the threshold sets $T_h(f)$ form a stack of nested binary images. A threshold set is defined as

$$T_h(f) = \{x \in E \mid f(x) \geq h\} \tag{3.1}$$

where E is the universal set and h a threshold [Ouzounis and Wilkinson, 2007, sec. 4.2]. A grey-scale version φ of an increasing connected filter ψ can then be defined by having φ assign h to every pixel x [Ouzounis and Wilkinson, 2011, sec. 2.3], where h is the highest threshold where x is retained, as expressed by the following *filter rule*:

$$\varphi(f)(x) = \max\{h \mid x \in \psi(T_h(f))\} \tag{3.2}$$

Equation (3.2) can be extended to non-increasing filters using the *subtractive rule* [Ouzounis and Wilkinson, 2011, sec. 2.4], defined as

$$\varphi(f)(x) = \int_0^\infty \chi(\psi(T_h(f)))(x)\,dh \tag{3.3}$$

where $\chi(X)(x) = 1$ if $x \in X$, else $\chi(X)(x) = 0$. Integration is replaced by summation in the practical case of discrete images. Note that eq. (3.2) is equal to eq. (3.3) if ψ is increasing.

Connected operators are commonly defined in terms of *peak components* P_h^x or *flat zones* F_h^x [Ouzounis and Wilkinson, 2011, sec. 2.3], which are defined as follows

$$P_h^x(f) = \Gamma_x(T_h(f)) \tag{3.4}$$

$$F_h^x(f) = \Gamma_x\{x \mid f(x) = h\} \tag{3.5}$$

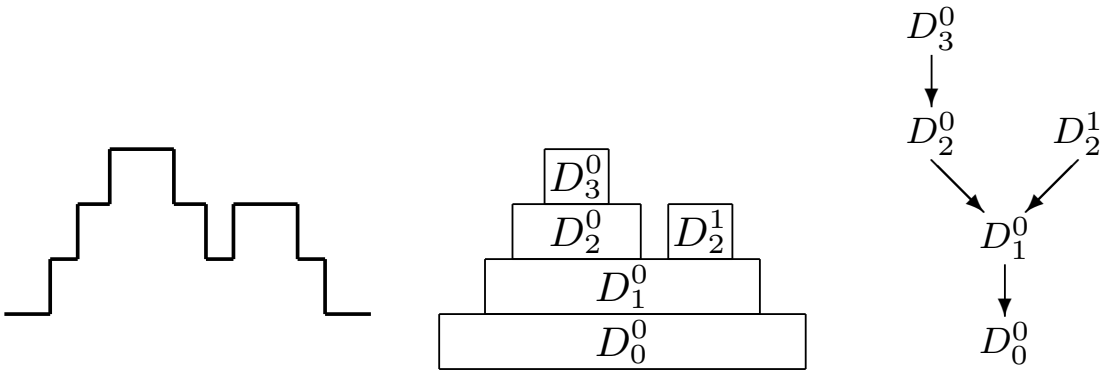

Figure 3.1: An illustration of the max-tree data structure. A 1D signal (left), its peak components (middle, denoted D_h^i here), and the corresponding max-tree (right) [Wilkinson, 2011].

where Γ_x is a *connectivity opening*, extracting the connected component to which x belongs. A flat zone F_h having no adjacent pixels of intensity larger than h is called a *regional maximum* M_h. Since a finite image has a finite number of peak components, flat zones, and regional maxima for each h, these sets can also be indexed, rather than be defined by a pixel x, for example P_h^i.

Finishing the description of max-trees, they store the hierarchy of peak components of an image. A node C_h^i of a max-tree contains those pixels such that $C_h^i = \{x \in P_h^i \mid f(x) = h\}$. Evidently, there is a one-to-one correspondence between max-tree nodes and peak components. Additionally, each max-tree node C_h^i (other than the root) stores a pointer to its parent $C_{h'}^j$, where $h' < h$, which is the node corresponding to the peak component one grey level lower [Ouzounis and Wilkinson, 2007, sec. 5.1]. An example max-tree is shown in fig. 3.1. This data structure allows for efficient computation of increasing attributes, as it is a matter of traversing the tree in a bottom-up fashion, and accumulating the values of child nodes. Non-increasing attributes can also be computed, if they can be decomposed in increasing attributes. Anti-extensive attribute filters can be implemented by applying the filtering rule of choice on each node.

3.1 Algorithms

On the implementation level, max-trees are not necessarily represented as a tree of peak components. Rather, a popular representation is to store one node for each pixel. Only the nodes that have a parent at a lower grey level are relevant for the structure of the tree; these nodes are called *level roots* [Moschini et al., 2015a] or *canonical elements* [Berger et al., 2007]. Several algorithms have been proposed to construct max-trees; these can be divided in two categories [Moschini et al., 2015a]: *bottom-up flooding methods* and *top-down merging methods*. Bottom-up flooding methods (examples include Salembier et al. [1998], Wilkinson [2011]) start with the root node (which has the lowest intensity) and construct the tree by traversing connected components at higher intensities in a depth-first manner. They often rely on a priority queue to keep track of the hierarchy. Top-down merging methods (an example includes Berger et al. [2007]) start with sorted (by grey value in a descending manner) singleton nodes and merge these into subtrees and

eventually complete the whole tree, often using the union-find algorithm.

Moschini et al. [2015a] introduce a parallel algorithm for max-tree construction, termed the diplomatic algorithm, as it uses both a top-down merging and a bottom-up flooding phase. The first phase uses a bottom-up flooding algorithm, which is simple to parallelise using spatial partitioning, to construct a so-called *pilot max-tree* of a quantised version of the image. The pilot max-tree contains a number of grey levels equal to the number of threads available and thus partitions the image based on grey levels. This partition is used to parallelise the algorithm of Berger et al. [2007] to produce the final *refined* max-tree. This algorithm was used to do most of the work for this thesis.

3.2 Mask-based connectivity

Ouzounis and Wilkinson [2007] present a new type of second-generation connectivity termed *mask-based connectivity*. The two traditional types of second-generation connectivity are *clustering-* and *contraction-based* connectivity, which add or remove members from a certain connectivity class $\mathcal{C}$ (for example, as defined by 4-connectivity in 2D), depending on some structural operator. Members of a connectivity class are called *connected sets*; connected sets of maximal extent are referred to as connected components.

Intuitively, clustering-based connectivity can be seen as merging connected components if their distance is small, while contraction-based connectivity splits connected components when they contain narrow, elongated structures. Smallness and narrowness depend on the size of the structural operator used. Formally, a clustering-based connectivity class $\mathcal{C}^\psi$, where ψ is a clustering, is defined as

$$\mathcal{C}^\psi = \{X \in \mathcal{P}(E) \mid \psi(X) \in \mathcal{C}\} \tag{3.6}$$

where E is the non-empty universal set and $\mathcal{P}(E)$ denotes all subsets of E. Clusterings are a class of increasing and extensive structural operators, including dilations and closings. A contraction-based connectivity class $\mathcal{C}^\varphi$ is instead based on a contraction φ, which belongs to a class of increasing, anti-extensive structural operators (such as openings). They are defined as

$$\mathcal{C}^\varphi = \{\varnothing\} \cap \mathcal{S} \cap \{X \in \mathcal{C} \mid \varphi(X) = X\} \tag{3.7}$$

where $\mathcal{S}$ is the set of all singleton sets in $\mathcal{P}(E)$.

Ouzounis and Wilkinson [2007] argue that the above framework is too limited, since the dependency on the properties of the structural operator causes many useful operators to fail to produce a valid connectivity class. Examples of such operators include alternating-sequential filters (which alternate a closing and an opening) and directional Minkowski additions (which use adaptive structural elements, based on the dominant direction of elongation). Mask-based connectivity solves this problem by eliminating the dependency on the structural operator, instead relying on a *connectivity mask* $M \subseteq E$ to define $\mathcal{C}$. This results in the following definition of a mask-based connectivity class $\mathcal{C}^M$

$$\mathcal{C}^M = \{\varnothing\} \cap \mathcal{S} \cap \{A \subseteq E \mid \exists_{x \in E} A \subseteq \Gamma_x(M)\} \tag{3.8}$$

where $\Gamma_x(M)$ is a connectivity opening of $\mathcal{C}$, extracting the connected component containing $e \in E$. $\mathcal{C}^M$ can be shown to be a valid connectivity class. It can be used to model clustering- and

contraction-based connectivity by taking $M = \psi(X)$ or $M = \varphi(X)$. There are no restrictions on how the mask M is created; any operator can be used or even an image of the same scene but acquired in a way different from the original image X.

Connectivity can be further generalised, as is done by Ouzounis and Wilkinson [2011], who describe so-called hyperconnectivity based on k-flat zones. A k-flat zone $F_{h,k}$ is similar to a flat zone, but the grey levels of its pixels are allowed to fluctuate between k and $h - k$. While no existing type of higher-order connectivity is used in this thesis, the novel mask-like technique discussed in chapter 4 is inspired by mask-based connectivity.

3.3 Discussion

Max-trees are a highly flexible image and volume representation, which allows attribute filters to be applied to connected/peak components and is therefore a building block for more sophisticated algorithms in image processing and related fields. Mask-based-connectivity extends this even further by allowing enhancement of existing connectivity classes. Fast algorithms like Moschini et al. [2015a]'s diplomatic algorithm allow these to be applied to practical problems requiring large volumes to be processed. All these properties make them an attractive representation for many image processing tasks, including the MT source finder.

4 Adaptive smoothing using mask-like connectivity

Most source finders rely on smoothing techniques to improve the signal-to-noise ratio of the input. Teeninga et al. [2013] explore the possibility of improving the performance of the MT objects method by applying a smoothing filter prior to the segmentation. As mentioned in section 1.1, this introduces a problem, as smoothing creates correlations between the noise pixels in the image, rendering the χ^2-test unusable. Determining the decision boundary using Monte Carlo simulations is less than ideal, as this is computationally expensive and needs to be repeated every time the filter is changed. Also, it makes it impossible to use adaptive filters, as they may behave very different on pure-noise images, compared to images containing a signal.

Therefore, using a form of mask-based connectivity to allow the segmentation to be enhanced by preprocessing the image, while at the same time not affecting the statistical model, is proposed here. Using mask-based connectivity as described by Ouzounis and Wilkinson [2007], would allow the connectivity of the image to be changed without restriction. The χ^2-test would still be usable, as the original intensity values corresponding to a peak component can be used.

Mask-based connectivity can be implemented elegantly using the algorithm presented by Wilkinson [2011], although some extra work would be required to combine it with the diplomatic max-tree construction algorithm of Moschini et al. [2015a]. An alternative to true mask-based connectivity is to build a max-tree of a mask image (likely created by applying some combinations of filters on the original image), ignoring the original image, but using the original intensity values to compute attributes of the peak components of the mask image's max-tree. The suggested name for this technique is *mask-like connectivity*. Since the intensity values used to compute the power remain unaltered, MT objects' χ^2-test still applies. Originally, the power(P) was defined as follows

$$\text{power}(P) = \sum_{x \in P} (f(x) - f(\text{parent}(P)))^2 \tag{4.1}$$

where P is a peak component and f is the image. With mask-like connectivity, the power computation is now defined as

$$\text{power}(P) = \sum_{x \in P} (f(x) - g(\text{parent}(P)))^2 \tag{4.2}$$

where g is the mask image. Since g appears in eq. (4.2), g cannot be any arbitrary transformation of f. Instead, the intensity values of g should be comparable to those of f. This is the case when g is a smoothed version of f.

In order to verify that the above approach works, some experiments were executed on an image used by Teeninga et al. [2013] (optical data). This image is shown in fig. 4.1, while the resulting object mask when applying the MT objects algorithm (described in section 1.1) is shown in fig. 4.3.

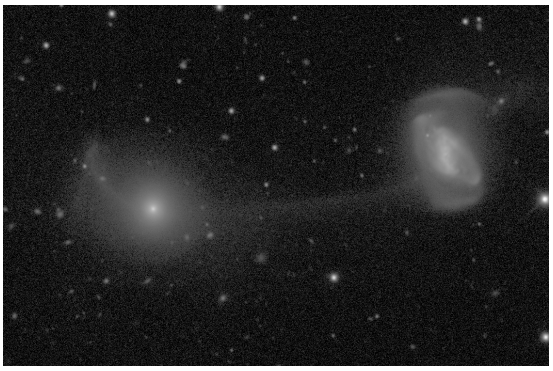

Figure 4.1: The input image for figs. 4.2 to 4.4. The visualised intensities follow a logarithmic scale.

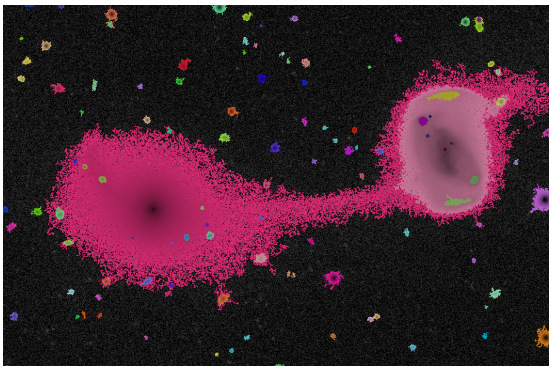

Figure 4.2: The object mask recovered when applying the mask-like approach with a Gaussian-smoothed max-tree mask.

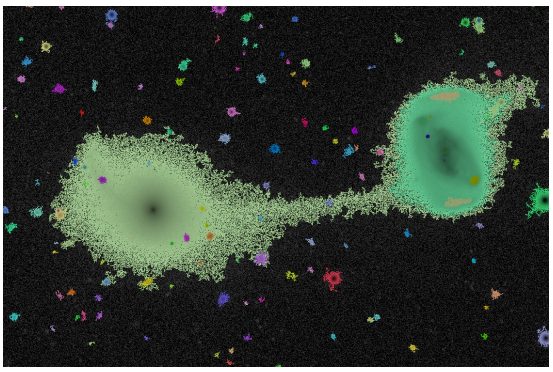

Figure 4.3: The object mask recovered using the χ^2-test, without any smoothing [Teeninga et al., 2015b, test 1].

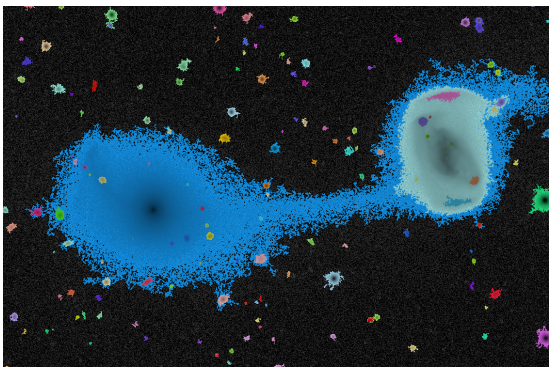

Figure 4.4: The object mask recovered using a simulated decision boundary, after Gaussian smoothing [Teeninga et al., 2015b, test 4].

Object identifiers are mapped to random colours. The parameters used are $g = 4.71$, $\lambda = 0.5$, and $\alpha = 10^{-6}$; no smoothing was used. Judging visually, the boundaries of the binary system object do not correspond too well to the true boundaries, suggesting smoothing the input image may be advantageous, to reduce the influence of noise on the max-tree structure. The results after applying Gaussian smoothing with $\sigma_G = 1/\sqrt{2 \ln 2} \approx 0.85$ are shown in figs. 4.2 and 4.4, using the mask-like approach and a simulated decision boundary, respectively. This Gaussian filter is practically identical to the smoothing filter applied by Source extractor [Teeninga et al., 2013, sec. 3.1], since its full width at half maximum (the distance between the two points at half maximum) is two. The results are very similar, indicating that the mask-like approach is a viable alternative to the simulated decision boundaries, although there are some faint differences. In both cases, the boundaries of the binary systems have a more natural shape than the result of fig. 4.3. The mask-like approach generally finds fewer objects than the simulated decision boundary (426 compared to 609 in this case), but the number of objects pixels found is similar (303154 compared to 318002), indicating that the missed objects are very small and may in fact be spurious.

4.1 Perona–Malik diffusion

Perona–Malik diffusion is an adaptive smoothing technique that may be useful in combination with the mask-like approach. Perona and Malik [1990] introduced this technique, also known as *anisotropic diffusion*, for building scale-spaces of images using a diffusion process. This scale space $I(x, y, t)$ for an image I is defined by the following second-order partial differential equation

$$\frac{\partial I(x, y, t)}{\partial t} = c(x, y, t)\, \Delta\, I + \nabla c \cdot \nabla I \tag{4.3}$$

where c is the conduction coefficient, Δ is the Laplace operator (divergence of gradient), and ∇ is the gradient (both the Laplacian and gradient are with respect to the spatial variables). The initial conditions $I(x, y, 0)$ are given by the original image. As noted by Perona and Malik [1990], using a variable conduction coefficient allows the process to be edge preserving. For example, by choosing

$$c(x, y, t) = \frac{1}{1 + \frac{|\nabla I(x,y,t)|^2}{K^2}} \tag{4.4}$$

for some constant K, less blurring occurs when the magnitude of the gradient is large. When using a constant c, eq. (4.3) reduces to Gaussian blurring, with $\sigma_G = \sqrt{2ct_{\max}}$ [Hummel and Moniot, 1989].

Instead of having the degree of blurring be dependent on the magnitude of the gradient, it can also be made dependent on other properties, such as the image intensity. In the case of radio volumes, the signal-to-noise ratio is better when the intensity is higher, so it is sensible to do this as it allows reducing the influence of noise on the resulting max-tree. Edges may be less well preserved, but that is not the primary aim here, as the edges of the galaxies are usually not well defined anyway. An additional advantage is that the dependency on the gradient estimate, which may be unstable, is reduced. If the conduction coefficient is defined as

$$c(x, y, t) = \frac{1}{1 + \frac{I^2(x,y,t)}{\kappa^2 \sigma^2}} \tag{4.5}$$

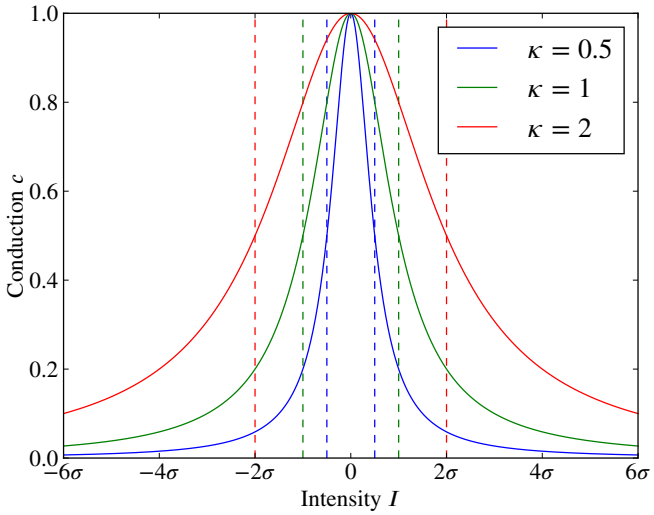

Figure 4.5: Visualisation of eq. (4.5) (intensity-based conduction coefficient). The solid lines represent the conduction function for different K, while the dashed lines indicate were the fall-off occurs.

for some user-defined parameter κ, most blurring occurs when $c(x, y, t) \geq 1/2$, which is the case when $I(x, y, t) \leq \kappa\sigma$. This means that it becomes difficult to detect objects below $\kappa\sigma$. Figure 4.5 visualises this function.

The above scheme was implemented using the discretisation described by Perona and Malik [1990], after applying the trivial extension to 3D, using a constant time step $\tau = 0.1$. The time step was fixed at a constant value, because varying the number of steps N is more or less equivalent to varying τ, since $t_{\max} = \tau N$. The discretisation is given by

$$I_{ij}^{n+1} = I_{ij}^{n} + \tau c_{ij}^{n} \left(\nabla_N I + \nabla_S I + \nabla_W I + \nabla_E I \right)_{ij}^{n} \tag{4.6}$$

where ∇_N is the finite difference operator in the north direction, ∇_S is the finite difference operator in the south direction, and so on. The discretisations of t, x, and y are n, i, and j, respectively. In order to stabilise the gradient approximation, a minimal degree of Gaussian smoothing with $\sigma_G = 0.85$ is applied before starting the diffusion process.

4.2 Results using intensity-driven diffusion

Two radio volumes used to demonstrate the effect of intensity-driven diffusion are shown in figs. 4.6 and 4.7, visualised with a grey-scale map. To ease the discussion, they are nicknamed the ‘ring’ and ‘pipe’ galaxy, based on their shape in the radio space. These noise-free volumes were generated using the CLEAN algorithm [Högbom, 1974]. The CLEAN algorithm is a deconvolution method that works as follows. It assumes the input is the sum of a number of point sources and noise, created by a known, ‘dirty’ point-spread function. Iteratively, it identifies the maximum in the image and subtracts a fraction of this maximum, convolved with the dirty point-spread function.

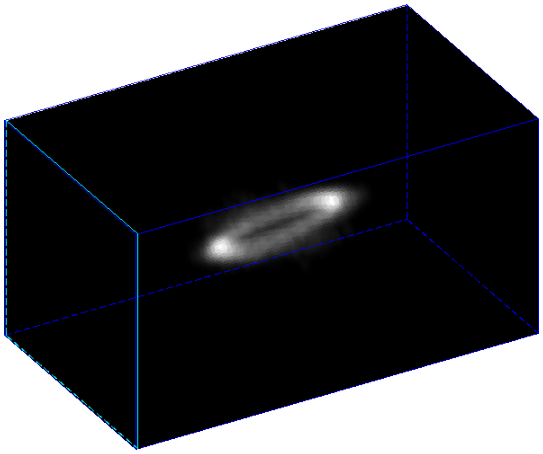

Figure 4.6: Radio volume of the ‘ring’ galaxy.

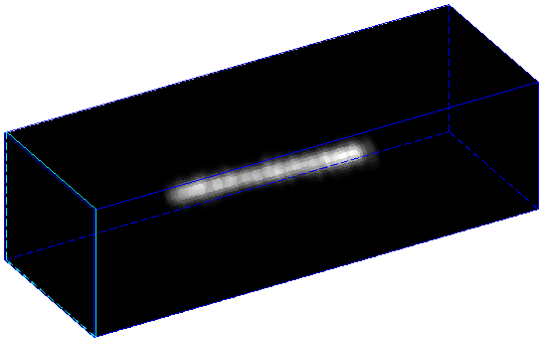

Figure 4.7: Radio volume of the ‘pipe’ galaxy.

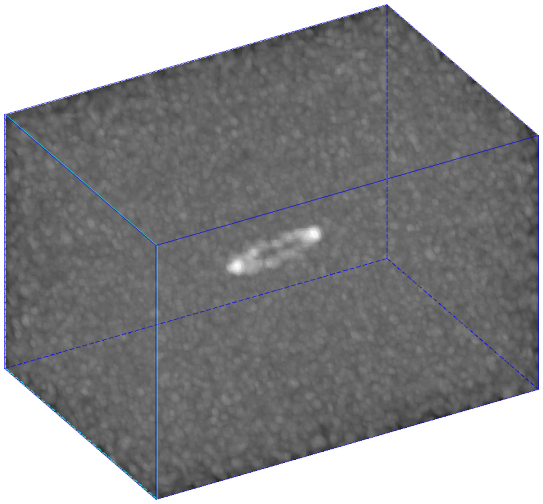

Figure 4.8: Volume of the ring galaxy after adding noise.

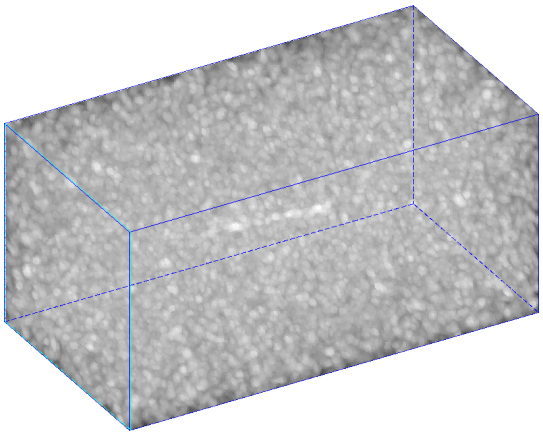

Figure 4.9: Volume of the pipe galaxy after adding noise.

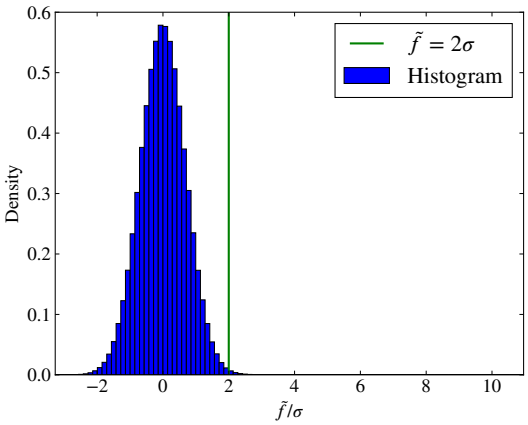

Figure 4.10: 100-bin histogram of fig. 4.8, after applying Gaussian smoothing with $\sigma_G \approx 0.85$ (smoothed values denoted $\tilde{f}$). The position of a possible K is shown in green.

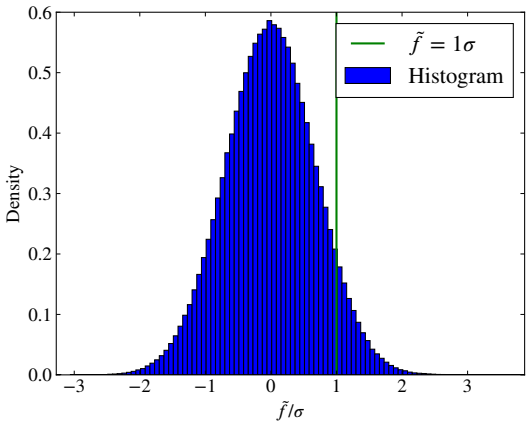

Figure 4.11: 100-bin histogram of fig. 4.9, after applying Gaussian smoothing with $\sigma_G \approx 0.85$ (smoothed values denoted $\tilde{f}$). The position of a possible K is shown in green.

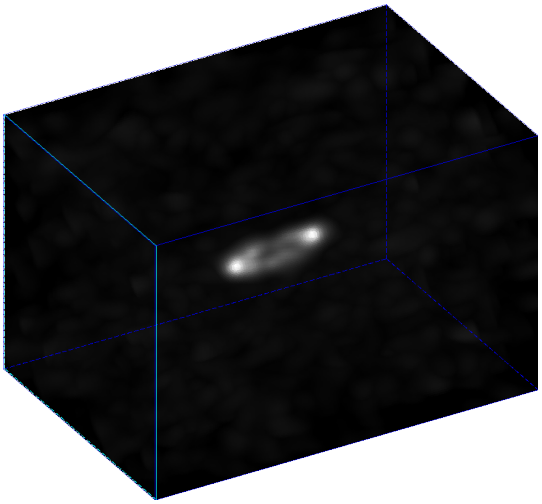

Figure 4.12: Noisy volume of the ring galaxy after applying intensity-driven diffusion with $\kappa = 2.0$ and $t_{\max} = 4.5$.

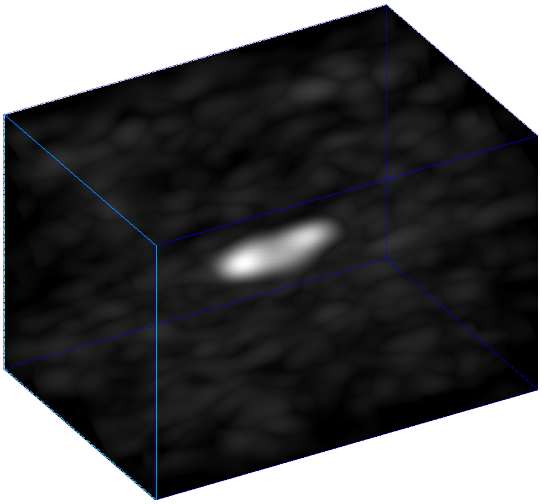

Figure 4.13: Noisy volume of the ring galaxy after applying Gaussian smoothing with $\sigma_G = 3.0$.

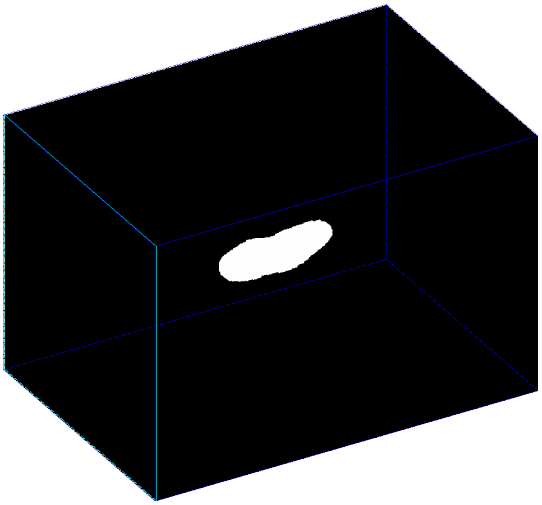

Figure 4.14: Segmentation recovered using fig. 4.12 as max-tree mask, using move-up factor $\lambda = 0.1$.

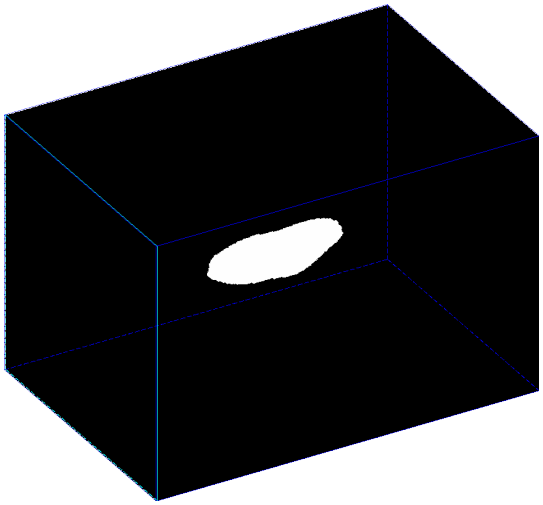

Figure 4.15: Segmentation recovered using fig. 4.13 as max-tree mask, using move-up factor $\lambda = 0.1$.

The iterative part terminates when the maximum of the image drops below a certain threshold. Then, a cleaned image can be reconstructed by convolving the removed maxima with a clean point-spread function (a Gaussian). Since the CLEAN algorithm assumes that the image only contains point sources, it is not guaranteed to work well on images containing extended sources.

Zero-mean noise with a standard deviation of $\sigma = 1.6 \times 10^{-3}$ was added to these volumes resulting in the volumes shown in figs. 4.8 and 4.9. The noise was not generated by a computer, but was instead obtained from an actual radio observation; it is the residual of a CLEAN operation. Radio data is sampled in the Fourier domain using an array of radio telescopes and then an inverse Fourier transform is used to get the representation used here; the inverse Fourier transform introduces noise correlations. Note that this 'pure' noise volume is not guaranteed to be completely free of sources (since the residual of a CLEAN operation), but it is the best thing available right now. Judging visually, the signal-to-noise ratio (SNR) is of the noisy ring galaxy volume is better than that of the noisy pipe galaxy volume, as the object is still easily distinguishable in the former case, but barely in the second case.

In all of the following results, the negative values of the max-tree masks were set to zero prior to segmentation. A significance level of $\alpha = 10^{-6}$ is used. The segmentation algorithm uses the flux density attribute described in section 5.3 and the move-up factor is used as in section 5.5.

Figure 4.12 shows the noisy ring galaxy volume after applying intensity-driven diffusion. Since the signal-to-noise ratio is good in this case, a relatively high value $\kappa = 2.0$ can be used. The histogram in fig. 4.10 supports this, as it seems to have a rather long tail past $\tilde{f} \geq 2\sigma$, which most likely represent the significant details (that is, the object) in the volume. After $t_{\max} = 4.5$ most noise seemed to have disappeared, while the details of the object have mostly been preserved. Applying smoothing with the corresponding Gaussian kernel with $\sigma_G = \sqrt{2 \times 4.5} = 3$ does not preserve the details of the object, as is shown in fig. 4.13, and also greatly lowers the intensity of

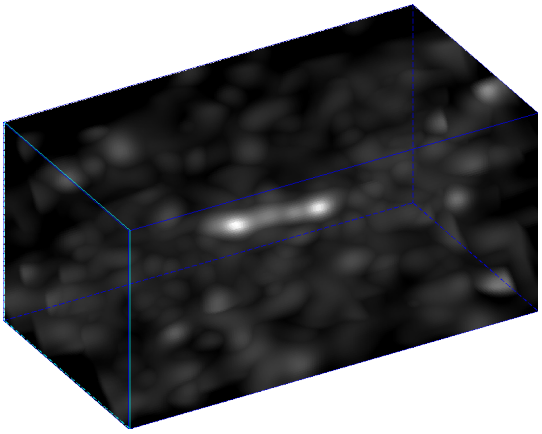

Figure 4.16: Noisy volume of the pipe galaxy after applying intensity-driven diffusion with $K = 1.0$ and $t_{\max} = 8.0$.

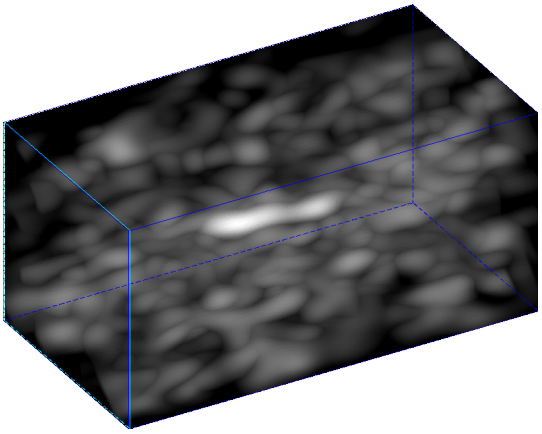

Figure 4.17: Noisy volume of the pipe galaxy after applying Gaussian smoothing with $\sigma_G = 3.0$.

the object with respect to the background. For both max-tree masks, smoothing is made twice as strong in the direction of the velocity axis, as radio sources are better resolved along the spectral axis.

The segmentation corresponding to the max-tree mask created by intensity-driven diffusion is shown in fig. 4.14, using $\kappa = 2.0$ and $\lambda = 0.1$. The segmentation recovered using the max-tree created by Gaussian smoothing is shown in fig. 4.15, which is almost identical to what is shown in fig. 4.14. In this case, changing the method of smoothing has no significant effect on the result.

The result of applying intensity-driven diffusion on the noisy pipe galaxy volume is shown in fig. 4.16. Since the signal-to-noise ratio is worse here, a lower value of $\kappa = 1.0$ was used, so there is less risk of smoothing important details away. As is evident from fig. 4.11, there is no particularly pronounced tail of values that are clearly significant, so a lower κ value is warranted. The time $t_{\max}$ was extended, to allow more of the noise to be smoothed out. Nevertheless, the resulting max-tree mask retains some visible noise, as the signal-to-noise ratio is bad. For comparison, a Gaussian-smoothed version with $\sigma_G = 3.0$ is shown in fig. 4.17; using $\sigma_G = \sqrt{2t_{\max}} = 4.0$ is clearly too aggressive. The segmentations recovered using these max-tree masks are shown in figs. 4.18 and 4.19. Similarly to the segmentations of fig. 4.8, the objects are detected almost perfectly.

In the examples considered here, the adaptive, diffusion-based smoothing methods perform similar to simple Gaussian smoothing. This is not surprising, since the volumes contain only a single object, and as such the smoothing parameters can be optimised on a per-object basis. The hypothesis is that using adaptive smoothing performs better for segmentation on more realistic radio volumes containing several objects with varying levels of brightness. The results in chapter 6 show that there is indeed a small improvement in the segmentation result when using adaptive smoothing.

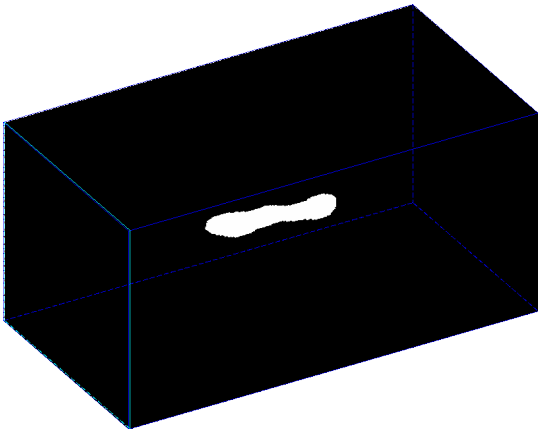

Figure 4.18: Segmentation recovered using fig. 4.16 as max-tree mask, using move-up factor $\lambda = 0.1$.

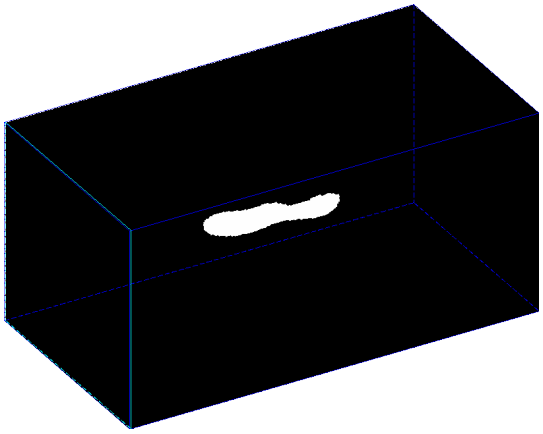

Figure 4.19: Segmentation recovered using fig. 4.17 as max-tree mask, using move-up factor $\lambda = 0.1$.

4.3 Results using gradient-magnitude-driven diffusion

The aim of this section is to assess the performance of the original, gradient-magnitude-driven Perona–Malik diffusion, compared to the intensity-driven diffusion discussed in section 4.2. One issue is the value of K to use in eq. (4.4). Choosing a value for K in the case of intensity-driven diffusion (eq. (4.5)) is intuitive, as one can select a factor κ of σ below which the noisy signal is smoothed most aggressively. In the case where the degree of smoothing is based on the gradient magnitude, choosing a value for K is less intuitive. One approach employed by Canny [1986], Perona and Malik [1990] is as follows: the magnitudes of the gradient of a volume are computed (here, after the initial gradient-stabilising smoothing step) and it is assumed that some percentage (say 80 to 90%) of the lower of the magnitudes are due to noise. In that case, it is sensible to use an estimate of the nth percentile as K. Here, a similar approach is taken, by simply looking at the histograms, plotted in figs. 4.20 and 4.21. Only the initial value is used here; it is not updated each iteration as Perona and Malik [1990] do instead.

The resulting max-tree mask after applying gradient-magnitude-driven diffusion with $K = 1.5 \times 10^{-3}$ and $t_{\max} = 4.5$ is shown in fig. 4.22. The result is similar to fig. 4.12 (intensity-driven diffusion max-tree mask), except that the interior of the object is smoothed as well, which may actually be desirable, as any details in the object are not important for the segmentation. Compared to fig. 4.13 (Gaussian smoothed max-tree mask), the intensity of the object is better preserved. The resulting segmentation when using this max-tree mask is very similar to fig. 4.14. Using gradient-magnitude-driven diffusion does not work as well when the SNR is lower, as shown in fig. 4.23, probably because the gradient magnitude of noise-induced edges is similar to that of true edges. When the SNR is good, gradient-magnitude-driven seems to do better at removing noisy peaks than intensity-driven diffusion does.

An alternative to using eq. (4.4) would be to determine the distribution of the noisy gradient magnitude analytically and use the probability density to steer the degree of smoothing, but that

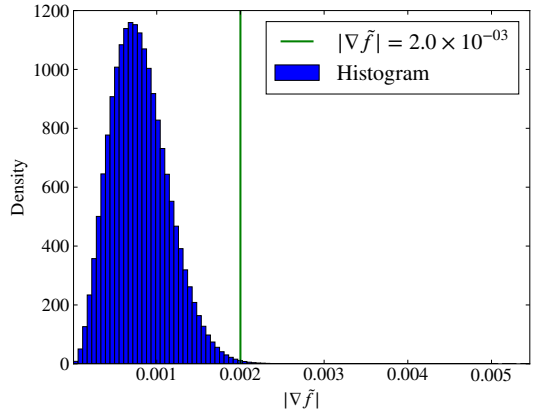

Figure 4.20: 100-bin histogram of the gradient magnitudes of fig. 4.8, after Gaussian smoothing with $\sigma_G \approx 0.85$ (smoothed values denoted $\tilde{f}$). The position of a possible K is shown in green.

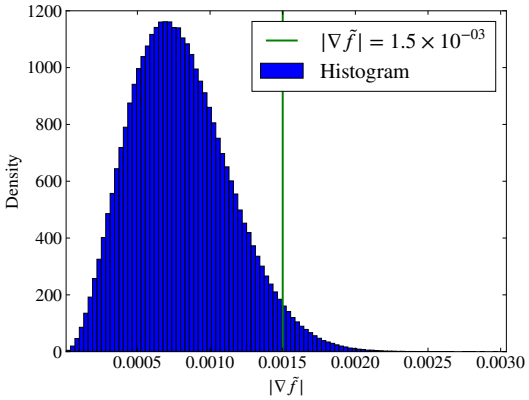

Figure 4.21: 100-bin histogram of the gradient magnitudes of fig. 4.9, after Gaussian smoothing with $\sigma_G \approx 0.85$ (smoothed values denoted $\tilde{f}$). The position of a possible K is shown in green.

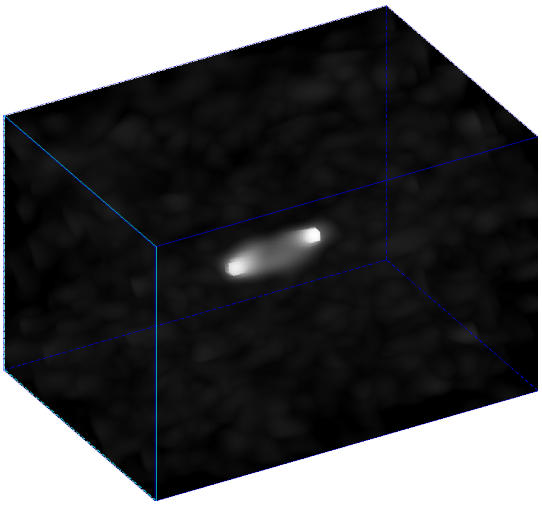

Figure 4.22: Noisy volume of the ring galaxy after applying gradient-magnitude-driven diffusion with $t_{\max} = 4.5$ and $K = 1.5 \times 10^{-3}$.

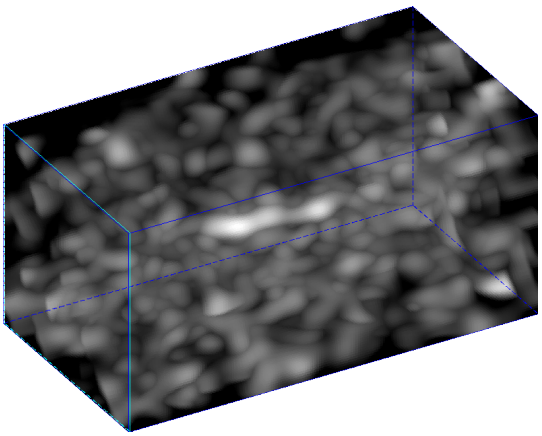

Figure 4.23: Noisy volume of the pipe after applying gradient-magnitude-driven diffusion with $t_{\max} = 4.5$ and $K = 2.0 \times 10^{-3}$.

would be difficult as the noise voxels in the radio datasets are not statistically independent.

4.4 Discussion

The mask-like approach to connectivity is a novel technique to combine pre-processing filters and the max-tree data structure that allows certain attributes to be computed from the original, unprocessed data. It was shown that this approach can be used to apply methods that require certain statistical properties to be preserved, in case they would be destroyed by the chosen pre-processing filter.

In this context, one promising technique is Perona–Malik diffusion. An alternative to the traditional, gradient-based version based on image intensity was proposed, which could be useful in case a higher image intensity indicates a better singal-to-noise ratio, that requires less smoothing. The performance of the MT source finder does not seem to be greatly affected when using Perona–Malik diffusion, rather than Gaussian smoothing, on a single source dataset. The differences are more evident in case several sources of varying signal-to-noise level are present, as mentioned in chapter 6, with gradient-magnitude-driven diffusion performing best.

5 Statistical models

Since the core of the MT source finder is a statistical test, to determine whether an excursion in brightness is significant, a sound statistical noise model is paramount. In the case of optical data, Teeninga et al. [2013] assume that the power attribute (as defined in section 1.1) follows a χ^2-distribution, when no pre-processing is applied to the data. Moschini et al. [2014] make the same assumption while studying radio volumes. This assumption allows the χ^2-test to be used to determine whether a max-tree node is significant.

Section 5.1 studies whether this assumption is warranted. Also included is an alternative statistical model for the power attribute (in section 5.2) and a possible model for the flux density attribute (in section 5.3). Finally, this chapter studies the problem of oversized objects, which is inevitable when combining an increasing attribute with a statistical test, in section 5.5.

5.1 Power attribute modelled using a chi-squared distribution

The χ^2-test applies under the following conditions. Under the null hypothesis, the statistic must be a sum of squared variables that are independent, normally-distributed, of zero mean, and of unit variance. The power divided by the volume variance is a sum of squared variables; to verify whether the other assumptions are met, power histograms were computed for max-trees constructed from pure noise volumes. One histogram was computed for each max-tree node volume. Two noise volumes were used, one containing radio noise acquired from a radio observations (from now on referred to as radio noise) and one containing independent Gaussian noise. In both cases, the mean of the noise is zero, while $\sigma = 1.6 \times 10^{-3}$. The radio noise volume contains correlations, as is demonstrated in fig. 5.1, a scatter plot of the auto-correlation of the radio noise cube with respect to voxel distance. This could cause problems when trying to apply the χ^2-test. Note that this radio noise volume is the same as the one used in chapter 4.

The same experiments were repeated using the mask-like approach, creating a max-tree mask by smoothing the input with Gaussian kernels with $\sigma_G \in \{1, 2\}$, in order to get insight in the effects of the mask-like approach. As in chapter 4, along the spectral axis, the smoothing was made twice as strong.

The results show that the current model does not fit the data very well. When not smoothing the max-tree mask, the histograms in figs. 5.2 and 5.3 show that the χ^2-distribution overestimates the power of the max-tree node for volume (degrees of freedom) $V = 100$. In fact, this happens consistently and the problems gets worse as the volume increases, as is shown in fig. 5.6. Figure 5.6 shows the estimated mean power $\hat{\mu}_{\text{power}}(V)$ per unit volume for max-tree nodes of different volume. If the statistic follows a χ^2-distribution, it follows that $\mu_{\text{power}}(V) = V$, but this is clearly not the case here, as the empirical value is lower. This may be because the parent intensity is subtracted when computing the power.

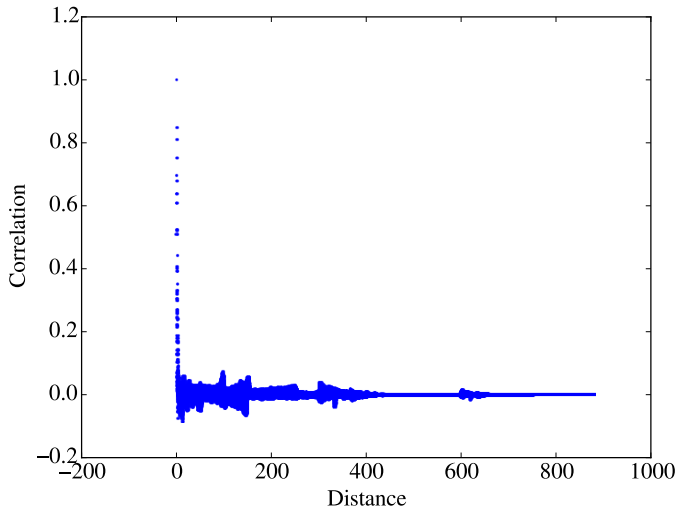

Figure 5.1: Correlation of the radio noise volume with respect to voxel distance, clearly showing correlations at non-zero distance are present.

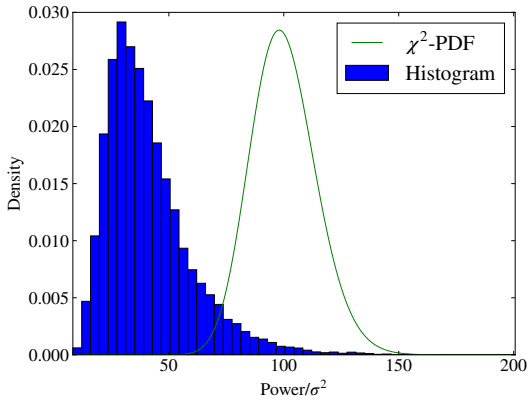

Figure 5.2: Power histogram (50 bins) of max-tree nodes with $V = 100$. The max-tree was computed from a radio noise volume. The histogram is shifted compared to the hypothetical distribution.

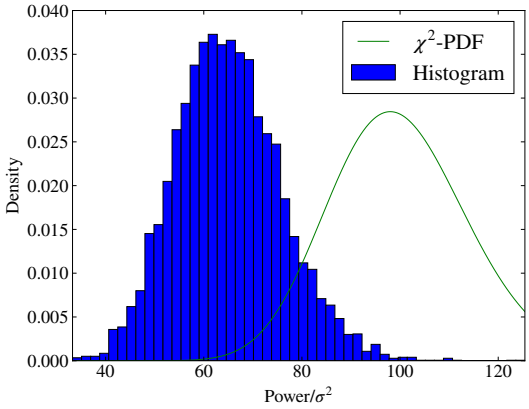

Figure 5.3: Power histogram (50 bins) of max-tree nodes with $V = 100$. The max-tree was computed from an independent Gaussian noise volume. The histogram is shifted compared to the hypothetical distribution.

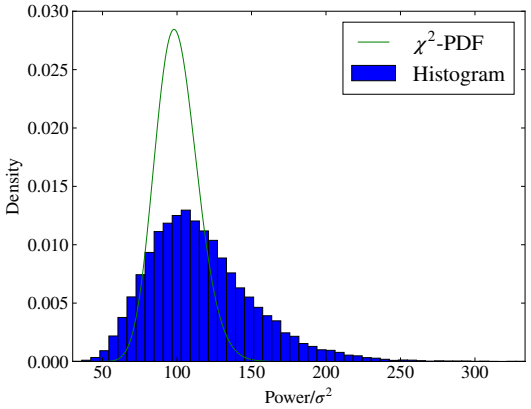

Figure 5.4: Power histogram (50 bins) of max-tree nodes with $V = 100$. The max-tree was computed from a radio noise volume, using a smoothed max-tree mask, with $\sigma_G = 1.0$. The histogram is wider than the hypothetical distribution.

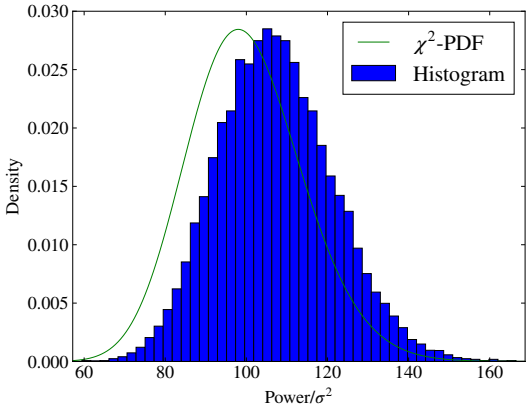

Figure 5.5: Power histogram (50 bins) of max-tree nodes with $V = 100$. The max-tree was computed from an independent Gaussian noise volume, using a smoothed max-tree mask, with $\sigma_G = 1.0$. The histogram and distribution are close.

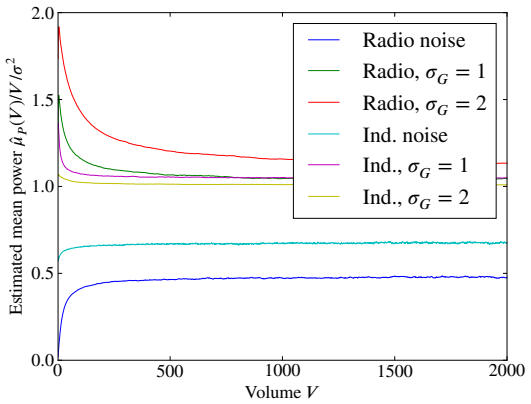

Figure 5.6: Mean power estimates with respect to volume, computed for max-tree nodes of pure noise volumes.

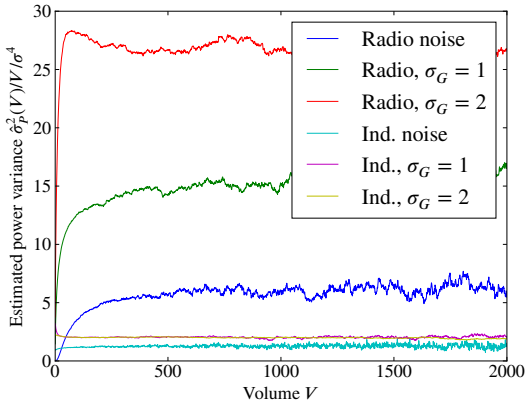

Figure 5.7: Power variance estimates with respect to volume, computed for max-tree nodes of pure noise volumes.

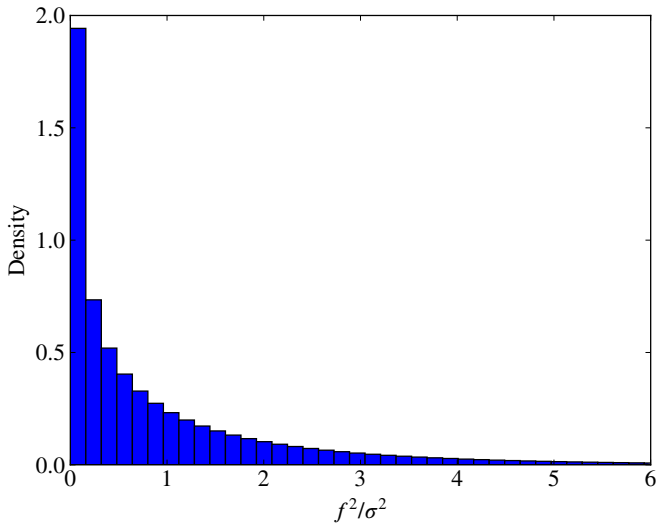

Figure 5.8: Distribution of squared intensities of the radio noise volume, visualised using a 200-bin histogram. This resembles an exponential distribution, although $\hat{\mu} \approx 2\hat{\sigma}/3$ instead of the expected $\mu = \sigma$.

If the mask-based approach is used, the subtraction of parent intensities is less of a problem, as is visible in figs. 5.4 and 5.5. This is because values below the parent mask intensity are allowed to contribute to the power. Still, the expected value of the χ^2-distribution is slightly different from the mean power of the max-tree nodes, as is visible in fig. 5.6. This is possibly caused by the fact that max-tree nodes are constructed to contain intensities above a certain threshold and thus the mean is larger than expected, although the effect is limited in the case of independent Gaussian noise. Additionally, in the radio noise case, the power variance is much larger than the expected value $\sigma^2_{\text{power}} = 2k$, as is shown in fig. 5.7 (which shows the estimated power variance per unit volume). This is likely caused by the correlations present in the radio noise volume. In the case of independent Gaussian noise, the variance seems to converge to the expected value $2k$, when max-tree mask smoothing is applied.

Combining the insights of fig. 5.6 and fig. 5.7 suggests that the power does not follow a χ^2-distribution, except maybe in the case of independent Gaussian noise with a smoothed max-tree mask. The latter observation is not very useful here, as the true radio noise is not independent, although it does show why the noise model works in the optical case, where the noise is independent. Also, it does not seem to be possible to correct for dependency effects by tweaking the number of degrees of freedom, since clearly $\mu_P(k) \neq \sigma^2_P(k)/2$.

5.2 Power attribute modelled using a Gamma distribution

As section 5.1 shows, the χ^2-distribution does not model the power attribute very well. Nevertheless, as figs. 5.6 and 5.7 show, it does seem that $\mu_P \propto V$ and $\sigma^2_P \propto V$ for larger V, which suggests an appropriate distribution could be a Γ-distribution. Γ-distributions are parametrised by *shape*

k and a *scale* θ, such that $\mu = k\theta$ and $\sigma^2 = k\theta^2$. Also, note that the Γ-distribution generalises the χ^2-distribution such that $\chi^2(k) \sim \Gamma(k/2, 2)$. Possibly, a model can be constructed with $k = f(V)$ and a constant θ.

Another reason to consider the Γ-distribution has to do with the fact that, if $k \in \mathbb{N}_1$, then the distribution describes the distribution of k independent exponential distributions with rate parameter θ. The distribution of the power of individual voxels is plotted in fig. 5.8, which resembles an exponential distribution. This is not a standard exponential distribution, as $\hat{\mu} \approx 2\hat{\sigma}/3$ instead of $\mu = \sigma$. Also, the power of neighbouring voxels is probably not independent, but perhaps the distribution of the sum is still similar to a Γ-distribution with some tweaks to the parameters.

Before trying to fit a function to $k_P(V)$, redefine the power attribute as:

$$\text{power}_{\text{alt}}(P) = \sum_{x \in P} (f(x) - g(\text{ancestor}(P)))^2 = \sum_{x \in P} f^2(x) \tag{5.1}$$

Replacing $g(\text{parent}(P))$ with $g(\text{ancestor}(P))$ in eq. (5.1) is done because including $g(\text{parent}(P))$ is dubious, as it does seem to have an intuitive meaning (not even when $f = g$). More sensible is to use $g(\text{ancestor}(P))$, where $\text{ancestor}(P)$ is the last *significant* ancestor of P and thus represents the containing object. This approach is also taken by Teeninga et al. [2015b], in their alternative definition of power. The radio volumes here are assumed not to contain any object nesting and therefore $g(\text{ancestor}(P)) = 0$.

What remains is to estimate the parameters of the Γ-distribution. This can be done using the maximum likelihood estimation (MLE) technique. The probability density function (PDF) of a Γ-distribution with shape parameter k and scale parameter θ is

$$p(x|k, \theta) = \frac{x^{k-1}}{\Gamma(k)\theta^k} e^{-\frac{x}{\theta}} \tag{5.2}$$

where the Γ-function is given by $\Gamma(t) = \int_0^\infty x^{t-1} e^{-x} dx \underset{t \in \mathbb{N}_1}{=} (t-1)!$. Following Minka [2002], the log-likelihood function is given by

$$\ln p(\vec{x}|k, \theta) = n(k-1)\overline{\ln x} - n \ln \Gamma(k) - nk \ln \theta - n\bar{x}/\theta \tag{5.3}$$

where $\vec{x}$ is the vector of samples and an overline indicates an average. It is not difficult to see that this function is maximised by $\theta = \bar{x}/k$, considering $d \ln p(\vec{x}|k, \theta)/d\theta = n\bar{x}/\theta^2 - nk/\theta$. So, the MLE is $\hat{\theta} = \bar{x}/\hat{k}$. Estimating k is more involved. Substituting $\theta = \bar{x}/k$ in eq. (5.3) gives the following:

$$\ln p(\vec{x}|k, \hat{\theta}) = n(k-1)\overline{\ln x} - n \ln \Gamma(k) - nk \ln \bar{x} + nk \ln k - nk \tag{5.4}$$

Apparently, one cannot maximise this function analytically. As explained by Minka [2002], a maximum can be found by iteratively maximising a linear lower bound. Unfortunately, this approach may take hundreds of iterations to converge, depending on k. A much faster approach is to use a generalised variant of Newton's method [Minka, 2000, sec. 2]. While Minka [2000] does not give the exact deviation, the following update step can be derived

$$\frac{1}{\hat{k}_{n+1}} = \frac{1}{\hat{k}_n} + \frac{\overline{\ln x} - \ln \bar{x} + \ln \hat{k}_n - \Psi(\hat{k}_n)}{\hat{k}_n - \hat{k}_n^2 \Psi'(\hat{k}_n)} \tag{5.5}$$

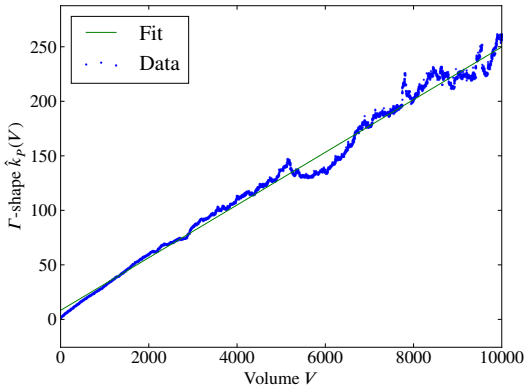

Figure 5.9: Shape estimates $\hat{k}$ of a possible Γ-distribution of the power of noise nodes, compared to node volume V. A line is fit to the data. The max-tree mask was smoothed using $\sigma_G = 3.0$.

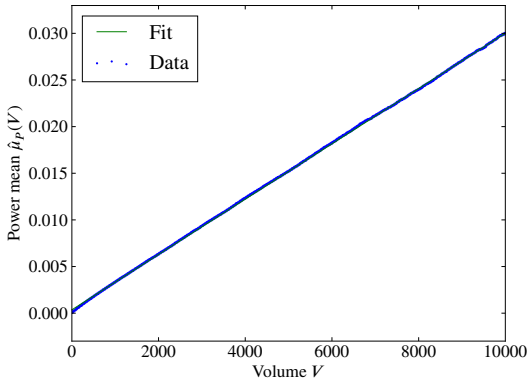

Figure 5.10: Mean power estimates $\hat{\mu}_P$ of noise nodes compared to node volume V. A line is fit to the data. The max-tree mask was smoothed using $\sigma_G = 3.0$.

where $\Psi(\hat{k}_n) = \Gamma(\hat{k}_n)/\Gamma(\hat{k}_n)$. An initial estimate $\hat{k}_0$ can be obtained as $\hat{k}_0 = 1/(2\ln\bar{x} - 2\overline{\ln x})$ [Minka, 2002, sec. 2].

Assuming the parameters of the power's distribution are a functions of volume, their values can be predicted by fitting a function to the values acquired by applying MLE on the max-tree nodes of a pure noise volume, when considering all nodes with the same volume separately. Figure 5.9 shows the Γ-shapes $\hat{k}_P$ acquired this way, while fig. 5.10 shows means acquired (which can be used to estimate the scale $\hat{\theta}_P$). Either function seems to represent a linear relation, which makes fitting a function to them using least-square-error estimation rather simple. For larger volumes, $\hat{k}_P(V)$ seems to deviate somewhat from linear; this may be because the number of samples used in the estimation is low (the larger the volume, the smaller the number of max-tree nodes). Another explanation is that the number of iterations used was too low. A constant number of ten iterations was used, which was assumed to be ample, as Minka [2002] indicates that four iterations is enough for converge for $k = 7.3$. This may have been overly optimistic, considering the range of $\hat{k}_P$ in fig. 5.9. Nevertheless, increasing the number of iterations was found to not make the relation appear more linear.

5.3 Flux density attribute

The power attribute was chosen by Teeninga et al. [2013] because it follows a χ^2-distribution when the noise is independent. It is entirely possible to use a different attribute with a statistical test, provided the distribution under the null hypothesis is known. Another possible statistic is the *flux density* or mean intensity of a node. This is similar to the power attribute, in the sense that it is also an aggregate of the intensities (flux) of a max-tree node, but is perhaps less sensitive to outliers as no squaring is involved (so the influence of outliers is lower). Additionally, overall

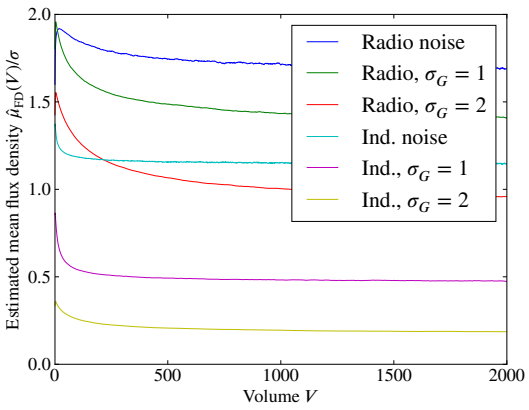

Figure 5.11: Mean flux density estimates with respect to volume, computed for max-tree nodes of pure noise volumes.

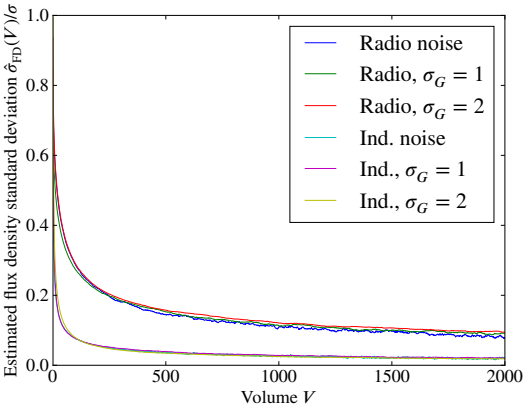

Figure 5.12: Flux density standard deviation estimates with respect to volume, computed for max-tree nodes of pure noise volumes.

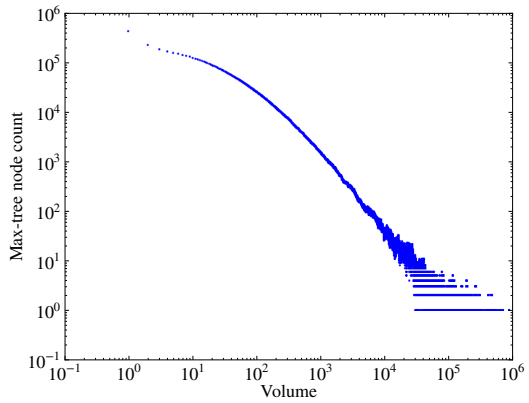

Figure 5.13: Number of nodes of a particular volume in the max-tree of the noise volume, constructed using a max-tree mask smoothed with $\sigma_G = 1.0$. When the volume goes above a certain threshold, say 10^4, the number of nodes drops to a very low value.

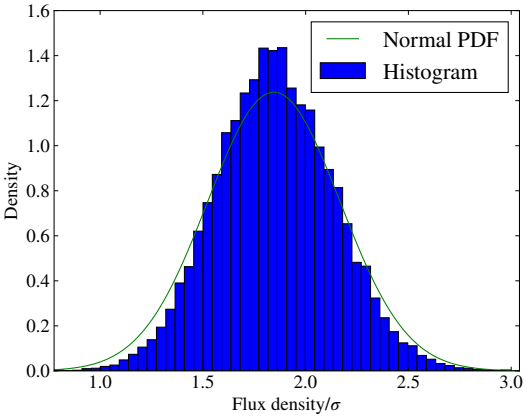

Figure 5.14: Flux density histogram (50 bins) of max-tree nodes with $V = 100$. The max-tree was computed from a radio noise volume, using a Gaussian-smoothed max-tree mask with $\sigma_G = 1.0$. The distribution matches the histogram quite well.

noise sensitivity is reduced as positive and negative noisy values get a chance of cancelling each other. The flux density F_{density} as investigated here is defined as

$$F_{\text{density}}(P) = \frac{F_{\text{total}}(P)}{V(P)} = \frac{1}{V(P)} \sum_{x \in P} f(x) \tag{5.6}$$

where P is a peak component of the max-tree mask g, $V(P)$ its volume, and f is the original volume. Note that the intensity of the parent node is not considered here, which makes the flux density in this formulation unsuited to find nested objects. A solution to this problem is left as future work, but it is likely easily corrected.

Determining the exact distribution of the mean intensity attribute is probably difficult, due to the correlation between noise voxels, and the nature of max-tree nodes, containing only voxels whose intensity is above a certain threshold. Assuming the noise voxels are independent enough, the mean intensity should follow a normal distribution with parameters μ_{FD} (mean flux density) and σ_{FD} (standard deviation). Estimates for these values as a function of volume are included in figs. 5.11 and 5.12, which were computed in the same way as the mean power and power variance in section 5.1. Once again, the differences in means between the radio and independent cases shows that the radio noise is correlated. While σ_{FD} seems to get a stable estimate for the two types of noise (radio versus independent Gaussian in the latter case it seems that $\sigma_{\text{FD}} = \sigma/\sqrt{V}$), whether mask smoothing is used or not, the same is not true for μ_{FD}. The mean flux density $\mu_{\text{FD}}(V)$ decreases for larger V when smoothing is introduced, as is also the case for the power attribute.

Assuming the K-parameter has been set appropriately, the noise in a volume smoothed by intensity-driven diffusion will be smoothed approximately as aggressively as it would be by Gaussian smoothing with $\sigma_G = \sqrt{2t_{\max}}$. Therefore, the estimates in figs. 5.11 and 5.12 can be used to parametrise the following statistical model. Under the null hypothesis, that is, the peak component to be considered is a noise artefact, the flux density follows a normal distribution with $\mu_{\text{FD},\sigma_G}(V)$ and $\sigma_{\text{FD}}(V)$. The null hypothesis is rejected when F_{density} is significantly larger than predicted by the normal distribution.

In order not to have to redo the entire simulation used to generate the mean and standard deviation estimates for radio noise in figs. 5.11 and 5.12 each time segmentation is performed, rational functions were fit to these estimates for several values of σ_G. For the experiments considered in chapter 6, $\sigma_G \in \{1, 2, 3, 4\}$ and the case of no smoothing were used. In case no approximation is available for the exact σ_G-value, the fit for the largest σ'_G for which $\sigma'_G < \sigma_G$ is true can be used as an upper bound for μ_{FD,σ_G}. A rational function is defined as

$$f(x) = \frac{\sum_{i=0}^{d_p} p_i x^i}{x^{d_q} + \sum_{i=0}^{d_q - 1} q_i x^i} \tag{5.7}$$

where $d_p \geq 0$ and $d_q \geq 0$ are the numerator and denominator degree, respectively, while p_i and q_i are the numerator and denominator coefficients. Note that, when $d_q = 0$, $f(x)$ degenerates to a polynomial. Also, $q_{d_q} = 1$, in order to guarantee that each set of coefficients produces a unique fit. Equation (5.7) was fit to the data acquired by simulation using the Levenberg–Marquardt algorithm for non-linear least-squares regression, as implemented in

Scipy's[1] `scipy.optimize.curve_fit` function. All estimates for which $V \in [1, 10^4]$ were used, which is the typical range of volumes of the objects considered here. Additionally, for larger volumes the number of max-tree nodes is very low, as shown in fig. 5.13 (the numbers drop below ten). This makes it hard to make good estimates. Appropriate initial values for the Levenberg–Marquardt algorithm can be acquired by picking $d_p + d_q + 1$ points from the data at random (denote them (x_i, y_i)) and fit these to the model by solving the following system of linear equations, retrieved by rewriting the model, for the unknown p_j and q_j, which will be the initial values:

$$y_i = p_0 + p_1 x_i + \cdots + p_{d_p} x^{d_p} - q_1 x_i y_i - \cdots - q_{d_q} x_i^{d_q} y_i \tag{5.8}$$

All combinations of $(d_p, d_q) \in [0, 3]^2$ were tried and most resulting fits produced a root mean-squared-error in the order of 10^{-5} for both the mean and the standard deviation, which is an inconclusive result when deciding on d_p and d_q. Instead, the simplest solution with resulting curves that are qualitatively similar to what is seen in figs. 5.11 and 5.12 was selected, which is $d_p = 1$ and $d_q = 1$. Figure 5.14 shows the resulting PDF for $V = 100$ and $\sigma_G = 1.0$, when using the rational functions to approximate the mean and standard deviation.

5.4 Alternative attributes

This chapter mentions two possible attributes (that is, power and flux density), but there are certainly many more attributes that can be used in one way or another to achieve segmentation; the only conditions are that it should be possible to compute it for max-tree nodes and there should be some sort of model (statistical or otherwise, for example a simple threshold). An attribute that is often used in astronomy to detect objects, is the peak (maximum) flux (the objects are then grown around their flux peak). This attribute was not considered here, because of the suspicion that it would be too sensitive too outliers – the peak flux is the outlier, after all.

5.5 Shrinking oversized objects

A problem with the current noise models is that not only nodes representing objects are marked as significant, but also some of their ancestor nodes, because the power of the object contributes to the power of the ancestor nodes. Currently, this is solved by the move-up step (which is similar to the k-absorption technique used by Ouzounis and Wilkinson [2011]). The move-up step has a very large impact on the resulting segmentation, which causes the importance of the supposedly intuitive parameter α to diminish. This is shown in fig. 5.16, whose segmentation was recovered using $\alpha = 1.0$. In effect, the statistical test was skipped, yet the result is still similar to fig. 5.15, which uses $\alpha = 10^{-6}$. The only use of the statistical test here seems to be removing local peaks.

The improved power model using a Γ-distribution does nothing to correct the aforementioned issues, so the move-up step as introduced by Teeninga et al. [2013] is still necessary. The model using the flux density attribute also has this problem, but less so, as an average is less sensitive to outliers than a sum of squares.

[1] `http://www.scipy.org`

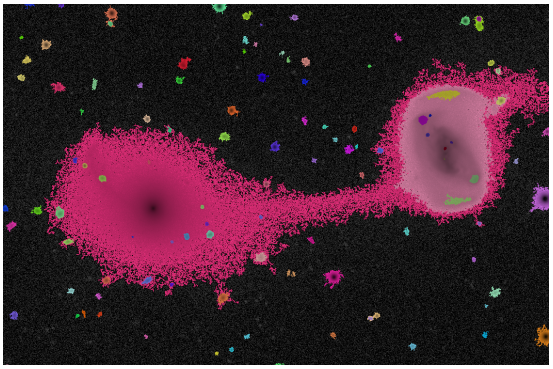

Figure 5.15: Same result as in fig. 4.2, uses significance level $\alpha = 10^{-6}$.

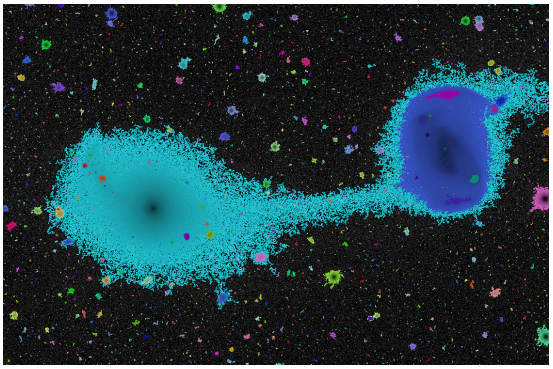

Figure 5.16: Similar to fig. 5.15, but uses significance level $\alpha = 1.0$. Other than a number of tiny spurious detections, the object mask remains the same.

Additionally, there is not yet a proper way to set λ other than trial and error. Rather than employing a move-up factor, an ideal solution would be if a noise model could be devised that solves the aforementioned problem of significant parent nodes. Since this is future work, the move-up step is studied in a bit more detail here. Teeninga et al. [2013] (in the optical case) move object identifiers up in the tree by an amount $\Delta f(O)$, defined as

$$\Delta f(O) = \lambda \sqrt{\frac{f(O)}{\text{gain}} + \sigma_B^2} \tag{5.9}$$

where O is the object under consideration, $f(O)$ is the base intensity of the object O, the gain is a property of the measuring apparatus, σ_B^2 is the background noise variance, and $\lambda \approx 0.5$ is a user-set parameter. This works well in the optical case. In the radio case, however, $g = \infty$ and consequently $f(O)$ has no effect on $\Delta f(O)$. This was found to give undesirable results, as faint detections are removed and brighter detections are hardly affected, retaining their spurious outer detail. Therefore, an alternative model is suggested here, which is also adapted to be used with mask-like connectivity

$$\Delta g(O) = \lambda \left(\max_{x \in O} g(x) - g(O) \right) \tag{5.10}$$

where g is the max-tree mask volume and $0 \leq \lambda \ll 1$. In effect, the lower fraction λ of the object's intensity range is removed. This means that faint objects are hardly affected, while brighter objects will have their spurious outer detail removed. Additionally, no detections are removed, which prevents the move-up step from dominating the segmentation. Some results are shown in figs. 5.17 and 5.18, using the max-tree masks of fig. 4.12 and the flux density model. One quirk of this approach is that the move-up is dependent on the max-tree mask and thus the smoothing used.

Since the range of λ is better defined now, it is easier to set this parameter. Values of 0.1 to 0.2 were found to give good results. Selecting a value that is too large will shrink objects too much, but this is not a major problem, as mask-growing algorithms are available [Popping et al., 2012].

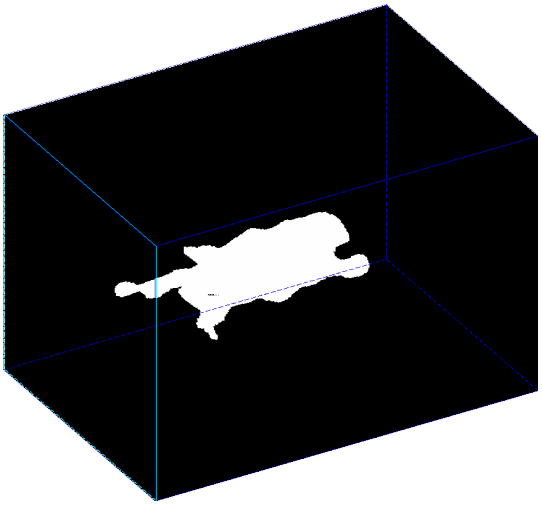

Figure 5.17: Segmentation recovered using fig. 4.12 as max-tree mask, using $\lambda = 0.0$. A large amount of spurious, outer detail is visible.

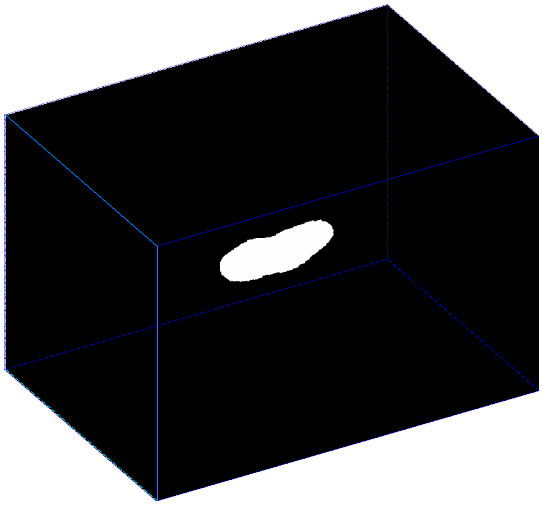

Figure 5.18: Segmentation recovered using fig. 4.13 as max-tree mask, using $\lambda = 0.1$. The spurious outer details have been removed.

5.6 Discussion

The radio noise was found to contain correlations, making statistical tests based on distributions that assume independent noise unusable. Two possible attributes and their distributions have been proposed that can be used given the correlated radio noise: flux density with a normal distribution and power with a Γ-distribution. The latter is a generalisation of the power attribute used by Teeninga et al. [2013], which assumed independent noise and a power attribute that follows a χ^2-distribution. The former is based on a new attribute, that is hypothesised to be less sensitive to outliers. Also, the move-up step was adapted to be dependent on the intensity range of detections, which gives better results for radio datasets.

Some disadvantages of the proposed statistical models are that they are not suited for nested features right now, which the optical model was to some extent, as it gave good results in terms of detecting nested objects. Possible further opportunities for investigation include using functions of attributes other than volume to predict model arguments, for example the mask intensity $g(P)$ of a max-tree node.

Estimating the parameters for the empirical models requires the availability of a pure noise volume. This may sound as a major drawback, but this need not be the case. For example, if the noise is symmetric and there are no absorption effects in the dataset, the negative part can be used as pure noise. Else, just using the entire volume will likely work as well, if it is sparse enough, as the objects present will have little effect on the parameter estimations. A third way could be to simulate the radio noise, if an appropriate model for that can be invented. In any case, since there is no suitable purely-analytic statistical model known, parameter estimation is necessary.

6 Segmentation results

This chapter aims to assess the performance of the MT source finder, that is, how well it detects objects. To quantify this, a few metrics should be defined. First, let D_{obj} be the number of objects in a reference object mask that were detected at least once. Second, let D_{true} be the number of detections that have overlap with at least one object in the reference. Finally, D_{tot} is the total number of detections.

The above allows more intuitive metrics to be defined, namely *completeness*, *reliability* (both are also used by Popping et al. [2012]), and *fragmentation*. Completeness, the fraction of objects that was detected, is defined as $D_{\text{obj}}/N_{\text{obj}}$, where N_{obj} is the number of objects actually present. Reliability, the fraction of detections that is correct (true), is defined as $D_{\text{true}}/D_{\text{tot}}$. Clearly, both completeness and reliability are in the range of zero to one, where higher is better. The final metric, fragmentation, is intended to prevent misinterpretation of the former metrics, as a method that greatly clusters or fragments detections can still score good at completeness or reliability. Fragmentation is defined as $D_{\text{true}}/D_{\text{obj}}$, where a score of one is perfect, a score smaller than one indicates a tendency to cluster objects, while a score larger than one indicates a tendency to fragment objects.

6.1 Results with different parameters

The segmentation algorithm was executed using both the power and flux density attributes on a $1464 \times 360 \times 360$ volume containing radio noise and 139 objects. A reference mask is available, meaning that the metrics indicated in the chapter introduction can be computed. The same volume was used by, among others, Serra et al. [2012].

The smoothing techniques used are intensity-driven diffusion, gradient-magnitude-driven diffusion and Gaussian smoothing. The diffusion times used were $t_{\max} \in \{0.5, 2.0, 4.5, 8.0\}$, with $K \in \{2.0\sigma, 3.0\sigma\}$ for intensity-driven diffusion, while gradient-magnitude-driven diffusion used $K \in \{1.7 \times 10^{-3}, 2.5 \times 10^{-3}\}$. These values for K were selected by looking at the histograms in figs. 6.1 and 6.2 and choosing a value close to the cut-off between the presumably noisy values and higher values, which probably represent objects (note the logarithmic density scale). The velocity smoothing factors are $c_{\text{vel}} \in \{1.0, 2.0\}$; when $c_{\text{vel}} = 2.0$, the volume is smoothed twice as aggressively in the velocity direction, which was suggested by astronomers to potentially result in an improvement. In all cases of Perona–Malik diffusion, the volume was pre-smoothed using Gaussian smoohing with a kernel of $\sigma_G \approx 0.85$. The significance level used for the statistical tests is $\alpha = 10^{-6}$ and the empirically determined move-up factor is $\lambda = 0.2$. Finally, a size filter was applied, discarding objects with a spatial extent smaller than three voxels or a spectral extent smaller than two voxels, as these detections are smaller than what a radio telescope can reliably measure and are thus indiscernible from noise.

Table 6.1: Segmentation results using different attributes and preprocessing techniques. The top 22 in terms of completeness (out of 246 total) are shown. Since most of these best results use the flux density attribute, it seems to be superior to the power in this context. A move-up factor of $\lambda = 0.2$ was used.

Smoothing	$t_{\max}$	K	c_{vel}	Attribute	Comp.	Rel.	Frag.
Diffusion (intensity)	2.0	2.0σ	1.0	Flux density	0.40	0.27	1.05
Diffusion (gradient)	2.0	1.7×10^{-03}	1.0	Flux density	0.40	0.27	1.07
Diffusion (gradient)	2.0	2.5×10^{-03}	1.0	Flux density	0.38	0.35	1.08
Diffusion (intensity)	2.0	3.0σ	1.0	Flux density	0.38	0.33	1.06
Gaussian	2.0		1.0	Flux density	0.38	0.19	1.06
Diffusion (intensity)	2.0	2.0σ	2.0	Flux density	0.37	0.65	1.08
Diffusion (gradient)	4.5	1.7×10^{-03}	1.0	Flux density	0.37	0.50	1.06
Gaussian	4.5		1.0	Flux density	0.37	0.44	1.06
Diffusion (intensity)	4.5	2.0σ	1.0	Flux density	0.37	0.44	1.12
Diffusion (intensity)	4.5	3.0σ	1.0	Flux density	0.36	0.50	1.12
Diffusion (gradient)	4.5	2.5×10^{-03}	1.0	Flux density	0.35	0.54	1.06
Diffusion (intensity)	0.5	2.0σ	1.0	Flux density	0.35	0.35	1.04
Diffusion (gradient)	0.5	1.7×10^{-03}	1.0	Flux density	0.35	0.39	1.04
Gaussian	0.5		1.0	Flux density	0.35	0.27	1.06
Diffusion (gradient)	0.5	2.5×10^{-03}	1.0	Flux density	0.34	0.37	1.04
Gaussian	2.0		2.0	Flux density	0.33	1.00	1.04
Diffusion (intensity)	0.5	3.0σ	2.0	Power	0.33	0.98	1.07
Diffusion (gradient)	2.0	2.5×10^{-03}	2.0	Flux density	0.33	0.84	1.07
Diffusion (intensity)	4.5	2.0σ	2.0	Flux density	0.33	0.83	1.04
Diffusion (intensity)	2.0	3.0σ	2.0	Flux density	0.33	0.82	1.07
Diffusion (gradient)	2.0	1.7×10^{-03}	2.0	Flux density	0.33	0.79	1.09
Diffusion (intensity)	0.5	3.0σ	1.0	Flux density	0.33	0.37	1.04

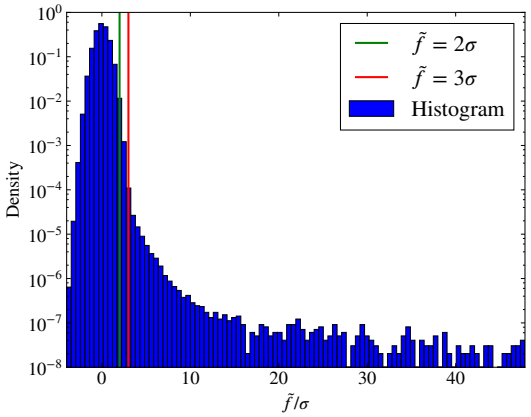

Figure 6.1: 100-bin histogram of the large radio dataset, after applying Gaussian smoothing with $\sigma_G \approx 0.85$ (smoothed values denoted $\tilde{f}$). The position of two possible Ks is shown, in green and red.

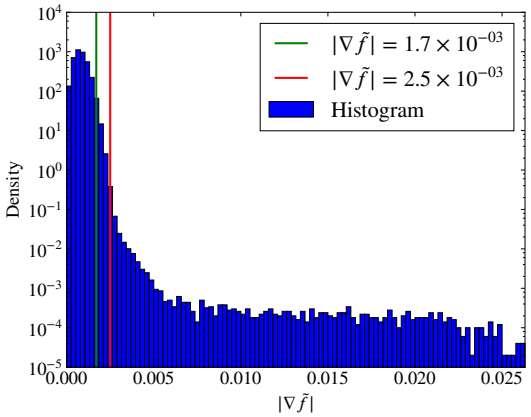

Figure 6.2: 100-bin histogram of the large radio dataset's gradient magnitudes, after applying Gaussian smoothing with $\sigma_G \approx 0.85$ (smoothed values denoted $\tilde{f}$). The position of two possible Ks is shown, in green and red.

Some results are shown in table 6.1. The results were filtered such that completeness is at least 1/3 and $1/2 \leq$ fragmentation ≤ 2. This means that results that are clearly nonsensical are not included. The entries are sorted by the completeness column and then by the reliability column. From table 6.1, the following conclusions can be drawn:

1. The flux density attribute outperforms the power attribute, the best (in terms of completeness) result using the power attribute achieving a completeness of only 0.33, compared to 0.40 of using the flux density attribute.

2. The optimal smoothing time seems to be around 2.0. At least the values 0.5 and 8.0 are likely too low and too high, respectively.

3. The different types of smoothing (intensity-driven Perona–Malik, gradient-magnitude-driven Perona–Malik, Gaussian) do not seem to produce very different results, when judging by the completeness. Reliability is slightly better using the adaptive variants, with more conservative K-values performing best.

4. Using $c_{\text{vel}} > 1$ seems to make the segmentation result worse, rather than better. This means that even better results may be possible if the parameter estimates for the statistical models are re-computed, as they were optimised for $c_{\text{vel}} = 2.0$.

5. The best completeness value attainable on this volume seems to be approximately 0.4, which is comparable to the state-of-the-art method discussed in section 6.2. The missed galaxies are all very faint and usually very distant.

6. For the selection of results in table 6.1, the fragmentation level is quite close to one, which is desirable. Also, the fragmentation level tends to be above zero, which is also good, as merging detections is likely easier than splitting them.

6.2 Comparison with other algorithms

To see whether the performance of the MT source finder is comparable to that of the existing methods, some experiments with different noise levels were executed, and the results were compared to those of the smooth-and-clip (S+C) source finder of the SoFiA[1] package. The volumes with different noise levels were acquired by multiplying the noise with $\sigma = 1.6 \times 10^{-3}$ by a constant $c_{\text{noise}} \in [0.2, 4.0]$ and then adding the objects. The reason that S+C was chosen as reference is two-fold. First, Popping et al. [2012, sec. 5.2] note that S+C is arguably the best source finder so far for extended (rather than point-like) sources, detecting all but the faintest sources while providing good reliability. Second, the S+C source finder is fast, making the execution of such experiments doable, while the wavelet-based methods and CNHI source finder may take hours or days to complete (this depends on the hardware and chosen parameters, of course). The fact that the latter source finders have more parameters only exacerbates this issue, as more experimentation is required.

The set of kernels used for the S+C source finder are SoFiA's defaults, as these already gave good results and similar kernels were used by Popping et al. [2012, sec. 4.2]. The threshold used is 4.0σ, where σ is estimated after smoothing, contrary to the MT source finder, where it is determined before smoothing.

The MT source finder uses the same parameters as in the case of the best result of section 6.1, which means using intensity-driven diffusion with $K = 2.0\sigma$ and $t_{\max} = 2.0$, which is isotropic along the spatial and spectral axes. Before applying Perona–Malik diffusion the volume was pre-smoothed. For the segmentation, a significance level $\alpha = 10^{-6}$ was used and a move-up factor of $\lambda = 0.2$. The same size filter as in section 6.1 was used, for both MT objects and S+C.

As an added bonus, the results of both source finders were ran through Serra et al. [2012]'s reliability filter, which is also implemented in SoFiA, using a reliability threshold of 0.9. This is to compare the performance when unreliable detections are removed. The MT source finder does not find any negative-total-flux detections, which are required for the reliability filter to work. This problem is solved by running the source finder twice, once on the inverted data cube and combining the results before they are processed by the reliability filter.

The results are plotted in figs. 6.3 to 6.5. In terms of completeness, which is arguably the most important metric (since good scores in the other metrics are mostly useless if the completeness is low), the MT source finder gets quite close to S+C, but consistently performs worse. While this is somewhat disappointing, it should be noted that S+C is considered a very good source finder. S+C's completeness curve is quite predictable, as it slowly decreases as the noise level increases. The curve of the MT source finder has a more surprising shape. Curiously, the performance *decreases* for the lowest noise level considered here ($c_{\text{noise}} = 0.2$) and there seems to be a local maximum at $c_{\text{noise}} = 1.0$. This is the noise level used to tune the parameters (using the results of section 6.1), which suggests that the parameters may have been overly tailored towards this

[1] `https://github.com/SoFiA-Admin/SoFiA`, revision 0020a4a3ce was used.

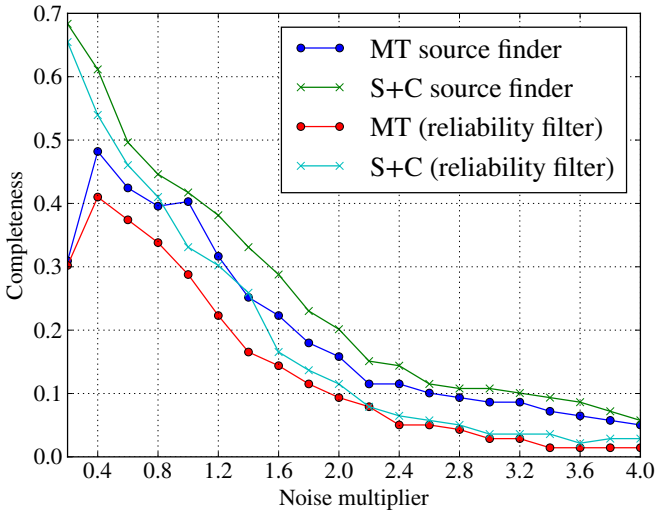

Figure 6.3: Comparison of completeness for the MT and S+C source finders for different noise levels. S+C appears to be more complete across all noise levels.

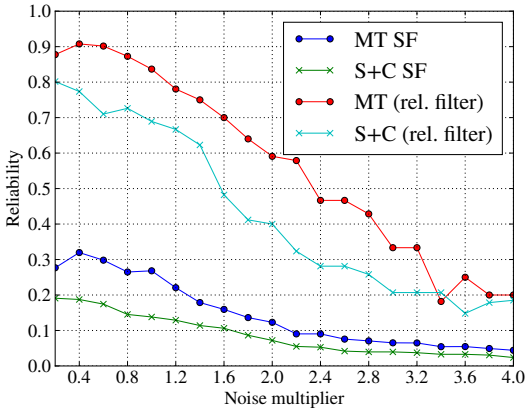

Figure 6.4: Comparison of reliability for the MT and S+C source finders for different noise levels. MT appears to be more reliable across all noise levels.

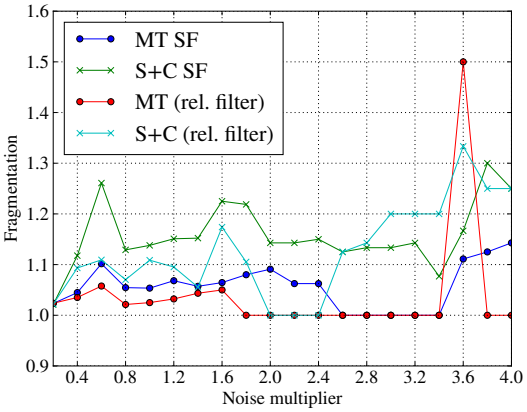

Figure 6.5: Comparison of fragmentation for the MT and S+C source finders for different noise levels. MT detections appear to be less fragmented than those of S+C.

noise level and that a σ-adaptive K is not enough to make up for this. It may be that $t_{\max} = 2.0$ is suboptimal for lower noise levels or that the amount of pre-smoothing is too aggressive (although even less aggressive smoothing might be problematic on discrete volume, as σ_G is already less than one voxel width). When the reliability filter is turned on, both source finders' completeness suffers, indicating some true detections are removed.

A comparison of the reliability is shown fig. 6.4 (negative-total-flux detections are not considered for either source finder). Here, the MT source finder fares much better in comparison to S+C, attaining reliability levels roughly twice as good. The reliability for both methods goes down when the noise level increases, because the number of true detections decreases, while the total number of detections remains stable. Curiously, the reliability results for S+C are much worse here than those mentioned by Popping et al. [2012] (see fig. 1.3) and Serra et al. [2012, sec. 4] (the latter paper considers the same radio volume, although a different set of kernels). This may be because the other authors use more aggressive size filtering, but this is conjecture. So, care should be taken in interpreting this results, as it may be simple to tune the S+C reliability to similar levels of the MT source finder. When the reliability filter is used, the reliability rates increase greatly, indicating the reliability filter does a good job at removing false detections. Relative to each other, the reliability rates seem unchanged.

Finally, fig. 6.5 shows the amount of fragmentation that occurs in the detected sources. Here, the MT source finder has a slight edge over S+C, although both methods perform well here and maintain similar fragmentation levels across the whole considered range of noise multipliers. The value for the MT source finder seems to increase drastically at the end, after being at the perfect level of one for a while, but this is due to the fact that very few detections remain, so one disconnection can lead to a drastic increase in fragmentation level. Turning on the reliability filter does improve the fragmentation level somewhat.

6.3 Qualitative assessment

Figures 6.6 to 6.9 show a few partial segmentation results of the MT source finder, corresponding to the best results from table 6.1 for intensity-driven-diffusion, gradient-magnitude-diffusion, and Gaussian smoothing. For comparison purposes, a segmentation result of the S+C source finder is also included (using threshold 4.0σ, as in section 6.2). All masks had their unreliable detections removed using Serra et al. [2012]'s reliability filter (using a threshold of 0.9). The spectral is aligned with the elongation of the objects.

The result using intensity-driven diffusion, shown in fig. 6.6, recovers two of the three shown objects only partially. The result using gradient-magnitude-driven diffusion, shown in fig. 6.7, suffers less from this problem, as the elliptical object is recovered completely. Figure 6.8, the result using Gaussian smoothing, misses the elongated object completely, but recovers the elliptical object fully. So, one could conclude that gradient-magnitude-driven diffusion and Gaussian smoothing give results that are qualitatively better than those of intensity-driven diffusion, because shapes of objects are better preserved.

S+C's result is shown in fig. 6.9. Unlike the MT source finder, S+C recovers all objects shown completely. Perhaps some type of anisotropic filtering is necessary to make the MT source finder recover the elongated object fully.

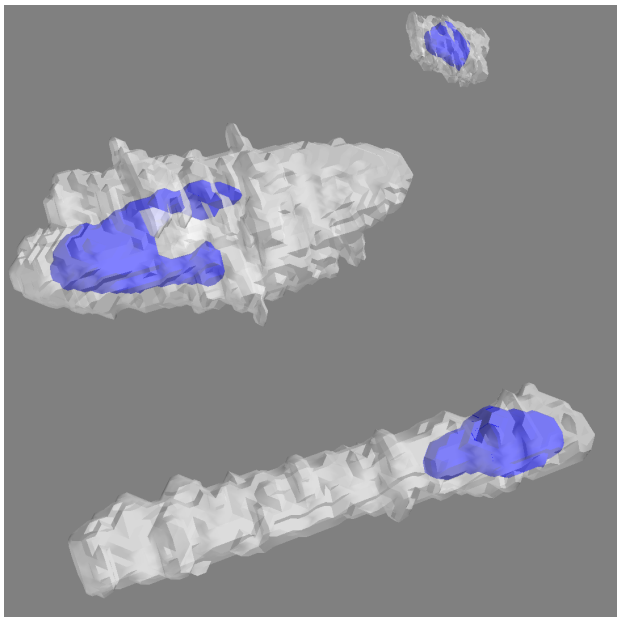

Figure 6.6: Contour plot of part of a segmentation result. The reference mask is shown in white and the MT source finder mask (using intensity-driven diffusion) is shown in blue. Both the elliptical and elongated object are only partially recovered.

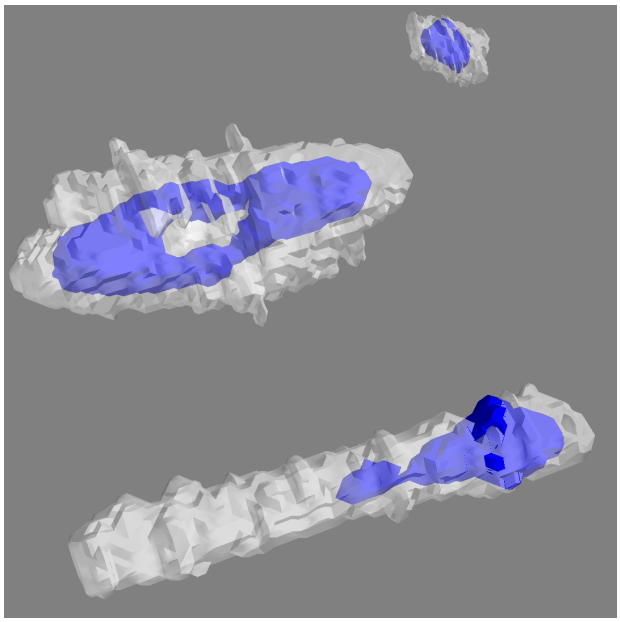

Figure 6.7: Contour plot of part of a segmentation result. The reference mask is shown in white and the MT source finder mask (using gradient-magnitude-driven diffusion) is shown in blue. The elongated object is only partially recovered.

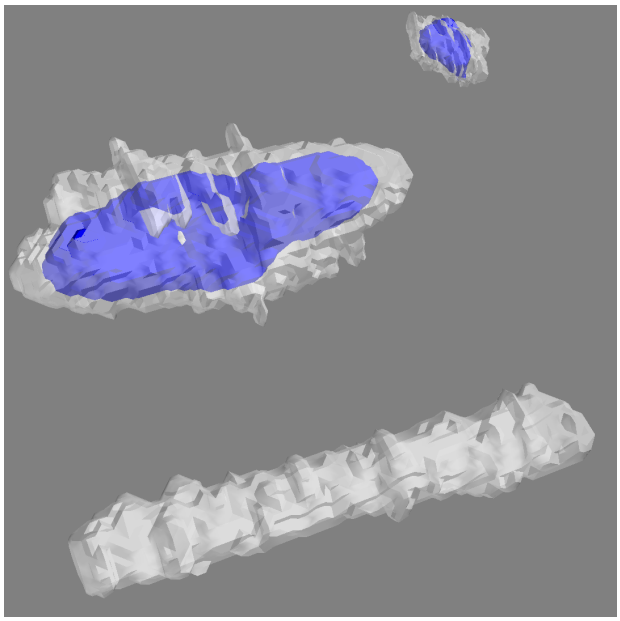

Figure 6.8: Contour plot of part of a segmentation result. The reference mask is shown in white and the MT source finder mask (using Gaussian smoothing) is shown in blue. The elongated object is missed.

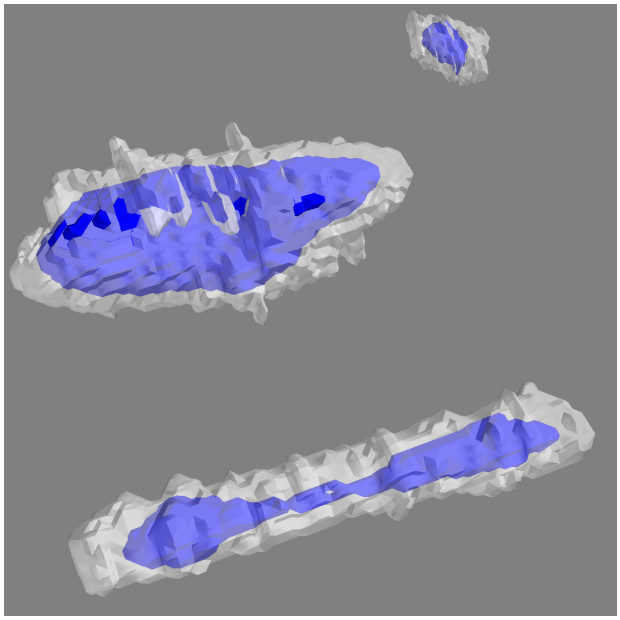

Figure 6.9: Contour plot of part of a segmentation result. The reference mask is shown in white and the S+C mask is shown in blue. All visible objects are recovered.

6.4 Discussion

The results achieved using the MT source finder are somewhat promising, as it gets close to a state-of-art method (S+C) in terms of completeness, although it does not yet outperform it. Therefore, some more research into the parameter selection and possible extensions to the algorithm is necessary. For example, the estimated parameters of the statistical models assume smoothing is anisotropic along the spectral axis (twice as strong). Since this did not seem to give the best result, the estimation should be redone using isotropic smoothing. The MT source finder does perform slightly better in terms of reliability, but post-processing is still required as the number of false detections is large. Additionally, like the existing methods, a large number of faint objects is missed.

Using adaptive smoothing in combination with the mask-like approach to connectivity does seem to improve the performance somewhat in comparison with simple Gaussian smoothing. In contradiction to the hypothesis of chapter 4, intensity-driven diffusion does not outperform gradient-magnitude-driven diffusion. Instead, they perform similarly with gradient-magnitude-driven diffusion resulting in qualitatively better object masks.

7 Identifying true detections

Most segmentation algorithms do some sort of post-processing on their detections to reject false detections. These types of post-processing include flux thresholds [Flöer and Winkel, 2012, sec. 4.4.1] and size filters [Whiting, 2012, sec. 7]. A size filter is also employed in chapter 6 to reduce the number of false detections of the various source finders and its use is justified by the fact that, due to the way the measuring apparatus works, an anomaly of a certain size is not always meaningful. Serra et al. [2012] mention that, "*a posteriori*, it would be easy to define a criterion to efficiently separate true from false detections", when discussing plots showing attributes of detected sources. Additionally, when using a method based on statistics, such as the MT source finder, one expects to find false detections based on the significance level. When considering a volume with in the order of 10^8 max-tree nodes, such as the one used in chapter 6, this means that there may be hundreds of false detections using $\alpha = 10^{-6}$. In practise this will only be exacerbated by the fact that the statical model is empirical in nature and not assumption-free.

This chapter aims to give a systematic approach to false detection elimination by employing some machine-learning techniques. For the best result of chapter 6 (the top one in table 6.1, which is better than the second result because of its fragmentation score), several attributes of the detected sources were computed. Experiments were executed to determine whether decision trees and nearest-prototype classification are useful here. If machine learning techniques can be used on their own for max-tree-based volume segmentation is an open question. This will likely prove challenging, as it is difficult to obtain a dataset that contains attributes for both objects and noise nodes that is not massively biased towards the latter.

In this chapter, 10-fold cross validation is used to make estimates for the performance of the different classifiers. Applications on real-word datasets may be somewhat problematic, as different datasets can possess greatly different characteristics, reducing the generalization ability of classifiers. Additionally, supervised methods require labelled input, which may not be available. One way around this could be to invert the dataset (make the negative values positive and vice versa), so that there are no more remaining true sources with a positive total flux. Then, plant artificial objects in the resulting volume, and apply the source finder of choice. Since the sources were added manually and most source finders do not consider negative-total-flux sources, identifying true and false detections in the segmentation is possible. The results can be used to train a classifier, which can then be applied on the results of processing the original dataset.

7.1 Useful attributes

The attributes used include the volume, total flux, and peak flux, which are all ubiquitous in astronomy. Additional computed attributes include *elongation* and *flatness*, which have the expected intuitive meaning for 3D shapes and are defined formally below (as by Westenberg et al. [2007]). Many of the sources that are known to be true have an elongated shape, for example

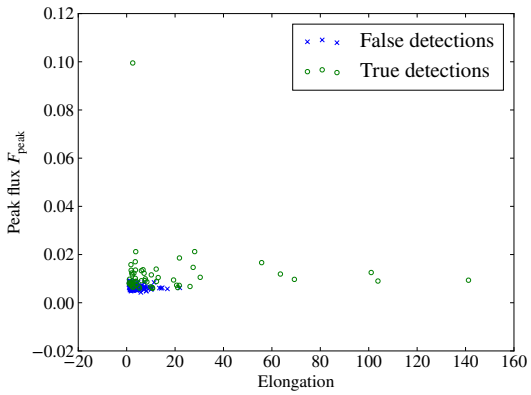

Figure 7.1: Scatter plot showing the peak flux and elongation of true and false detections. This shows both peak flux and elongation could be useful in identifying true detections.

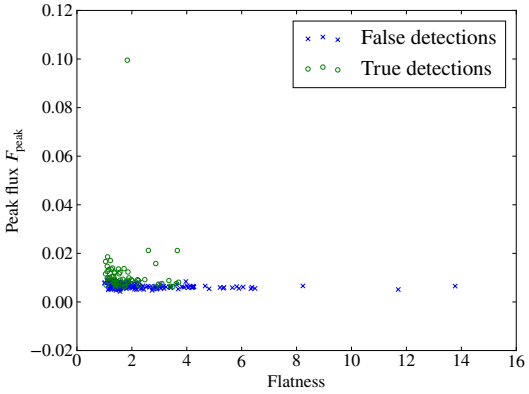

Figure 7.2: Scatter plot showing the peak flux and flatness of true and false detections. This shows flatness could be useful in identifying false detections.

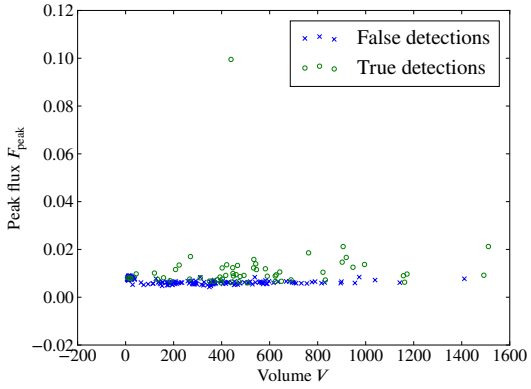

Figure 7.3: Scatter plot showing the peak flux and volume of true and false detections. This shows volume is likely not useful in distinguishing true from false detections.

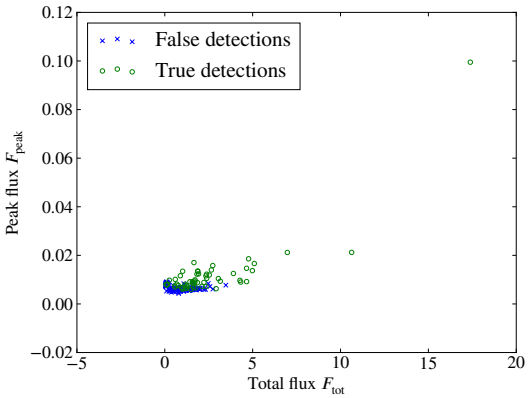

Figure 7.4: Scatter plot showing the peak flux and volume of true and false detections. This shows that total flux, like peak flux, could be useful in separating true from false detections.

those considered in chapter 4. Flatness is simple to compute in addition to elongation and could reveal some additional insight in the properties of true sources. More possible attributes that are excluded are spatial area, spectral width, sparseness [Westenberg et al., 2007], or anything that can be computed for a max-tree node. Judging by figs. 7.1 to 7.4, which plot detections' peak flux versus the other attributes, it may be possible to construct a decision boundary that separates at least some of the true detections from the false detections. The plots also suggest that some true detections may not be distinguishable from false detections. Since the completeness of all source finders is quite low for this dataset, it may be that these are indeed among those very hard to detect and the fact that they were detected is a coincidence.

The elongation and flatness are computed as in Westenberg et al. [2007, sec. III]. Given a max-tree node C, the moment of inertia tensor $\boldsymbol{I}(C)$ is defined as

$$\boldsymbol{I}(C) = \begin{pmatrix} I_{xx}(C) & I_{xy}(C) & I_{xz}(C) \\ I_{xy}(C) & I_{yy}(C) & I_{yz}(C) \\ I_{xz}(C) & I_{yz}(C) & I_{zz}(C) \end{pmatrix} \tag{7.1}$$

with

$$I_{xx}(C) = \sum_C (x - \bar{x})^2 + V(C)/12 \tag{7.2}$$

$$I_{yy}(C) = \sum_C (y - \bar{y})^2 + V(C)/12 \tag{7.3}$$

$$I_{zz}(C) = \sum_C (z - \bar{z})^2 + V(C)/12 \tag{7.4}$$

$$I_{xy}(C) = \sum_C (x - \bar{x})(y - \bar{y}) \tag{7.5}$$

$$I_{xz}(C) = \sum_C (x - \bar{x})(z - \bar{z}) \tag{7.6}$$

$$I_{yz}(C) = \sum_C (y - \bar{y})(z - \bar{z}) \tag{7.7}$$

where $\bar{x}$, $\bar{y}$, $\bar{z}$ are the centre-of-mass coordinates of C. The term $V(C)/12$ in the diagonal elements is to correct for the cubic nature of the voxels. Let $|\lambda_1| \geq |\lambda_2| \geq |\lambda_3|$ be the eigenvalues of $\boldsymbol{I}(C)$. Then the elongation of C is defined as $\mathcal{E}(C) = |\lambda_1/\lambda_2|$ and flatness is defined as $\mathcal{F}(C) = |\lambda_2/\lambda_3|$. Since all elements of $\boldsymbol{I}(C)$ can be decomposed into $\sum x^2$, $\sum y^2$, $\sum z^2$, $\sum xy$, $\sum xz$, $\sum yz$, $\sum x$, $\sum y$, $\sum z$, and $V(C)/12$, which are all increasing, they can easily be computed for max-tree nodes.

7.2 Classification trees

One way to do classification is using *classification trees*, a type of decision tree. A classification tree is a tree where each node gives a criterion about which branch to take, eventually leading to a leaf that determines the classification. One advantage of classification trees is that it is easy to understand how they work, unlike certain types of neural network. One popular algorithm to construct such a tree is the *C4.5 algorithm*, also used by Moschini et al. [2015b] to separate merging from overlapping galaxies.

The C4.5 algorithm works in the following recursive manner, starting with the complete training set and recurseing on subsets:

Base case 1 All examples in the current training subset belong to the same class. Create a leaf node labelled with this class.

Base case 2 No split of the current training subset achieves a decrease in entropy or the current training subset is empty. Create a leaf node with the expected (most common) class for the full training set.

Recursive case Compute the information gain for all possible splits of each attribute. Split the current training subset on the attribute that achieves the largest information gain. Recurse on the resulting subsets.

Here, information gain is defined as

$$\mathrm{IG}(S) = H(S) - \sum_i \frac{|T_i|}{|S|} H(T_i) \tag{7.8}$$

where S is the current training subset, T_i are the subsets of S after splitting, and $H(S)$ is the entropy. Entropy is defined as $H(S) = -\sum_{c \in C} p(c) \lg p(c)$, where C is the set of classes and $p(c)$ is the overall probability of encountering class c. Pruning techniques can be applied to remove subtrees that do not result in a decrease in test error or that seem to be the result of overfitting (for example, when only a few elements remain in T_i).

Two techniques employed to improve the performance of classifiers are *boosting* and *bagging*. The performance of these techniques with respect to the C4.5 algorithm is discussed by Quinlan [1996], who concluded that, while both boosting and bagging generally improve the performance of C4.5, boosting is usually more effective. Note that boosting does have problems with certain datasets, the reason of which was not fully understood at the time of writing of Quinlan [1996]'s paper.

Both bagging and boosting work by creating T different classifiers, which then work in tandem using a majority vote. The difference lies in how these classifiers are trained. Bagging re-samples the training set (of size N) into T different training sets, with replacements. This means that the individual training subsets may miss examples or contain duplicates. Boosting takes a different approach; a weight vector for the examples is maintained, which is updated after each training trial, such that the weight of misclassified examples is increased. The weight vector determines how much the examples influence the next classifier. The final aggregate classifier weights the votes based on the accuracy of the constituent classifiers.

In boosting (AdaBoost.M1 is used here), the weight vector $\vec{\omega}^t$ is initialised such that $\omega_x^1 = 1/N$. The composite classifiers are trained assuming that the probability of example x occurring is ω_x^t. The error ε^t of the (aggregate) classifier C^t after iteration t is also determined in terms of the weight vector, as it is defined as the sum of the weights of the incorrectly classified examples. After trial t is completed, the weight vector is updated by multiplying the weights of correctly classified examples by a factor $\varepsilon^t/(1-\varepsilon^t)$ and then normalising. If $\varepsilon^t = 0$, the training is terminated (because the training error is zero) and if $\varepsilon^t > 0.5$ the training is also terminated and the current trial is discarded, as it can no longer be guaranteed that continuing will result in an improvement.

Table 7.1: Results of applying C4.5 classification trees to separate true from false detections, without and with boosting and bagging. 10-fold cross validation is used to get unbiased metrics. A paired t-test revealed that there are no significant differences between the rows.

Technique	Error	TPR	FPR	Precision
Simple C4.5	0.14	0.61	0.05	0.85
Boosting	0.15	0.67	0.08	0.78
Bagging	0.13	0.66	0.06	0.82

Using the Weka machine learning software[1], the C4.5 algorithm was used on the attributes discussed in the chapter introduction to classify the detections as either true or false. C4.5 was used on its own, as well as with bagging and AdaBoost.M1, both using ten trials. 10-fold cross validation was used to compute the error estimates, which was again repeated ten times for a total of a hundred calls of the algorithm. Branches that do not result in a decrease in test error were pruned, as well as singleton leaves (containing only a single example). Many different measures to quantify the quality of a binary classifier exist. The following selection was made here: error (one minus accuracy), true positive rate (TPR, also hit rate, recall rate, or sensitivity), false positive rate (FPR, also false alarm rate), and precision. These measures are shown in table 7.1 and are defined as follows (where N is the size of the test set, TP the number of true positives, FP the number of false positives, TN the number of true negatives, and FN the number of false negatives):

$$\text{Error} = \frac{\text{FP} + \text{FN}}{N} \tag{7.9}$$

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{7.10}$$

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}} \tag{7.11}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{7.12}$$

Intuitively, the error is the fraction of detections that is classified incorrectly. This is the simplest metric, but can give a misleading intuition if the test set is biased, which it is here, as there are more false than true detections (since the reliabilities recovered in chapter 6 are below 0.5). The TPR is the fraction of true detections that is actually classified as true, while the FPR is the fraction of false detections that is classified as true. Finally, the precision is the fraction of detections classified as true that actually are true. Note that the true negative rate (also correct rejection rate or specificity) and false negative rate (also miss rate) can easily be computed as one minus the false positive rate and true positive rate, respectively. A measure similar to precision can also be computed for false detections.

Table 7.1 shows that the results are all very similar for the different techniques. Using the paired t-test built in Weka with a significance level of $\alpha = 10^{-6}$ proves that there is indeed no statistically significant difference in any of these quantities. Arguably, since the primary

[1] `http://www.cs.waikato.ac.nz/~ml/weka`

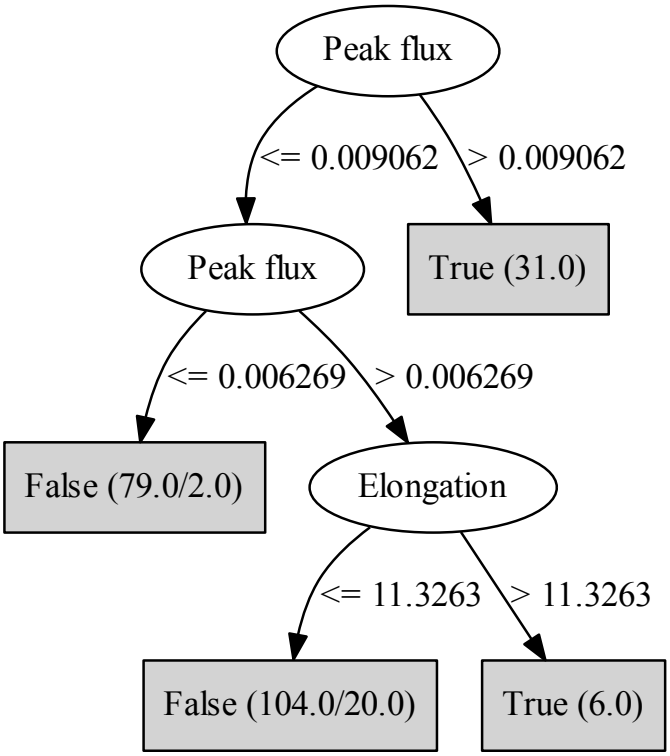

Figure 7.5: Decision tree resulting from the C4.5 algorithm, grown using all available data. Singleton leaves where pruned. Given in brackets, the first number is the number of instances reaching that leaf and the second number (if present) gives the number of misclassifications.

interest is in identifying true detections, the most import metrics are the TPR and the precision. All in all, using a more complex algorithm does not seem to result in a significant increase in performance; this may be because the classification problem is too simple, so the boosting and bagging techniques are too complex. Figure 7.5 shows the decision tree resulting from the simple C4.5 algorithm applied on the whole training set, which only has three inner nodes and could also have been determined manually from fig. 7.1.

7.3 Nearest-prototype classification

An alternative to classification trees is k-nearest-neighbour classification. In this classification algorithm new examples are classified by performing a majority vote on their k nearest neighbours in a dataset with known class labels. While this algorithm could be used here, it could get unwieldy as a potentially very large number of known examples is required. An answer to this is to use *nearest-prototype classification*. Here, instead of using a large set of known examples, a small set of carefully-chosen prototypes representing the classes is used, and as a new example is presented, it is assigned to the class of the nearest prototypes. Note that, contrary to k-nearest-neighbour classification, a training phase is required here to construct the prototypes. Nearest-prototype classification is intuitive, like a classification tree, as the prototypes can be viewed as data instances themselves.

There are several algorithms that can be used for constructing prototypes, both supervised and unsupervised. One unsupervised algorithm is the k-means algorithm, which attempts to partition a dataset in k clusters, where each example belongs to the cluster with the nearest mean. The constraint is that the clustering minimises the sum of squared distances of the examples to the corresponding cluster means. A popular algorithm to realise k-means is Lloyd's algorithm, which is also used here. It was first used in a signal processing context [Lloyd, 1982]. Here, this is extended to prototype construction by taking the cluster means as prototypes and assigning labels to them by a majority vote within the cluster. A brief sketch of Lloyd's algorithm is as follows. Given N examples $\vec{x}^i$ and k (random) initial cluster means $\vec{\mu}^j$, iteratively execute the following steps:

Assignment step Assign each $\vec{x}^i$ to the cluster S_j with nearest cluster mean $\vec{\mu}^j$ in terms of (squared) Euclidean distance.

Update step Update the cluster means such that $\vec{\mu}^j = \text{mean}(S_j)$.

These steps are repeated until the cluster means stop moving (significantly). Clearly, the sum-of-squared distances to the cluster means is decreased (or stable) every iteration, so it is guaranteed to find a local minimum of the sum of squared distances. The implementation used here is the one included in scikit-learn[2].

A supervised algorithm to select prototypes is *learning vector quantisation* (LVQ). LVQ1 is used here, which means that only one prototype is updated each iteration. One problem when using LVQ1 is selecting the initial prototypes to use per class. This is solved here by taking the k-means and their class labels as assigned above. Other problems are selecting the number of

[2] `http://scikit-learn.org`

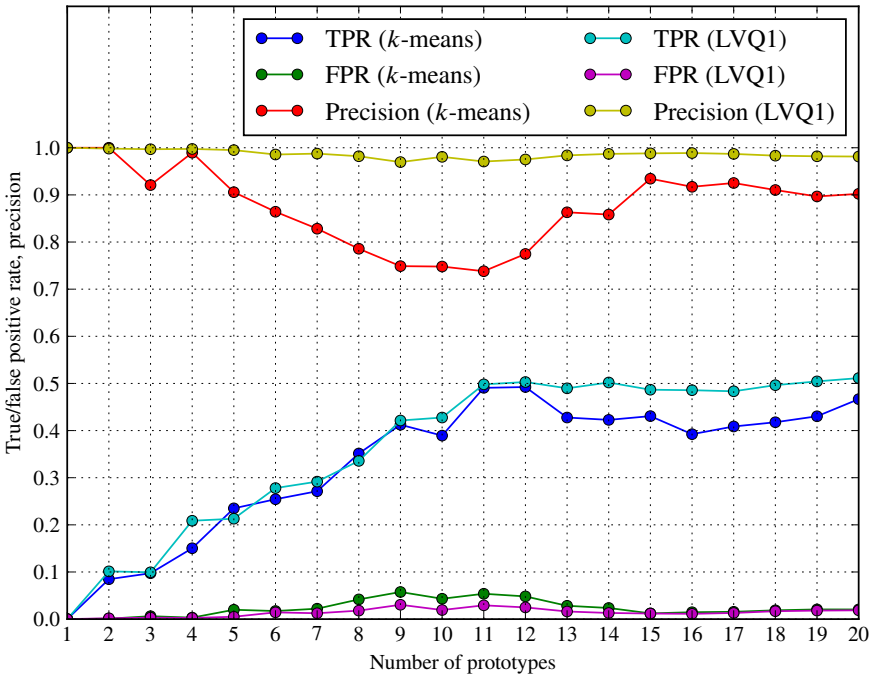

Figure 7.6: True/false positive rates and precision when using nearest-prototype classification, with prototypes constructed using k-means and LVQ1. LVQ1 realises a small improvement over k-means for TPR and FPR, and a large improvement for precision.

prototypes (several are tried) and a distance measure (only Euclidean distance is considered here). A custom LVQ1 implementation is used here that performs Q *randomised sweeps* through the dataset. This means that the examples $\vec{x}^i$ are considered in a random order and each $\vec{x}^i$ is presented Q times, but not again until all other examples have been considered. When an example $\vec{x}^i$ with label X^i is presented, the following steps are executed:

Prototype identification step Identify the prototype $\vec{p}^j$ with label P^j closest to the example $\vec{x}^i$.

Update step Update the prototype $\vec{p}^j$. If $P^j = X^i$, $\vec{p}^j$ is attracted towards $\vec{x}^i$: $\vec{p}^j \leftarrow \vec{p}^j + \eta(\vec{x}^i - \vec{p}^j)$. Else, it is repelled: $\vec{p}^j \leftarrow \vec{p}^j - \eta(\vec{x}^i - \vec{p}^j)$.

The k-means and LVQ1 algorithm were used in conjunction with nearest-prototype classification and stratified 10-fold cross validation to create figs. 7.6 and 7.7. For LVQ1, a learning rate $\eta = 0.1$ was used and ten randomised sweeps through the training set were performed. As in section 7.2, this was repeated ten times using different splits for 10-fold cross validation and the results were averaged. Figure 7.8 shows the error rate per prototype for eleven prototypes, when using the whole dataset as training set. Such a plot could be used to assess the reliability of a classification if the prototype is known.

About eleven prototypes are necessary for the nearest-prototype classifier to be able to compete with the decision trees in terms of TPR. With this number of prototypes the results stabilise and

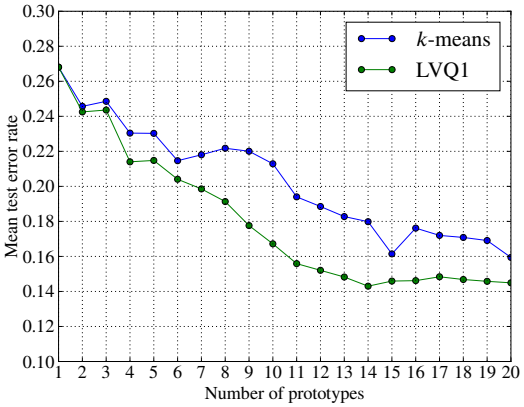

Figure 7.7: Test error rates when using nearest-prototype classification for different numbers of prototypes, constructed using k-means and LVQ1. LVQ1 shows a modest improvement.

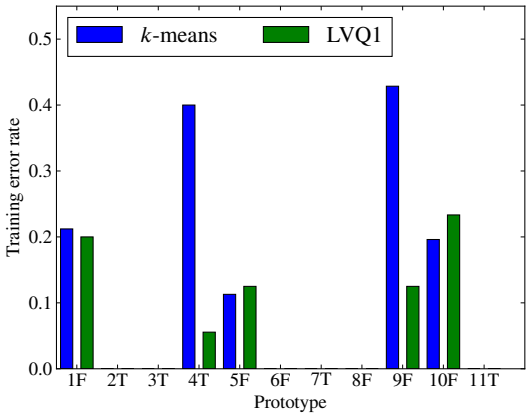

Figure 7.8: Comparison of the training error rates per prototype for nearest-prototype classification. This could possibly be used to assign a reliability to a particular classification, based on which prototype it is assigned to.

adding more does not seem to result in an improvement (although there is no guarantee that there will never be an improvement if the number of prototypes is increased even further). Using eleven prototypes gives a true positive rate of approximately 0.5 (compared to 0.6 for the simple C4.5 tree) and a false positive rate comparable to that of the simple C4.5 tree (approximately 0.05). The overall error rate is slightly worse (approximately 0.16 for LVQ1 compared to 0.14 for the simple C4.5 tree), although k-means performs worse still. LVQ1 does shine when it comes to precision, achieving a value close to one over the entire range of prototype counts considered. The k-means-based method has no such property and this puts the LVQ1-based classifier well ahead.

7.4 Discussion

Machine learning techniques, such as classification trees and prototype-based classifiers, may very well be a valuable tool in separating true from false detections in the output of astronomical source finders. On the considered dataset, a simple C4.5 classification tree is able to attain a TPR of 0.61 and a precision of 0.85. On the other hand, an LVQ1-based nearest-prototype classifier with eleven prototypes achieved a TPR of approximately 0.5, with a precision close to one. Thus, the choice between a classification tree and prototype-based classifier may boil down to a choice between TPR and precision. Since the MT source finder can attain a completeness of 0.4 on the considered dataset, taking into account the TPR and precision of the LVQ1-based classifier, about 0.2 of all sources can be identified with certainty.

Possible future work includes evaluating the performance of generalisations of LVQ. Additionally, LVQ can be biased so the user can modify the trade-off between the different metrics

Table 7.2: Error rate, true positive rate, false positive rate, and precision for Serra et al. [2012]'s reliability filter as applied in section 6.2 for $c_{\text{noise}} = 1.0$, using a threshold of 0.9.

Error	TPR	FPR	Precision
0.08	0.77	0.06	0.69

manually.

For comparison, a few numbers relevant to Serra et al. [2012]'s reliability filter are included in table 7.2. This technique achieves much better error and true positive rates than the classification tree and the LVQ1-based prototype classifier, but suffers in terms of precision (some tuning could be done using the threshold). So, while this technique excels at TPR, the LVQ1-based prototype classifier may still be preferable when a high precision is required. Note that no 10-fold cross validation was used on the reliability filter so the results may be biased, although such an effect is likely limited, as this method always trains on the set to be classified itself and the training is not stochastic.

There may be some practical problems when applying supervised machine learning techniques to the type of datasets considered here, as reference data may be unavailable (although it could possibly be generated by using the negative part of a dataset and added artificial sources). Still, using k-means as a purely unsupervised method to identify clusters of true detections could be useful. Techniques that may be useful but have not been considered here, include unsupervised outlier detection. Since most of the detections are false, it is tempting to attempt to find outliers in the full set of detections, as they are likely true.

8 Performance analysis

This chapter analyses the theoretical and observed performance of the max-tree source finder and discusses how these compare.

8.1 Time complexity and memory usage

First, consider Perona–Malik diffusion. Given a volume with V voxels, it is evident from eq. (4.6) that the number of computations per iteration is proportional to cV, where c is the connectivity, usually six, a small constant. So, the time complexity of Perona–Malik diffusion using the descretisation discussed in section 4.1 is $O(NV)$, where N is the number of iterations. When the process is parallelised over T threads, by spatially splitting the volume each iteration in equally-sized partitions, a speed-up of T is expected, as there are no interdependencies between voxels. The memory usage is $O(V)$. From eq. (4.6) it follows that multiple voxels I_{ij}^{n+1} have a dependency on I_{ij}^{n}; the current implementation manages this by storing both $\boldsymbol{I}^n$ and $\boldsymbol{I}^{n+1}$. This results in a total memory usage of approximately $2Vd$, where d is the bit depth of the volume (for example 32 bits for single-precision floating point numbers).

Next, consider the time complexity of the max-tree construction algorithm. Moschini et al. [2015a, sec. 8] argue that the so-called diplomatic algorithm has linearithmic time complexity. Memory-wise, the max-tree construction algorithm requires $O(V)$ of storage: the max-tree mask, a quantised version of the max-tree mask, an array of sorted voxels, a union-find data structure, the pilot max-tree, and the final max-tree. This results in a memory consumption of approximately $6Vd$, assuming that intensity values and indices have the same bit depth (for example, single-precision floating point numbers for intensities and 32-bit integers for indices). Note that the number of max-tree nodes is $O(V)$; this is not just an upper bound, as floating-point volumes of volume V may very well contain V different floating-point values, requiring V max-tree nodes. This does not include the space required to store attributes. The current implementation stores several of those, including: volume (64-bits integer), nine attributes required to compute the moment-of-inertia tensor (64-bits integers), six attributes to facilitate the size filter (16-bits integers), the sum of intensities (64-bits floating-point number), the sum of squared intensities (64-bits floating-point number), the maximum intensity (32-bits floating-point number), and the maximum max-tree mask intensity (32-bits floating-point number). Taking all this together, for the volume of chapter 6, results in $(6{\times}32{+}9{\times}64{+}6{\times}16{+}2{\times}64{+}2{\times}32){\times}V/(8{\times}2^{30}) \approx 23.3$ gigabytes. While not all of this needs to be in memory at the same time, it likely will not fit in the memory of a regular desktop PC. Being more selective about which attributes to store could reduce memory usage; for example, if the LVQ1 classifier is used, attributes that have no significant effect on the classification can be omitted.

Finally, there is the segmentation step. The time complexity is at least $O(V)$, as all $O(V)$

max-tree nodes need to be visited in order to test them for significance. Additionally, it is necessary to find the closest significant ancestor for each node, to prevent any significant node with significant ancestor from being marked as an object. For this, a union-find algorithm is used that is of amortised quasi-linear time complexity. Also, the dominant descendant for each significant node needs to be determined, to facilitate the move-up step, but this is trivial to do in linear time once the closest significant ancestor is known for all nodes. All in all, the segmentation step is of a quasi-linear time complexity. Some parts of the segmentation can easily be parallelised, such as the significance tests, the move-up step, and size filter. Others, such as the identification of significant ancestors, cannot and were left sequential for this project. No additional memory is required, except when the results are written to files, some intermediate storage may be necessary. The move-up step and size filter can be done in $O(V)$.

In conclusion, no part of the algorithm has a quadratic (or worse) dependency on the number of voxels V. This indicates that this algorithms should scale and allow very large inputs to be processed, given enough processing power.

8.2 Performance measurements

The MT source finder was implemented in C and parallelised with a combination of POSIX threads and OpenMP. Time measurements were performed on a shared-memory Dell R815 rack server with four 16-core AMD Opteron processors and 512 GB RAM. It has 32 floating-point units (FPUs), as this CPU model shares one FPU per pair of cores. Two sets of performance measures were performed on chapter 6's radio volume (using the same parameterisation that resulted in highest completeness), one where the number of threads is varied and one where the resolution is varied. In the latter case, the volume is sliced over the x-axis in order to simulate datasets of different resolutions. When the number of threads is varied, the full volume is used, while only one thread is used when the volume is sliced. The measurements do not include computing the elongation and flatness attributes, nor any I/O time. Each measurement was done five times and the minimum was retained.

The results are shown in figs. 8.1 to 8.3. The pre-processing time is the time spent on the initial Gaussian filter. This uses Scipy's implementation and is not parallelised (fast, parallel implementations are available). Also included are the times spent on the Perona–Malik diffusion, max-tree construction, and segmentation. Total time is the sum of the previous timings. Figure 8.1 shows that the running time seems to increase linearly with the number of voxels, although the max-tree construction step experiences a slowdown at the highest resolution considered. The max-tree construction is the dominant part of the algorithm.

Figure 8.2 shows the running times when varying the number of threads and fig. 8.3 shows the resulting speed-up. The Perona–Malik diffusion step, which is most easily parallelised, shows the greatest speed-up. The speed-up is not optimal, which is especially noticeable when the number of threads is larger than eight. This is somewhat dissappointing, as there is little interaction between the threads for this step (only a barrier after each iteration), so a large speed-up is expected. The segmentation step shows the worst speed-up, which is unsurprising, since not all parts of it are parallelised, so once the sequential parts start to dominate, the speed-up will not increase significantly. The max-tree construction step shows a modest speed-up. The diplomatic max-tree

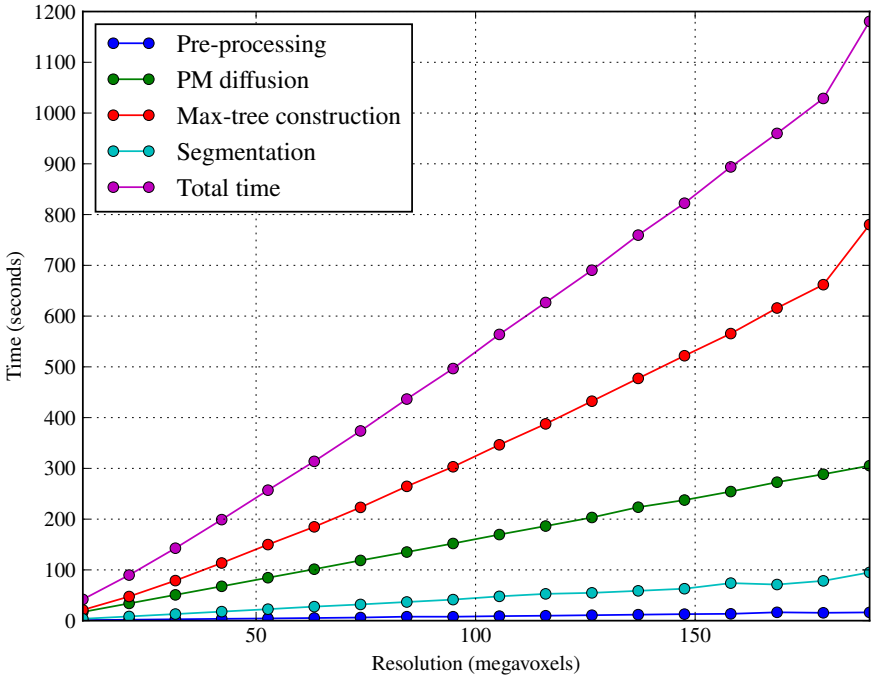

Figure 8.1: Running times of the MT source finder for volumes of different resolutions. This plot confirm the no-worse-than linearithmic time complexity with respect to the number of voxels.

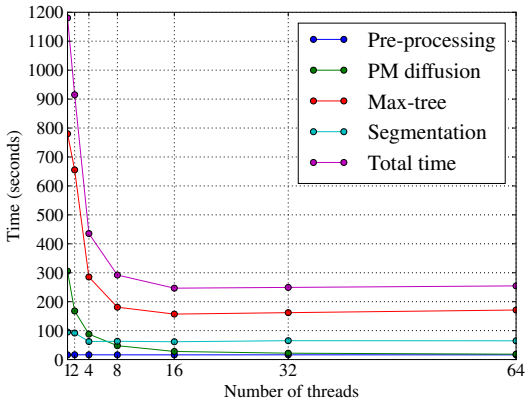

Figure 8.2: Running times using the MT source finder for different numbers of threads.

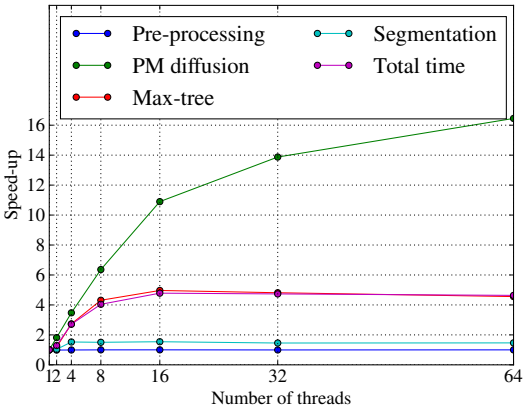

Figure 8.3: Speed-up resulting from parallelisation. Perona–Malik shows the largest speed-up. The theoretical optimum is not attained.

construction algorithm distributes voxels over threads using grey level, which is not optimal here, as the radio volumes do not have a uniform distribution of grey values (refer to fig. 6.1), lower values being more common. In fact, since the mask-tree mask has all of its negative values set to zero, one may even suspect a speed-up of at most two. Clearly, this is an oversimplified picture, as the actual speed-up is greater. The max-tree construction step seems to experience a *slowdown*, when increasing the number of threads beyond sixteen. This is likely caused by additional multi-threading overhead after the speed-up has been exhausted. Additionally, the limited number of FPUs could have a detrimental effect on performance for the case of 64 threads. Here, this probably is not a problem, as the best speed-up is already obtained with sixteen threads.

Executing SoFiA's S+C source finder on the full volume took approximately five minutes seconds (without I/O), on the same hardware as above. This implementation is not parallelised, although it is likely possible to parallelise at least the smoothing filters. As mentioned in chapter 6, the wavelet-based algorithms are much slower, taking hours if not days to complete (without parallelisation) on the hardware considered here. It should be noted that the running time of S+C and the wavelet-based methods are highly dependent on the number of scales (or smoothing kernels) used. For S+C's, 12 were used here (the default setting). S+C's running time is comparable to that of the MT source finder with parallelisation turned on.

8.3 Discussion

The MT source finder should be able to attain linearithmic performance in the number of voxels of the input volume, indicating that it should scale to process larger datasets. Also, its performance is not much worse than that of a comparable method, namely S+C. The speed-up resulting from parallelisation is not optimal, as a speed-up of about five it attained for sixteen threads. Therefore, investigating further performance improvements could prove worthwhile. Possible directions for improvement include a better distribution of voxels over threads in case of a skewed grey value distribution with many duplicates. Techniques for more efficient memory access can also be investigated.

9 Conclusion and future work

Referring back to the research questions in the introduction, the following conclusions can be drawn.

Smoothing It is possible to incorporate smoothing in the MT source finder, using the novel technique termed mask-like connectivity. Intensity-driven and gradient-magnitude-driven (that is, traditional) Perona–Malik diffusion were found to be usable in this context, although their performance in terms of the final segmentation is not too different from simple Gaussian smoothing. Nevertheless, intensity-driven diffusion was found to give the best results in terms of completeness, reliability, and fragmentation of the final segmentation results. However, qualitatively, gradient-magnitude-driven and simple Gaussian smoothing were found to give better results, as the shape of objects is better preserved by these methods.

Statistical models Compared to the use case of optical datasets, the radio datasets contain correlated noise and as a result, a simple χ^2-based statistical model for the power attribute no longer suffices. Two improved models were proposed, one based on the flux density attribute and a normal distribution, as well as one based on the sum of squared fluxes, using a Γ-distribution. The flux-density-based model was found to offer superior performance, mainly because it is much less sensitive to outliers (as they are averaged out). Both of these models are empirical in nature and parameters are estimated from noise volumes.

Segmentation results The results of the MT source finder are somewhat similar to the results of a state-of-the-art method (S+C), albeit slightly worse in terms of completeness. The adaptive smoothing methods produce slightly better results than simple Gaussian smoothing, but have more parameters to tune. The value of the K-parameter can be tuned according to the distribution of the input volume. The $t_{\max}$-parameter is more difficult, although a relation to the σ_G of Gaussian smoothing exists for low-intensity regions of the input. The move-up factor is another difficult parameter, but its effect on the segmentation is mostly cosmetic, so its value is less critical. All in all, more research into the parameters is necessary, plus possible algorithmic enhancements, to make the MT source finder more competitive compared to competing methods.

Identifying true detections Two machine learning techniques were considered as an answer to the huge number of false detections that source finders produce (which is expected due to the size of the input), namely classification trees and an LVQ1-based nearest-prototype classifier. While both methods fare less well than Serra et al. [2012]'s reliability filter in terms of true positives (true detections classified as true), the LVQ1-based nearest-prototype classifier attains an almost perfect score in terms of precision. If the proposed training scheme based on negative-total-flux

detections and artificially-planted sources turns out to work, it could be a valuable tool for high-precision source identification.

Computational performance Computational performance was not investigated in great detail, but was found to be linearithmic in nature with respect to the number of voxels and comparable to the S+C method, while wavelet-based methods are far slower. In terms of memory usage, a regular desktop PC will likely not suffice in the near future, but this is unavoidable without greatly reducing the precision of the attributes computed. Computational performance can possibly be improved, as the parallel speed-up has not yet reached close-to-optimal levels.

9.1 Future work

While the method already performs similarly to some existing methods, perhaps its greatest potential lies in its extensibility. The max-tree data structure allows many attributes to be computed in a convenient manner and it contains a hierarchical view of the features in the input. As mentioned above, the most important work that is left is a more detailed analysis of all the parameters involved and development of any algorithmic enhancements that make the MT source finder a competitive alternative for the existing methods. Additionally, the implementation leaves some room for performance improvements. More suggestions for future work are done below:

Better validation So far, the performance of the MT source finder has only been tested on one radio volume, which is somewhat artificial as objects detected using another segmentation algorithm were planted in a pure-noise radio volume, which is itself not guaranteed to be clean. More experimentation, preferentially on real datasets, should confirm the utility of the MT source finder. They should also be able to confirm whether the machine learning techniques considered can be applied in practice (on datasets without reference). Should the LVQ1-based nearest-prototype classifier give promising results, generalised or optimised variants of LVQ can be considered.

Nested features Given a suitable classification model, the MT source finder can readily be used to detect nested objects, which is something that has not yet been considered for radio volumes, in the author's knowledge. While this possibility was not explored here, as the radio volumes were assumed to be free of object nesting, it could perhaps be useful in the future. Moreover, it has already been successfully applied in the optical case.

Alternative input filters The mask-like approach to connectivity allows arbitrary filters to be used to drive the creation of the max-tree. While there is a strong focus on (adaptive) smoothing filters in this thesis, any kind of image filter can be used. A possibility that could be explored are morphological operators, perhaps based on viscous filters. Care should be taken however, when computing attributes that depend on the mask intensity. Here, a local average is used, which ensures that the mask and original intensities remain comparable. Staying with smoothing, more variants of Perona–Malik diffusion can be tried, for example using iteration-adaptive K-values.

Additionally, since radio volumes are acquired in the Fourier domain and then transformed to get the spatial representation, perhaps noise filtering would be more effective in the Fourier

domain. Although in that case, there is the problem that sources are not localised in the Fourier domain, perhaps making it harder to construct techniques that do not damage relevant parts of the signal.

More sophisticated statistical models There are also alternative statistical models or changes to the existing models that can be experimented with. For example, this project assumes that volume determines the statistical parameters of a max-tree node. Other attributes could be used here, for example max-tree mask intensity. Instead of empirically motivated models, a more analytic approach could perhaps be applied as well. Excursion sets can be used to describe the probability that all elements of a set exceed a particular value (a reference on excursion sets is Adler [2000]). This could be used in conjunction with true mask-based connectivity, as the max-tree nodes are excursion sets in that case. One problem that needs to be taken care of are the noise correlations, as excursion set theory usually considers independent noise.

If the empirical models are retained, it would be a good idea to quantify how well the proposed models fit the noise, using a goodness-of-fit measure. This measure needs to be robust to the estimated nature of the parameters.

Bibliography

Robert J. Adler. On excursion sets, tube formulas and maxima of random fields. *Annals of Applied Probability*, pages 1–74, 2000.

Ch. Berger, T. Geraud, R. Levillain, N. Widynski, A. Baillard, and E. Bertin. Effective component tree computation with application to pattern recognition in astronomical imaging. In *Image Processing (ICIP), 2007. IEEE International Conference on*, volume 4, 2007. doi: 10.1109/ICIP.2007.4379949. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4379949`.

Emmanuel Bertin and S. Arnouts. SExtractor: Software for source extraction. *Astronomy and Astrophysics Supplement Series*, 117(2):393–404, 1996.

John Canny. A computational approach to edge detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(6):679–698, 1986. doi: 10.1109/TPAMI.1986.4767851. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4767851`.

Lars Flöer and Benjamin Winkel. 2D–1D wavelet reconstruction as a tool for source finding in spectroscopic imaging surveys. *Publications of the Astronomical Society of Australia*, 29(3): 244–250, 2012.

J.A. Högbom. Aperture synthesis with a non-regular distribution of interferometer baselines. *Astronomy and Astrophysics Supplement Series*, 15:417, 1974.

R. Hummel and R. Moniot. Reconstructions from zero crossings in scale space. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 37(12):2111–2130, 1989. doi: 10.1109/29.45555. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=45555`.

Russell Jurek. The characterised noise Hi source finder: Detecting Hi galaxies using a novel implementation of matched filtering. *Publications of the Astronomical Society of Australia*, 29 (3):251–261, 2012.

S. Lloyd. Least squares quantization in PCM. *IEEE Transactions on Information Theory*, 28(2): 129–137, 1982. doi: 10.1109/TIT.1982.1056489. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=1056489`.

Thomas P. Minka. Beyond Newton's method. Online, 2000. URL `http://research.microsoft.com/en-us/um/people/minka/papers/minka-newton.pdf`.

Thomas P. Minka. Estimating a gamma distribution. Online, 2002. URL `http://research.microsoft.com/en-us/um/people/minka/papers/minka-gamma.pdf`.

Ugo Moschini, Paul Teeninga, Michael H.F. Wilkinson, Nadine Giese, Davide Punzo, Jan M. van der Hulst, and Scott C. Trager. Towards better segmentation of large floating point 3D astronomical data sets: first results. In *Proceedings of the 2014 conference on Big Data from Space (BiDS'14)*, pages 232–235. Publications Office of the European Union, 2014.

Ugo Moschini, Arnold Meijster, and Michael H.F. Wilkinson. A diplomatic shared memory parallel max-tree algorithm for high dynamic-range images. Unpublished, 2015a.

Ugo Moschini, Paul Teeninga, Scott C. Trager, and Michael H.F. Wilkinson. Parallel 2D local pattern spectra of invariant moments for galaxy classification. Unpublished, 2015b.

G.K. Ouzounis and M.H.F. Wilkinson. Mask-based second-generation connectivity and attribute filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(6):990–1004, 2007. doi: 10.1109/TPAMI.2007.1045. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4160950`.

G.K. Ouzounis and M.H.F. Wilkinson. Hyperconnected attribute filters based on k-flat zones. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(2):224–239, 2011. doi: 10.1109/TPAMI.2010.74. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=5432219`.

P. Perona and J. Malik. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639, 1990. doi: 10.1109/34.56205. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=56205`.

Attila Popping, Russell Jurek, Tobias Westmeier, Paolo Serra, L. Flöer, Martin Meyer, and Baerbel Koribalski. Comparison of potential ASKAP Hi survey source finders. *Publications of the Astronomical Society of Australia*, 29(03):318–339, 2012.

J. Ross Quinlan. Bagging, boosting, and C4.5. In *AAAI/IAAI, Vol. 1*, pages 725–730, 1996.

P. Salembier, A. Oliveras, and L. Garrido. Antiextensive connected operators for image and sequence processing. *IEEE Transactions on Image Processing*, 7(4):555–570, 1998. doi: 10.1109/83.663500. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=663500`.

P. Serra, R. Jurek, and L. Flöer. Using negative detections to estimate source-finder reliability. *Publications of the Astronomical Society of Australia*, 29(03):296–300, 2012.

Paul Teeninga, Ugo Moschini, Scott C. Trager, and Michael H.F. Wilkinson. Bi-variate statistical attribute filtering: a tool for robust detection of faint objects. In *11th International Conference "Pattern Recognition and Image Analysis: New Information Technologies" (PRIA-11-2013)*, pages 746–749. IPSI RAS, 2013.

Paul Teeninga, Ugo Moschini, Scott C. Trager, and Michael H.F. Wilkinson. Improving background estimation for faint astronomical object detection. In *Image Processing (ICIP), 2015. IEEE International Conference on*, 2015a. Submitted.

Paul Teeninga, Ugo Moschini, Scott C. Trager, and Michael H.F. Wilkinson. Improved detection of faint extended astronomical objects through statistical attribute filtering. Unpublished, 2015b.

M.A. Westenberg, J.B.T.M. Roerdink, and M.H.F. Wilkinson. Volumetric attribute filtering and interactive visualization using the max-tree representation. *IEEE Transactions on Image Processing*, 16(12):2943–2952, 2007. doi: 10.1109/TIP.2007.909317. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=4376245`.

Matthew T. Whiting. DUCHAMP: a 3D source finder for spectral-line data. *Monthly Notices of the Royal Astronomical Society*, 421(4):3242–3256, 2012.

M.H.F. Wilkinson. A fast component-tree algorithm for high dynamic-range images and second generation connectivity. In *Image Processing (ICIP), 2011. 18th IEEE International Conference on*, pages 1021–1024, 2011. doi: 10.1109/ICIP.2011.6115597. URL `http://ieeexplore.ieee.org/stamp/stamp.jsp?arnumber=6115597`.