Assessing The Performance of Sub-Millimetre Compact Object Detection Algorithms

Mark E. Watson

Submitted to the University of Hertfordshire in partial
fulfillment of the requirements of the degree of
M.Sc. by Research in Astrophysics

March, 2010

Abstract

Sub-millimetre astronomy is about to be transformed by the deployment of new detectors that can map larger images than previously possible. A particular issue in sub-millimetre astronomy is the automated detection of compact, irregular regions of emission known as clumps. There are numerous clump detection software packages freely available yet there is little consensus as to which is the most appropriate to use, as each package has its own systematic bias when performing clumpfinding. The purpose of this investigation was to investigate a number of these clumpfinding packages and determine where some of these biases may lie.

The CUPID package is designed for the creation and detection of clumps within images. There are four algorithms for the detection of clumps; ClumpFind, FellWalker, GaussClumps, and Reinhold. Each algorithm was individually investigated using data from SCAMPS (the SCUBA Massive Precluster Survey), (Thompson et al., 2005) to determine the effect of changing their parameters; the algorithms were then compared against each other to examine how the results differed between them.

Using Monte Carlo simulations, Gaussian artificial clumps (with known peak, size, location, and integrated flux) were inserted into an image and the algorithms were tested to determine which algorithm extracted the information with the greatest accuracy, and where the completeness limits lie with each algorithm. ClumpFind, FellWalker, and Reinhold detected a lower integrated flux level than was inserted; this effect was more evident in large, flat clumps. Due to the profile of the clump it was expected that GaussClumps would detect the integrated flux more correctly, as was the proven case.

Acknowledgments

Foremost I would like to thank my supervisors; Dr Mark Thompson and Dr Jason Stevens, for their invaluable assistance throughout the course. The STRI and its members for providing me with all the opportunities and resources that have helped me in all my endeavours. Also to David Berry, the creator of the CUPID package, who was a great help in understanding how the package works and resolving any issues I encountered with CUPID.

I would also like to thank my parents; firstly for my existence, then the financial support and encouragement ever since which has helped alleviate any potential difficulties I might have faced without their continued assistance.

Finally I would like to thank Tobias, my cat, without whom inspiration would have certainly been lacking.

Contents

Abstract **i**

Acknowledgments **ii**

List of Figures **vi**

List of Tables **vii**

1 Introduction **1**

1.1 Overview 1
1.2 What is CUPID? 1
1.3 What is a Clump? 2
1.4 Why Use Automated Programs? 3
1.5 SCUBA 3
1.6 SCAMPS 4
1.7 Previous Investigations of Clumpfinding Algorithms 5

2 The Algorithms **7**

2.1 Overview 7
2.1.1 ClumpFind 7
2.1.2 FellWalker 8
2.1.3 GaussClumps 10
2.1.4 Reinhold 11
2.2 The Algorithm Parameters 12
2.2.1 ClumpFind 13
2.2.2 FellWalker 14
2.2.3 GaussClumps 15
2.2.4 Reinhold 15

3 Algorithm Parameter Investigation **17**

3.1 Overview 17
3.2 Varying Parameters 17
3.2.1 Parameter Value Constraints for all Algorithms 18
3.2.2 ClumpFind 19
3.2.3 FellWalker 24
3.2.4 GaussClumps 35
3.2.5 Reinhold 44
3.3 Conclusions 56

4 Algorithm Cross Comparison 58
4.1 Overview . . . 58
4.2 The Comparison . . . 62
4.2.1 Peak Value . . . 62
4.2.2 Size of The Clump . . . 66
4.2.3 Integrated Flux Value . . . 70
4.3 Conclusion . . . 74

5 Tests Using Artificial Clumps 76
5.1 Overview . . . 76
5.2 Implementation . . . 76
5.2.1 The Image . . . 76
5.2.2 The Artificial Clump . . . 77
5.3 Artificial Clump Properties . . . 79
5.3.1 Algorithm Completeness . . . 79
5.3.2 Peak Value and Clump Size . . . 80
5.3.3 Peak Depth . . . 86
5.3.4 Clump Location . . . 89
5.4 Monte-Carlo Simulations . . . 91
5.4.1 Recovery of an Input Flux Distribution . . . 91
5.4.2 Results . . . 93
ClumpFind . . . 93
FellWalker . . . 96
GaussClumps . . . 99
Reinhold . . . 102
Fraction of Missing Flux . . . 105
5.5 Conclusion . . . 108

6 Conclusions and Future Work 112
6.1 Conclusions . . . 112
6.2 Future Work . . . 115

References 116

A RMS Values for SCAMPS Images 119

B Artificial Clump IDL Code 122

C SCAMPS Image Catalogue 124

List of Figures

1.1 SCAMPS image g1015-034 2

2.1 ClumpFind clump detection process 8
2.2 FellWalker route process 9
2.3 FellWalker MinDip parameter 10
2.4 GaussClumps Gaussian profile 11

3.1 ClumpFind DeltaT results 20
3.2 ClumpFind TLow results 21
3.3 ClumpFind IDLAlg results 23
3.4 FellWalker Noise results 25
3.5 FellWalker FlatSlope results 27
3.6 FellWalker CleanIter results 29
3.7 FellWalker MinDip results 31
3.8 FellWalker MaxJump results 34
3.9 GaussClumps NPad results 36
3.10 GaussClumps MaxNF results 38
3.11 GaussClumps ModelLim results 41
3.12 GaussClumps FwhmStart results 42
3.13 GaussClumps Thresh results 43
3.14 Reinhold Noise results 45
3.15 Reinhold MinLen results 47
3.16 Reinhold Thresh results 48
3.17 Reinhold FixClumpIterations results 49
3.18 Reinhold CAIterations results 51
3.19 Reinhold FlatSlope results 53
3.20 Reinhold CAThresh results 55

4.1 Clump profiles for a 450μm image for each algorithm using the cross comparison parameters 60
4.2 Clump profiles for a 850μm image for each algorithm using the cross comparison parameters 61
4.3 Peak value matching for 450μm data 63
4.4 Peak value matching for 850μm data 64
4.5 Algorithm peak value cross comparison 65
4.6 Clump size matching for 450μm data 67
4.7 Clump size matching for 850μm data 68
4.8 Algorithm clump size cross comparison 69
4.9 Flux value matching for 450μm data 71

4.10 Flux value matching for 850μm data . . . 72
4.11 Algorithm flux value cross comparison . . . 73

5.1 SCAMPS image used for Monte-Carlo simulations . . . 77
5.2 3D plot showing the completeness of the ClumpFind algorithm . . . 81
5.3 3D plot showing the completeness of the FellWalker algorithm . . . 82
5.4 3D plot showing the completeness of the GaussClumps algorithm . . . 83
5.5 3D plot showing the completeness of the Reinhold algorithm . . . 84
5.6 Peak depth for each algorithm using 450μm data . . . 87
5.7 Peak depth for each algorithm using 850μm data . . . 88
5.8 SCAMPS image for Monte Carlo simulation with contours . . . 89
5.9 Image of boundary area for the insertion of artificial clumps . . . 90
5.10 Constant flux, artificial clump plots, ClumpFind 450μm . . . 94
5.11 Constant flux, artificial clump plots, ClumpFind 850μm . . . 95
5.12 Constant flux, artificial clump plots, FellWalker 450μm . . . 97
5.13 Constant flux, artificial clump plots, FellWalker 850μm . . . 98
5.14 Constant flux, artificial clump plots, GaussClumps 450μm . . . 100
5.15 Constant flux, artificial clump plots, GaussClumps 850μm . . . 101
5.16 Constant flux, artificial clump plots, Reinhold 450μm . . . 103
5.17 Constant flux, artificial clump plots, Reinhold 850μm . . . 104
5.18 Plots showing the constant flux percentage difference for the 450μm results . . . 106
5.19 Plots showing the constant flux percentage difference for the 850μm results . . . 107
5.20 Constant flux Gaussian noise cut-off . . . 108
5.21 Graphical representation of the clump mass function . . . 111

C.1 Catalogue of the SCAMPS images used in this investigation. . . . 124

List of Tables

4.1 Parameter values for the ClumpFind algorithm that gave the best concordance with the other algorithms for the cross comparison. 59
4.2 Parameter values for the FellWalker algorithm that gave the best concordance with the other algorithms for the cross comparison. 59
4.3 Parameter values for the GaussClumps algorithm that gave the best concordance with the other algorithms for the cross comparison. 59
4.4 Parameter values for the Reinhold algorithm that gave the best concordance with the other algorithms for the cross comparison. 59

5.1 RMS values of the image used for the insertion of the artificial clump. 77
5.2 Range of Peak values used with the corresponding input flux values for the 450μm investigation. 92
5.3 Range of Peak values used with the corresponding input flux values for the 850μm investigation. 92

A.1 RMS values for the 450μm images as calculated by CUPID, and myself using GAIA. 120
A.2 RMS values for the 850μm images as calculated by CUPID, and myself using GAIA. 121

Chapter 1

Introduction

1.1 Overview

Throughout this project clumpfinding algorithms in the CUPID package (Berry et al., 2007)[1] were tested to determine their accuracy and completeness. Data collected in SCAMPS (the SCUBA Massive Precluster Survey) (Thompson et al., 2005) was used initially. Later in the investigation artificial sources were created to allow for a more thorough analysis of these algorithms. This chapter discusses the basics of clumpfinding in sub-millimetre astronomy and why it is required.

1.2 What is CUPID?

CUPID is a Starlink[2] package created by David Berry for the purpose of clump identification and analysis. The aim of the package was to bring together numerous clump identification algorithms from the literature thus allowing an easy and direct comparison between them. To enable this, the output for each algorithm is designed to be of a similar and consistent format. CUPID contains five main commands, which are:

Extractclumps calculates the parameters for the clumps within an array provided a mask array is supplied which identifies all of the pixels within each clump.

Findback spatially filters the array to remove features that are smaller then a specified size. This produces an estimate of the background level within the image.

Findclumps finds clumps within an array. This outputs a mask with each clump indexed and identifies all the pixels contained within the clump.

Makeclumps creates an array containing random Gaussian clumps, and noise.

[1]Complete details of the CUPID software can be found in the CUPID users manual by David Berry, this can be found at http://starlink.jach.hawaii.edu/docs/sun255.htx/sun255.html (Version 0.0-38, created 6th March 2009).

[2]Starlink software is freely available astronomical data reduction and analysis software, available from http://starlink.jach.hawaii.edu.

Outlineclump draws an outline on an image around a two-dimensional clump.

This investigation focused on the findclumps command. Findclumps contains four different algorithms for detecting clumps; ClumpFind (Williams et al., 1994), FellWalker (Berry et al., 2007), GaussClumps (Stutzki & Guesten, 1990) and, Reinhold (Berry et al., 2007). Further algorithms can easily be added to the software should the user deem it necessary. ClumpFind and GaussClumps have been previously published and used independently, see §1.7. In order to implement ClumpFind and GaussClumps into the CUPID package, they needed to be re-written to perform the clumpfinding process in the same way and give similar results to the original algorithms, (Berry et al., 2007). FellWalker and Reinhold are newer algorithms and CUPID is their first implementation.

Dent et al. (2009) used the CUPID package, particularly the GaussClumps algorithm for the process of clumpfinding of regions within the Rosette Molecular Cloud. The results from the clumpfinding were used to determine the clump mass function of the whole map, inner region, and the expanding ring.

1.3 What is a Clump?

There is much debate as to the true nature of a clump, but it is generally understood that a clump is an area of high density within a larger area of lower density. This investigation will therefore assume this particular definition of a clump to be true.

Star formation is found in all types of molecular cloud, these clouds have been determined to be inhomogeneous in their density with pockets of high density matter being separated by low density interclump gas e.g. (Blitz & Shu, 1980). These high density areas have been shown to be brighter in their emission (Blitz & Stark, 1986) and this localised emission is what is currently being considered to be a clump. Each image may either contain only a single source or a more complex cloud with multiple sources and therefore multiple clumps. Figure 1.1 shows one of the images which contains many clumps. To perform well the clumpfinding algorithm must locate each of these individual clumps and accurately measure their emission. A catalogue of all the SCAMPS images used in this investigation can be found in Appendix C.

1.4 Why Use Automated Programs?

The mass of a clump is calculated from the total flux of the clump, for a Gaussian clump the total flux can be calculated from the peak flux and the size of the clump (Williams et al., 1994). If a clump is sufficiently isolated it is expected that the profile of the clump could be accurately determined and catalogued by eye. As the image becomes more crowded with clumps there arises the problem of blended emission. In this case subjective biases become more evident as each person could perceive the data differently and thus identify different clumps. Automated

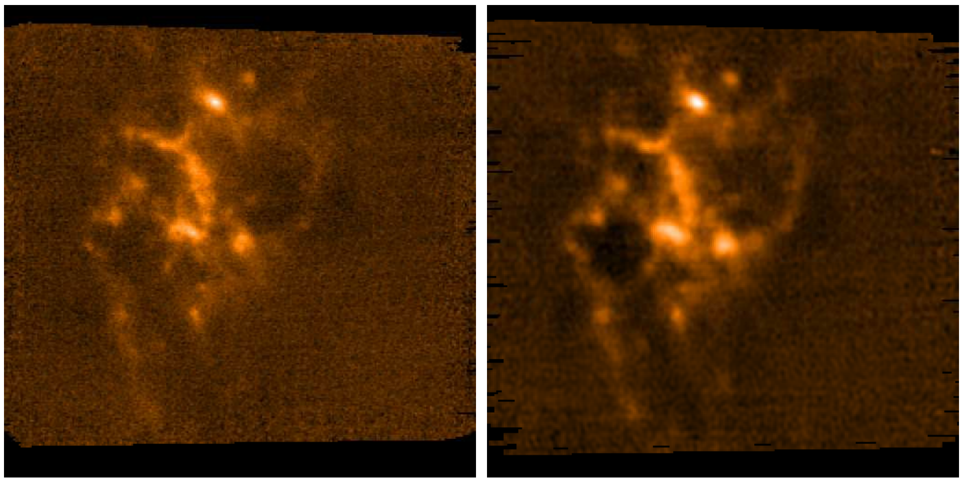


Figure 1.1: Figure shows one of the 31 SCAMPS fields, g1015-034, left; 450μm image, right; 850μm image.

clump detection algorithms remove the individual bias to provide consistent and objective results for different investigations.

Until recently current sub-millimetre detectors have only been able to produce a relatively small number of images that require clumpfinding to be performed on them. Although clumpfinding could have been performed by eye on these images it would be quicker, easier, and more objective to use an automated program. With the installation of new instruments such as SCUBA-2 and Herschel the number of images requiring clumpfinding will dramatically increase, as will the number of clumps that are detected. This increase would be too great to deem performing the clumpfinding process by eye a feasible option, therefore automated clumpfinding programs are required to cope with the increased amount of information being supplied.

Automated algorithms remove the human bias only if the same algorithm is used by everyone with the parameters set at the same values. Presently this is not the case as there are many different clumpfinding algorithms in use, each algorithm with its own bias, and many studies using slightly different combinations of parameters. Though, since different results are being achieved there is constant competition to be more accurate, thereby causing improvements in the clump detection software to be made regularly.

1.5 SCUBA

SCUBA (the Sub-millimetre Common-User Bolometer Array) was mounted upon the JCMT (James Clerk Maxwell Telescope), located near the summit of Mauna Kea in Hawaii. A brief outline of SCUBA will be given here, however it is fully described in Holland et al. (1999). At the time of its conception it was the most powerful and versatile sub-millimetre camera of its generation. It contained a dual-waveband imaging system along with a three-band photometer. The dual-waveband consisted of a short wave array and a long wave array; the short wave array contained 91 detectors, the long wave array 37 detectors along with the three photometric pixels. Its operating wavelength range was between 350μm and 2mm with two wavelengths being able to be acquired simultaneously; one short e.g. 450μm, and one long e.g. 850μm. The SCUBA data-reduction pipeline was based upon the UKIRT ORAC system. This system could be used off-line allowing for automatic data reduction of the SCUBA images.

SCUBA was able to run in four different observing modes:

Photometry was used for observing an isolated point source with a known location. For this either a single detector in the array was used, or one of the photometry pixels.

Jiggle-mapping was used either for extended sources, that are smaller than the full field of view of the arrays, or for the detection of a point source when its exact location was not precisely known.

Scan-mapping was used for the mapping of regions larger than the field of view of the array.

Polarimetry was used to measure the polarization of the emission; when used additional equipment was required i.e. a photo-lithographic analyser.

Shortly after SCUBA was installed on the JCMT the filter wheel in SCUBA became stuck. This resulted in the measurable wavelengths being fixed at 450μm and 850μm.

1.6 SCAMPS

SCAMPS was a survey performed using SCUBA, the purpose of which, was to locate a number of massive precluster cores using their sub-mm emission, and from the data collected determine the physical properties of these cores.

SCAMPS imaged 32 Ultra-Compact (UC), HII regions using SCUBA in the scan-mapping observation mode (Thompson et al., 2005). These regions were imaged in both 450 and 850μm simultaneously. Each image contained at least one bright source and many contain multiple sources, Figure 1.1 shows one of these images. Each image is a 256$\times$256 pixel two-dimensional array with each pixel 3$\times$3 arcseconds in size. The scan mapping process works by scanning across a region whilst chopping in a fixed direction of the sky to produce a differential map of the source (Holland et al., 1999). From the work of Emerson (1995) data is taken in six different chop configurations; chop throws of 20, 30, and 65 arcseconds each with a chop position of zero and 90 degrees are recommended (Jenness & Lightfoot, 2003). Although all six of these configurations give the best coverage for the spatial frequencies, using four gives an acceptable map. With a created map the next stages consist of the baseline and sky removal, this is not a simple task since most of the bolometers view at least part of the source at some point during the mapping procedure. To determine the sky level the original source is removed from the images, this is done using calcsky [3]. The process of chopping

[3]Calcsky is a function for determining the background sky value by first removing any sources in the image, full details can be found at: http://starlink.jach.hawaii.edu/docs/sun216.htx/node59.html (SCUBA User Reduction Facility 1.6, created 3rd April 2003).

whilst scanning produces an image containing a positive and a negative image of the source, to resolve this, the chop must be deconvolved from the original chop by rebinning the images from each chop configuration. Once this is done the images are processed using remdbm [4]. This completes the data reduction performed by SCUBA. The images used in this investigation also underwent a flattening procedure to remove the large scale background fluctuations due to the largest chopthrow, this produced images as shown in Figure 1.1. The flattening was performed by first removing any sources brighter than five sigma, after this was done the image was smoothed using a Gaussian twice the size of the largest chop size. This smoothed image was then subtracted from the original image, as is described in Johnstone et al. (2000) and Reid & Wilson (2005).

The aims of the SCAMPS survey were:

- To carry out an unbiased census of warm and cold massive dust cores located in the same star-forming complexes as known UC HII regions.
- To estimate their temperatures from the MSX and SCUBA SED, and hence measure their dust masses and densities.
- To identify cold massive MSX-dark precluster candidates for further study.

1.7 Previous Investigations of Clumpfinding Algorithms

This investigation is concerned with the accuracy and completeness of the clumpfinding algorithms contained within CUPID. Previous investigations have used these algorithms, some have tested their completeness whereas others have used them for the detection of clumps and used the results as part of a further investigation.

ClumpFind is one of the more widely used algorithms for clump detection. Its first implementation was by the creators of the algorithm in Williams et al. (1994). since then ClumpFind has been investigated and used many times for the process of clump detection in, for example, Enoch et al. (2006), Rathborne et al. (2009), Kainulainen et al. (2009), Reid & Wilson (2005), Massi et al. (2007), Friesen et al. (2009), Pineda et al. (2009), and Muñoz et al. (2007). Each investigation chose their own values for each of the parameters described in §2.2 (specifically the DeltaT and TLow parameters). Rathborne et al. (2009), Kainulainen et al. (2009), and Pineda et al. (2009) also performed an investigation into varying the parameters and observing their effects. The results showed that larger contour gaps were able to determine large bright structures more accurately but narrower contour levels provided an increased level of detail for dimmer structures. Enoch et al. (2006) performed investigations into the completeness of the ClumpFind algorithm and details how ClumpFind interpolates clumps at various peak values. They found that fewer faint sources were detected with ClumpFind compared to the

[4]Remdbm is a function for the reduction of scan map images by the use of the Fast Fourier techniques as described in Emerson (1995).

Peak-find algorithm, also that many of the brighter sources were broken up into multiple clumps by ClumpFind.

GaussClumps being the oldest of the algorithms in the package was first published in Stutzki & Guesten (1990) having been created by the authors. GaussClumps has been used both independently and in conjunction with ClumpFind for the clumpfinding process. Independent investigations have been performed by Schneider et al. (1998) and Lo et al. (2009). Schneider et al. (1998) used GaussClumps to determine a value for the clump mass spectrum of the Rossette Molecular Complex whereas Lo et al. (2009) used GaussClumps to examine the clumping within the Giant Molecular Cloud.

Though working by different methods; ClumpFind contouring the data and determining the clumps from the contour information, and GaussClumps using the data to model and fit Gaussian clump profiles. They have both been widely used, with GaussClumps being used in a greater number of investigations than ClumpFind, though this could be due to the longer period that GaussClumps has been in existence. Investigations involving both GaussClumps and ClumpFind have be detailed in Mookerjea et al. (2004) and Schneider & Brooks (2004). Both of these investigations used ClumpFind and GaussClumps separately to determine the number of clumps and the clump mass spectrum. The results indicated that both algorithms, although working by different means and obtaining a different number of clumps (GaussClumps detecting almost ten times as many clumps as ClumpFind), obtained a similar mass spectrum of the detected clumps.

Chapter 2

The Algorithms

2.1 Overview

CUPID can be used for clump identification and analysis of two and three-dimensional data arrays; a clump being an area of detected emission within an image. This investigation focused on the identification of clumps in two-dimensional arrays, specifically using the 'findclump' routine from CUPID. The CUPID findclumps process requires an input of an NDF (extensible N-dimensional Data Format) file (input file) with the original data image and outputs an NDF file (output file) showing the area of the clumps and in most cases the clump index, where each pixel belonging to each clump is identified and numbered. Findclumps also outputs a fits table (outcat file) with the relevant clump information contained i.e. peak positions, peak value, positions of the centre of the clump, dimensions of the clump, area/volume of the clump and the total measured flux of the clump. The different clumpfinding algorithms in CUPID all strive for the same end result but achieving it in different ways.

2.1.1 ClumpFind

The ClumpFind algorithm was originally developed by Jonathan Williams, and is fully described in Williams et al. (1994).

In brief, the algorithm locates the position of the peak emission values within the image, from there it contours the data starting from the peak down to a minimum contour level. The contour levels are user defined and usually multiples of the RMS (Root Mean Square) value of the background noise in the image.

The peaks are determined by looking at the array and determining the highest value pixel in the array, once that has been located the process descends down from that peak value. The amount descended by is the value of the contour spacing; at each contour level it continues the profile of any previously found clumps in addition to detecting any new peaks. These new peaks are the highest value pixels within the contour boundaries that are not already associated with a clump. If an area of pixels is assigned to multiple clumps then the pixels in that area are divided between the clumps, the association of each pixel is given to the closest clump, this association is to the next highest contour boundary level and not the position of the peak value in the clump. This splitting of the pixel areas uses the 'friends-of-friends' algorithm.

The process continues working through each contour level until it reaches the specified minimum contour level, new clumps found below this level are ignored as they are assumed to be noise but clumps that have been detected at higher contours are continued down to this minimum value, see Figure 2.1.

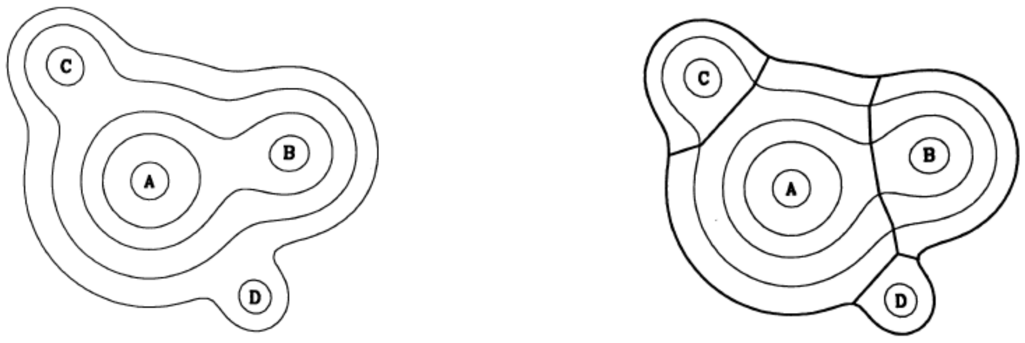


Figure 2.1: Graphical representation of the clump detection and separation process as performed by ClumpFind, firstly peak A is determined, peak B at the next contour level followed by peak C then peak D. The determination of which pixel belongs to which clump and where each clump meets is done by the friends of friends algorithm (Image obtained from Williams et al. (1994) with permission from the author).

2.1.2 FellWalker

The FellWalker algorithm was developed by David Berry, and as its name suggests works in a similar way to a fellwalker making a route through a hilly area. FellWalker is a relatively new algorithm, because of this it has not been documented in the literature for the use of clumpfinding.

A fellwalker would start his/her route by ascending the steepest part of a hill in order to reach the top, this provides the most certain way of reaching the peak using the most direct route. The program works with two or three-dimensional data but for the purposes of explanation it is easier to consider it as if the data were a two-dimensional array, the same as is visually displayed on a geographical contour map. See Figure 2.2 for a graphical representation of this.

The algorithm looks at each pixel individually, if the pixel is below a minimum defined value then it is ignored and the algorithm moves onto the next pixel. Once the algorithm finds a pixel above this minimum value the route begins, the surrounding pixels are observed and the algorithm moves onto the pixel which would provide the greatest ascending gradient, this process continues until a peak is located. A peak is defined as a pixel where every surrounding pixel is of a lower value, therefore a descending gradient. Sometimes this situation may be caused by a noise spike in the data and as such FellWalker checks the extended neighbourhood of pixels to see if there is a pixel of greater value nearby. The size of this neighbourhood is defined by one of the parameters, MaxJump, see §2.2.2. If no such pixel is found then the original pixel is a confirmed data peak, if a pixel of greater value is located within the defined area then the algorithm jumps to that new location and the process continues until a confirmed peak is found. With the peak location confirmed every pixel travelled during that route is considered part of one clump and is given an integer value to identify those pixels as being part of that specific clump, this is the clump number. The algorithm returns to the location where its walk started, moves onto the next pixel and continues the process until all pixels have been investigated. If during the process of making a new route it intersects with a previous route the present journey is terminated and all the pixels of that journey are identified as belonging to the previously determined clump, and are numbered to indicate this.

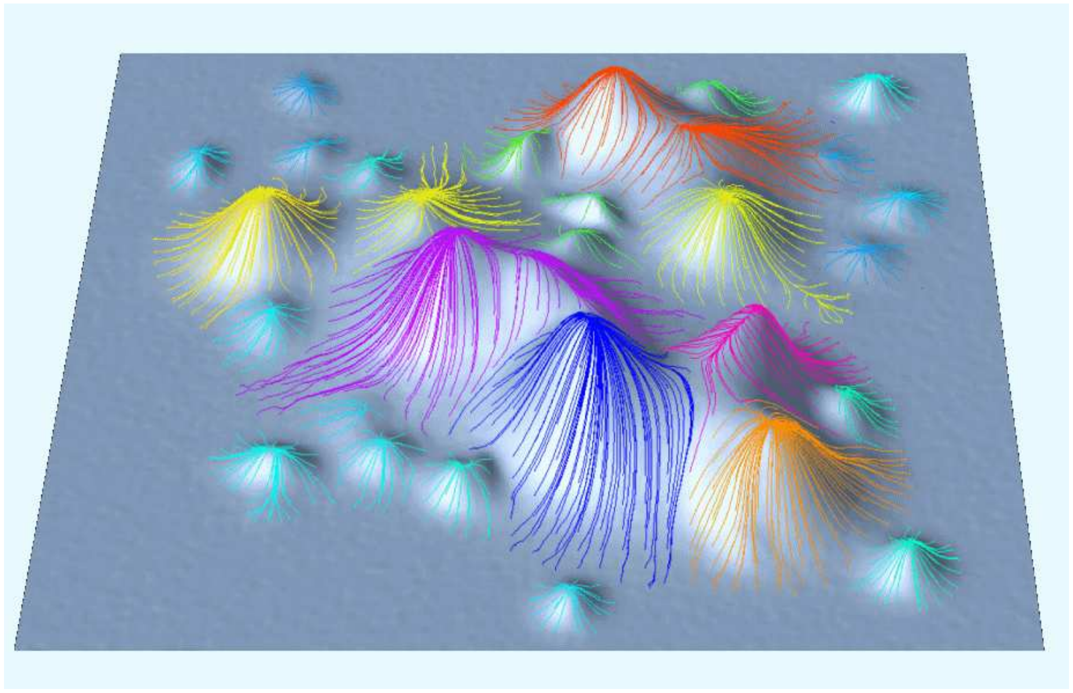


Figure 2.2: Graphical representation of how the fell walking route process determines each peak and clump (Image obtained from http://starlink.jach.hawaii.edu/starlink/fellwalker.html (last edited 24th September 2009) with permission from the author).

During the route, if the process finds a local peak, jumps to another pixel of higher value within the extended neighbourhood and continues the journey, the local peak is considered to be noise and as such is part of the initial clump. If that peak was not caused by noise but by blended emission, then the program needs to separate these two sources and not consider them to be all of one clump. Another parameter, MinDip, see §2.2.2, is used to distinguish between what would be considered as a noise spike and another source, this parameter generally assumes noise spikes to be of relative size to the noise in the rest of the image, see Figure 2.3.

In certain situations the initial part of the journey may have a very shallow gradient which becomes steeper, depending on this gradient the initial part of the journey may be ignored until the route reaches a point, over four pixels long, where the gradient is steep enough for

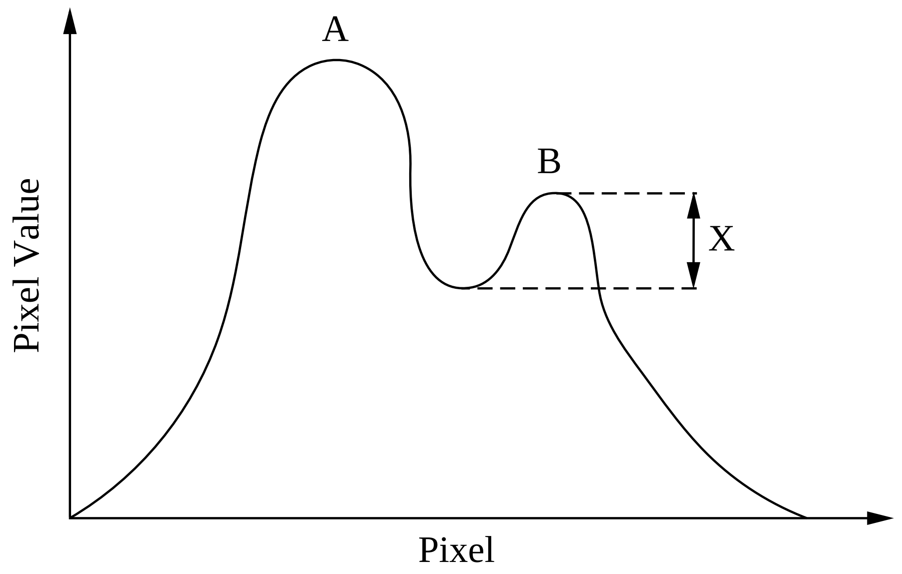


Figure 2.3: Graphical representation of the MinDip parameter where A and B are separate sources but the value for X must be greater than MinDip for source B to be considered independent of source A.

it to be confirmed as the edge of a clump, this process is ignored at higher pixel values where the entire length of the route is considered as part of the clump, this gradient value is a user defined parameter; FlatSlope see §2.2.2.

Once all the pixels have been investigated by FellWalker and the clumps have been allocated, a cleaning process is performed to smooth the clump edges, this looks at each pixel associated with a clump and assigns each pixel the same clump number as the most commonly occurring value of the surrounding pixels.

2.1.3 GaussClumps

The GaussClumps algorithm is fully described and demonstrated in Stutzki & Guesten (1990), who were the original developers of the GaussClumps algorithm. The version used in CUPID is a re-write of the original algorithm, it follows the same principles but with some minor adjustments of the termination criteria for the algorithm. (See below, and the CUPID users manual (Berry, 2009).)

The algorithm starts by fitting a Gaussian profile onto the brightest peak within the data array, the fit is then subtracted from the data. This process is performed many times, each time the new highest peak value in the data is used, until the integrated flux of all the determined clumps equals the integrated flux of the entire image. The process will run until one the following termination criteria are met:

- All the information in the data array has been assigned to clumps.
- The number of detected clumps reaches a user defined maximum, MaxClumps.

- The number of peaks under a user defined value, Thresh, reaches a user defined maximum, NPad.
- The algorithm fails to perform a fitting to a certain clump after attempting a user defined number of times, MaxSkip.

Once one of these termination criteria has occurred, all the clumps that have been detected are deemed to be real and output into the resulting catalogue. A clump is deemed to be false if its size or peak is below a value as determined by the parameters mentioned in §2.2.3, this is due to potential noise and the chosen value prevents the erroneous detection of noise spikes as real clumps.

In GaussClumps the clumps are permitted to overlap, therefore a single pixel can be associated with multiple clumps unlike the other CUPID algorithms, where each pixel can only be associated to one clump. This should give a more accurate size and flux value for the identified clumps since blended emission is most likely as a result of multiple clumps overlapping. If each pixel can only belong to one clump then the true sizes of the clumps would be inaccurate. This may result in an incorrect flux value for the clump, see Figure 2.4. Consequently, the clump image produced by GaussClumps does not contain indexed clumps as ClumpFind and FellWalker do, but instead shows the summation of the fitted Gaussians. Since the algorithm fits Gaussian arrays to the data there are a large number of parameters which weight or adjust the fitting of the Gaussian.

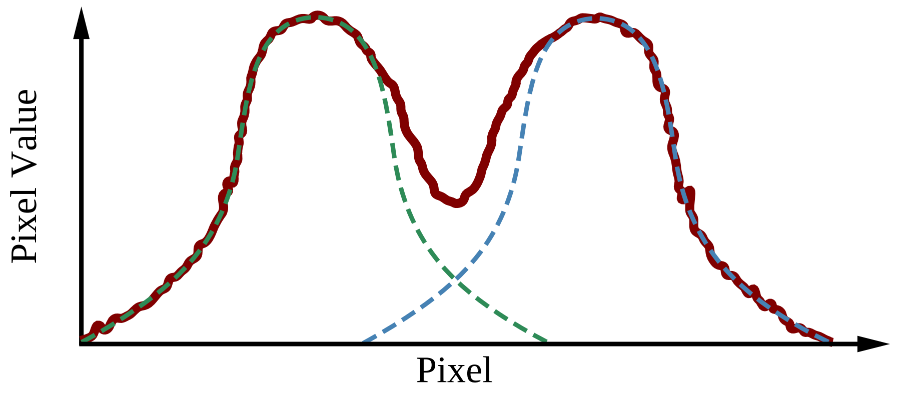


Figure 2.4: Graphical representation of how GaussClumps performs its Gaussian fitting and how a single pixel can belong to multiple clumps, the red line shows the clump profile for the data and the blue and green lines are two separately determined clumps with pixel values for each clump according to the Gaussians.

2.1.4 Reinhold

The Reinhold algorithm was developed by Kim Reinhold at the Joint Astronomy Centre in Hilo, Hawaii. Its original purpose was for determining the spatial features of cells in the human body by modelling our understanding of how the human eye works. Her method converts the original two or three-dimensional data arrays into many one-dimensional arrays, the length of each one-dimensional array is equal to the length of each axis in the original multi-dimensional data array. The algorithm identifies the highest value data point in that array and assigns that to be a peak, provided that the pixel value is above a user defined minimum peak value. If it is not above this minimum then that value is ignored and the algorithm decides that there was no real peak in that array, and the process moves onto the next array. If the peak value is above the minimum then the program moves out from this peak in both directions along the array until it reaches a pixel that fulfils the criteria for being an edge pixel, see below.

- The algorithm reaches a pixel that has already been associated with another clump.
- Two adjacent pixels are below the pre-specified minimum value, Thresh.
- The average gradient over three adjacent pixels drops below the specified minimum value, FlatSlope.
- The end of the data array is reached and none of the previous criteria have been met.

The data arrays are re-combined into the original two or three-dimensional arrays with the clump edges now determined, this then produces a number of ring or shell like structures which trace the outline of the clump. Since the algorithm is looking at the edges, the size of the clump is determined at lower pixel values and therefore the detection is more susceptible to noise, consequently the structures need to be cleaned up.

This cleaning process uses a number of cellular automata which dilate then erode the structures. This is done by determining exactly which pixels are the edges, each adjacent pixel is also considered as an edge. Within the edge pixels there are central regions, if the number of edge pixels around the central region is equal or below a certain value then the central region pixel would not be considered as an edge pixel, the edge pixels are then eroded away. this process can be run multiple times depending on certain parameters see §2.2.4. After the cleaning process all the pixels within the structure are then assumed to belong to the clump associated with that structure.

After the clump edges have been cleaned, each clump is assigned a unique integer value, if the cleaning process has left gaps in the edge of the clump this process leaks out to pixels outside of the clump. Due to this a single pixel may be claimed by two separate clumps and to resolve this issue the clumps go through another cleaning process, whereby if a pixel is associated with multiple clumps then the pixel is claimed by the clump with the greater number of adjacent pixels to the pixel under dispute.

2.2 The Algorithm Parameters

Listed in this section are some of the parameters of each method along with a description as to their effect on the data. These are not all the parameters but are either the ones that were adjusted for this investigation, ones that have been previously mentioned concerning how the algorithm works, or ones that are important as to the workings of the algorithm for the purposes of this investigation. Each parameter has a default value which is used when the parameter value is not altered, these are noted in square brackets after the description of each parameter.

2.2.1 ClumpFind

ClumpFind is one of the more widely used algorithms for clumpfinding, as such there is extensive documentation as to recommended values for some of the parameters, (Williams et al., 1994), these were thoroughly tested in this investigation. The main parameters of ClumpFind are:

AllowEdge determines whether a clump touching the edge of the image array should be included in the results, if set to zero the clump is rejected, non-zero the clump is retained [0].

DeltaT is the numerical distance between the contour levels, this can be an absolute value or in respect to the RMS noise value [2*RMS].

IDLAlg is a parameter introduced after numerous confusion since the on line IDL version of the algorithm distributed by Jonathan Williams on the 28th April 2006 differed to the one published in Williams et al. (1994), therefore CUPID has both versions of the algorithm and this parameter allows you to choose which to use. If zero the published version is used, if non-zero the on line version is used [0].

MaxBad is the number of pixels in a single clump that can be adjacent to a bad pixel, if this value is exceeded then the clump is rejected [4].

MinPix is the minimum number of pixels a clump must contain for it to be accepted and included in the results [3, 7, and 16 (for one, two, and three-dimensional data arrays respectively)].

Noise is the value where any pixel less than this is considered to be background noise and therefore ignored, this can be an absolute value, or one in respect to the RMS [2*RMS].

RMS is the RMS noise value of the image, since this changes for each image there is no set default value and the parameter must either be specified beforehand or input during the operation of the algorithm. CUPID is able to calculate the RMS value and provides it as a suggestion to the user during the operation of the algorithm [varies].

Tlow is the lowest value to which the algorithm performs contouring, this can be either an absolute value or one in respect to the RMS [2*RMS].

FwhmBeam is the FWHM (Full Width Half Maximum) of the instrument beam, in pixels. The output clumps are reduced (in quadrature) by this amount if the application parameter DECONV is set to true. Should a direct comparison be required between other implementations of the ClumpFind algorithm then DECONV should be set to false [2.0]. (For this investigation DECONV was set to false).

Naxis determines what is considered an adjacent pixel when contouring the data. If set to one then only points at the centre of a cube face are considered, if two then those pixels at the centre of the edges are also considered, and if three then the pixels at the corners of a cube are also considered. The Naxis value is ignored if the application parameter, perspectrum is set to true [1 for 1-D arrays, 2 for 2-D arrays, and 3 for 3-D arrays].

2.2.2 FellWalker

FellWalker uses an entirely new process with a number of useful parameters to determine the profile of the clump. The main parameters of FellWalker are:

AllowEdge determines whether a clump touching the edge of the image array should be included in the results, if set to zero the clump is rejected, non-zero the clump is retained [1].

CleanIter gives the number of times the cleaning process should be performed [1].

FlatSlope is the minimum gradient over the initial four pixels for the section to be included into the clump, this can be an absolute value, or one in respect to the RMS [1*RMS].

MaxBad is the number of pixels in a single clump that are allowed to be adjacent to a bad pixel, if this value is exceeded then the clump is rejected [4].

MinDip is the minimum value a dip can be between two peaks, this measurement is taken from the peak of the smaller peak and the dip before it as a result of the larger peak. If the dip is less than this value then both peaks are considered to be part of the same clump with the lesser peak most likely considered noise, this can be an absolute value, or one in respect to the RMS [3*RMS], see Figure 2.3.

MinHeight is the minimum value a peak can be, if less than this it is ignored, this can be an absolute value, or one in respect to the RMS [MinDip + Noise].

MinPix is the minimum number of pixels a clump must contain for it to be accepted and included in the results [3, 7, and 16 (for one, two, and three-dimensional data arrays respectively)].

MaxJump defines the size of the pixel neighbourhood where the algorithm looks for higher values once it reaches a peak, the neighbourhood size is a square/cube with sides equal to twice the size of the value given [4].

Noise is the value where any pixel less than this is considered to be background noise and therefore ignored, this can be an absolute value, or one in respect to the RMS [2*RMS].

RMS is the RMS noise value of the image, since this changes for each image there is no set default value and the parameter must either be specified beforehand or input during the operation of the algorithm. CUPID is able to calculate the RMS value and provides it as a suggestion to the user during the operation of the algorithm [varies].

2.2.3 GaussClumps

GaussClumps is one of the oldest clumpfinding algorithms, as with the other algorithms we started from the beginning and tested out all the important parameters, which include:

FwhmBeam is the FWHM in pixels of the instrument beam, the fitted Gaussians are not allowed to be smaller than this value [2].

FwhmStart is an initial guess at the observed clump size to beam width ratio. This gives the starting point for the algorithm to find the best fitting curve for each clump. If no value is given the algorithm makes an initial guess based on the local profile around the peak pixel [no value].

MaxBad is the number of pixels in a single clump that are allowed to be adjacent to a bad pixel, if this value is exceeded then the clump is rejected [4].

MaxClumps is one of the termination criteria for the algorithm, once it has determined this number of clumps the process ends [unlimited].

MaxNF is the maximum number of evaluations for the fitting of the Gaussian for each clump [100].

MaxSkip is one of the termination criteria for the algorithm, if the number of consecutive failures when fitting the Gaussian to a clump reaches this value then the process ends [10].

ModelLim is the relative value at which the Gaussian models are truncated to zero, the value is multiplied by the RMS noise value before it is used, model values below this are then treated as zero [0.5].

NPad is one of the termination criteria for the algorithm, if this number of consecutive clumps are evaluated all with a peak value below that of the Thresh parameter then the process stops [10].

RMS is the RMS noise value of the image, since this changes for each image there is no set default value and the parameter must either be specified beforehand or input during the operation of the algorithm. CUPID is able to calculate the RMS value and provides it as a suggestion to the user during the operation of the algorithm [varies].

Thresh is the minimum peak value a clump can have, the value is multiplied by the RMS noise value before it is used [2].

2.2.4 Reinhold

Reinhold is also a relatively new and unused algorithm, as described in Berry et al. (2007). This investigation was one of its first thorough trial and evaluations against other, better known algorithms. Reinhold's main parameters are:

CAIterations gives the number of times to perform the first cleaning process on the clumps [1].

CAThresh controls the cellular automata used to clean the clump edges. If the number of edge pixels in a 3x3 square or 3x3x3 cube is greater that this value then the central pixel is considered to be an edge pixel, otherwise it is not. [8, 26 (for two, and three-dimensional data arrays respectively)].

FixClumpiterations is the number of times to perform the second cleaning process, after the filling process has occurred [1].

FlatSlope is the minimum gradient the clump must have going from the peak pixel to the pixel location, once the gradient drops below this, the clump is considered to end, this can be an absolute value, or one in respect to the RMS [1*RMS].

MaxBad is the number of pixels in a single clump that can be adjacent to a bad pixel, if this value is exceeded then the clump is rejected [4].

MinLen is the minimum number of pixels a peak must span for it to be included [4].

MinPix is the minimum number of pixels a clump must contain for it to be accepted and included in the results [3, 7, and 16 (for one, two, and three-dimensional data arrays respectively)].

Noise is the value where any pixel less than this is considered to be background noise and therefore ignored, this can be an absolute value, or one in respect to the RMS [2*RMS].

RMS is the RMS noise value of the image, since this changes for each image there is no set default value and the parameter must either be specified beforehand or input during the operation of the algorithm. CUPID is able to calculate the RMS value and provides it as a suggestion to the user during the operation of the algorithm [varies].

Thresh is the minimum value a peak can be, if less than this it is ignored, this can be an absolute value, or one in respect to the RMS [Noise + 2*RMS].

Chapter 3

Algorithm Parameter Investigation

3.1 Overview

As the algorithms all work using different methods it is not possible to say which algorithm works best without fully understanding the differences between them and how each one operates. To this end I looked into the effects of altering the parameters from their default values. Once I had a full understanding of each of the algorithms in turn it was then possible to compare one algorithm against another (see Chapter 4). In this chapter I discuss the effect of changing each of the algorithms parameters in turn thus determining which ones have an effect on the results, and what values may be suited for these parameters if it is not the default value.

3.2 Varying Parameters

There is little documentation as to the robustness of each of the parameters, and as such the algorithms default values have generally been used when the clumpfinding process has been performed in the literature. To this end it was unknown as to the precise effect that changing these parameters would have. In the investigation, only one parameter at a time was changed. Any differences observed would be solely due to that one parameter, since the remaining parameters were left at their default values or at a constant value for the entirety of the investigation.

To perform the investigation findclumps was performed on all 62 SCAMPS images (31 of each wavelength) with each of the parameter variations. This was undertaken using a Perl script to run the process on each of the images in turn, starting with the 450μm data then the 850μm. Once the clumpfinding process had been performed I had 62 outcat data tables containing the details for each clump that was detected. So that all the data could be analysed more easily the files were concatenated together, this was done using fitscopy and tabmerge from the CFITSIO package, (Pence, 1999)[1]. This process left me with an output of each of

[1]A NASA based software library for FITS file subroutines, this can be found at http://heasarc.gsfc.nasa.gov/fitsio/ (accessed June 2008).

the clump index images, an outcat data table for each of the files and a table with all the outcat information for each wavelength. The different parameter values were saved into a text file which was accessed by the script in turn, once all 62 images had been run using one set of parameters it then automatically ran with the next set. Each time the process ran, the RMS values of each image (see appendix A) were accessed from a text file.

The outcat data table contained all the relevant information for each clump (peak position and peak value of the clump, central position of the clump, size of the clump, and integrated flux of the clump) in each image so the effects of changes in the parameters could be easily seen.

For each of the Figures in this section (Figures 3.1 - 3.20) the concatenated data for all 31 images at each wavelength was used to compare the number of clumps detected for each of the different parameter combinations. The size and total integrated flux values of the clumps are compared using histograms, and an example of one of the 62 output images is given to show the clumps that the algorithm detected in that particular image; this image was g1015-034_scamps_sho_flat.sdf.

3.2.1 Parameter Value Constraints for all Algorithms

Certain parameters were held constant for all the algorithms but changed for each of the two wavelengths, one of these was the MinPix parameter. This parameter is dependant on the resolution of the data that is used. In the default case, the MinPix value would be seven but due to the beam size in relation to the pixels this value needed to be changed. The beam width is three and five pixels for the 450μm and 850μm data respectively, the smallest a clump should be is the square of this value, therefore the chosen values of MinPix were nine for the 450μm data and twenty-five for the 850μm data.

The RMS parameter also had to be set as it varies with each image. CUPID can automatically determine a value for this using min-max background estimation, (Berry et al., 2007) though it is believed this gives a value lower than the true value for the image. Consequently we decided to determine our own value for the RMS parameter, this was achieved by selecting a minimum of five areas on each image and determining their average standard deviation from the mean using GAIA. The calculated value was supplied to each algorithm for each individual image. For a list of the calculated and CUPID RMS values for each image see appendix A.

For a number of the parameters investigated the default value was at the extreme range of the investigation, i.e. either the highest value or the lowest value investigated. With certain parameters this had to be the case, e.g. CAThresh with Reinhold has a maximum value of eight for two-dimensional images, this is also its default. For other parameters (e.g. those that dictated the minimum peak value of a clump) the lowest value investigated was two times the RMS value of the noise in the image, this was the default value for the algorithms. Lower values for these parameters were not investigated because if a value of one times the RMS value of the noise in the image was used then a large number of spurious clumps would be detected. These clumps would in fact be nothing more than spikes in the background noise and the detection of them would drastically alter the results.

3.2.2 ClumpFind

For the ClumpFind algorithm the main parameters altered for investigation were; DeltaT, TLow, and IDLAlg. Previous investigations have favoured a value of 2*RMS, (Williams et al., 1994), (Friesen et al., 2009) for the first two parameters so we started the investigation using values of 2*RMS, 2.5*RMS, and 3*RMS for each of these parameters.

It became clear from the results that as the value for DeltaT increases, the number of clumps detected decreases by almost a factor of two, this is more evident in the 450μm data than the 850μm, see Figure 3.1. From the clump index images it appears that the total area in which the clumps in the images are found does not alter that much, yet the number of clumps contained in that area decreases. From the index images and the histograms of the integrated flux and size of the clumps for each of the parameter values it was determined that the reduction in the number of clumps detected is specifically due to smaller low flux clumps that are no longer detected. There are a greater number of larger high flux clumps detected as DeltaT increases, these large clumps are not likely to be newly detected ones but just smaller clumps at lower values of DeltaT, which have been merged into a larger clump at higher DeltaT values. This information leads to the conclusion that lower values of DeltaT distinguish the finer structure of clumps, causing the detection of small clumps that are close to a much larger clump. Increasing DeltaT causes smaller clumps to be merged together creating fewer larger clumps. Lower DeltaT values would suffer greater effect from noise spikes and tend to determine a single clump to be multiple clumps i.e. one main large clump with a number of smaller satellite clumps around the edges of the main clump.

TLow shows a similar behaviour, i.e. as the value of this parameter increases, the number of clumps detected decreases. This is also more evident in the 450μm data than the 850μm and from the clump index images, see Figure 3.2, it appears that the total area of all the clumps decreases as the value of TLow increases. The histograms in Figure 3.2 show that as the value of TLow increases the number of small clumps detected decreases by a factor of over two, this decrease continues into larger clump sizes. The decrease in small clump detections is probably due the clumps becoming smaller as TLow increases, many of which may become too small such that they are discarded due to MinPix. As for the reduction in the number of large clump detections it can be determined that as TLow increases the size of the clumps decreases and as such there is a shift, such that large clumps detected at lower TLow values are slightly smaller than when detected at higher TLow values. This shift is also true when looking at the integrated flux of the detected clumps: As the value of TLow increases the number of clumps detected with low flux values also decreases. Most of this decrease should be simply due to the fewer number of total clumps detected.

When the IDLAlg parameter was altered the number of clumps detected was greater for the IDL version over the originally published version, see Figure 3.3. This is again more evident in the 450μm data than the 850μm. Although the number of clumps that are detected

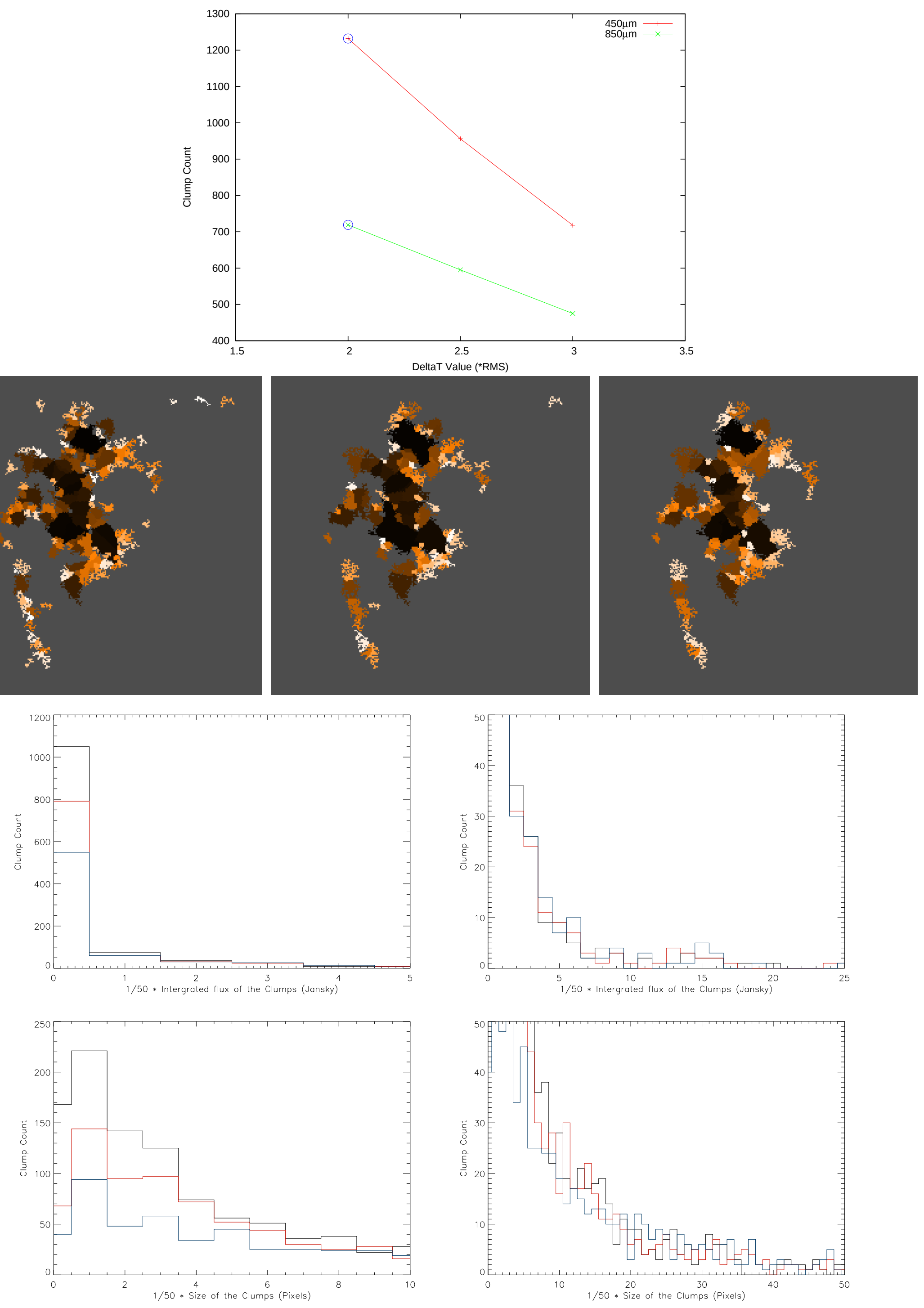


Figure 3.1: Top; Plot of how the number of clumps detected in all the images changes with different values for DeltaT using the ClumpFind algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of DeltaT changes, from left the values of DeltaT are; 2, 2.5, and 3 times the RMS value. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

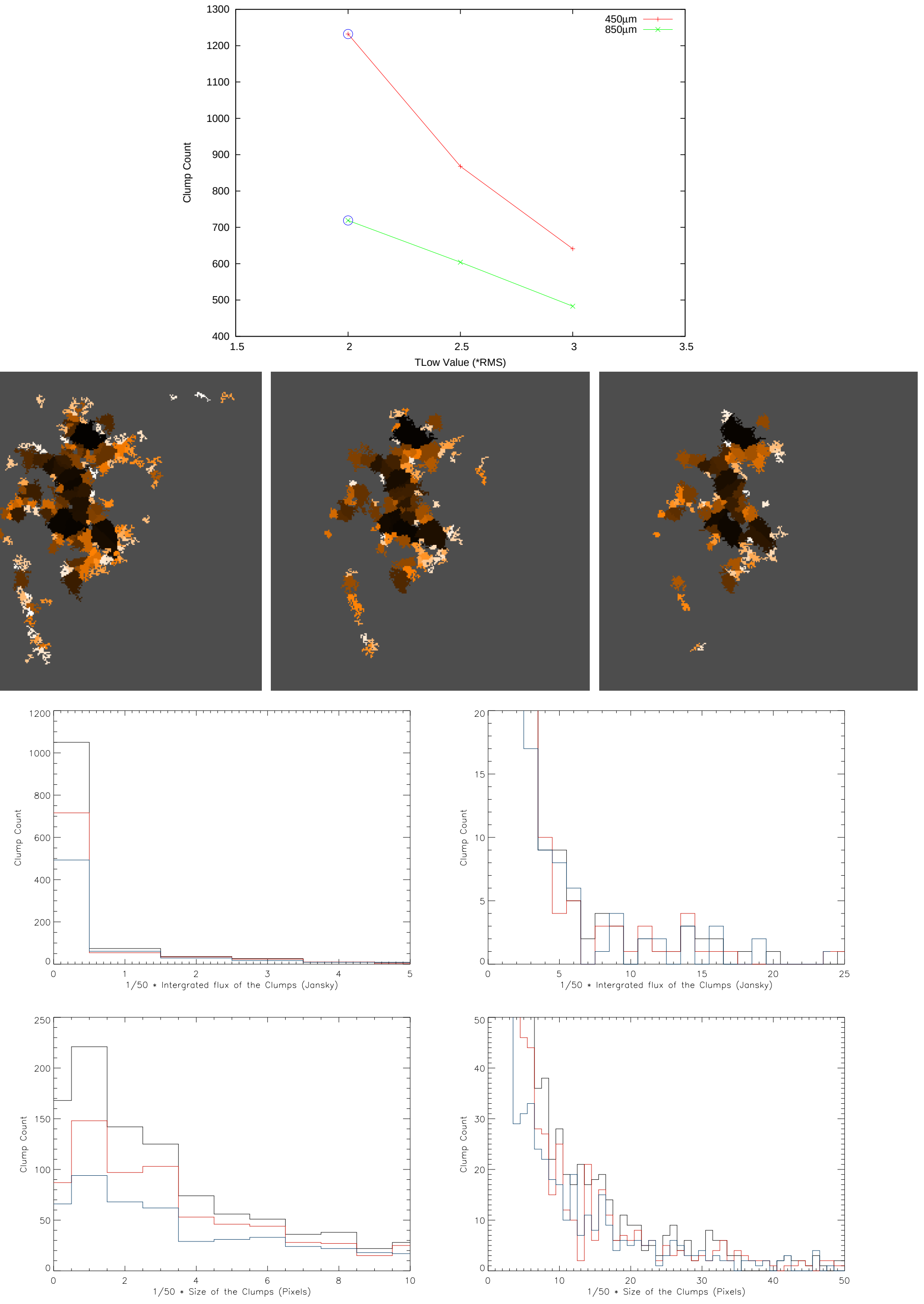


Figure 3.2: Top; Plot of how the number of clumps detected in all the images changes with different values for TLow using the ClumpFind algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of TLow changes, from left the values of TLow are; 2, 2.5, and 3 times the RMS value. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

decreases, the total area covered in the clump index images remains mostly unchanged. The histograms clearly show that the difference in the number of clump detections is primarily for smaller clumps. The IDL version detects many more smaller clumps. As the size of the detected clumps increases the difference between the two versions of ClumpFind reduces to a point where the published version of ClumpFind starts to detect more clumps than the IDL version of the largest clumps. These larger clumps are clumps that are determined to be smaller with the IDL version, possibly due to this version breaking larger clumps. This observation leads to the conclusion that the IDL version should better distinguish the finer structure of the clump area and locate small clumps that are close to a much larger clump. However this greater sensitivity to small clumps would suffer greater effect from noise spikes and may show a single clump to be multiple clumps, i.e. one main large clump with a number of smaller satellite clumps around the edges of the main clump. Therefore, using the IDL version of the algorithm leads to the detection of more clumps within the same area and hence their size and flux to be smaller than the published version.

ClumpFind showed a large difference in the number of detected clumps as the parameter values change, in most cases the variance is with the number of small clumps that are detected. These additional clumps may have been small sources around a much larger source and therefore only a portion of the smaller source was visible or it could be that the algorithm is sensitive to background noise especially at lower clump values. This could cause the algorithm to determine noise spikes to be real peaks and therefore breaking up a single large clump into multiple smaller clumps, most likely with one main clump and a number of satellite clumps replacing the edges of what would be the original clump. This result has also been concluded in other investigations into the ClumpFind algorithm, (Enoch et al., 2006). ClumpFind causes more clumps to be found resulting in the calculated sizes and fluxes of the clumps to be smaller. This is due to the satellite clumps having taken up some of the area of the original clump, consequently the size and flux of the original clump was calculated as a lower value than it should have been. The breaking up of the main clumps is reduced if the spacing between the contours (DeltaT) is increased. This has the unfortunate side effect of causing the algorithm to be unable to pick up the fine structure of the area and potentially missing ”true” small clumps that are located around the edges of a much larger clump.

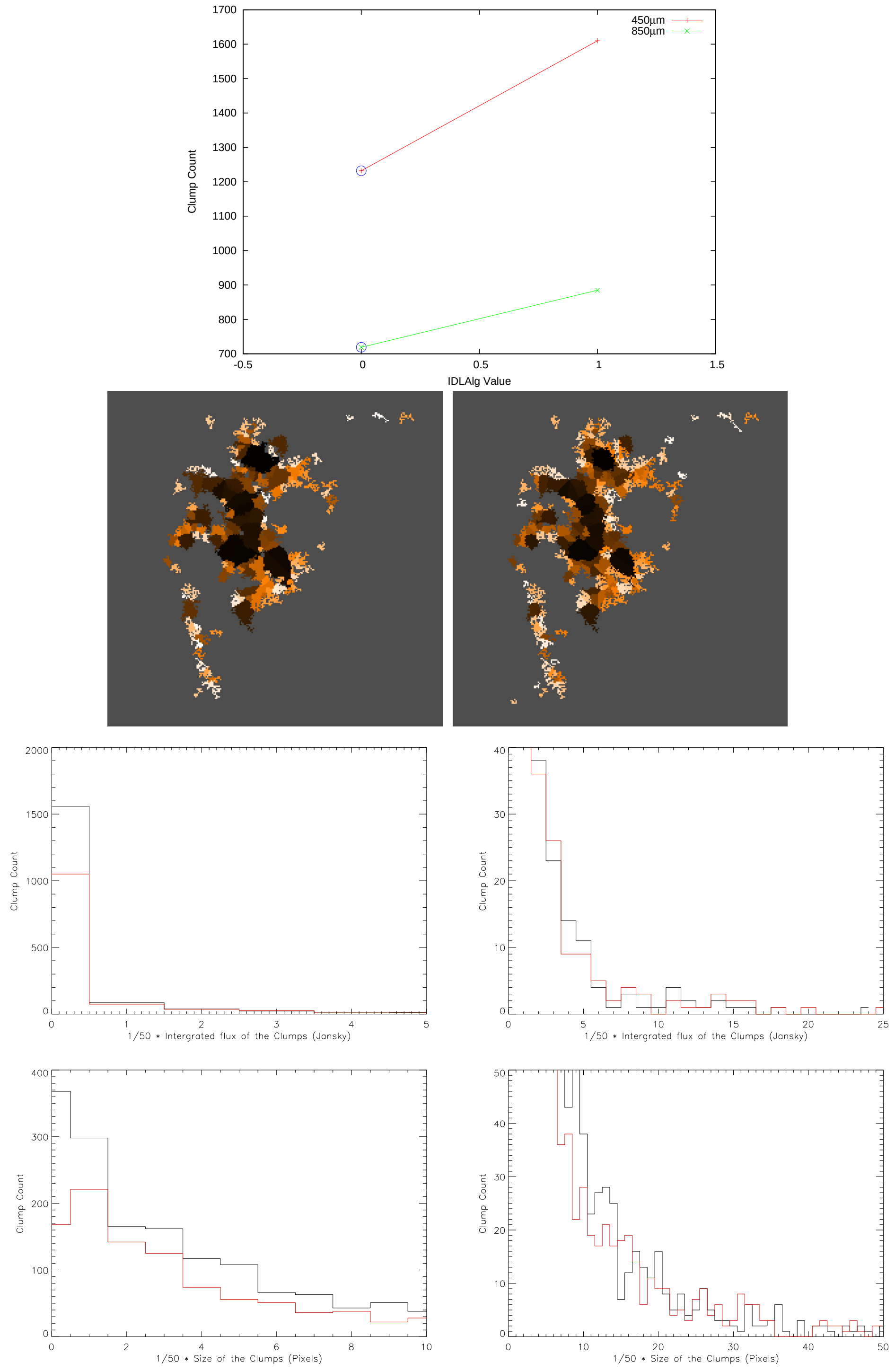


Figure 3.3: Top; Plot of how the number of clumps detected in all the images changes with different values for IDLAlg using the ClumpFind algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of IDLAlg changes, from left the values of IDLAlg are; 0 (meaning the published version of ClumpFind is used), and 1 (meaning the IDL version of the ClumpFind program is used). Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 1 (black line) and 0 (red line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 1 (solid line) and 0 (dashed line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

3.2.3 FellWalker

In the investigation into the FellWalker algorithm the parameters changed were; AllowEdge, CleanIter, Noise, FlatSlope, MinDip, and MaxJump. AllowEdge was changed so that clumps touching the edge were not included in the results, this is not the default for FellWalker and was set to this value for all the different parameter configurations investigated. As with ClumpFind the Noise parameter was changed to investigate the differences between a value of 2*RMS and 3*RMS. FlatSlope was investigated between 1*RMS and 3*RMS as was MinDip, MaxJump was investigated between values of 1 and 10, and CleanIter was investigated between values of 1 and 10.

AllowEdge was changed since the data used had objects centered in the image and it is unlikely that any real clumps would be found touching the edges of the image. Towards these edges the images become noisier and the algorithm could incorrectly detect a particularly noisy area in the image to be a clump.

Increasing the value of the Noise parameter has the effect of decreasing the number of clumps found, though this reduction in the number of clumps was relatively small (between a 10 and 15 percent decrease over the values investigated). From the clump index images it can be seen that as the value for the Noise increases the size of the remaining clumps appears to decrease also. The histograms show that as the value of Noise increases the number of small clumps detected decreases, see Figure 3.4. For larger clumps the opposite briefly becomes true and beyond a certain limit there is great variation as to which parameter value causes greater detections at a particular clump size. The most probable result of increasing the value of the Noise parameter is that the size of the clumps reduces, therefore the smallest clumps become too small for detection and the large clumps become slightly smaller. This causes a shift in the clump size histograms so that where a peak in clump sizes is present at one parameter value, that peak appears further to the left (at lower clump sizes) for a higher parameter value. From the histogram of the number of clumps detected at different integrated flux values we can see that there are fewer clump detections at low flux values when the value of Noise increases, but for higher flux values the number of detected clumps increases with increasing values of Noise.

The FlatSlope parameter has little effect on the number of detected clumps. As the value of the parameter increases there is a small increase in the number of clumps detected, see Figure 3.5. From the clump index images it appears that altering FlatSlope has only a minimal effect of the size of the clumps. The histograms show that although the number of clumps does not change greatly as the FlatSlope value is changed, the number of clumps at different sizes does change. A pattern can not be determined for the smaller clumps, but it is clear that the large clumps are reduced in size as the parameter value increases. This causes a shift to the left on the bottom histograms in Figure 3.5 showing what was detected as a large clump with a low FlatSlope value is detected as a marginally smaller clump when a higher value for FlatSlope is used. The flux histograms show that varying the value of FlatSlope has little effect on the number of clumps within each range of flux values, there are slight variations but these could

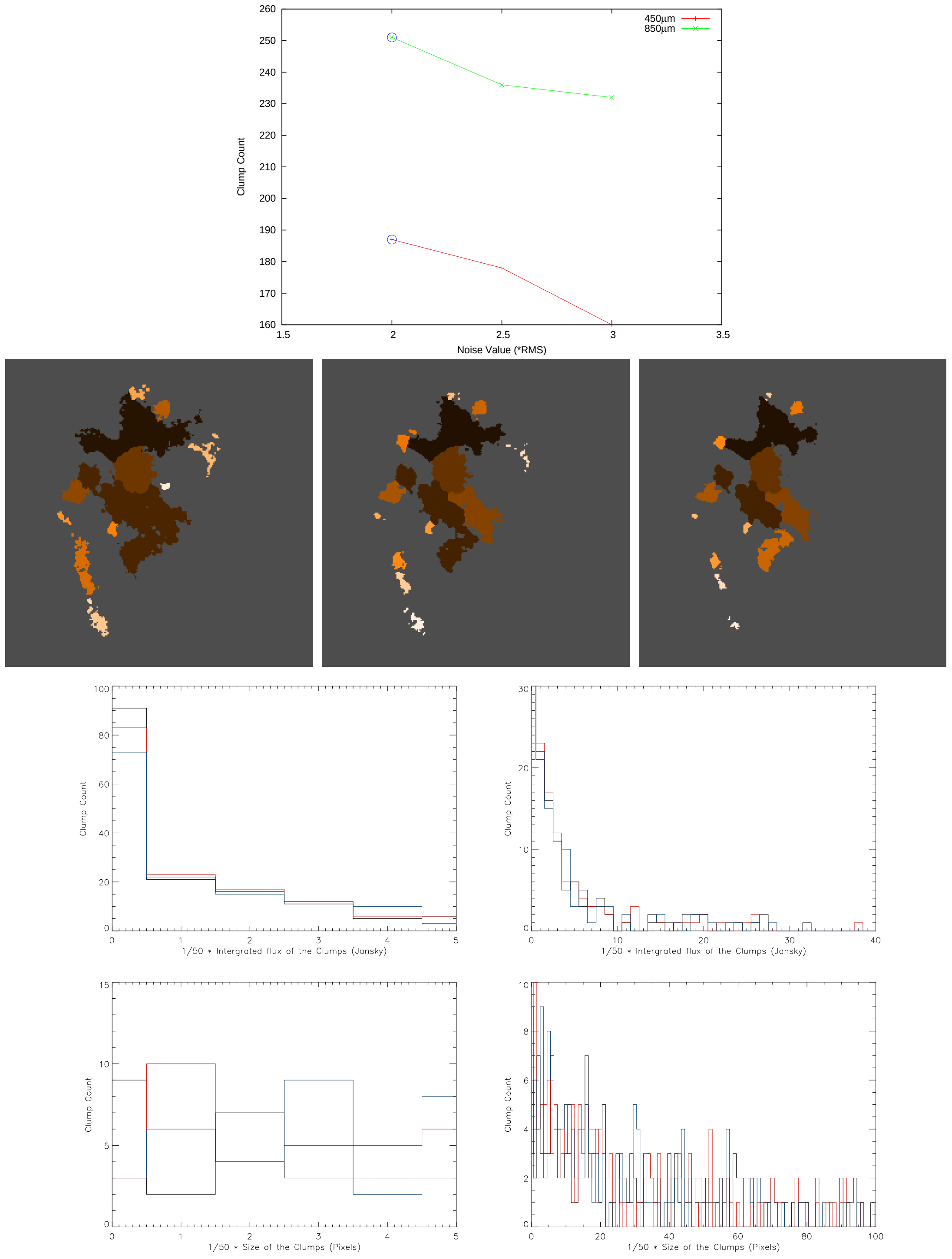


Figure 3.4: Top; Plot of how the number of clumps detected in all the images changes with different values for Noise using the FellWalker algorithm, the blue circles showing the default values. second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of Noise changes, from left the values of Noise are; 2, 2.5, and 3 times the RMS value. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

easily be due to the additional number of clumps found as the parameter value increases.

The CleanIter parameter controls the cleaning of the clumps after the detection process. The number of clumps detected decreases rapidly at first as the value increases from zero, but this rate decreases as the value increases until a point is reached where increasing the parameter value has minimal or no further effect on the number of clumps detected. From the clump index images it can be seen that no cleaning gives clumps with more ragged edges compared to when more cleaning is performed. As the cleaning process is performed the clump edges become eroded away and get smoother, causing the clumps to become smaller. As with the number of detected clumps this effect, although noticeable between low parameter values, shows no noticeable effect at high parameter values. From the histograms of the clump size in Figure 3.6 it can be seen that without the cleaning process the total number of clumps detected is higher, particularly small clumps, than when the cleaning is performed. For larger clumps it is difficult to obtain a definite conclusion between the effects of performing the cleaning process multiple times. In the flux histograms no cleaning again causes a higher number of low flux clumps to be detected than when it is performed and each time it is performed the number of low flux clumps that are detected decreases. This rate decreases as the CleanIter value increases. At higher flux values the effect of changing the CleanIter values is minimal.

MinDip determines if spikes in already present clumps are just noise or are in fact another clump with its emission mostly obscured by a larger source. Increasing the value of MinDip decreases the number of clumps detected. From the clump index images in Figure 3.7 it can be seen that the total area occupied by the clumps does not alter as the parameter value changes. This is because the clumps that are no longer detected are assumed to be noise spikes present in a larger clump, causing the small clumps to then be merged into a larger clump. As the number of detected clumps decreases the size of the remaining clumps thus increases by taking the merged clumps into account. The clump size histograms show that as the value of MinDip increases, the number of small clumps detected decreases, at larger clump sizes increasing MinDip causes a shift to the right, causing already large clumps to become larger. This increase in size of the clumps is linked to the reduction in the number of small clumps, as the MinDip value increases small clumps are assumed to be noise and their clump profile is absorbed, creating a larger clump. Fewer small clumps are detected as a result and the large clumps are increased in size. As the value of MinDip increases the number of low flux value clumps that are detected decreases, this effect lessens until a point where the number of clumps detected remains similar for different flux values irrespective of the MinDip value used.

MaxJump is the distance in pixels that the algorithm will examine for a higher value pixel if a peak has already been found. As the value of MaxJump increases the number of clumps detected increases rapidly, up to a point, after that the clump count starts to decrease but at a lower rate than the initial increase. From the index images in Figure 3.8 it can be seen that the total area of clump detection does not alter much but the number of clumps detected within that area changes with the same pattern as noticed in the clump count plot. The decrease

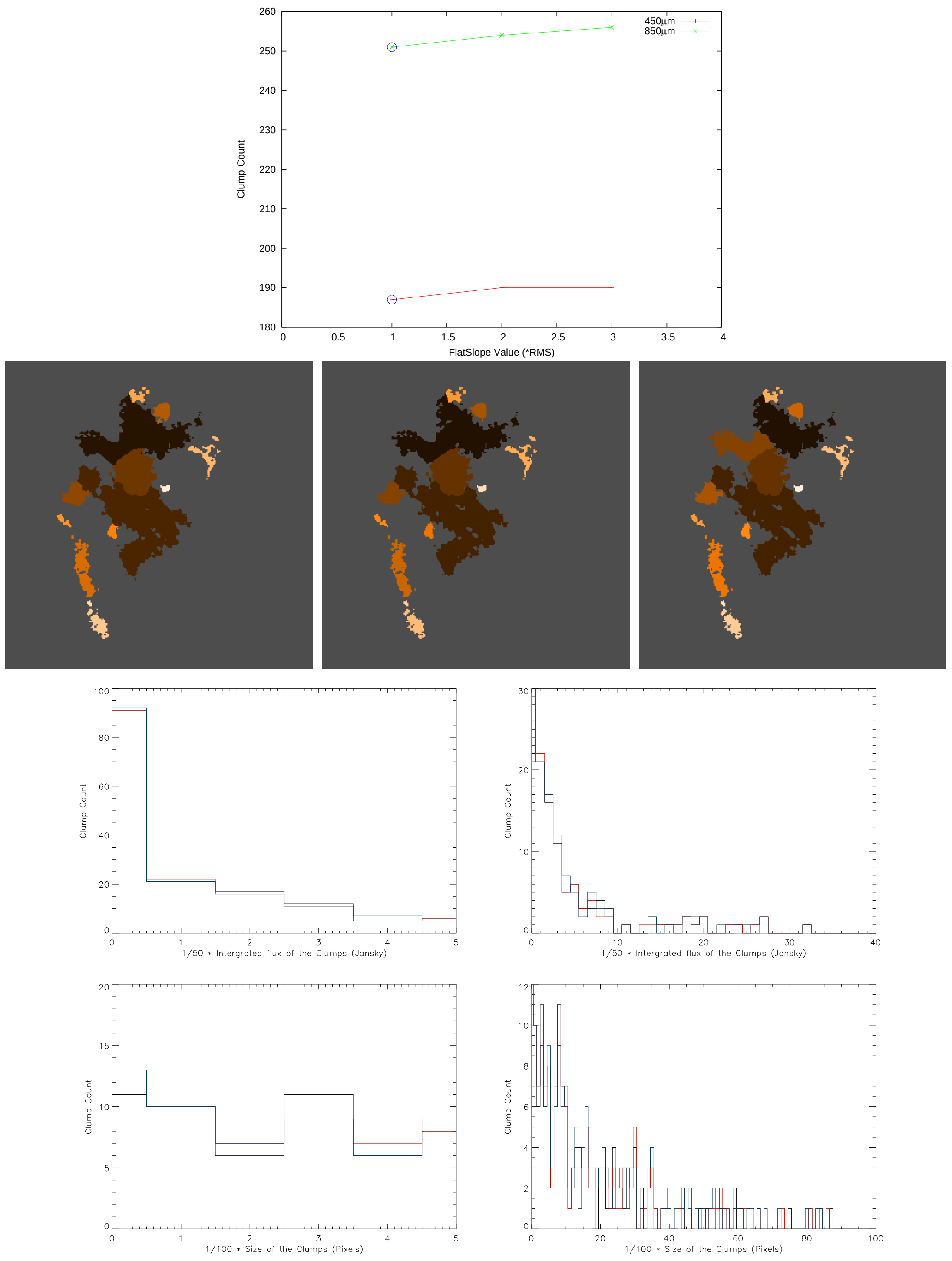
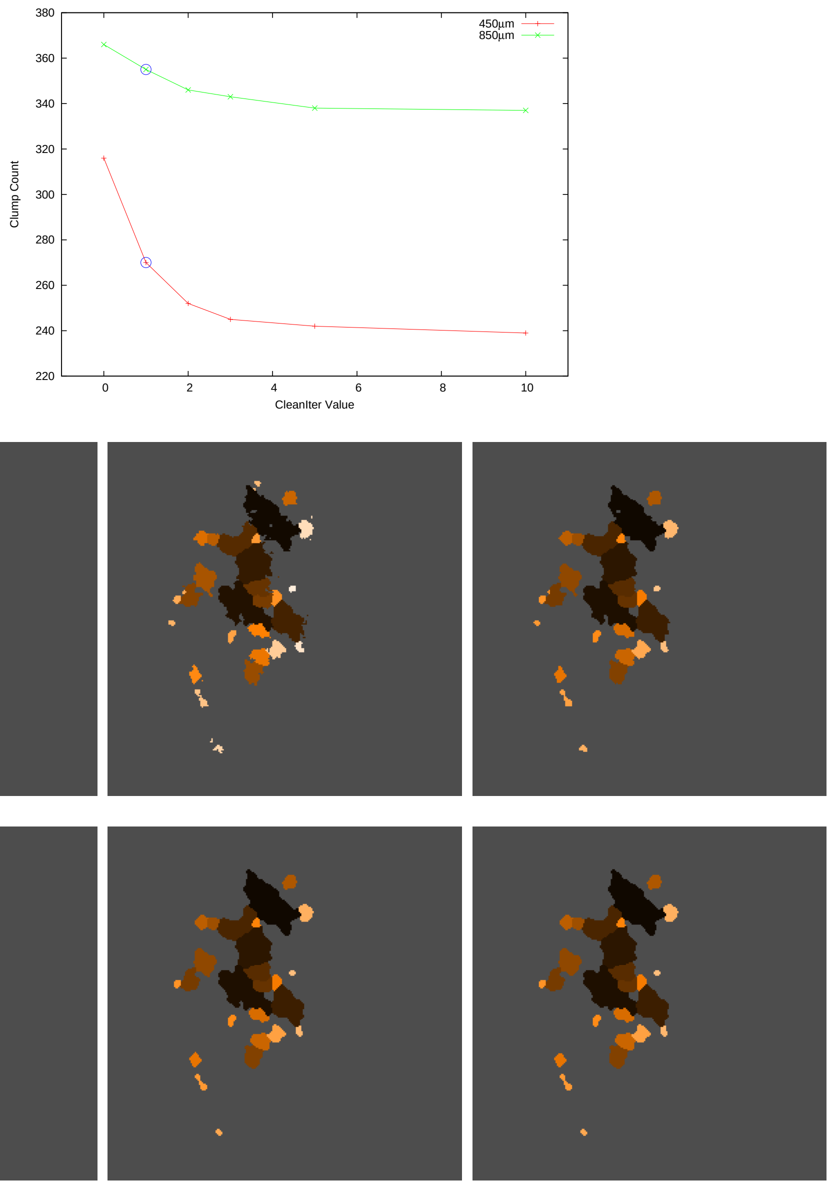


Figure 3.5: Top; Plot of how the number of clumps detected in all the images changes with different values for FlatSlope using the FellWalker algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of FlatSlope changes, from left the values of FlatSlope are; 1, 2, and 3 times the RMS value. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 2*RMS (black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.



450µm
850µm
Clump Count
CleanIter Value
380
360
340
320
300
280
260
240
220
0
2
4
6
8
10

Figure 3.6:

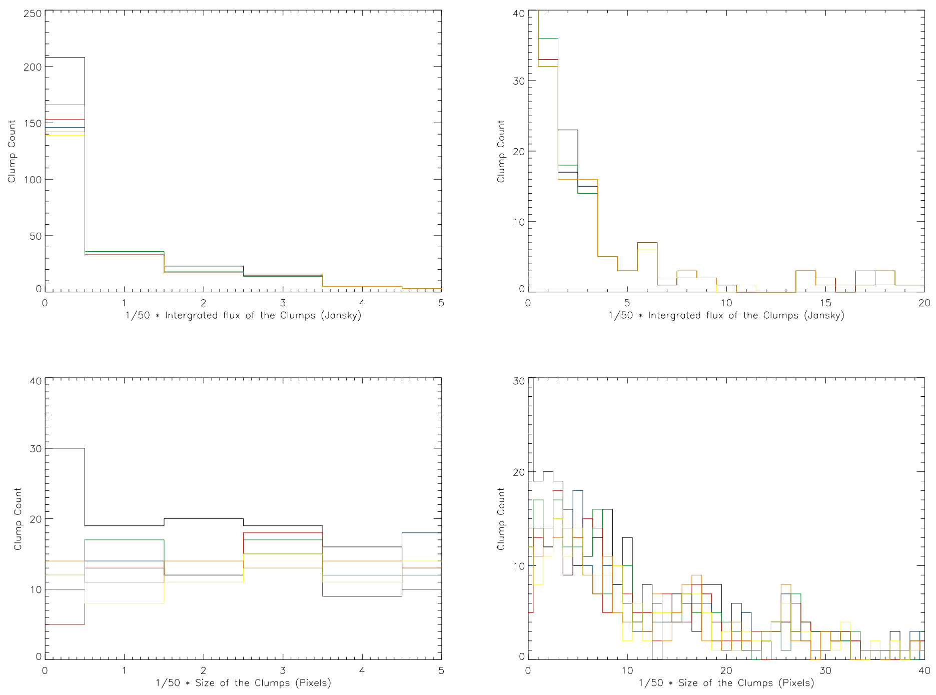


Figure 3.6: Previous Page; Top; Plot of how the number of clumps detected in all the images changes with different values for CleanIter using the FellWalker algorithm, the blue circles showing the default values. Second and bottom line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of CleanIter changes, from middle left the values of CleanIter are; 0, 1, and 2, from bottom left the values are; 3, 5, and 10. This page; Top; Histograms showing the integrated flux values of the 450μm data for the different parameter values 0(black line), 1(green line), 2(red line), 3(blue line), 5(orange line), and 10(yellow line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 0(black line), 1(green line), 2(red line), 3(blue line), 5(orange line), and 10(yellow line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

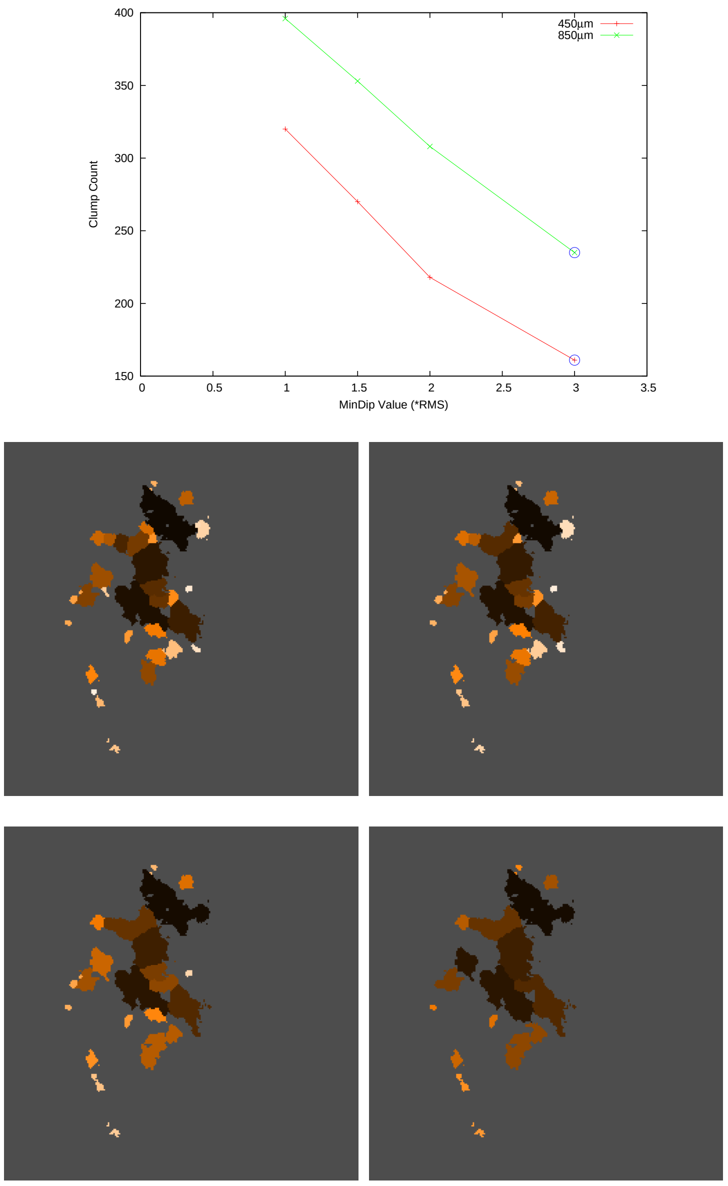

450µm
850µm
Clump Count
MinDip Value (*RMS)
400
350
300
250
200
150
0
0.5
1
1.5
2
2.5
3
3.5

Figure 3.7:

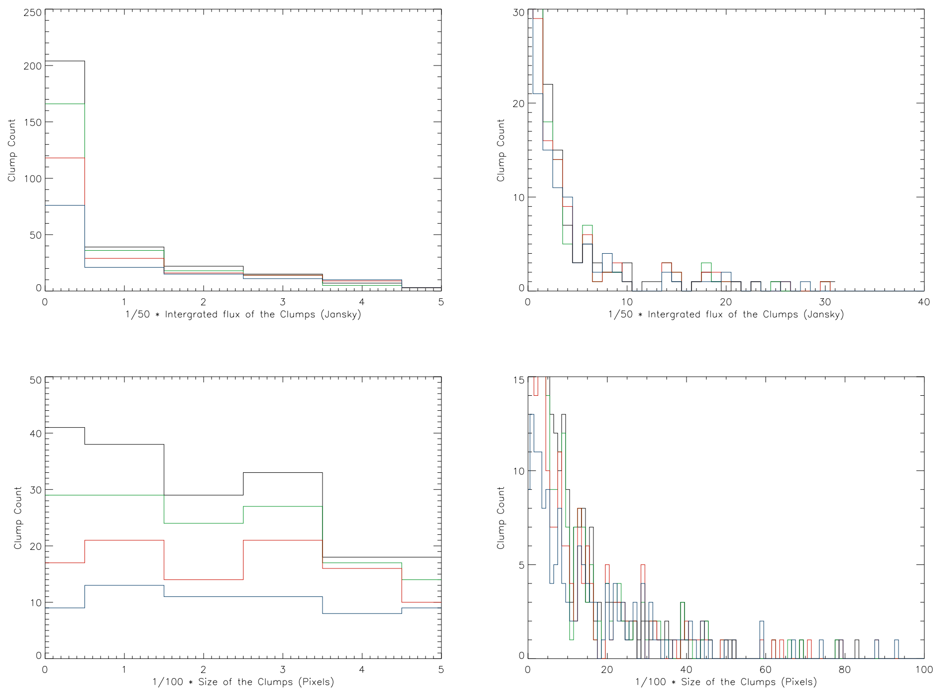


Figure 3.7: Previous page; Top; Plot of how the number of clumps detected in all the images changes with different values for MinDip using the FellWalker algorithm, the blue circles showing the default values. Second and bottom lines; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of MinDip changes, from second left the values of MinDip are; 1, and 1.5 times the RMS value, from bottom left the values are; 2, and 3 times the RMS value. This page; Top; Histograms showing the integrated flux values of the 450μm data for the different parameter values 1*RMS (black line), 1.5*RMS(green line), 2*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 1*RMS (black line), 1.5*RMS(green line), 2*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

would be due to the program being able to look further for a higher value pixel, therefore more likely to find one and as such, small individual peaks are assumed to be nothing but noise spikes and are therefore merged with a high peak value clump creating a single larger clump, due to this reasoning the turnover would not be expected at lower values and the number of clumps detected should continue to increase as the parameter value decreases. A possible explanation for this turnover could be due to the cleaning process CleanIter; where the index number of a pixel is changed to the same value as the majority of its surrounding pixels, as such large numbers of small clumps (a few pixels in size) would be merged into larger clumps. Setting CleanIter equal to zero may prevent this, but as only the MaxJump value was changed CleanIter was able to cause the merging of the clumps. A further explanation could be due to the MinPix parameter where clumps under a certain size are discarded and as the MaxJump value decreases the size of the clumps would probably decrease as well thusly some may be discarded. As the Max Jump value increases from one to three the number of clumps detected increases, from the histograms in Figure 3.8 the newly detected clumps can be seen to be smaller clumps, this is due to larger clumps being broken up into multiple smaller clumps, because of this the number of larger clumps decreases. With MaxJump values of four to eight the number of detected clumps starts to decrease, many of these clumps are the smaller clumps again being merged into fewer larger clumps as such there is an increase in the number of larger clumps detected as the MaxJump value increases. The flux shows a similar trend with a dramatic increase in low flux clumps from MaxJump values one to three but then a more gradual decrease in the number of low flux clumps as the MaxJump value increases further. These low flux clumps coming from to merging into larger clumps therefore also having a notable effect on the higher flux values.

Further to the MaxJump investigation the process was run again for the lower values but with the CleanIter value set equal to zero to prevent the conglomeration of smaller clumps into a larger clump though MinPix was left the same. The results of this further investigation did not show a turnover in the number of clumps detected when the MaxJump value is less than three. Instead a flattening off of the results was seen, demonstrating that the maximum number of possible clumps was detected just below the default value. It is possible some of these may be spurious detections and in fact be noise spikes though the MinPix parameter may prevent too many of these.

The plots in Figures 3.4 - 3.8 showing the number of clumps detected for each parameter value display a reversal in which of the two wavelengths detect more clumps when compared to Figures 3.1 - 3.3. With ClumpFind the 450μm results show a consistently greater number of clump detections than the 850μm results, with FellWalker it is the 850μm results with the greater number of clumps detected. Possible reasons for this occurrence are described in §3.3.

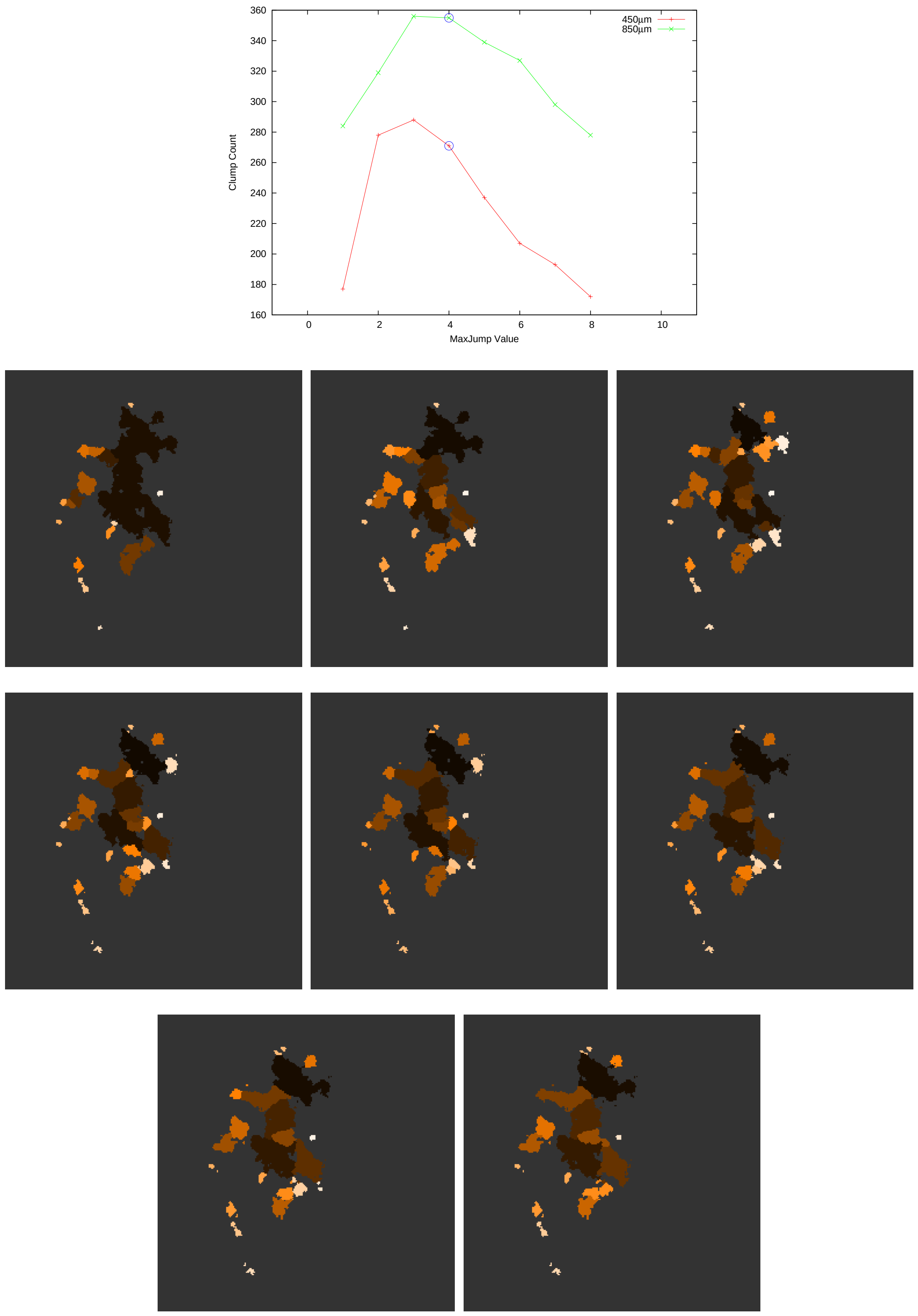


Figure 3.8: Top; Plot of how the number of clumps detected in all the images changes with different values for MaxJump using the GaussClumps algorithm, the blue circles showing the default values. second to bottom line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of MaxJump changes, from second line, left the values of MaxJump are; 1, 2, and 3, from third line, left the values are; 4, 5 and 6, from forth line, left the values are; 7, and 8.

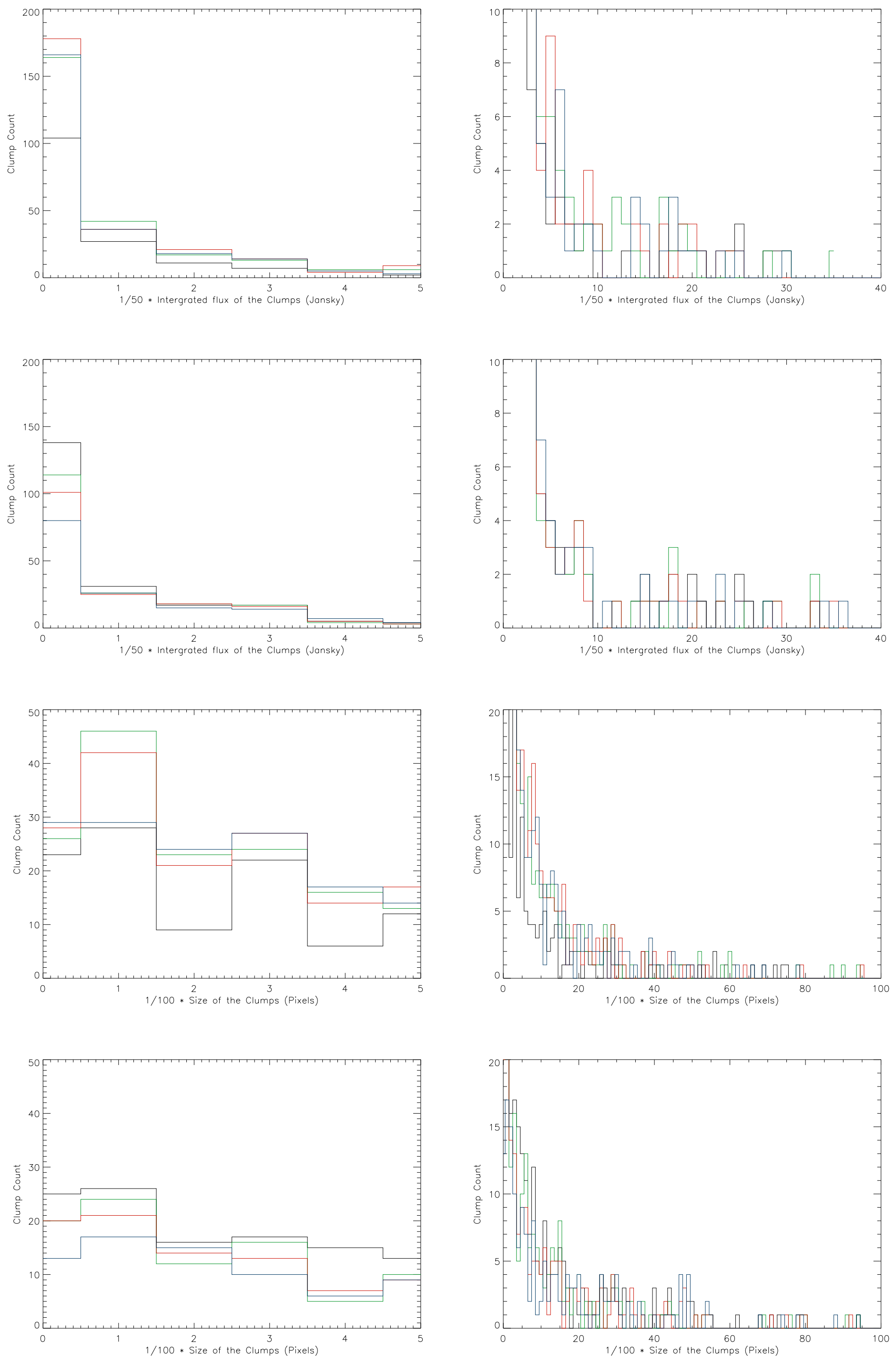


Figure 3.8: Top and second lines; Histograms showing the integrated flux values of the 450μm data for the different parameter values, top line; 1(black line), 2(green line), 3(red line), 4(blue line), second line; 5(black line), 6(green line), 7(red line), 8(blue line), the left histograms being a magnified view of the low integrated flux clumps, the right ones showing the full range of fluxes but limited clump count. Third and bottom line; Histograms showing the size of the clumps of the 450μm data for the different parameter values, third line; 1(black line), 2(green line), 3(red line), 4(blue line), bottom line; 5(black line), 6(green line), 7(red line), 8(blue line) the left histograms being a magnified view of the small clumps, the right ones showing the full range of sizes but limited clump count.

3.2.4 GaussClumps

The parameters investigated in GaussClumps were; FwhmBeam, MaxNF, ModelLim, Thresh, FwhmStart, and NPad. The FwhmBeam parameter depends upon the beam width in pixels, just like the MinPix parameter. This parameter was altered for all the different parameter configurations to take into account the different beam widths at 450μm and 850μm. The FwhmBeam value is taken to be three pixels for the 450μm data and five pixels for the 850μm data. The Thresh and FwhmStart values we investigated between two and three, ModelLim between zero and two, NPad between two and twenty, and MaxNF between ten and two-hundred.

Due to the way the GaussClumps algorithm works, the clump index image as output from CUPID is not as informative as with the other three methods. In the other methods you are able to see how the algorithm allocates individual pixels to each clump showing their shape and position. Since with GaussClumps a single pixel could belong to multiple clumps it is impossible to produce a similar index image. Because of this only a general view of the entire clump area can be seen, making any assessment based purely on the output image limited.

Altering NPad had no effect on the data, either the number of clumps detected, their sizes and hence fluxes, irrespective of the value of NPad, see Figure 3.9.

MaxNF causes a decrease in the number of clumps detected and their size when a value less than the default is used, with a value greater than the default the decrease quickly stops, the number of clumps becomes constant and the shape of each clump becomes consistent with the other higher values of the parameter. From the Flux histograms in Figure 3.10 it can be seen that there are variations between the number of low flux value clumps detected when the MaxNF value changes, these variations decreases as MaxNF value increases, past the default value there is little difference between the number of low flux clumps detected. As the flux value increases the difference between different parameter values becomes nil. The clump size histograms show that again there is a lot of variation in the number of clumps detected at all clump sizes when the MaxNF value is changed, the only noticeable trend is when the parameter value is 75(green line) where almost no very small clumps are detected but the number of larger clumps detected is often greater than with all the other parameter values.

Altering ModelLim causes a sharp increase in the number of clumps detected as the parameter value increases from zero, this quickly trails off as the parameter value increases further, eventually leading to a gradual decrease in the number of clumps detected. This decrease is not smooth but contains many fluctuations, see Figure 3.11. The clump index images appear similar at parameter values greater than zero, though there is continual variability such that no two images look the same and it is not possible to determine the exact differences. The flux histograms show that the increase or decrease in the number of clumps occurs mostly with clumps of a low flux value. As such there is an increase in the number of detected clumps as the ModelLim value increases, this becomes a gradual decrease in the number of detected clumps as ModelLim continues to increase. Changing the value of ModelLim has no effect on the number of clumps of higher flux values. It is difficult to determine a specific trend in the

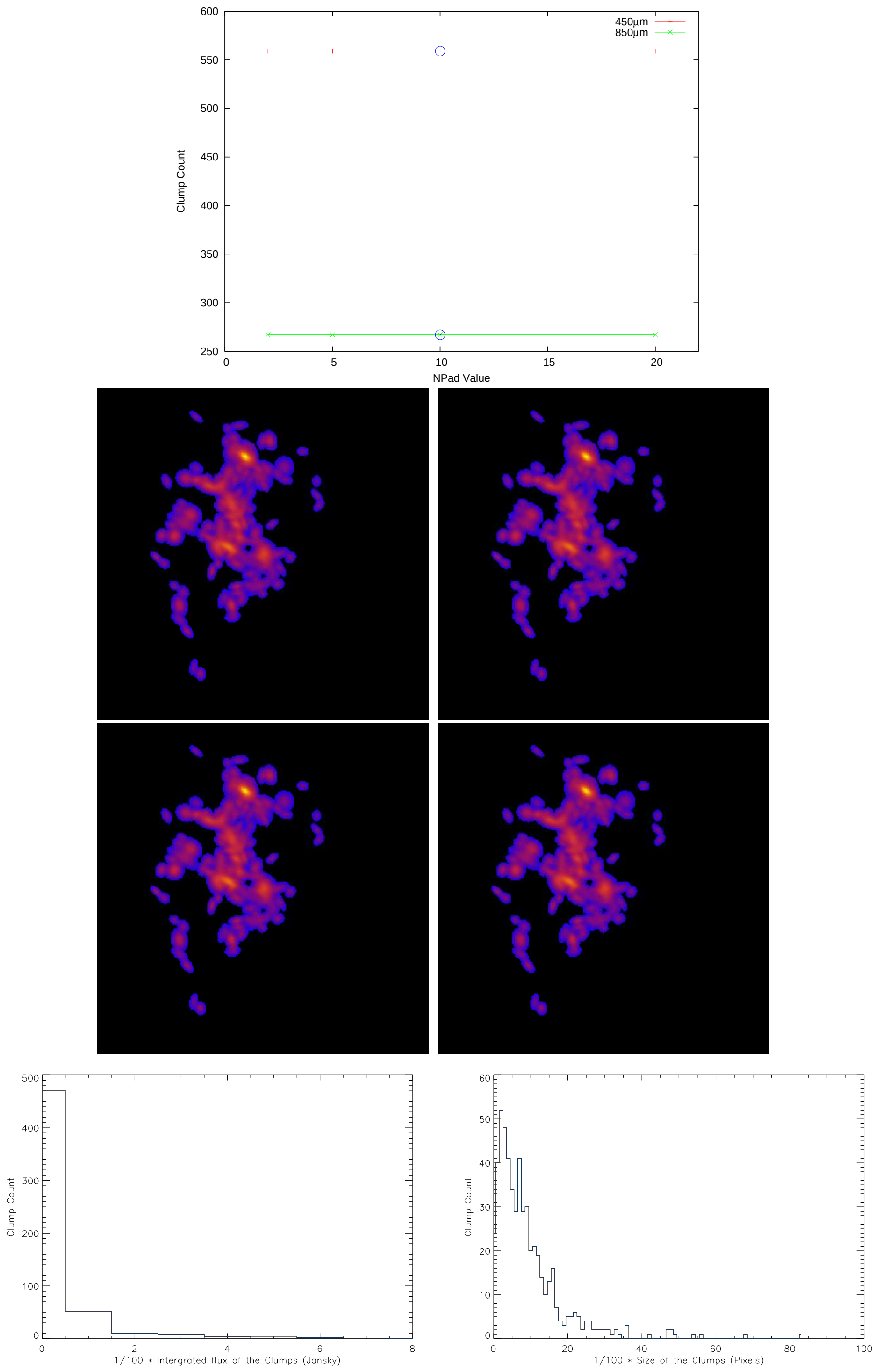
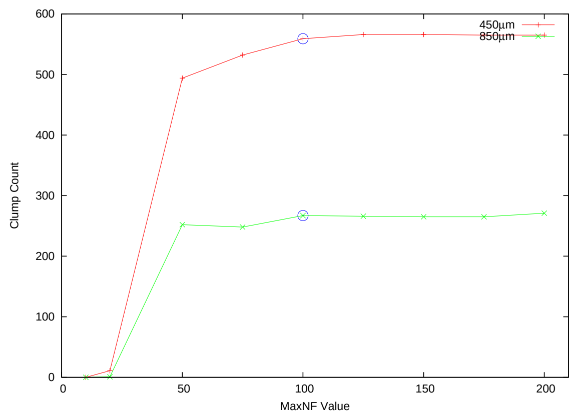


Figure 3.9: Top; Plot of how the number of clumps detected in all the images changes with different values for NPad using the GaussClumps algorithm, the blue circles showing the default values. Second and third; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of NPad changes, from second line left the values of NPad are; 2, and 5, from third line left the values are; 10, and 20. Bottom line; Histograms showing, from left the integrated flux of the detected clumps of the 450μm data, then the size of the detected clumps of the 450μm data for each of the different values for the NPad parameter.


450µm
850µm
Clump Count
MaxNF Value
0
100
200
300
400
500
600
0
50
100
150
200

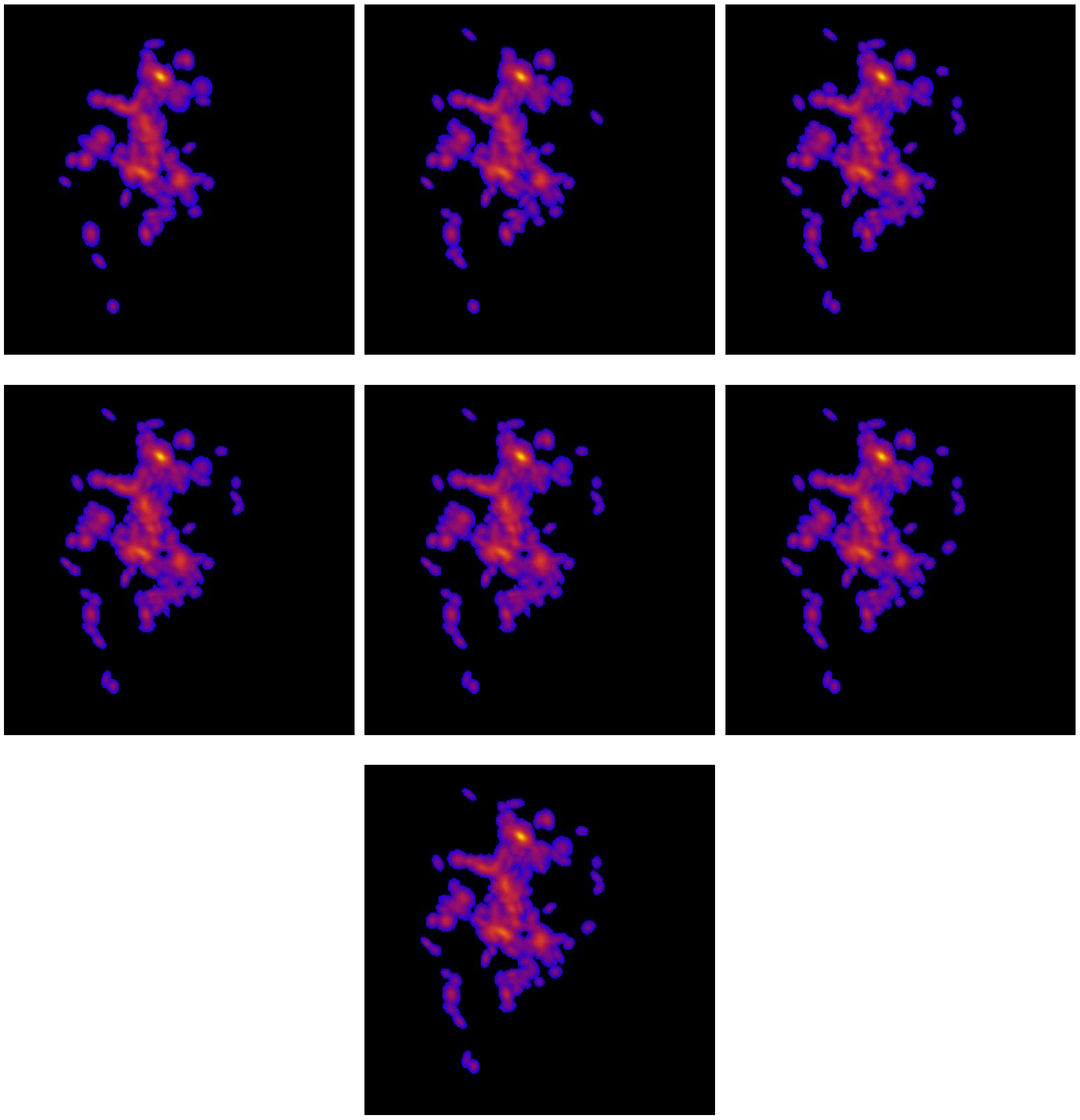


Figure 3.10:

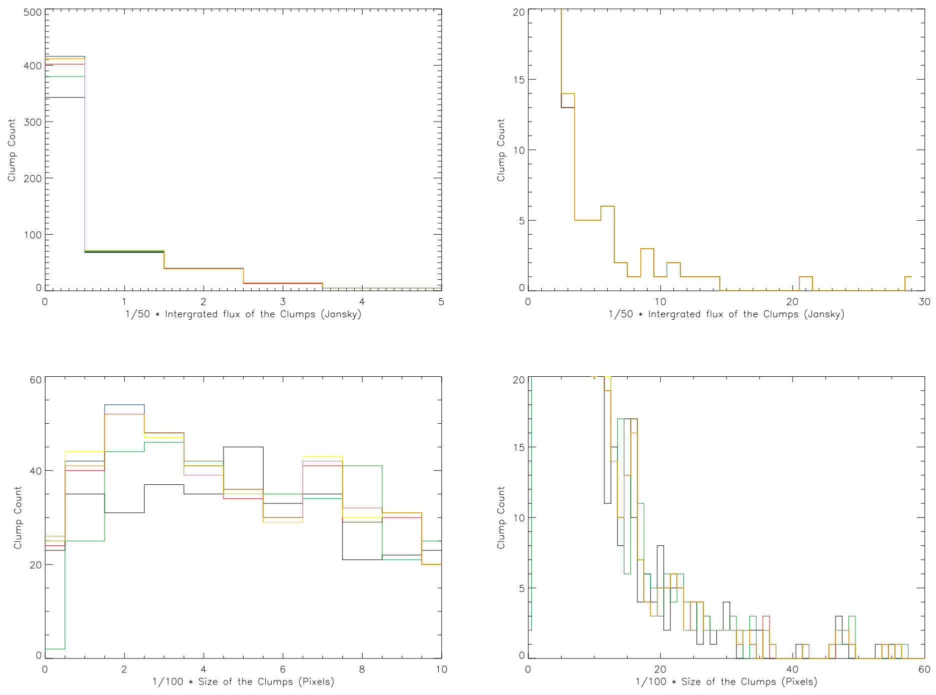


Figure 3.10: Previous page. Top; Plot of how the number of clumps detected in all the images changes with different values for MaxNF using the GaussClumps algorithm, the blue circles showing the default values. second to bottom line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of MaxNF changes, from second line, left the values of MaxNF are; 50, 75, and 100, from third line, left the values are; 125, 150 and 175, forth line value is; 200. This Page; Top; Histograms showing the integrated flux values of the 450μm data for the different parameter values 50(black line), 75(green line), 100(red line), 125(blue line), 150(orange line), and 175(yellow line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 50(black line), 75(green line), 100(red line), 125(blue line), 150(orange line), and 175(yellow line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

size of detected clumps due to the high amount of variability between the results for each of the parameter values.

Altering the value of FwhmStart has an effect on the number of clumps that are detected. With the 850μm data the number of detected clumps decreases as the value of the parameter increases, whereas for the 450μm data as the parameter value increases there is initially a decrease in the number of clumps, after which there is an increase in the number of detected clumps, see Figure 3.12. From the clump index images it appears that as the value of FwhmStart increases the number of clumps detected also increases and the clumps that were detected at low parameter values appear to be larger in size than at higher parameter values. From the histograms it appears that as FwhmStart increases the number of small clumps detected in general decreases and the number of larger clumps detected increases. This trend is the same with the flux values; as FwhmStart increases the number of low flux value clumps decreases and the number of higher flux value clumps that are detected increases. This leads to the conclusion that although increasing FwhmStart results in fewer clumps being detected the difference is with the small, low flux clumps, i.e. the effect is to cause smaller clumps to be detected as larger clumps.

Altering Thresh had no effect on the data, either the number of clumps detected, their sizes and hence the flux of the detected clumps. see Figure 3.13. This was due to the fact that none of the clumps detected had a peak value of less than three times the RMS value as such altering the parameter would have no effect. If the parameter value was increased much higher then a drop in the number of clump detected would be expected.

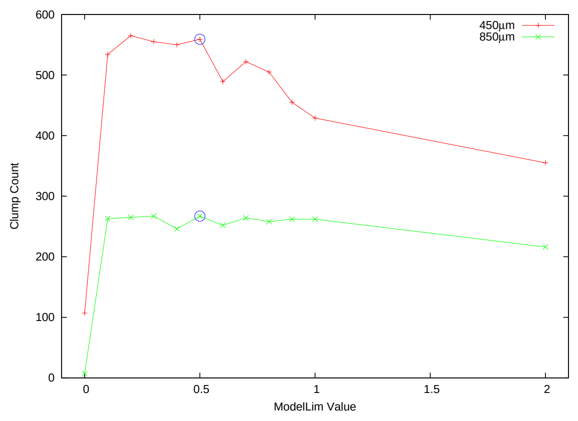


Figure 3.11: Plot of how the number of clumps detected in all the images changes with different values for ModelLim using the GaussClumps algorithm, the blue circles showing the default values.

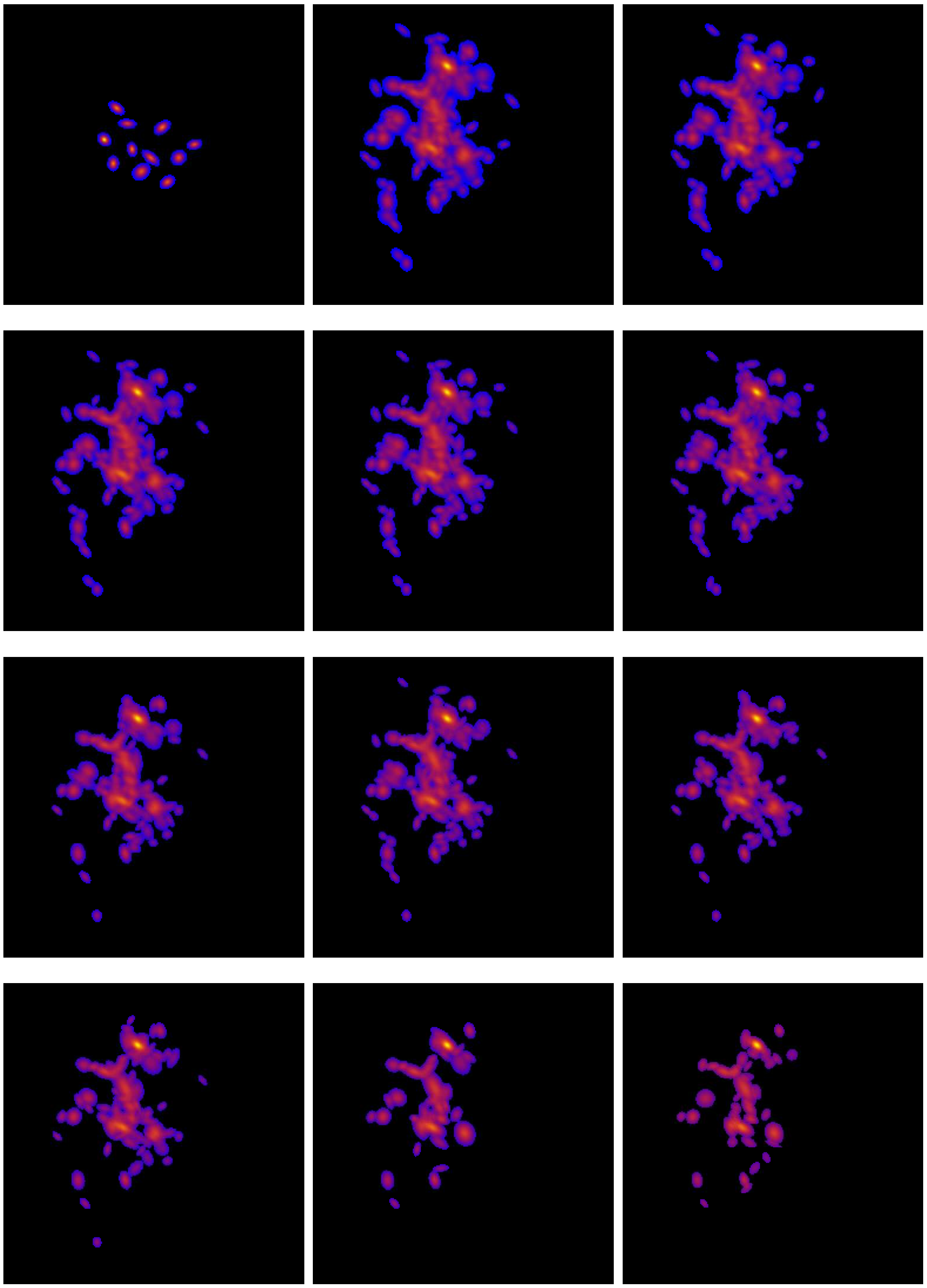


Figure 3.11: Top to bottom line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of ModelLim changes, from top line, left the values of ModelLim are; 0, 0.1, and 0.2, from second line, left the values are; 0.3, 0.4 and 0.5, from third line, left the values are; 0.6, 0.7, and 0.8, from fourth line, left the values are; 0.9, 1.0, and 2.0.

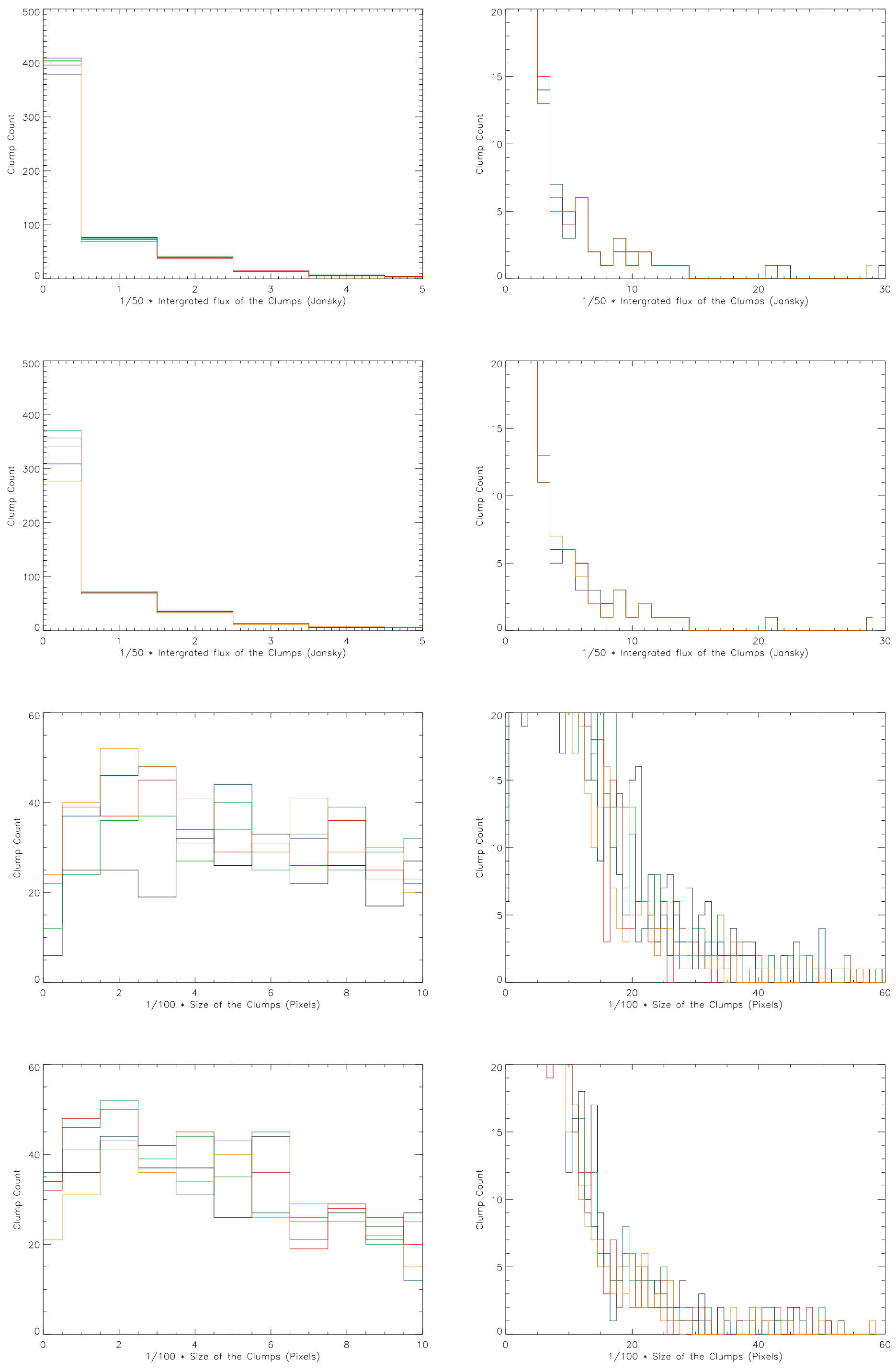
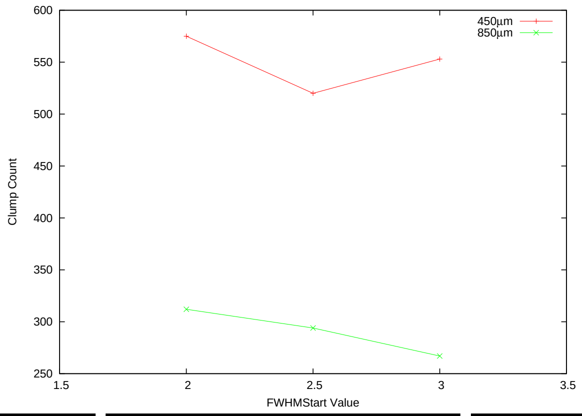
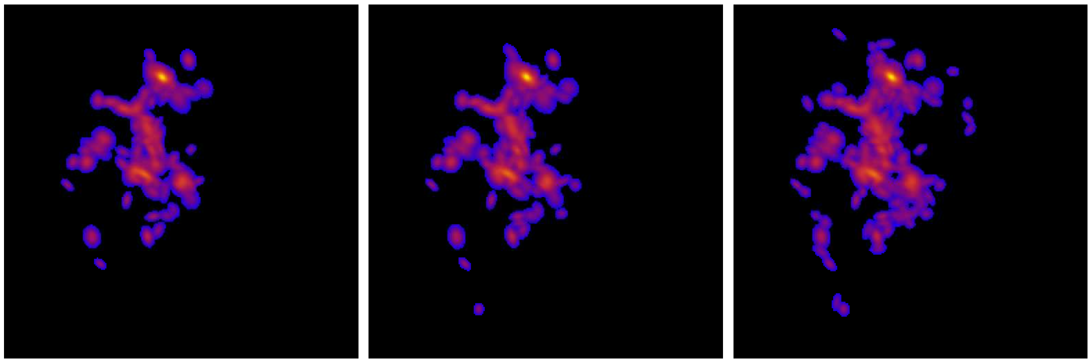


Figure 3.11: Top and second lines; Histograms showing the integrated flux values of the 450μm data for the different parameter values, top line; 0.1(black line), 0.2(green line), 0.3(red line), 0.4(blue line), and 0.5(orange line), second line; 0.6(black line), 0.7(green line), 0.8(red line), 0.9(blue line), and 1.0(orange line), the left histograms being a magnified view of the low integrated flux clumps, the right ones showing the full range of fluxes but limited clump count. Third and bottom line; Histograms showing the size of the clumps of the 450μm data for the different parameter values, third line; 0.1(black line), 0.2(green line), 0.3(red line), 0.4(blue line), and 0.5(orange line), bottom line; 0.6(black line), 0.7(green line), 0.8(red line), 0.9(blue line), and 1.0(orange line), the left histograms being a magnified view of the small clumps, the right ones showing the full range of sizes but limited clump count.





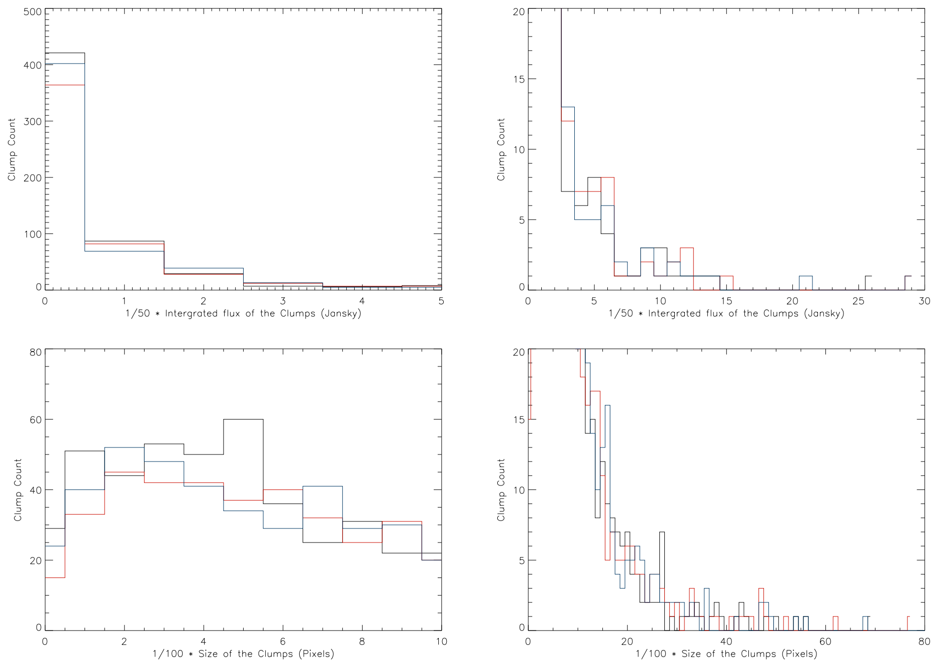

Clump Count
1/50 * Intergrated flux of the Clumps (Jansky)
1/100 * Size of the Clumps (Pixels)

Figure 3.12: Top; Plot of how the number of clumps detected in all the images changes with different values for FwhmStart using the GaussClumps algorithm. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of FwhmStart changes, from left the values of FWHMStart are; 2, 2.5, and 3. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 2(black line), 2.5(red line), and 3(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 2(black line), 2.5(red line), and 3(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

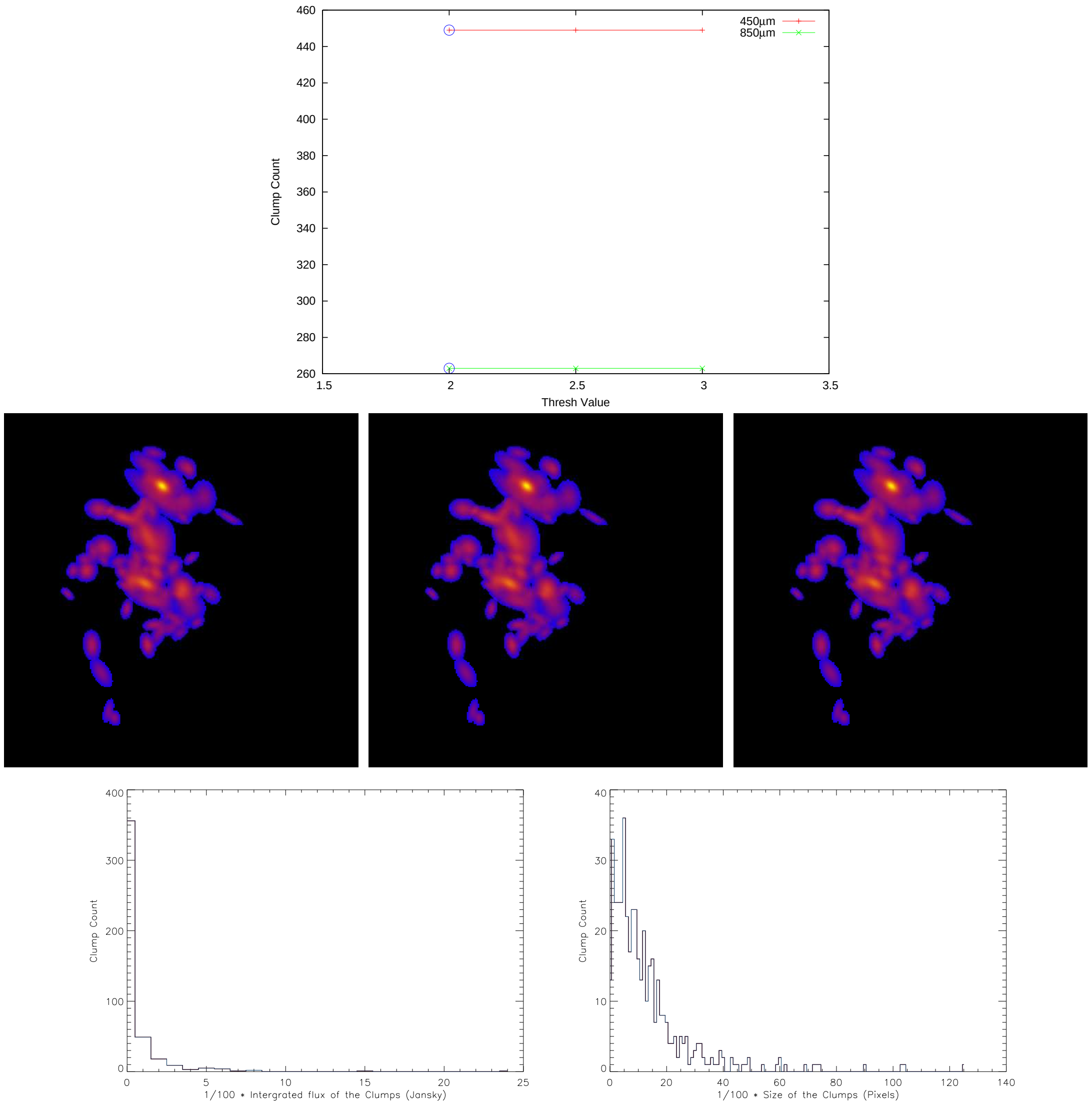


Figure 3.13: Top; Plot of how the number of clumps detected in all the images changes with different values for Thresh using the GaussClumps algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of Thresh changes, from left the values of Thresh are; 2, 2.5, and 3. Bottom line; Histograms showing, from left the integrated flux of the detected clumps of the 450μm data, then the size of the detected clumps of the 450μm data for each of the different values for the Thresh parameter.

3.2.5 Reinhold

In the Reinhold algorithm investigation the parameters altered were: CAIterations, CAThresh, FixClumpIterations, FlatSlope, MinLen, Noise, and Thresh. As with ClumpFind and Fell-Walker the Noise parameter was changed to investigate the differences between a value of 2*RMS and 3*RMS. FlatSlope was investigated between 0*RMS and 3*RMS, CAIterations and FixClumpIterations between zero and two, CAThresh and MinLen between zero and eight, and Thresh between 0*RMS+Noise and 3*RMS+Noise.

Altering the Noise parameter did not cause the number of clumps detected to change by a large amount, with the 850μm data the number of clumps detected decreased as the value for Noise increased, with the 450μm data the number of detected clumps initially increased before decreasing again, see Figure 3.14. The clump index images indicate the detection of a lower number of clumps but no further information could be obtained from them. The flux histogram shows that both changes in the detected number of clumps are only for clumps of a low flux value, as such higher flux values are unaffected by changing the Noise value. The clump size histogram on the other hand shows no continual trend for the entire range of clump sizes.

If the value of the MinLen parameter is less than the default value of four then the results are the same as the default in both the number of clumps detected and the clump index images. Once the value becomes greater than the default value then the number of clumps detected quickly decreases. The clump index images also show this decrease but with more irregularly shaped clumps as MinLen increases. Additional protrusions extend out of one clump in particular, see Figure 3.15. The flux histogram shows there is little variation in the results when the MinLen value is at the default value or less. At values greater than the default, the number of clumps detected for most of the flux histogram bins decreases, this decrease continues as the value for MinLen increases further. For values less than the default the clump sizes are mostly the same but at the default value a larger number of smaller clumps are detected and a lower number of larger clumps are detected. For values of MinLen that are greater than the default the number of clumps detected drops for all clump sizes as the parameter value continues to increase.

Altering Thresh had no effect on the data, either on the number of clumps detected, their sizes, and therefore the flux of the detected clumps, see Figure 3.16. This was probably to the fact that none of the clumps detected had a peak value of less than five times the RMS value as such altering the parameter would have no effect. If the parameter value was increased much higher then a drop in the number of clump detected would be expected.

Altering FixClumpIterations had no effect on the data, either on the number of clumps detected, their sizes, and therefore the flux of the detected clumps, see Figure 3.17.

CAIterations gives very different values for the number of clumps detected and their profile. For the 450μm data, values other than the default give similar results to each other but fewer clumps are detected than when the default value is used. With the 850μm data the trend is similar although if the value for CAIterations used is less than the default the number of

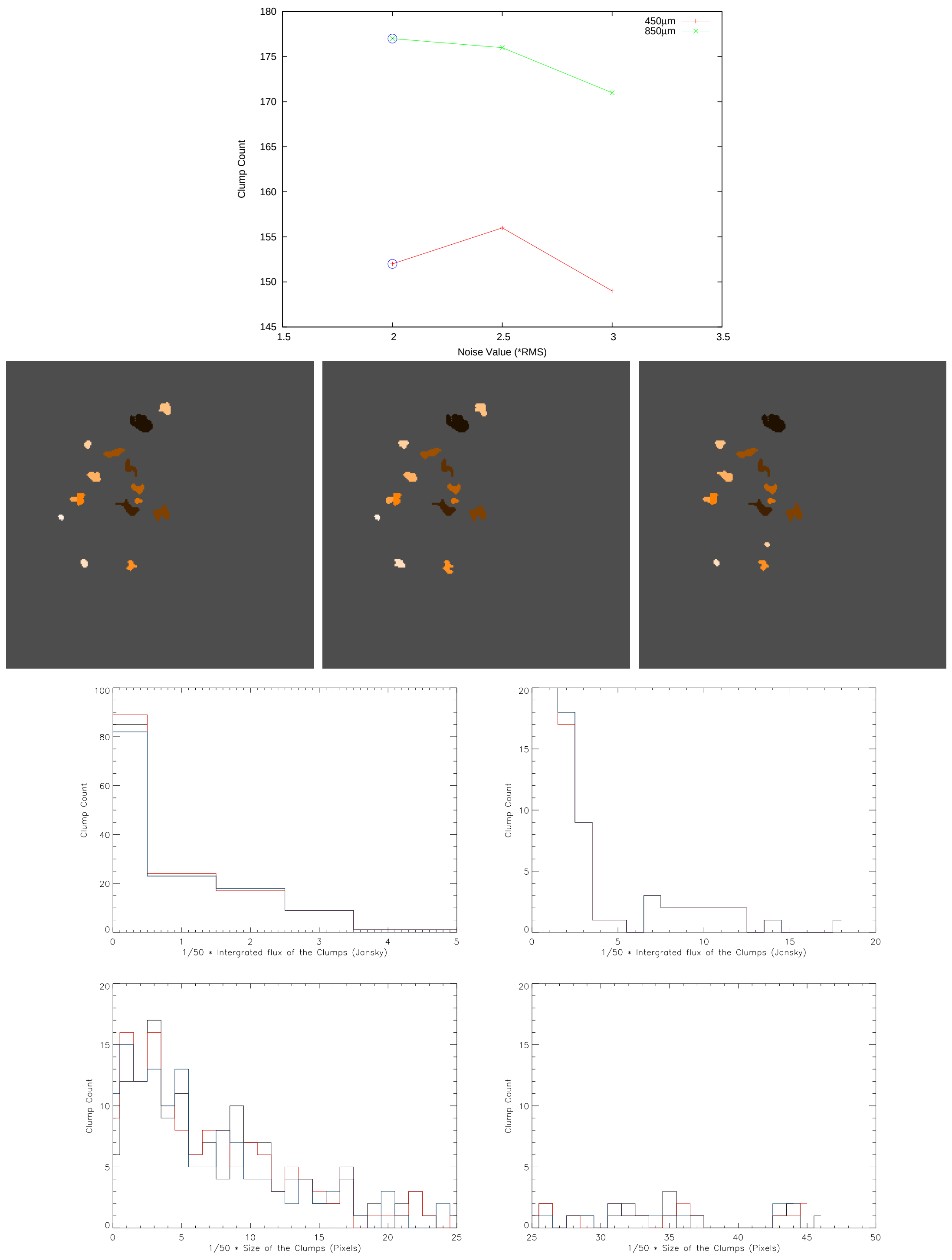


Figure 3.14: Top; Plot of how the number of clumps detected in all the images changes with different values for Noise using the Reinhold algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of Noise changes, from left the values of Noise are; 2, 2.5, and 3 times the RMS value. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 2*RMS(black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 2*RMS(black line), 2.5*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

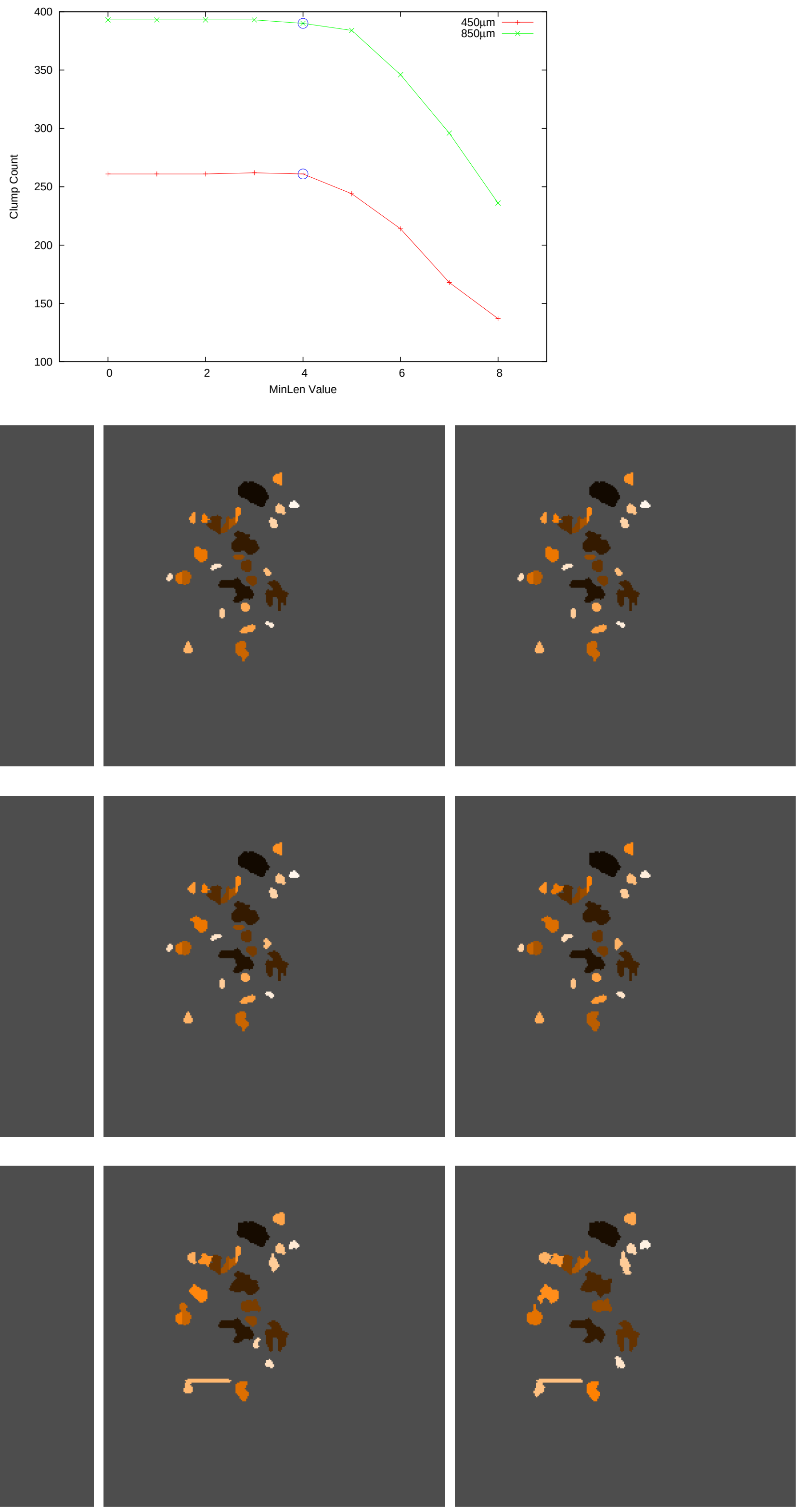


Figure 3.15: Top; Plot of how the number of clumps detected in all the images changes with different values for MinLen using the Reinhold algorithm, the blue circles showing the default values. second to bottom line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of MinLen changes, from second line, left the values of MinLen are; 0, 1, and 2, from third line, left the values are; 3, 4 and 5, from forth line, left the values are; 6, 7, and 8.

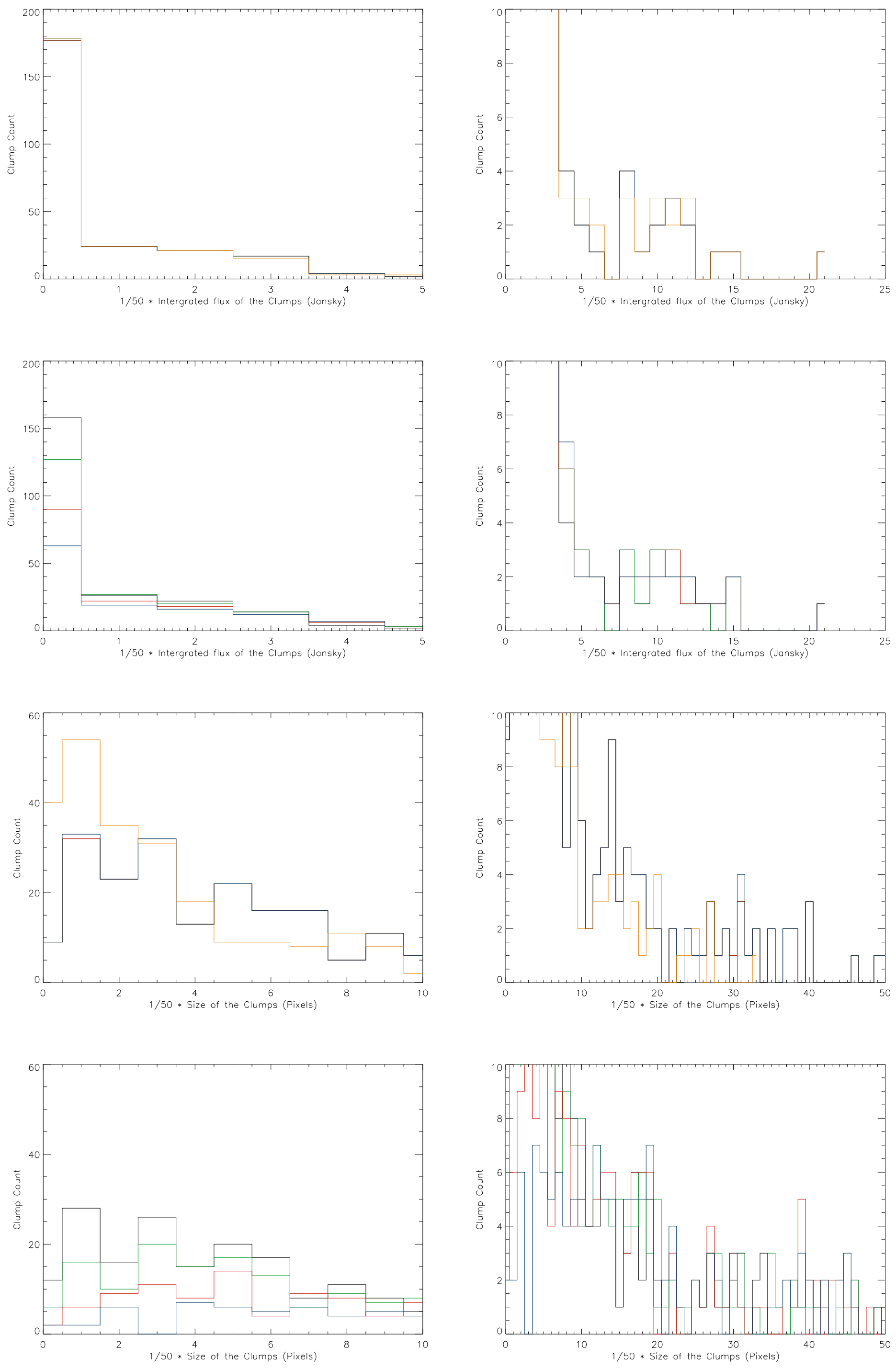
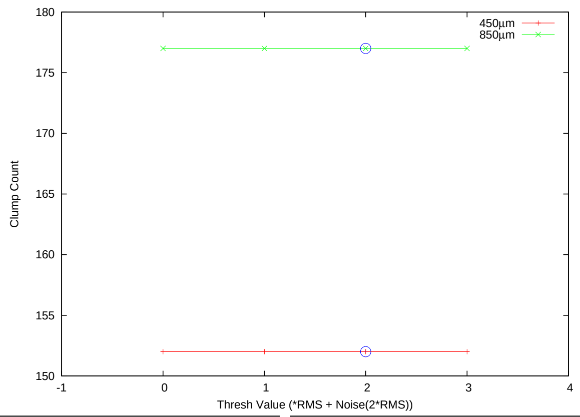
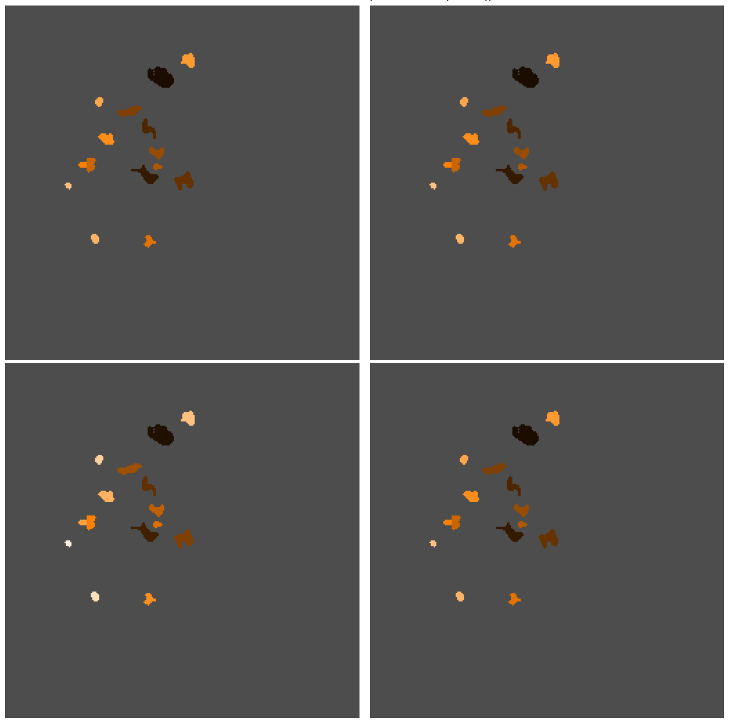


Figure 3.15: Top and second lines; Histograms showing the integrated flux values of the 450μm data for the different parameter values, top line; 0(black line), 1(green line), 2(red line), 3(blue line), and 4(orange line), second line; 5(black line), 6(green line), 7(red line), and 8(blue line), the left histograms being a magnified view of the low integrated flux clumps, the right ones showing the full range of fluxes but limited clump count. Third and bottom line; Histograms showing the size of the clumps of the 450μm data for the different parameter values, third line; 0(black line), 1(green line), 2(red line), 3(blue line), and 4(orange line), bottom line; 5(black line), 6(green line), 7(red line), and 8(blue line)), the left histograms being a magnified view of the small clumps, the right ones showing the full range of sizes but limited clump count.





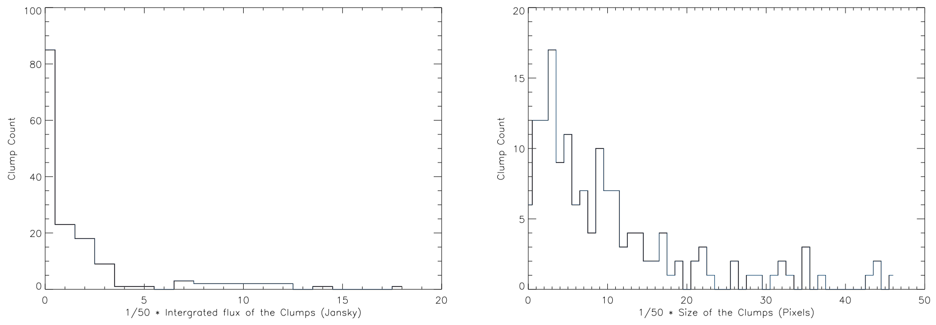

Clump Count
1/50 * Intergrated flux of the Clumps (Jansky)
Clump Count
1/50 * Size of the Clumps (Pixels)

Figure 3.16: Top; Plot of how the number of clumps detected in all the images changes with different values for Thresh using the Reinhold algorithm, the blue circles showing the default values. Second and third lines; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of Thresh changes, from middle left the values of Thresh are; 0, and 1 times the RMS value, from bottom left the values are; 2, and 3 times the RMS value. Bottom line; Histograms showing, from left the integrated flux of the detected clumps of the 450μm data, then the size of the detected clumps of the 450μm data for each of the different values for the NPad parameter.

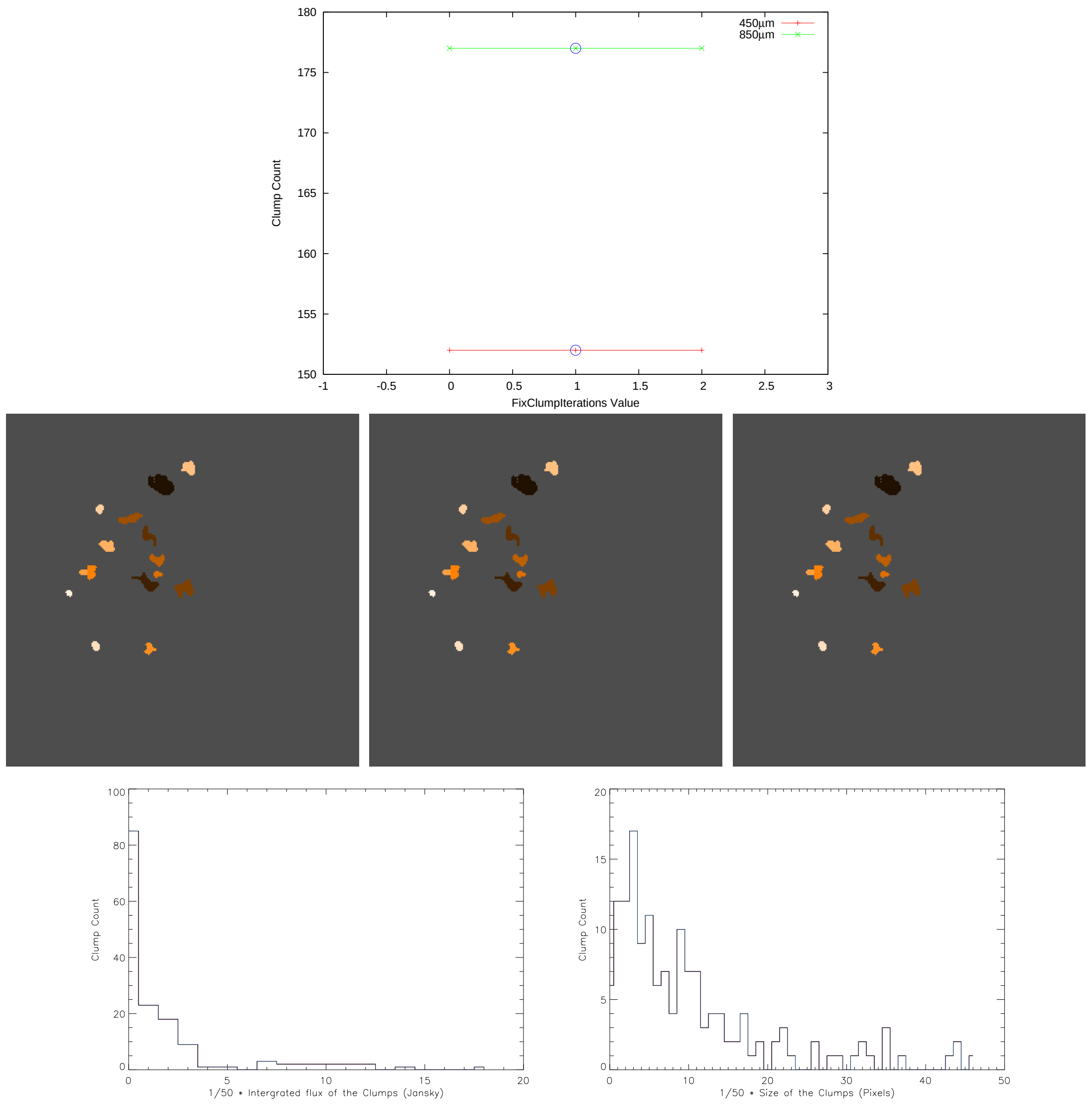


Figure 3.17: Top; Plot of how the number of clumps detected in all the images changes with different values for FixClumpIterations using the Reinhold algorithm. Middle line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of FixClumpIterations changes, from left the values of FixClumpIterations are; 0, 1, and 2. Bottom line; Histograms showing, from left the integrated flux of the detected clumps of the 450μm data, then the size of the detected clumps of the 450μm data for each of the different values for the NPad parameter.

clumps detected is similar to the number when the default value is used. The default and lower values of CAIterations give a greater number of detections than the number of clumps detected when the value used is greater than the default. The clump index images show this change not only in the number of clumps detected but also show that if the value used is greater than the default then the clump morphology becomes highly irregular for a large proportion of the clumps, see Figure 3.18. The histogram of the clump size shows that if the CAIterations value is less than the default the number of clumps detected is less for all clump sizes than at the default but still follows a similar trend with many small clumps being detected but fewer larger clumps. If the parameter value is greater than the default then the number of detected clumps is again lower but the trend changes in that there are only a few small clumps detected but more larger clumps. The flux histograms follow the same trend with parameter values less than the default having the same trend as the default but with fewer detections and parameter values greater than the default resulting in an opposite trend. These results match what can be seen in the clump index images showing that above the default value the clumps become highly irregular in their shape, causing a corresponding change in the clump sizes and their fluxes.

As the FlatSlope value increases the number of clumps detected decreases by a factor of seven over all the values investigated, see Figure 3.19. The clump index images not only show this decrease in the number of clumps detected, but also that as the parameter value increases the few remaining clumps start to develop an irregular shape with protrusions. The histograms also show that as the value of FlatSlope increases the number of clumps detected decreases for all ranges of clump size and clump flux. As FlatSlope increases further the number of clumps detected continues to decrease throughout the range of size and flux values.

For CAThresh the number of detected clumps increases as CAThresh increases, the clump index images also show this effect and there are still some noticeable irregular protrusions from a few clumps as the parameter value increases, see Figure 3.20. The irregularities are less severe than those noticed when altering other parameters such as FlatSlope and CAIterations. The histograms show that as the value of CAThresh increases the number of clumps detected increases for all ranges of clump size and clump flux, as CAThresh increases further the number of clumps detected still continues to increase throughout the range of size and flux values.

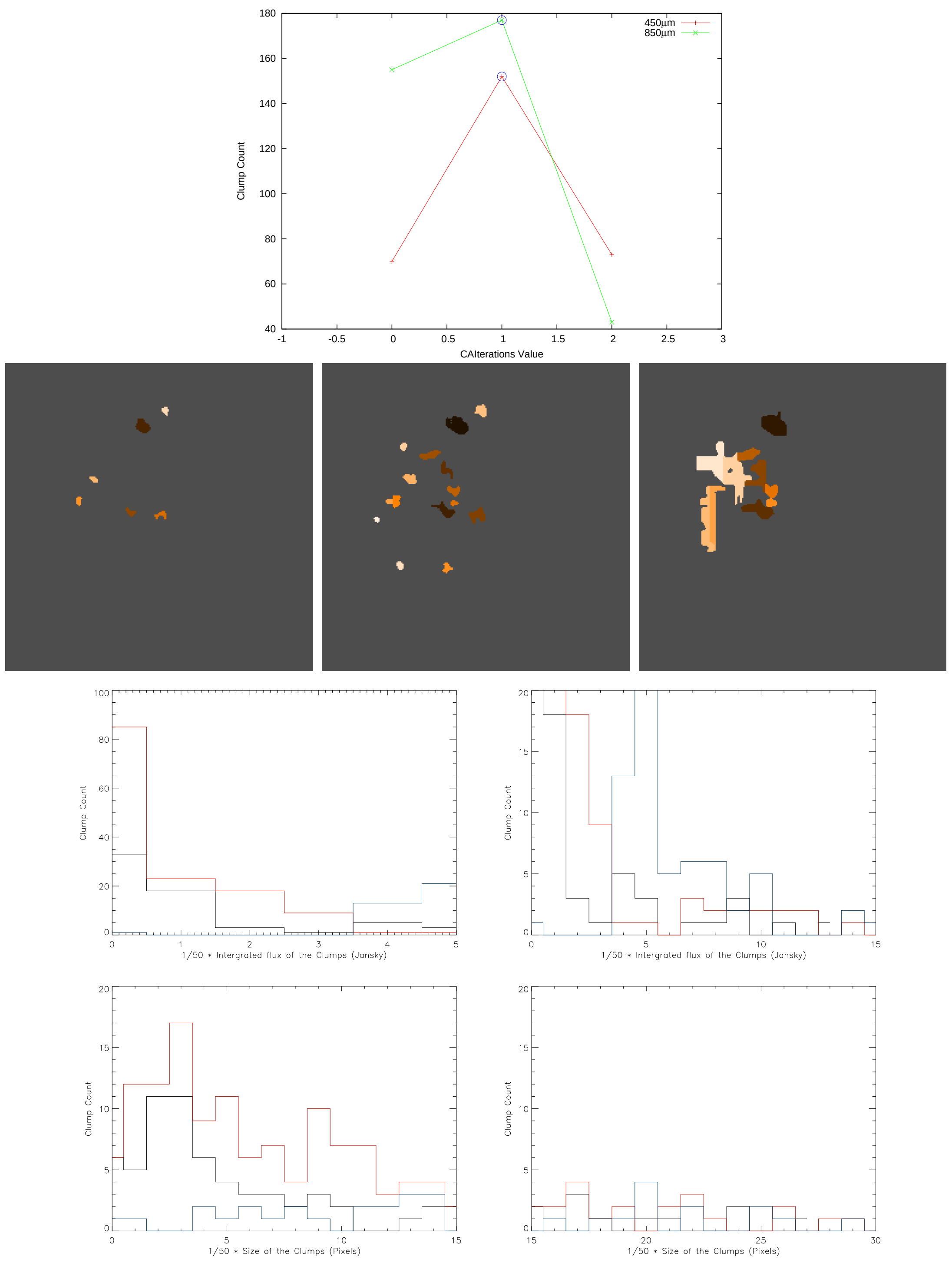


Figure 3.18: Top; Plot of how the number of clumps detected in all the images changes with different values for CAIterations using the Reinhold algorithm, the blue circles showing the default values. Second line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of CAIterations changes, from left the values of CAIterations are; 0, 1, and 2. Third line; Histograms showing the integrated flux values of the 450μm data for the different parameter values 0(black line), 1(red line), and 2(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 0(black line), 1(red line), and 2(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.

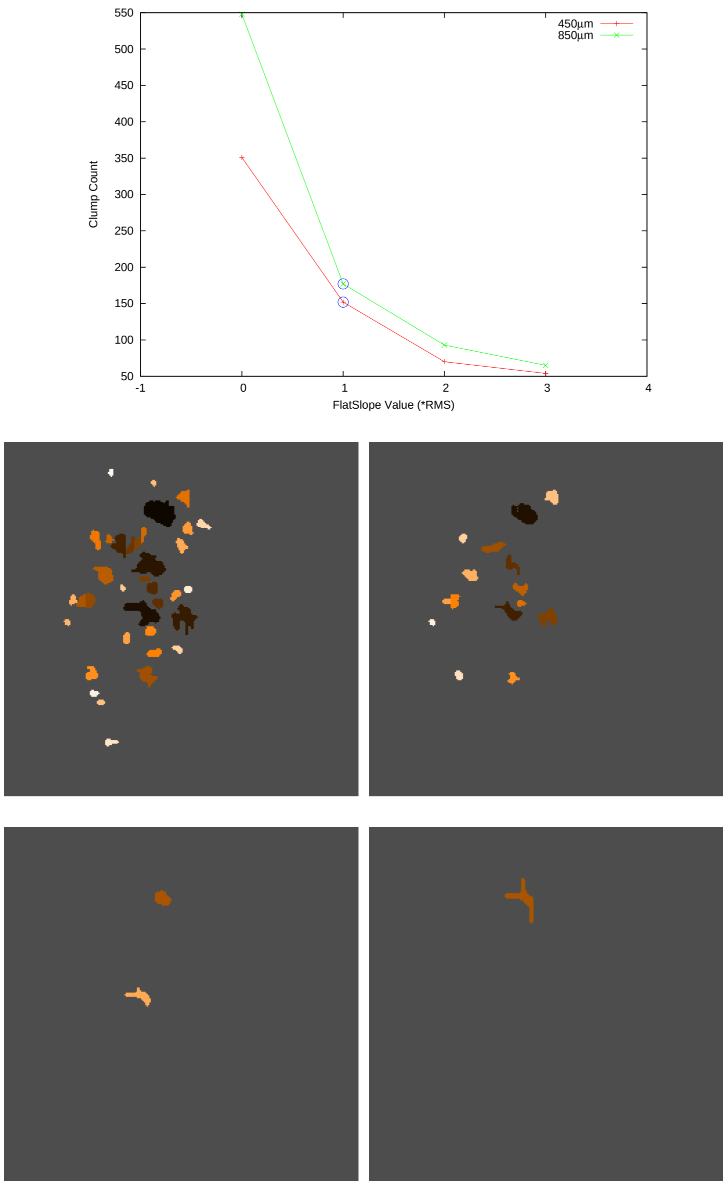


Figure 3.19: Top; Plot of how the number of clumps detected in all the images changes with different values for FlatSlope using the Reinhold algorithm, the blue circles showing the default values. Middle and bottom; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of FlatSlope changes, from middle left the values of FlatSlope are; 0, and 1 times the RMS value, from bottom left the values are; 2, and 3 times the RMS value.

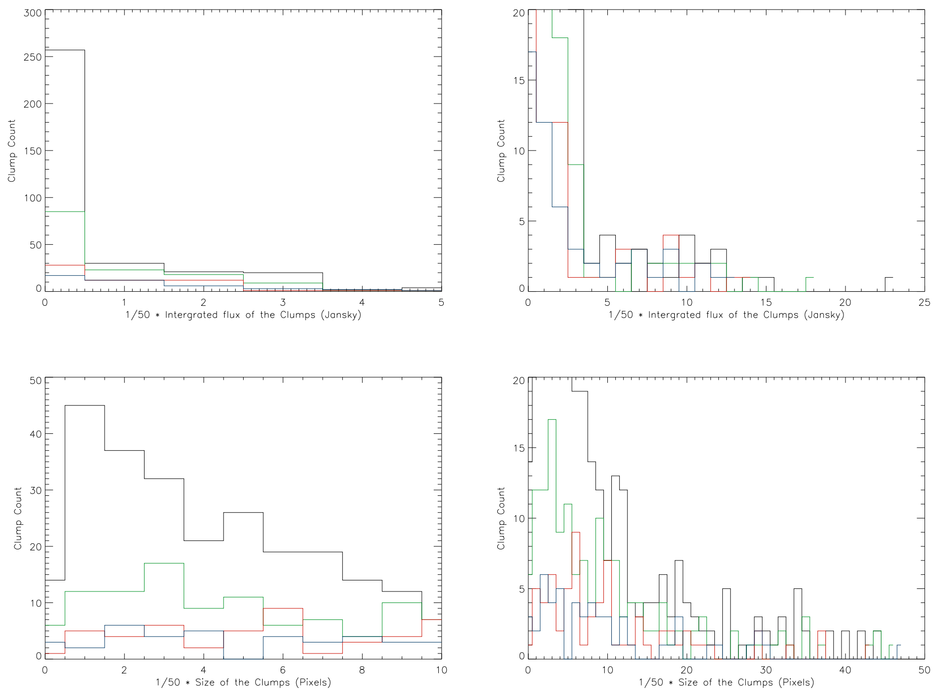
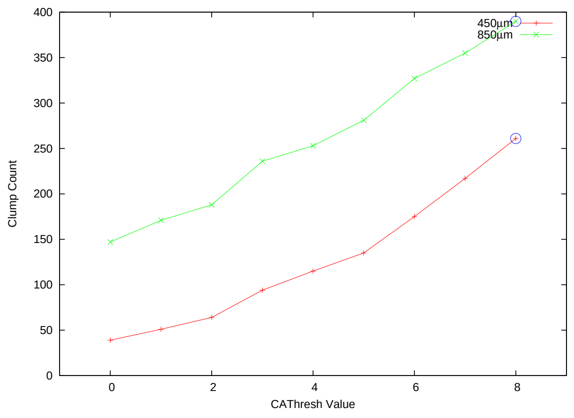


Figure 3.19: Top; Histograms showing the integrated flux values of the 450μm data for the different parameter values 0*RMS(black line), 1*RMS(green line), 2*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the low integrated flux clumps, the right showing the full range of fluxes but limited clump count. Bottom; Histograms showing the size of the clumps of the 450μm data for the different parameter values 0*RMS(black line), 1*RMS(green line), 2*RMS(red line), and 3*RMS(blue line), the left histogram being a magnified view of the small clumps, the right showing the full range of sizes but limited clump count.



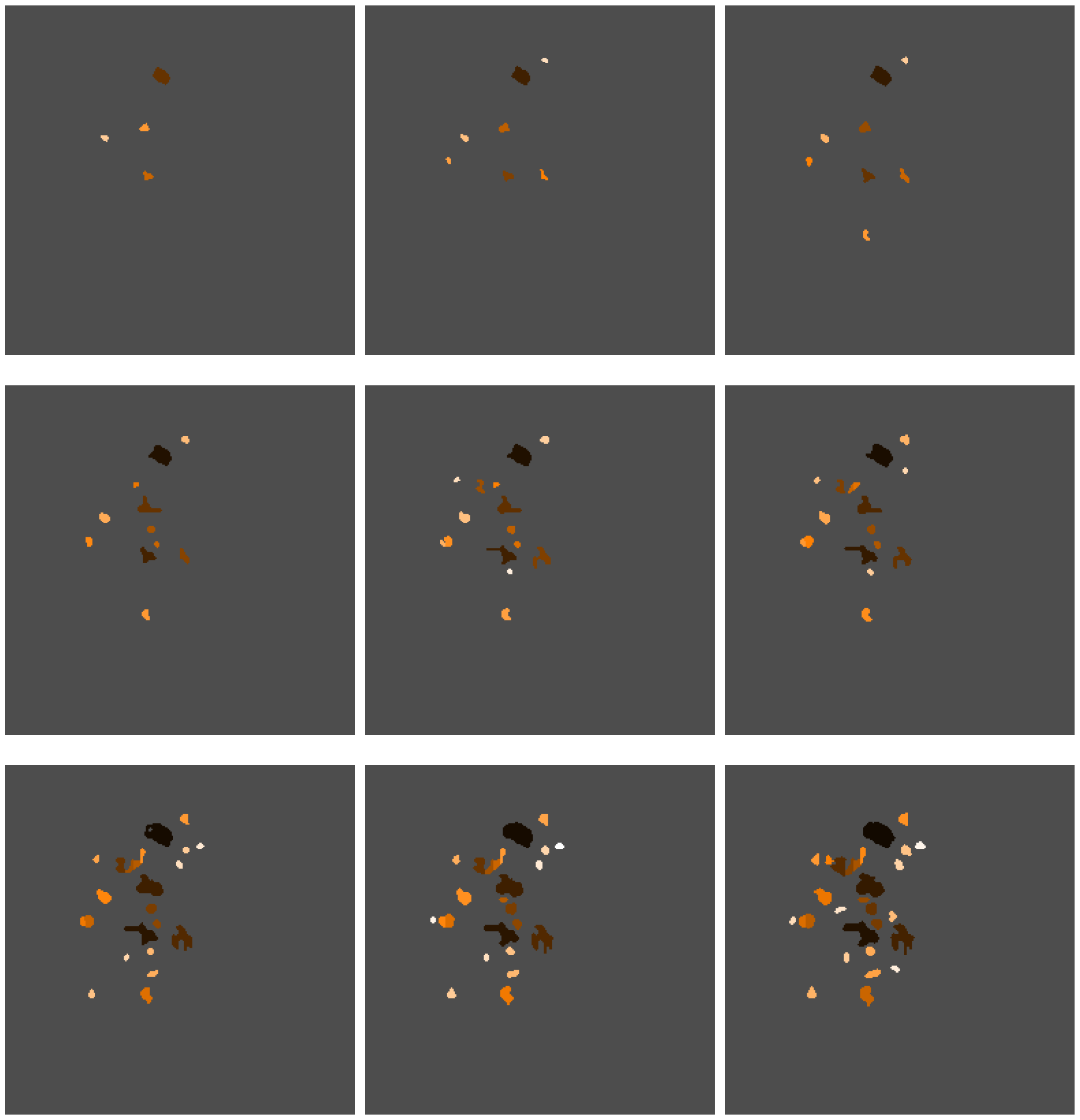


Figure 3.20: Top; Plot of how the number of clumps detected in all the images changes with different values for CAThresh using the Reinhold algorithm, the blue circles showing the default values. second to bottom line; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' showing the profiles of each of the detected clumps as the value of CAThresh changes, from second line, left the values of CAThresh are; 0, 1, and 2, from third line, left the values are; 3, 4 and 5, from forth line, left the values are; 6, 7, and 8.

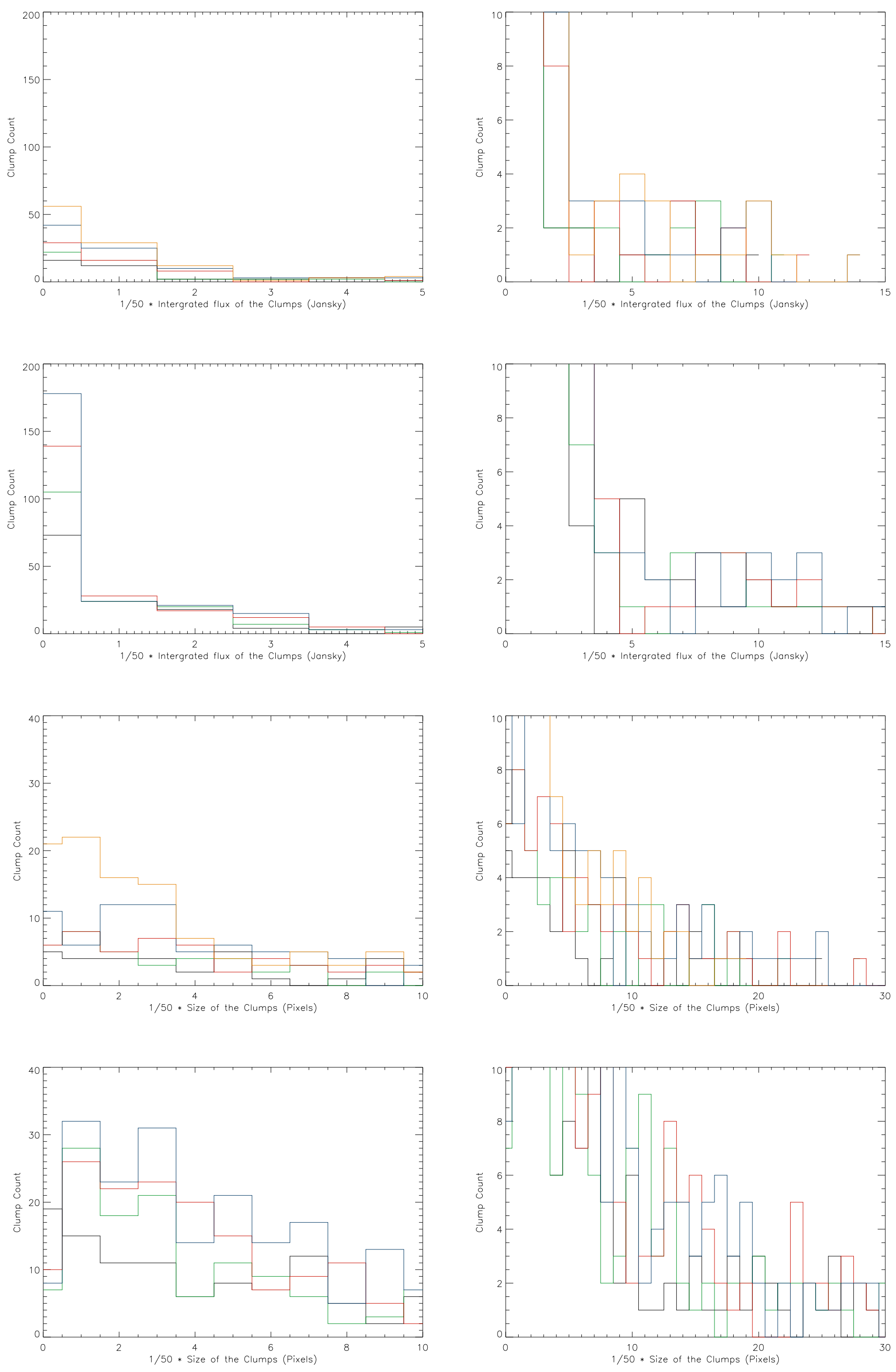


Figure 3.20: Top and second lines; Histograms showing the integrated flux values of the 450μm data for the different parameter values, top line; 0(black line), 1(green line), 2(red line), 3(blue line), and 4(orange line), second line; 5(black line), 6(green line), 7(red line), and 8(blue line), the left histograms being a magnified view of the low integrated flux clumps, the right ones showing the full range of fluxes but limited clump count. Third and bottom line; Histograms showing the size of the clumps of the 450μm data for the different parameter values, third line; 0(black line), 1(green line), 2(red line), 3(blue line), and 4(orange line), bottom line; 5(black line), 6(green line), 7(red line), and 8(blue line)), the left histograms being a magnified view of the small clumps, the right ones showing the full range of sizes but limited clump count.

3.3 Conclusions

Of the four algorithms Fellwalker experienced the least amount of change in the number of clumps detected when altering the parameters. MinDip and MaxJump both had a large difference between the highest and lowest clump detection count, but these were no more than a factor of two where as the other algorithms experienced changes up to a factor of seven. Taking this into account FellWalker could be deemed to be a more robust algorithm, as there are no large fluctuations in the values obtained from the same data when the parameters are changed. FellWalker does provide the opportunity to make adjustments to many parameters though altering them has a small effect of the results obtained. Since the results do not change by a large amount when the parameters are changed then it is possible FellWalker gives values that are consistently near a ”true” value, while with the other algorithms this range would be greater owing to greater variation in the results.

One notable point from the results is that ClumpFind and GaussClumps consistently found a greater number of clumps with the 450μm data than with the 850μm data. FellWalker and Reinhold were the reverse of this, finding a greater number of clumps in the 850μm data than in the 450μm data. It is to be expected that the algorithms would detect different values for the number of clumps, but it would be assumed that a particular wavelength would consistently provide a greater number of detected clumps. A possible reason why ClumpFind and GaussClumps detected more with the 450μm data than with the 850μm data is that the 450μm images are of a higher noise level than the 850μm images. This leads to larger clumps being broken up into smaller clumps due to noise spikes being assumed to be real peaks. FellWalker and Reinhold may not be so affected by the greater noise levels, therefore clumps are not broken up leading to the detection of fewer clumps in the 450μm data. Another theory could be due to the beam size, where the 450μm images contain a greater resolution than the 850μm images, to this end larger clumps may be again broken up or smaller details are more defined making the clumps more distinct and noticeable, especially small clumps. Therefore more clumps can be detected in the 450μm data than with the 850μm data. Once again ClumpFind and GaussClumps support this hypothesis yet it does not account for the reversal when using FellWalker or Reinhold. An explanation why FellWalker and Reinhold detect more clumps in the 850μm data than in the 450μm data could be due to the fact that the 850μm data provides a deeper image, therefore clumps that would be too faint in the 450μm images are more noticeable in the 850μm images which FellWalker and Reinhold detect yet ClumpFind and GaussClumps are unable to do so.

Reinhold rarely detected clumps that touched each other. The other algorithms would fill the entire clump area and consist of a number of clumps next to each other, whereas Reinhold’s clumps, although in similar locations as the other algorithms were smaller in size to such an extent that very few clumps came edge to edge. Also the Reinhold clump shape was often unexpected and contained protrusions. The reason for this is likely to be due to the cleaning processes that occur during the clumpfinding procedure when Reinhold is used. The edges of the clumps are eroded away to remove unusual protrusions from the clumps that might have been detected, this would make the detected clumps generally smaller than the algorithm originally detected them to be. After the first cleaning process the clumps are then indexed. Unfortunately once the initial cleaning has been performed the clumps may contain gaps around their edges, because of this when the clumps are indexed the algorithm does not know where to stop as there is no edge to the clump and the indexing continues outside of the clump area. This fault could lead to the protrusions and the unusual shapes of the detected clumps. After the indexing has been performed another cleaning process is performed causing the clump sizes to be reduced still further. A further issue with Reinhold was the FlatSlope parameter, its default value being 1*RMS. If the default value of FlatSlope was used the number of clumps detected appeared low when compared with Figure 1.1, when the parameter was increased from the default value then the number of clumps detected decreased and their profile became highly irregular (Figure 3.19). When the FlatSlope value was reduced to zero the number of detections increased to what appeared to be more accurate, therefore the default value for FlatSlope appears to be set too high and requires alteration. As a result of these issues the Reinhold algorithm in its present state is considered to be unsuitable for the purpose of accurate clump detection, particularly using SCUBA images. Despite this I continued to analyse the algorithm in parallel with the other algorithms for the remainder of this investigation to examine how it compared with each of them.

With a number of the parameters for each of the algorithms the default was often a limit such that if the value went one side of this default value the clumps detected did not alter from that of the default value but if the parameter value were be the other side of the default value then there would be a rapid change in the number of clumps, and their profile. To this extent the default of such parameters is advisable since deviation can cause a significant variation in the results.

It was mentioned that ClumpFind could be breaking up large single clumps into multiple clumps with one main clump and many satellite clumps, this effect has also been observed in Enoch et al. (2006). It is also possible that ClumpFind is determining the clumps correctly and other algorithms, such as FellWalker, are merging the clumps therefore giving a number of clumps fewer than are truly present. Using one of the original images, as shown in Figure 1.1 and comparing with the clump index images shown in Figures 3.1 - 3.20 it can be seen by eye that ClumpFind does indeed appear to detect more clumps than what appears to be present in the original image. Specifically finding a number of clumps where only one large clump may be present. It was also noted that the CleanIter parameter in FellWalker eroded the edges of clumps away and in certain cases made the clump small enough that they were then assumed to no longer be valid by the MinPix parameter. This effect would reduce the number of clumps detected by FellWalker over the other algorithms, by looking at the Figures 1.1 and 3.6 it can be seen that the CleanIter parameter can remove clumps that are clearly present in the original image. This process will only effect small and isolated clumps and having the CleanIter parameter value as low as possible helps to reduce the effect of clumps being eroded away.

Chapter 4

Algorithm Cross Comparison

4.1 Overview

Having investigated how changing the parameters of each algorithm has an effect on the clumps that are detected, the next step was then to compare the algorithms against one another. To do this the parameters would be kept constant at values that provided concordance between the algorithms such that the comparison would be fair. This was done using the number of clumps detected and the index images as shown in Figures 3.1 - 3.20. The parameter values that were chosen were ones that when combined gave a clump count and a clump profile that was similar for each algorithm, thus giving the best direct comparison between the algorithms. If a single parameter was present in multiple algorithms or if each algorithm had a parameter that performed the same function as another parameter for a different algorithm then they were matched together and their values kept constant. If a parameter had been found to result in a great amount of variation in the results when its value was altered from the default then that value was kept at its default value for these comparisons.

Listed in tables 4.1 - 4.4 are the parameters chosen for each algorithm. If a parameter is not mentioned in the table then it was left as the default value as given in §2.2. As few parameters were changed as possible so that the algorithms were compared against each other with parameters that are similar to those that would normally be used for the clumpfinding process. Figures 4.1 and 4.2 show the clump index images for each algorithm when using these parameters, each one is an image of the file 'g1015-034_SCAMPS'. The Figures also show the original image overlaid with the minimum contour and a contour spacing of three times the RMS background level of the image.

Table 4.1: Parameter values for the ClumpFind algorithm that gave the best concordance with the other algorithms for the cross comparison.

Parameter	Default Value	450μm	850μm
MinPix	7	9	25
DeltaT	2*RMS	3*RMS	3*RMS
TLow	2*RMS	3*RMS	3*RMS

Table 4.2: Parameter values for the FellWalker algorithm that gave the best concordance with the other algorithms for the cross comparison.

Parameter	Default Value	450μm	850μm
AllowEdge	1	0	0
MinPix	7	9	25
Noise	2*RMS	3*RMS	3*RMS
FlatSlope	1*RMS	3*RMS	3*RMS
MinDip	3*RMS	1.5*RMS	1.5*RMS

Table 4.3: Parameter values for the GaussClumps algorithm that gave the best concordance with the other algorithms for the cross comparison.

Parameter	Default Value	450μm	850μm
MinPix	7	9	25
FwhmBeam	2	3	5
FwhmStart	n/a	3	3
Thresh	2	3	3

Table 4.4: Parameter values for the Reinhold algorithm that gave the best concordance with the other algorithms for the cross comparison.

Parameter	Default Value	450μm	850μm
MinPix	9	9	25
Noise	2*RMS	3*RMS	3*RMS
FlatSlope	1*RMS	0	0

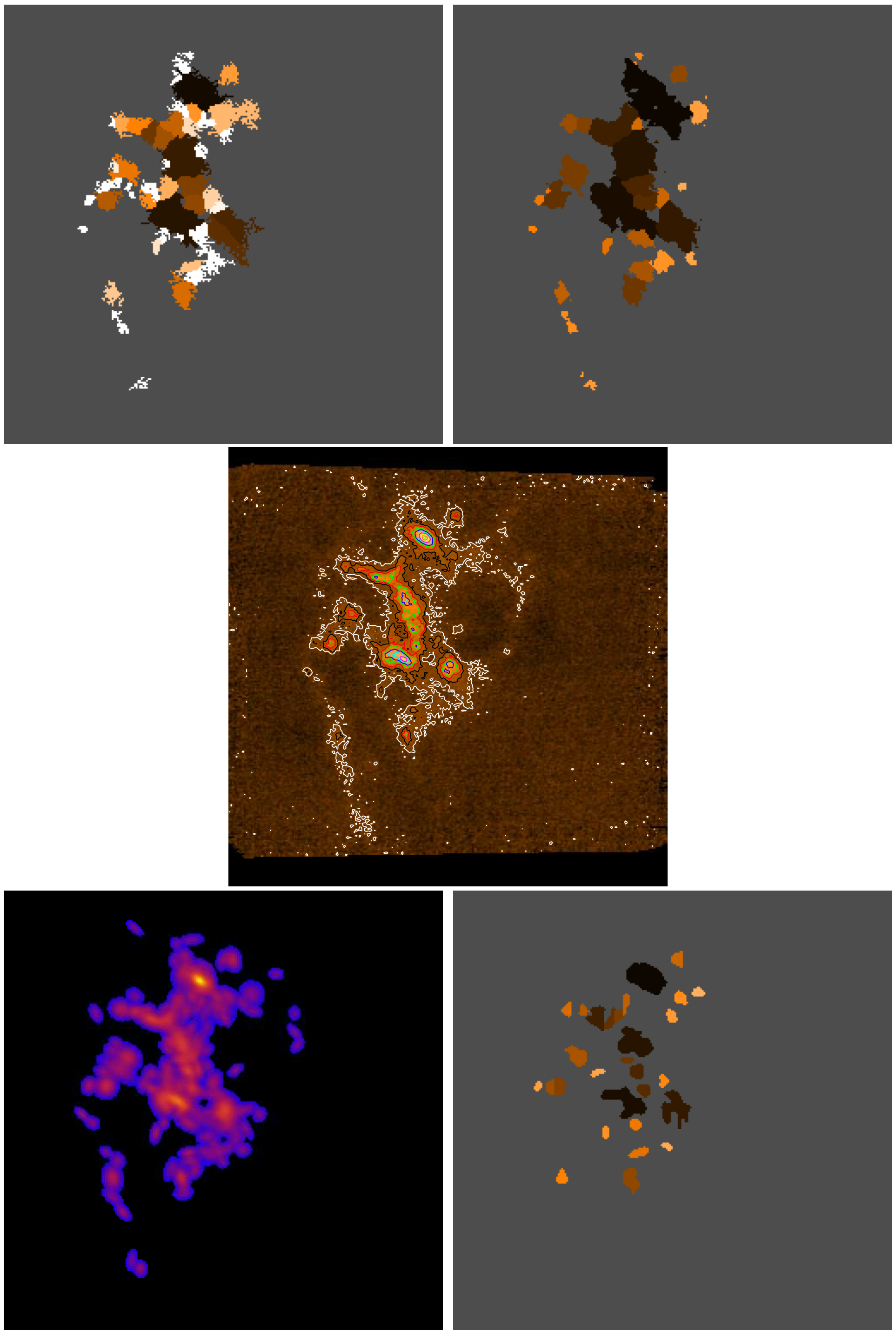


Figure 4.1: Top, from left; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' for ClumpFind (left) and FellWalker (right) using the parameter values as mentioned in Table 4.1 and Table 4.2. Middle; g1015-034_scamps_sho_flat.sdf with contour levels starting from three times the RMS value with spacing between the contours of three times the RMS value. Bottom; Output files from CUPID of the image 'g1015-034_scamps_sho_flat.sdf' for GaussClumps (left) and Reinhold (right) using the parameter values as mentioned in Table 4.3 and Table 4.4.

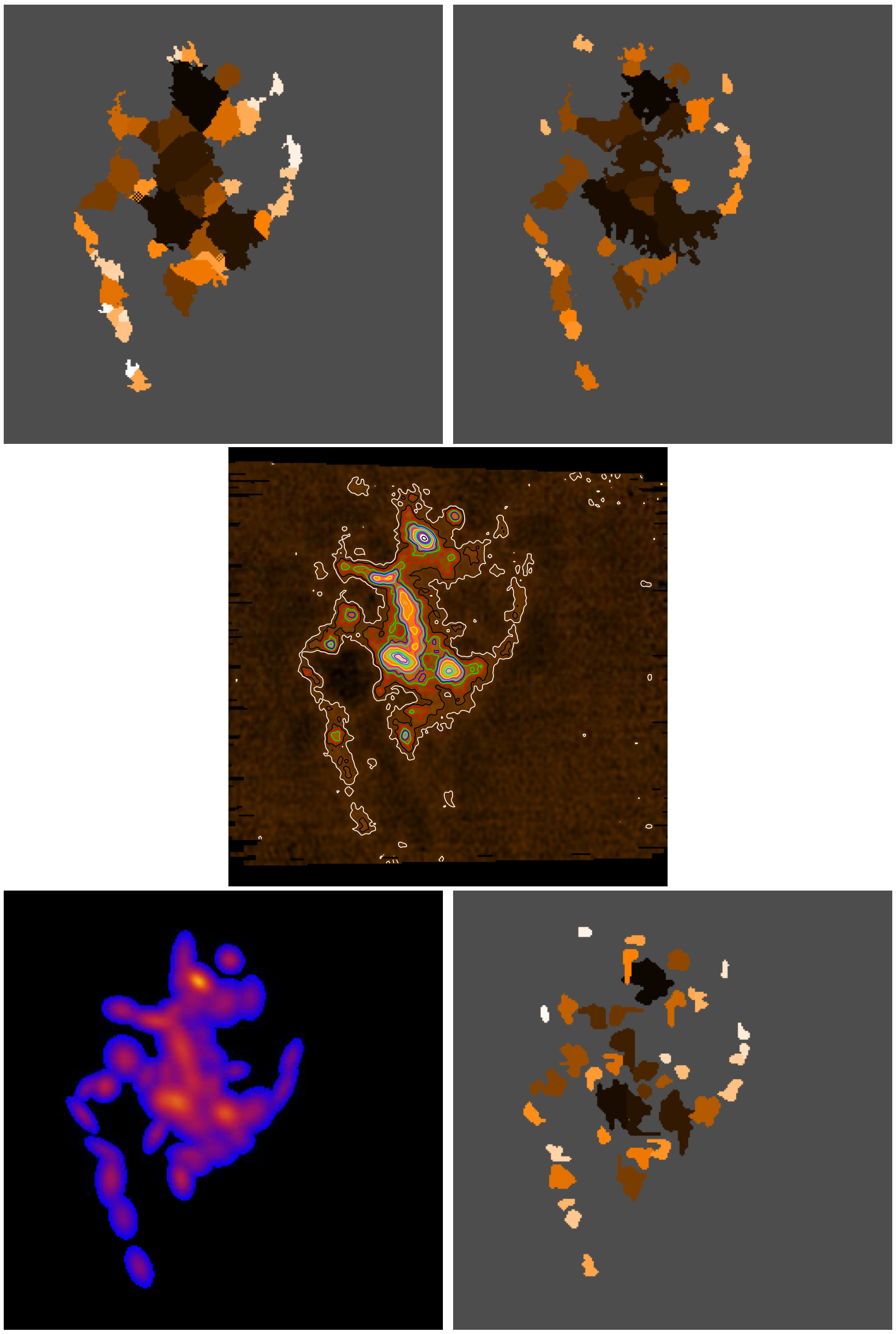


Figure 4.2: Top, from left; Output files from CUPID of the image 'g1015-034_scamps_lon_flat.sdf' for ClumpFind (left) and FellWalker (right) using the parameter values as mentioned in Table 4.1 and Table 4.2. Middle; g1015-034_scamps_lon_flat.sdf with contour levels starting from three times the RMS value with spacing between the contours of three times the RMS value. Bottom; Output files from CUPID of the image 'g1015-034_scamps_lon_flat.sdf' for GaussClumps (left) and Reinhold (right) using the parameter values as mentioned in Table 4.3 and Table 4.4.

4.2 The Comparison

With the parameters set so that each algorithm has a similar configuration it was now possible to directly compare each algorithm against its counterparts. Before the clump catalogues from each algorithm could be compared they needed to be matched together, this was achieved with Topcat (Taylor, 2005) using the peak positions of the data sets and pairing each peak with a corresponding peak at the same or nearby (within 5 arc-seconds) position. This process only matched corresponding peaks, therefore any additional peaks in one data set over another are ignored for part of this investigation. There are clearly different numbers of clumps detected by each method and ignoring these additional clumps would not give an accurate analysis of these algorithms, to this end a further investigation involving all the data was performed using histograms. From this, trends in the number of clumps can be seen.

4.2.1 Peak Value

ClumpFind, FellWalker, and Reinhold give similar peak values when the data are matched, as seen in Figures 4.3 and 4.4. This would be expected since they should all determine the peak position to be the same pixel and therefore its corresponding value ought to be the same. GaussClumps on the other hand gave lower peak values than the other methods, this becomes more noticeable at higher values and is more evident in the 450μm over the 850μm data. It is possible GaussClumps is automatically subtracting the background level from the clump and that is why its values are consistently lower than the other algorithms. Another possible explanation could be due to the fitting of the Gaussian curve causing the peak of the Gaussian to be a pixel near the peak of the clump and not the peak pixel itself. Though it is not known whether this is a more accurate determination of the peak of the clump.

When looking at the histograms in Figure 4.5 it is clear that ClumpFind and GaussClumps find more clumps than the other methods in the 450μm data, these additional clumps are found to have low peak values. As the peak value of the clumps increases the four algorithms start to match with each other and are likely detecting the same clumps at these values. The 850μm is slightly different with all four algorithms giving the same trend, a few high value peaks, some medium value peaks, and many low value peaks. The only difference is the number of clumps at these levels with GaussClumps giving consistently lower clump numbers than the other methods. The additional clumps may be due to the process of breaking up the clump, i.e. having satellite clumps around a main clump. These satellite clumps would have much lower peak values as they are located around the edge of larger clumps, therefore giving a much higher number of clumps at a lower peak value. See Figure 4.5

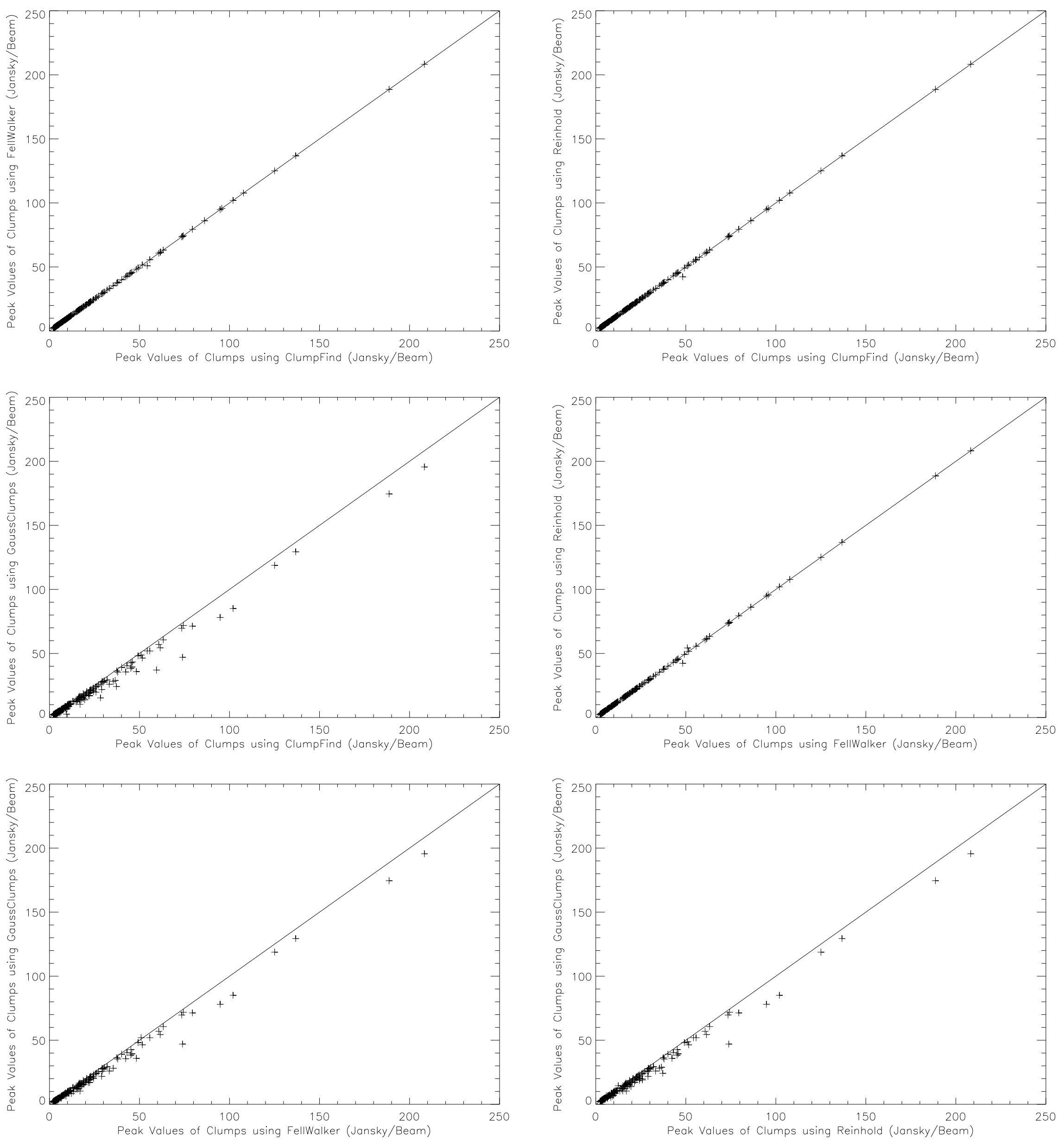


Figure 4.3: Topcat matchings for the peak value of the clumps for each of the algorithms using the 450μm data, Top from left; ClumpFind matched with FellWalker, ClumpFind matched with Reinhold. Middle from left; ClumpFind matched with GaussClumps, FellWalker matched with Reinhold. Bottom from left; FellWalker matched with GaussClumps, Reinhold matched with GaussClumps. The line in each plot follows the function of y=x.

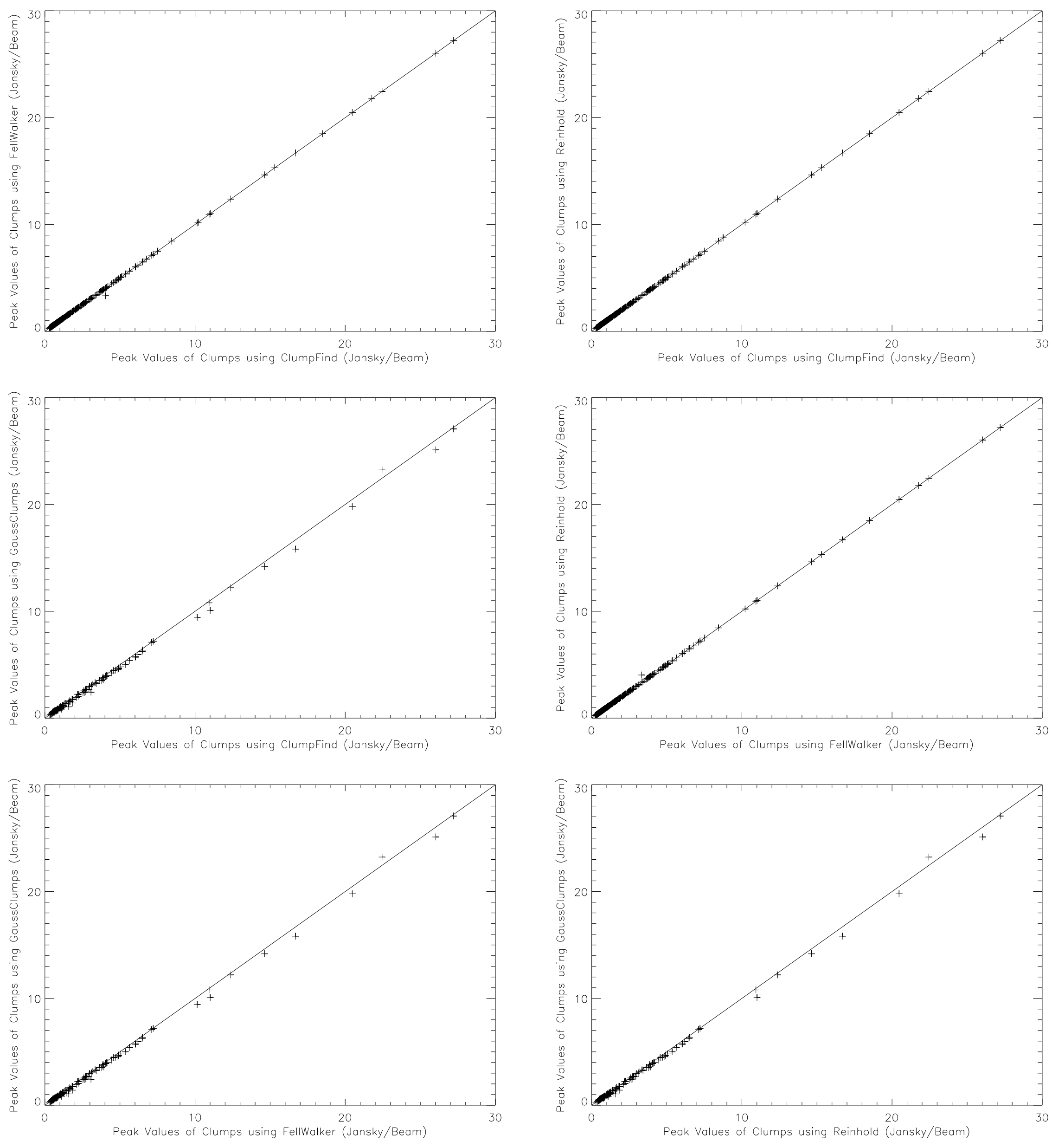


Figure 4.4: Topcat matchings for the peak value of the clumps for each of the algorithms using the 850μm data, Top from left; ClumpFind matched with FellWalker, ClumpFind matched with Reinhold. Middle from left; ClumpFind matched with GaussClumps, FellWalker matched with Reinhold. Bottom from left; FellWalker matched with GaussClumps, Reinhold matched with GaussClumps. The line in each plot follows the function of y=x.

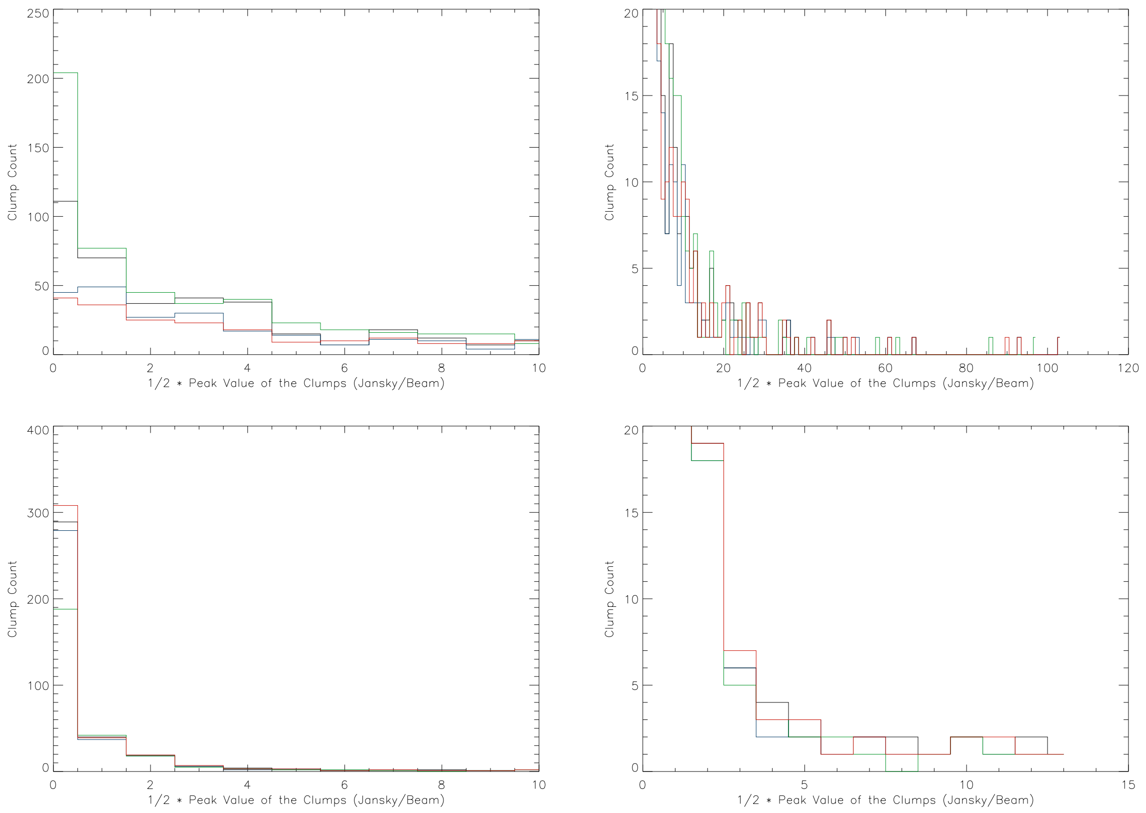


Figure 4.5: Histograms showing the peak values for each of the different algorithms; ClumpFind(black line), FellWalker(blue line), GaussClumps(green line), and Reinhold(red line), the left histograms being a magnified view of the low peak value clumps, the right ones showing the full range of peak values but limited clump count. Top; 450μm data, bottom; 850μm data.

4.2.2 Size of The Clump

The size of the clump is the total area of the clump within the boundary edges determined by the clump-finding algorithms. This is the total number of pixels contained within the clump. Since SCAMPS images have the axis is arc-seconds, where each pixel is three arc-seconds in both axis, the area is output in square arc-seconds. Each algorithm gives the same units for the size of the clump but since the algorithms work by different means the boundary edges of the clumps are different for each algorithm.

GaussClumps often gives a result of larger clumps than the other algorithms, as demonstrated in Figures 4.6 and 4.7. This is likely to be due to it using a completely different process for clumpfinding; each pixel in the image can belong to multiple clumps therefore the size of a clump is not limited by a neighbouring clump. Reinhold on the other hand returns consistently smaller clumps, this is probably due to the cleaning processes it performs subsequent to the clumpfinding process. The shape and size of the clumps detected by the Reinhold algorithm are often unusual, especially when compared to the other methods. ClumpFind and FellWalker give mostly similar results, with the FellWalker clumps sometimes being larger than the ClumpFind ones. ClumpFind and FellWalker do have a number of larger clumps than GaussClumps, this result is more evident at generally greater clump sizes.

The histograms in Figure 4.8 show there is a large variation in the number of different sized clumps detected. Reinhold has a large number of small clumps, ClumpFind and GaussClumps find many small clumps but also some medium sized clumps, FellWalker finding many small, some medium sized, and a few large clumps. This pattern is true of both the 450μm and 850μm data. See Figure 4.8.

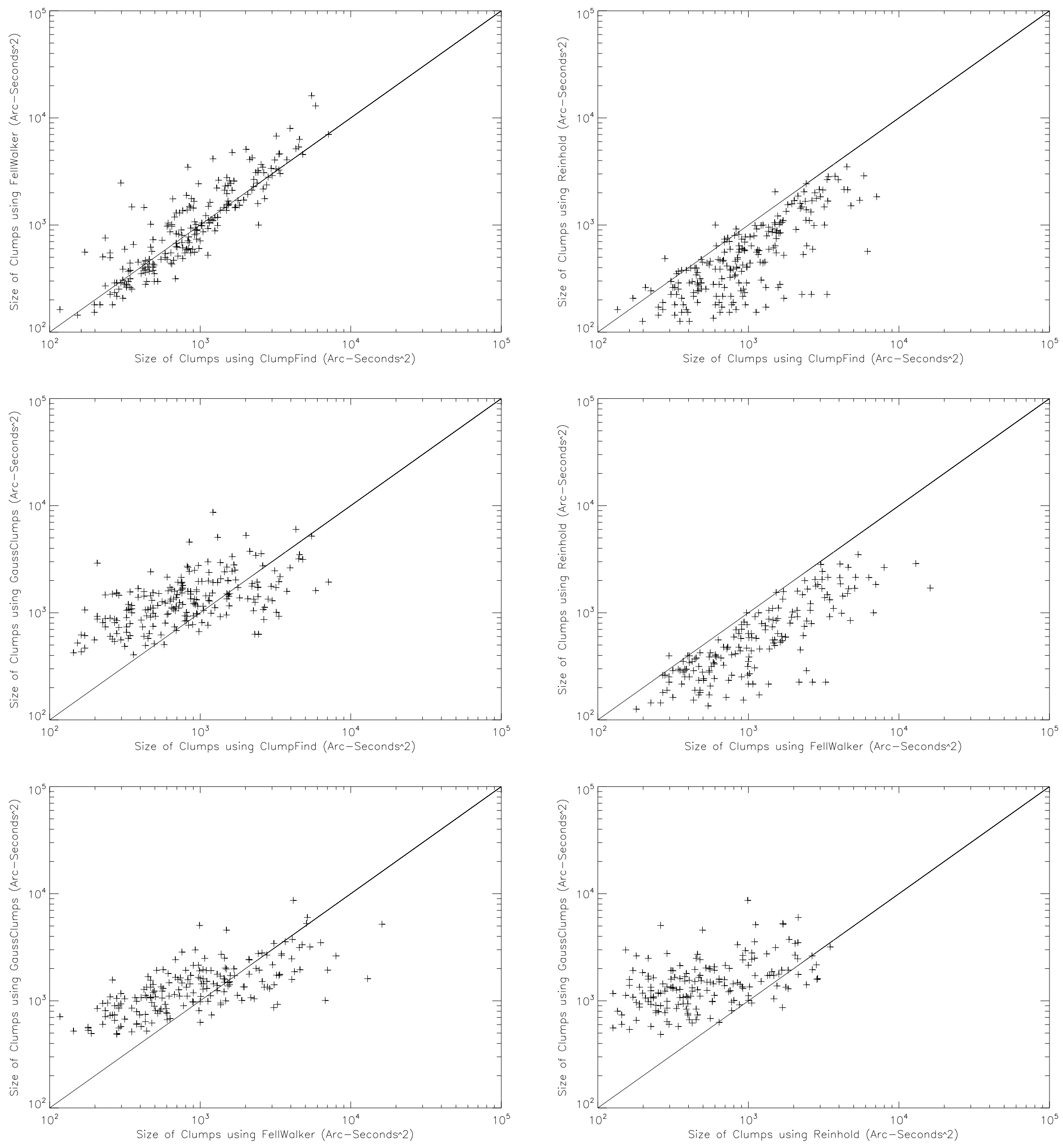


Figure 4.6: Topcat matchings for the size of the clumps for each of the algorithms using the 450μm data, Top from left; ClumpFind matched with FellWalker, ClumpFind matched with Reinhold. Middle from left; ClumpFind matched with GaussClumps, FellWalker matched with Reinhold. Bottom from left; FellWalker matched with GaussClumps, Reinhold matched with GaussClumps. The line in each plot follows the function of y=x.

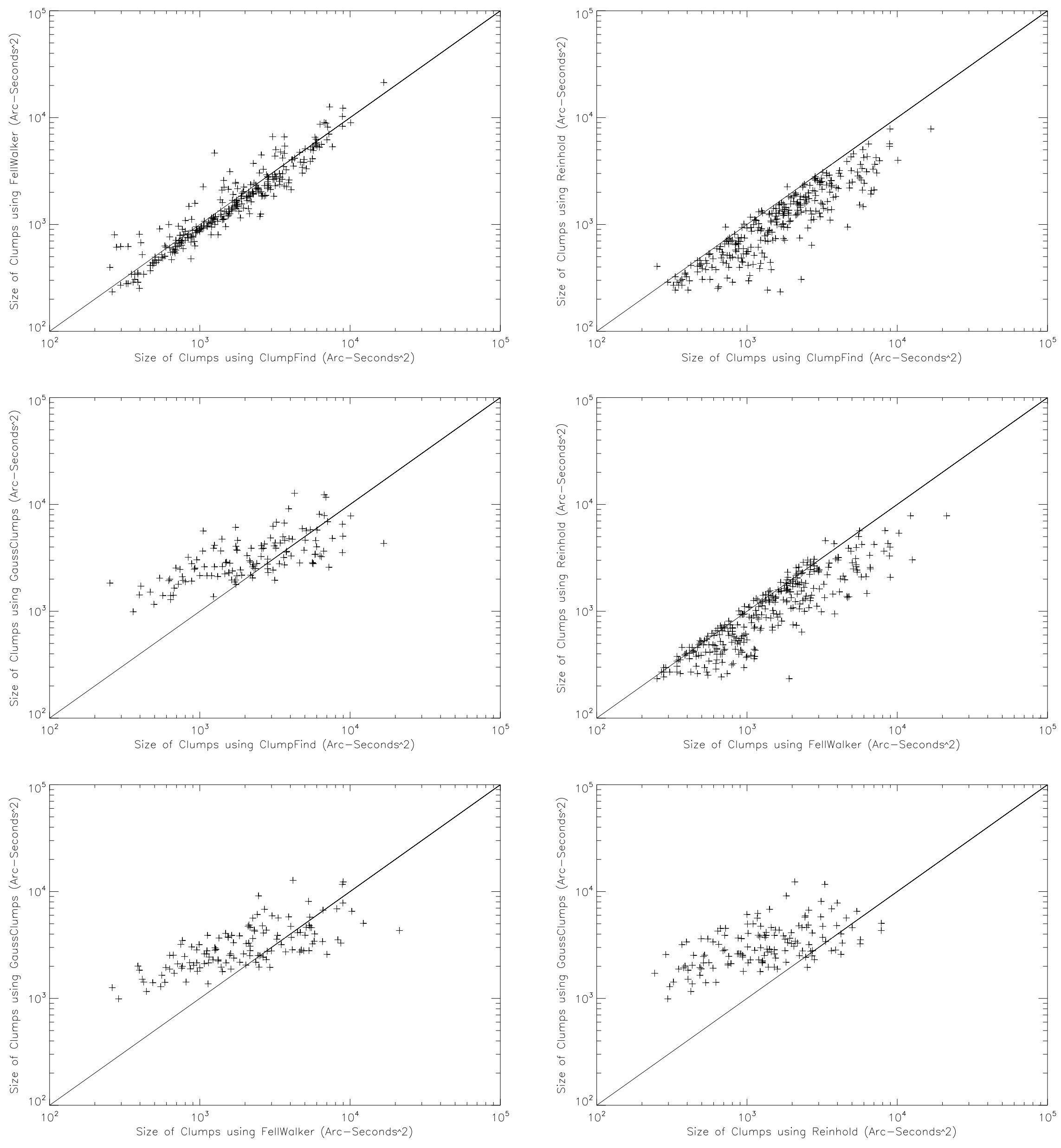


Figure 4.7: Topcat matchings for the size of the clumps for each of the algorithms using the 850μm data, Top from left; ClumpFind matched with FellWalker, ClumpFind matched with Reinhold. Middle from left; ClumpFind matched with GaussClumps, FellWalker matched with Reinhold. Bottom from left; FellWalker matched with GaussClumps, Reinhold matched with GaussClumps. The line in each plot follows the function of y=x.

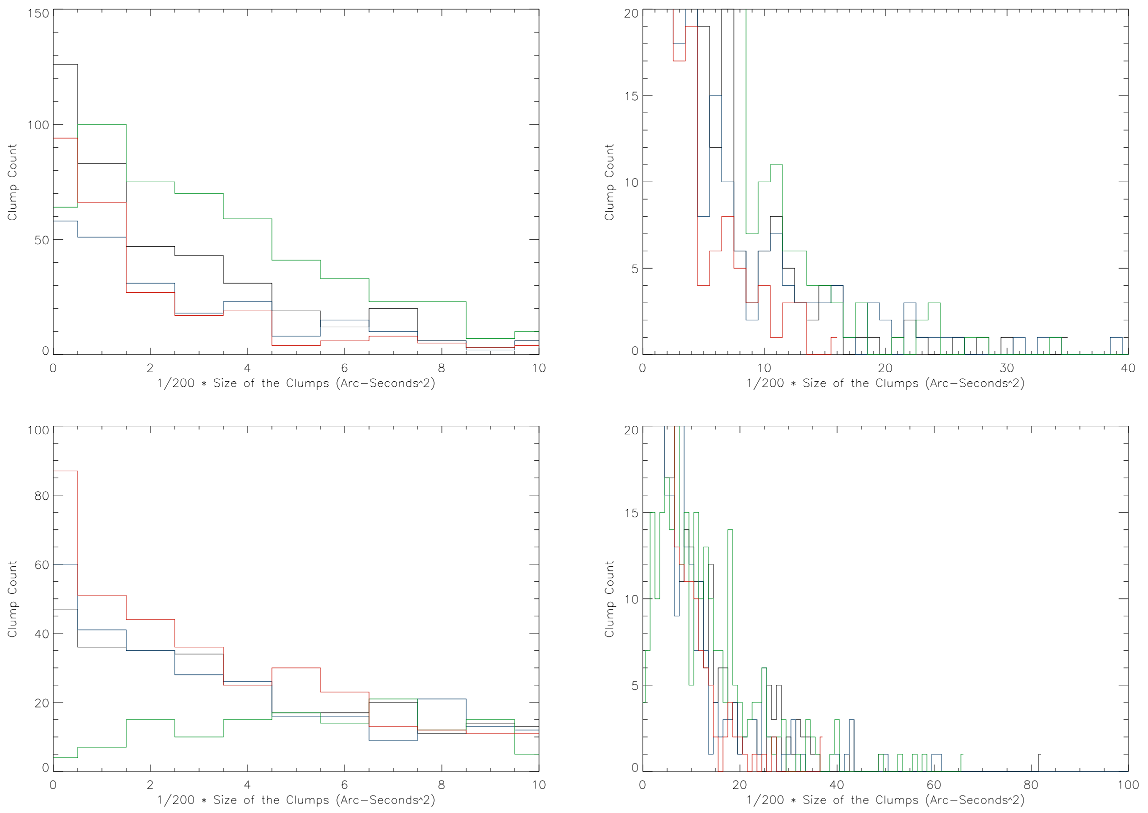


Figure 4.8: Histograms showing the clump sizes for each of the different algorithms; ClumpFind(black line), FellWalker(blue line), GaussClumps(green line), and Reinhold(red line), the left histograms being a magnified view of the small clumps, the right ones showing the full range of sizes but limited clump count. Top; 450μm data, bottom; 850μm data.

4.2.3 Integrated Flux Value

The integrated flux of a clump depends on the size of the clump and the value assigned to each pixel, therefore a large clump with a high peak is likely to have a high integrated flux value. If an algorithm gives consistently low or high values for either or both of the size and peak values then the flux values would also be affected. Since ClumpFind and FellWalker give broadly similar peak values and sizes of clumps they also give similar values for the flux of the clumps. For some clumps FellWalker gives slightly higher clump sizes than ClumpFind and therefore often reports higher clump flux values (See Figures 4.9 and 4.10). Due to FellWalker determining larger clumps it therefore gives the highest clump integrated flux values out of the four algorithms. The clumps detected by GaussClumps were often larger than FellWalker, however as GaussClumps consistently detected lower peak values its flux values are regularly lower than both ClumpFind and FellWalker. Reinhold gave similar peak values to ClumpFind and FellWalker but consistently smaller clump sizes compared to the other three algorithms, therefore its flux values are consistently lower and regularly the lowest out of the four algorithms (see Figure 4.8).

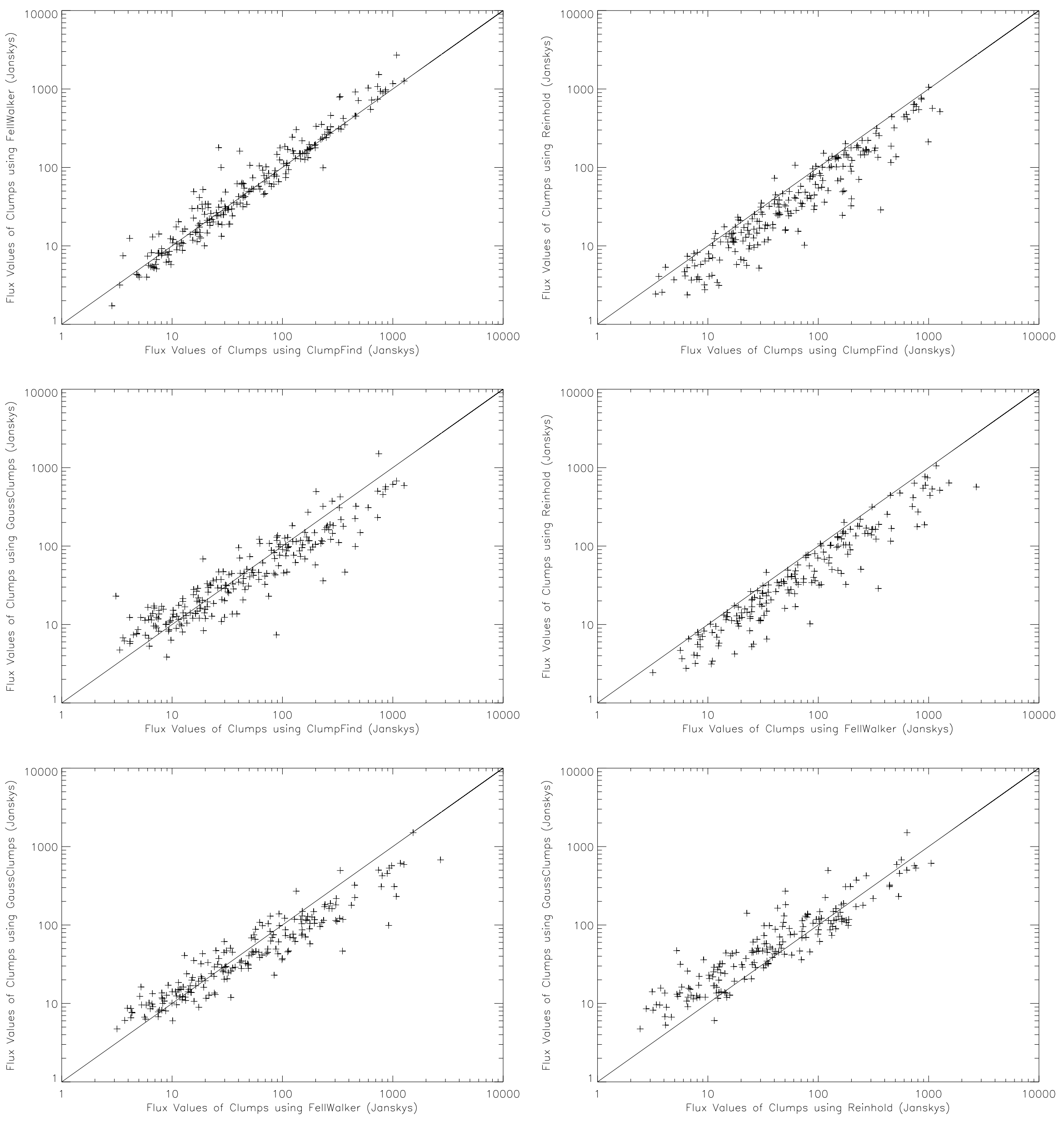


Figure 4.9: Topcat matchings for the flux value of the clumps for each of the algorithms using the 450μm data, Top from left; ClumpFind matched with FellWalker, ClumpFind matched with Reinhold. Middle from left; ClumpFind matched with GaussClumps, FellWalker matched with Reinhold. Bottom from left; FellWalker matched with GaussClumps, Reinhold matched with GaussClumps. The line in each plot follows the function of y=x.

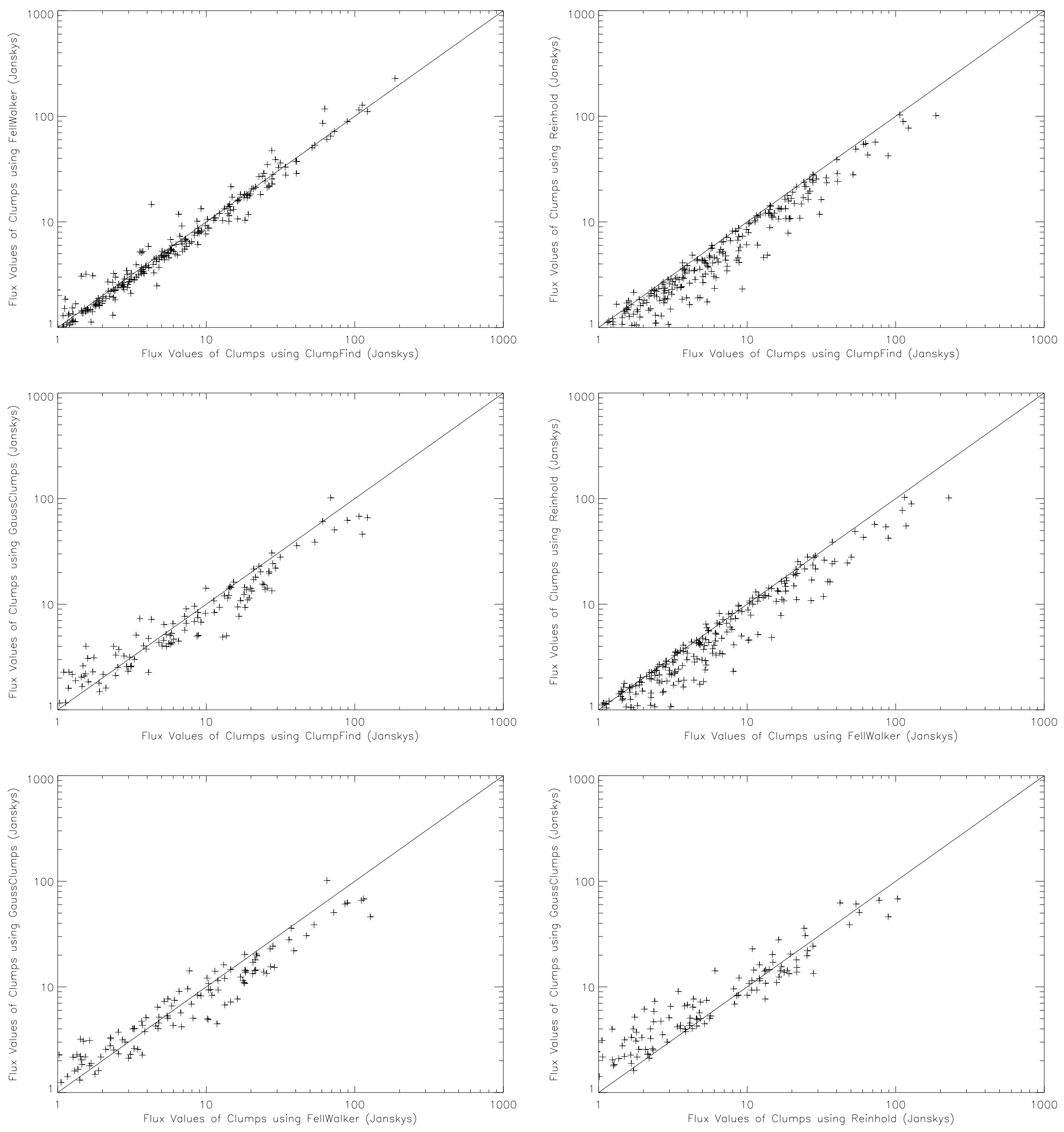


Figure 4.10: Topcat matchings for the flux value of the clumps for each of the algorithms using the 850μm data, Top from left; ClumpFind matched with FellWalker, ClumpFind matched with Reinhold. Middle from left; ClumpFind matched with GaussClumps, FellWalker matched with Reinhold. Bottom from left; FellWalker matched with GaussClumps, Reinhold matched with GaussClumps. The line in each plot follows the function of y=x.

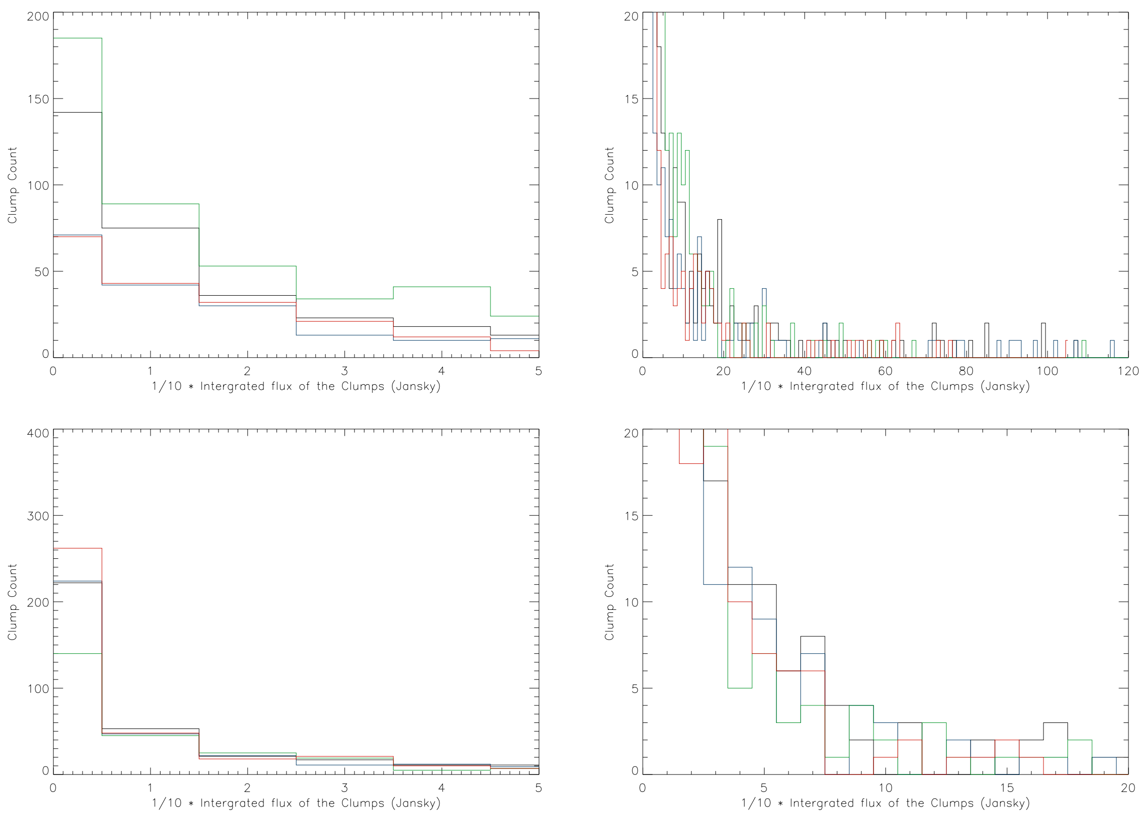


Figure 4.11: Histograms showing the flux values for each of the different algorithms; ClumpFind(black line), FellWalker(blue line), GaussClumps(green line), and Reinhold(red line), the left histograms being a magnified view of the low integrated flux clumps, the right ones showing the full range of fluxes but limited clump count. Top; 450μm data, bottom; 850μm data.

4.3 Conclusion

Since the Topcat matchings are only able to pair clumps with similar peak positions a number of clumps are ignored. If an algorithm is breaking large clumps into multiple smaller clumps or the reverse where many small clumps are merged into a single large clump then the size and therefore flux for that clump would vary between the different algorithms. This could be seen in the plots where the size and flux of clumps are consistently larger or smaller for one algorithm than another.

ClumpFind often detects a larger number of clumps than the other algorithms. If ClumpFind is detecting more clumps than are present then it is due to the process breaking up a large clump into a smaller clump with a number of satellite clumps surrounding it. This effect would cause a lower flux value for the real clumps, thus producing not only an incorrect number of clumps but also incorrect sizes and fluxes for the detected clumps.

FellWalker detects a lower number of clumps than ClumpFind. The peak pixel matches with most of the other methods. FellWalker does appear to detect larger clumps than the other methods on a number of occasions, this could be due to it simply finding fewer clumps over the same area and therefore not breaking up the main clump. This effect can be seen in Figures 4.1 and 4.2 where by looking at the contour image and comparing with the CUPID output images where ClumpFind detects many more clumps over the same area as FellWalker, yet FellWalker appears to match with the contoured image. The larger clump sizes means that FellWalker also gives greater flux values.

GaussClumps detects a larger number of clumps in the 450μm data than the 850μm data. The peak value is often lower than the other algorithms and also the location of the peak pixel varies compared with the other algorithms. The size of the clumps seems to be restricted to a range between 500 and 5000 Arc-Seconds^2 (other algorithms having a greater range between 100 and 10000 Arc-Seconds^2) with very few small or large clumps. This may be caused by the fact that GaussClumps can detect small overlapping clumps in a crowded area. Therefore large clumps do not dominate over smaller clumps as in the other algorithms which require pixels to be assigned to individual clumps. This can be seen in Figures 4.1 and 4.2 where the size of each ellipse is similar to the others. Since a pixel can belong to multiple clumps it is difficult to examine the profile of the clumps and as such we can not determine their exact shape or size visually. It is possible for the output image file to contain only ellipses to provide a visual for the location of the clumps and how they overlap but this was not possible at the time of this investigation, and our analysis for GaussClumps is restricted to the data matching and the histograms.

Reinhold detects a similar number of clumps as FellWalker with similar peak values and locations as ClumpFind and FellWalker but the size and shape of the clumps can be highly erratic. The size is generally smaller than the other algorithms but the shape of the clump is completely unique and often irregular, which causes the flux values to also be erratic and smaller than the other algorithms.

The four algorithms are able to determine similar locations for the matched clumps. There is little variation between the algorithms for locating a source in an uncrowded image, though as the structure becomes more complex additional clumps may be detected. ClumpFind detects a greater number of clumps than the other algorithms which could be due to ClumpFind being able to determine the finer structure within the image and detect more embedded sources, although these sources could be noise spikes. This can be avoided by having a higher DeltaT value, though doing so causes the fine structure in the image to be overlooked. Therefore to have both accurate results for the main clump and also to observe the fine structure the algorithm would need to be run multiple times with different parameter values as done by Rathborne et al. (2009). FellWalker contains a parameter (MinDip) that helps reduce the effect of noise on the larger clumps while still allowing the finer detail of the smaller clumps to be detected.

Where each of the algorithms results differ is in the clump size and therefore its total flux, with ClumpFind, FellWalker, and GaussClumps all giving similar clump sizes to each other when compared with Reinhold. This is true not only for the output value of the clump size but also the appearance of the clump in the clump index images. ClumpFind clearly detects more clumps than the other methods, it is not possible to determine if it is detecting more clumps than are actually there or if the algorithm is able to detect clumps that the other algorithms are unable to do so. This increased clump detection is also true of GaussClumps yet to a lesser extent. The sizes of the Reinhold clumps are regularly erratic with a highly suspect shape in certain instances.

It is not possible to know from the SCAMPS images the true and exact location, size, and integrated flux of each clump. To test the reliability of the algorithms in finding ”true” sources they must be run using known clump sources that are artificially generated. With artificial clumps we are able to know the exact peak location, size, and integrated flux value for each clump. Therefore we can directly compare the results given by the algorithms with the known true values. This comparison was undertaken using Monte-Carlo simulations and the results are described in Chapter 5.

Chapter 5

Tests Using Artificial Clumps

5.1 Overview

For this investigation we created artificial Gaussian sources and combined them with a SCAMPS image. With the previous investigation into the parameters in §3 there was no certainty as to the true number of clumps in an image or their properties. Knowing the details of these input sources we can then compare the results from each of the algorithms and determine how correctly and accurately the algorithm interpreted the data, thereby giving us an indication of possible faults or biases the algorithms have. We also wanted to know the completeness of the algorithms; what clump profiles (size and peak value) would the algorithms either be unable to detect at all, occasionally detect, or consistently detect. This chapter details how the investigation was implemented and the results obtained.

5.2 Implementation

5.2.1 The Image

The investigation used one of the SCAMPS data files to insert the artificial Gaussian source into, instead of inserting an artificial source into a blank field with artificial Gaussian noise. The problem with using a blank field with Gaussian noise is that the noise in SCUBA images is not Gaussian and is likely to be correlated between pixels (**?**). Using Gaussian noise would thus not give an accurate indication as to how the algorithms respond to real data and to do so would require the artificial source to be inserted into a real data image. One particular image file needed to be chosen to insert the artificial clump onto, it was deemed best to use an uncrowded image to prevent any confusion, preferably with only a single clear source being present in the centre of the image. This was the desired situation as, with a simpler image it would be possible to determine how well the algorithms could simply detect a single source and obtain its properties. With a crowded image there comes the possibility of blended emission and as such it becomes more difficult to determine the accuracy of the results. Once the initial investigation had yielded its results, and a suitable evaluation of the algorithms had been made it would then be the logical next step to use a more crowded image to investigate the effects caused by clump confusion. The image chosen for this initial investigation was g1084-259 of the SCAMPS survey, see Figure 5.1

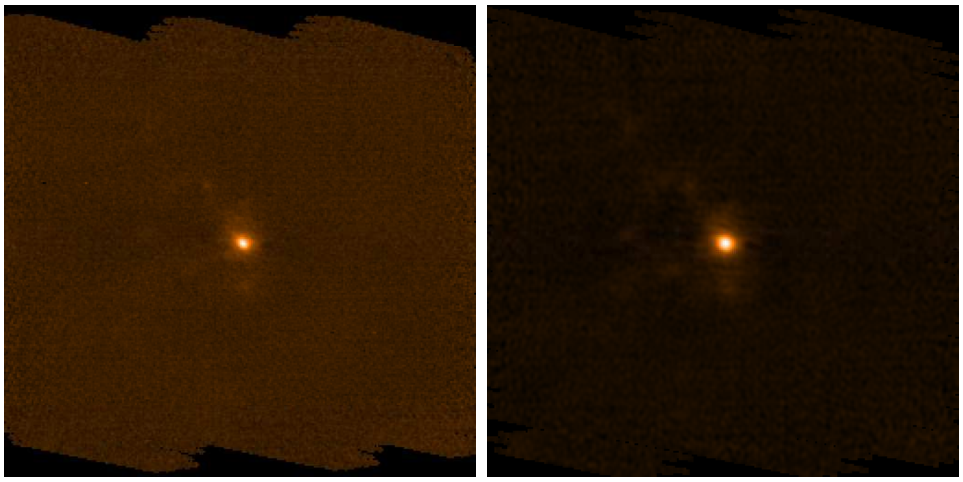


Figure 5.1: From left; g1084-259_SCAMPS_sho_flat.sdf, g1084-259_SCAMPS_lon_flat.sdf.

5.2.2 The Artificial Clump

For this investigation IDL was principally used in the creation of the artificial sources and the insertion of them into the image, see Appendix B for the full code of the program used. The clumpfinding process was then performed on this image by CUPID using the same scripts as in §3.

To create the artificial clump in IDL the data had to firstly be read in, this was done using the readfits command[1]. Readfits requires the '.fits' extension yet CUPID uses NDF format so the files were converted to NDFs after the artificial source was inserted. With the file now read in, the sky function[2] was performed to determine the background noise value for the image, this gave the RMS value which can then be used later in the program. The RMS value given by the sky function was similar to that calculated by hand using GAIA and as such it was preferable to the value given by CUPID, which is generally much lower (see table 5.1). With all these processes completed the file was now ready for the artificial clump to be inserted.

Table 5.1: RMS values of the image used for the insertion of the artificial clump.

Image	SKY RMS	GAIA RMS	CUPID RMS
g1084-259_SCAMPS_sho_flat.sdf	0.350963	0.3000043	0.25309130
g1084-259_SCAMPS_lon_flat.sdf	0.0511509	0.04596421	0.01540634

[1]Readfits function is used by IDL to read in a fits file so further analysis can be performed upon it. Details on this function can be found at http://idlastro.gsfc.nasa.gov/ftp/pro/fits/readfits.pro (accessed October 2008).

[2]The sky function is used by IDL to determine the sky level in an image, this gives the value for the RMS parameter. Details on this function can be found at http://idlastro.gsfc.nasa.gov/ftp/pro/idlphot/sky.pro (accessed October 2008).

The artificial clump was created as an array the same size as the original image array, the two arrays were then combined creating a single array with the data from both. The artificial clump array had a value of zero for all data points except for the position of the clump, therefore the other points on the original data array were not affected by the addition of the artificial clump array. To create this artificial clump array we used the IDL function psf_Gaussian[3], which created an artificial clump with a Gaussian flux profile. The central position of the artificial clump within the image was controlled by the function RandomU[4], see §5.3.4. The Gaussian also required a value for its FWHM (Full Width at Half Maximum) this value was again created at random using RandomU. From early testing we were able to determine that there was a minimum value of the FWHM for detection by the algorithms but also after a greater value, the artificial clump became unrealistically large (see §5.3.2). This is due to the scan-map process see §1.6, as the largest chop size in the SCAMPS images was 65 arcseconds any clump larger than this is unlikely to be imaged well by the data reduction process, this equates to a clump size of 22 pixels in any one direction. The value for FWHM was chosen to be between six and ten pixels and is equivalent to a FWHM range between 18 and 30 arcseconds. We now had an array the same size as the original image with a Gaussian source randomly positioned excluding certain areas and with a random size between predetermined values. Gaussians created by psf_Gaussian are initially normalised to a peak value of one, this needed to be changed, firstly the array was multiplied by the RMS value determined by the sky function, this then gave a peak equal to the RMS value of the image, the array was again multiplied to give a peak value that was a multiple of the RMS value. The value by which it was multiplied depended on the total flux required and the size of the clump. The program performed the artificial source process a number of times, each time a new copy of the original data array was used.

Once the artificial source had been created the images were converted from fits format to NDF format using the Starlink package fits2ndf[5]. With the images now in NDF format it was possible to remove the original sources in the images so that they were not considered by the clumpfinding process, this was done using the KAPPA[6] command, ardmask[7]. With the images ready for the clumpfinding process CUPID was run on each of the artificial source images. Each algorithm was used in turn using the parameters described in §4, (tables 4.1 -

[3]Psf_Gaussian is an IDL function, it's purpose is to create an array with a Gaussian profile within it, the array and Gaussian can be of any size and the Gaussian can be positioned anywhere within the array. Full details on this function can be found at http://idlastro.gsfc.nasa.gov/ftp/pro/image/psf_Gaussian.pro (accessed October 2008).

[4]RandomU is an IDL function used to create a random number between zero and one, full details on this function can be found at http://idlastro.gsfc.nasa.gov/idl_html_help/RANDOMU.html (accessed October 2008).

[5]Fits2ndf is a Starlink program for converting a .fits format image into .sdf format, full details on this program is available at, http://starlink.jach.hawaii.edu/docs/sun55.htx/node19.html (accessed October 2008).

[6]KAPPA is a Starlink package designed for processing images of the Starlink standard format (NDF). The KAPPA users guide can be found at http://starlink.jach.hawaii.edu/docs/sun95.htx/sun95.html (version 1.10, created on 9th July 2009).

[7]Ardmask masks out an area of an image and replaces the pixels in that area with bad pixels, the area designated for masking is specified in a separate file. Full detail on ardmask can be found at http://starlink.jach.hawaii.edu/docs/sun95.htx/node225.html (accessed February 2009).

4.4).

5.3 Artificial Clump Properties

We wanted to see which sources the algorithms detected and if they detected them correctly. To help facilitate this, certain aspects needed to be investigated with initial test simulations. These simulations allowed me to decide on the appropriate ranges for the FWHM and peak values, along with determining the area on the image that produced the least amount of noise that could interfere with the main investigation. The initial simulations involved the insertion of a single source into the image, that source having a known FWHM and peak value but the location of the centre of the source being random. The FWHM value ranged between one and eight pixels and the peak value ranged between one and ten times the RMS noise value of the image. The process was repeated 100 times for each combination of FWHM and peak value, giving a range of results, forming a grid of FWHM and peak values and their respective number of artificial clump detections (shown in Figures 5.2 - 5.5).

5.3.1 Algorithm Completeness

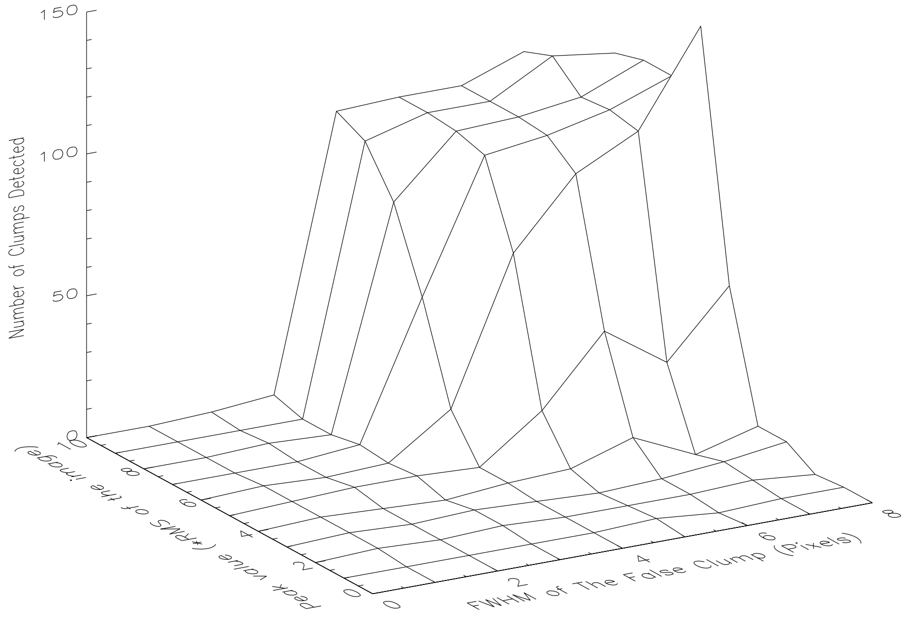
From Figures 5.2 - 5.5 we can see how many clumps each algorithm detected for different peak values and FWHM values. In these figures 100 artificial source images were used, again with g1084-259_SCAMPS_sho_flat.sdf and g1084-259_SCAMPS_lon_flat.sdf (the clump count axis in the plots were adjusted to remove the number of clumps that were found in the image without an inserted artificial source). The peak values ranged between one and ten times the RMS value of the image, the FWHM ranged between one and eight pixels. The four algorithms display the expected S-shaped completeness trend (Coppin et al., 2006); small, low peak clumps are not detected, the detections increase as the size and peak value of the clumps increase, then this increase levels off as the maximum detections possible were obtained. GaussClumps, Figure 5.4, on the other hand not only detected the artificial clump as it should have, but it also detected an additional clump in the original 450μm image. GaussClumps did not detect this additional clump all the time. It was detected when the peak value and FWHM of the artificial clump were at high values, although the artificial clump was not located near the additional clump such that its profile would be effected making it more noticeable. The additional clump was detected at nearly all peak values, providing the FWHM size was equal to or greater than four, and also detected at high peak values when the FWHM was two pixels. It is unclear at this stage as to why this extra clump was detected. Additionally there was a non smooth gradient for the detection count with GaussClumps, which is discussed below. The 850μm image using GaussClumps did not locate an additional clump but the plot shows results that indicate a much greater amount of variation in the number of clumps detected for particular sizes and fluxes than with the other algorithms

From the figures the general trend can be easily seen, for low peak and low FWHM values, high peak and low FWHM values, or low peak and high FWHM values there were no detections of the artificial clump. As each of the values increase the detection count remains at zero until a FWHM and RMS value of four or five, at which point there is a sharp increase in the detection count. When both values reach six there is at least 50% completeness (half of the number of inserted clumps were detected) for each of the algorithms, though this is more variable when using GaussClumps, particularly the 850μm image. 100% completeness is achieved soon after, at values of six for the FWHM of the clump and seven for the RMS value.

After 100% completeness was obtained there should be little or no further variability in the number of clumps detected. This was not the case with the exception of FellWalker. ClumpFind, GaussClumps, and Reinhold have completeness greater than 100%. I do not believe that the algorithms are detecting any additional clumps (except the one previously mentioned with GaussClumps) but instead when the clumps are sufficiently large enough the algorithm was influenced by noise, and hence breaks up the single artificial source into multiple sources. This does not occur for every artificial clump but a sufficient number to be noticeable and cannot be excused experimentally away as an occasional noise spike. This breaking up would not only cause incorrect results for the number of clumps but also the size and flux of the erroneously detected ones. In this particular situation the detected clumps would be smaller and have a lower flux. As mentioned FellWalker did not exceed the 100% completeness level but, there were occasionally fewer than 100% detections at levels that would normally be 100% complete. This investigation was performed without the boundary limits mentioned in §5.3.4, therefore the artificial clump may have been inserted anywhere on the image. This could have resulted in a clump touching the edge or being placed on top of emission already present in the image, making it indistinguishable from the original source. If the artificial clump was placed in one of those locations it would have not been detected by FellWalker. The other algorithms would have been unable to detect the artificial clump but with them breaking up many of the clumps it is difficult to determine if this really was the case.

5.3.2 Peak Value and Clump Size

From the initial simulations described above it was decided that further investigations of the algorithms would be set so that the minimum peak height of a clump was three times the RMS value of the image. Any artificial source with a peak value below this was unlikely to be detected due to the parameter values used. If that clump was positioned over an area of high background noise or a spike in the noise then the pixel values of the clump would increase. If a clump were to have a peak slightly higher than this minimum value then it is still doubtful whether the program would consider the artificial clump as viable since one of the parameters (MinPix) required a minimum clump size. Due to the Gaussian nature of the artificial clump not only did the peak had to have a value large enough so it was above the minimum value, but also a portion of the clump with an area equal or more than the minimum clump size had to be over the minimum peak value. Consequently only artificial clumps with a peak value of more than three times the RMS noise of the image would be detected as the algorithm



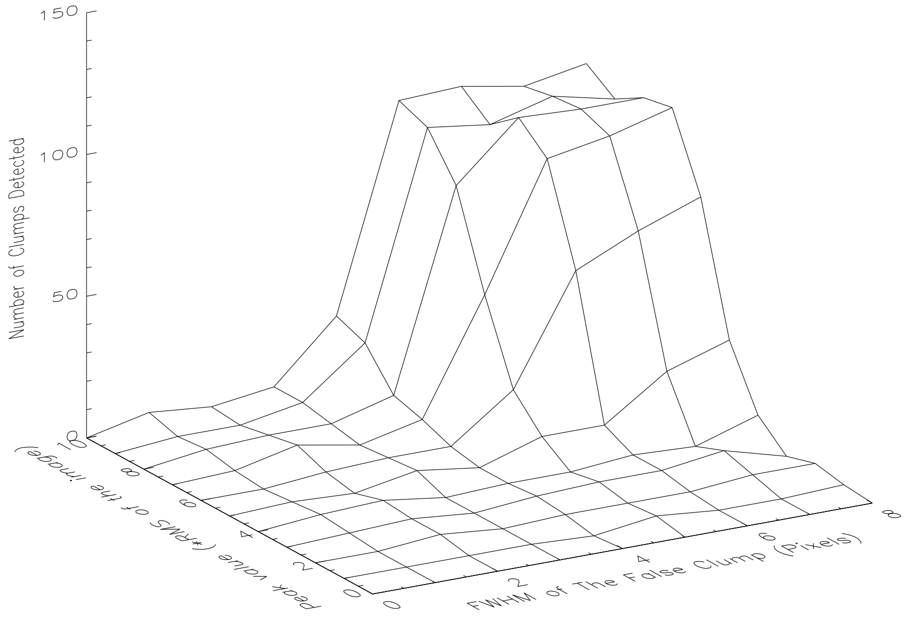
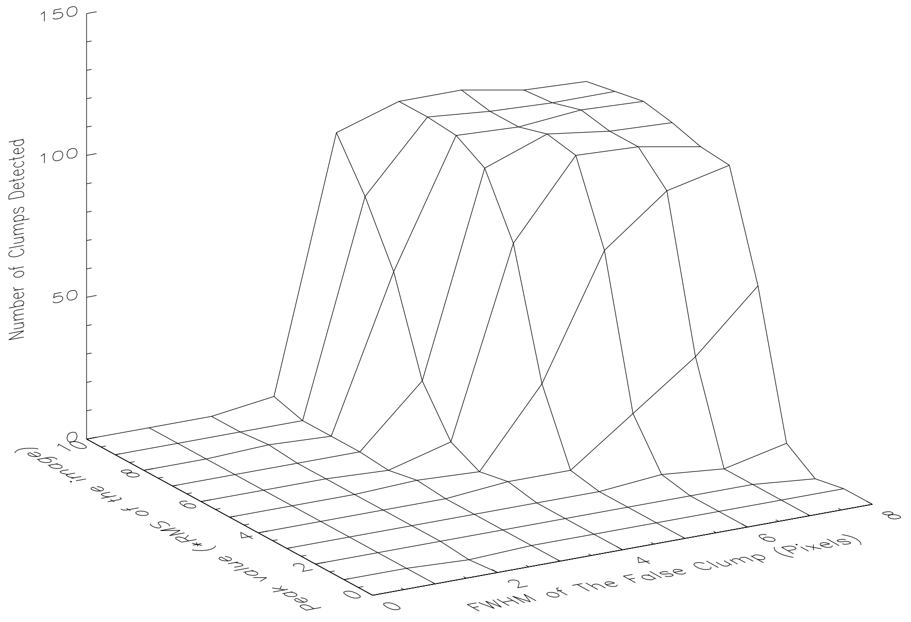


Figure 5.2: Three-dimensional plots to show how well ClumpFind detects artificial Gaussian sources at different peak values and FWHM sizes. Top; using a 450μm image as the original. Bottom; using an 850μm image as the original.



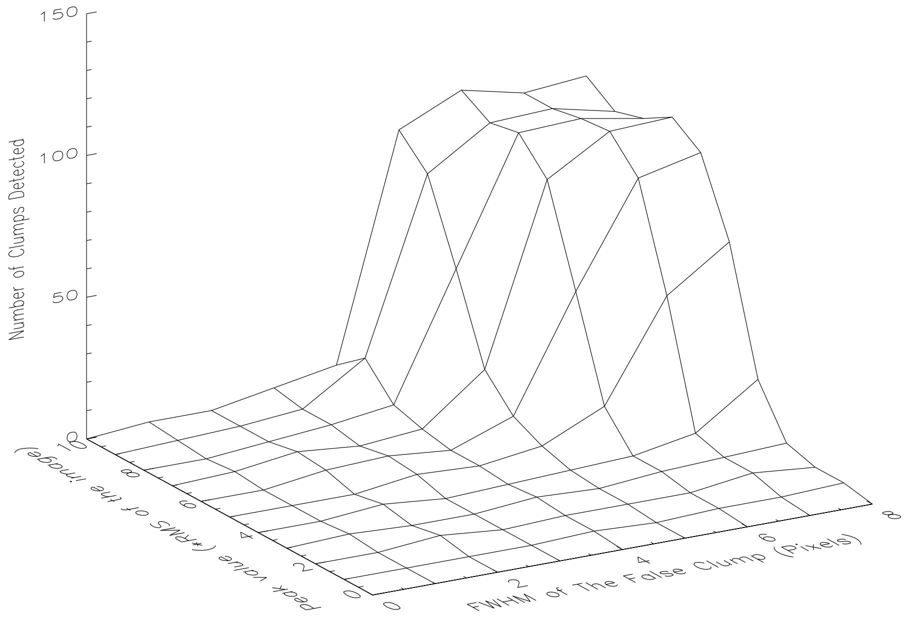
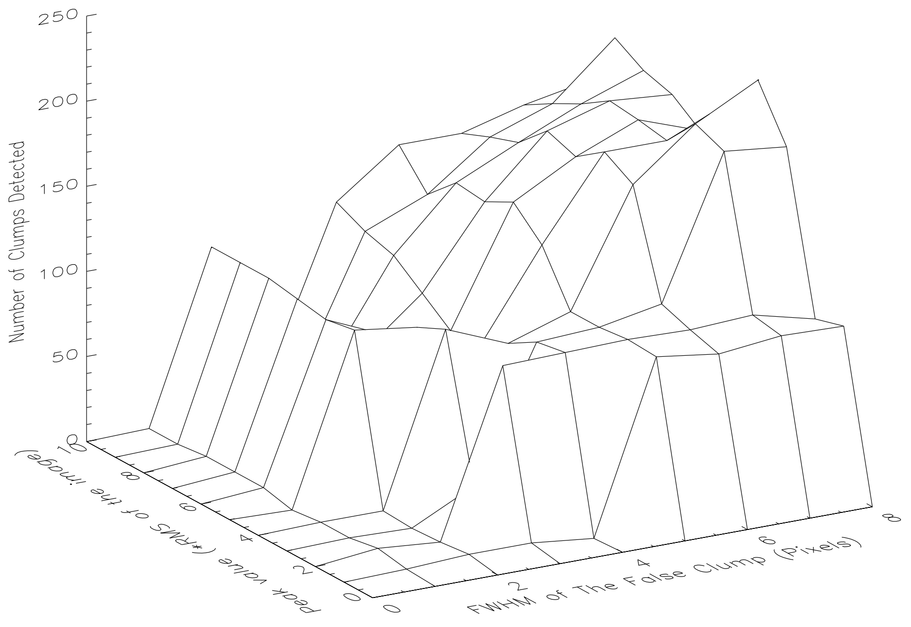


Figure 5.3: Three-dimensional plots to show how well FellWalker detects artificial Gaussian sources at different peak values and FWHM sizes. Top; using a 450μm image as the original. Bottom; using an 850μm image as the original.



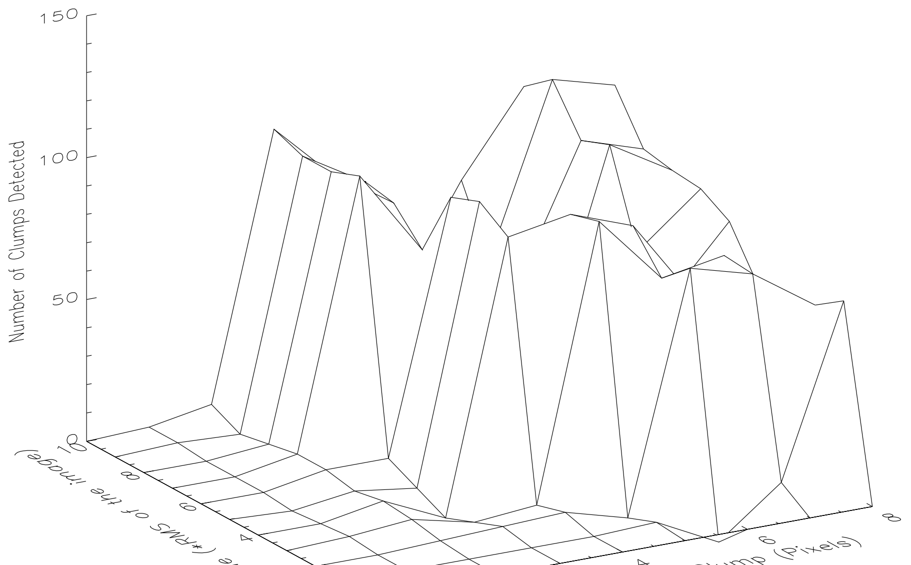
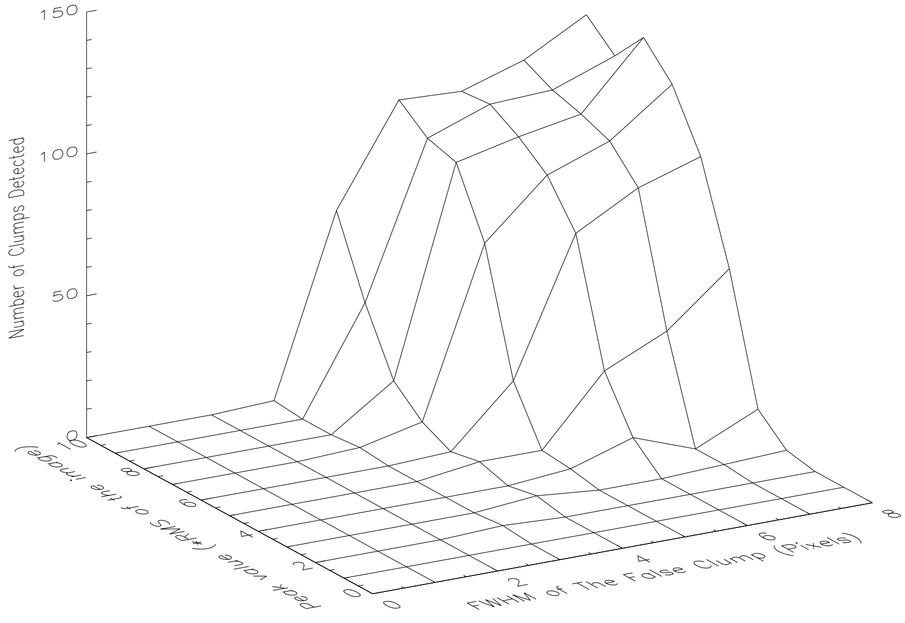


Figure 5.4: Three-dimensional plots to show how well GaussClumps detects artificial Gaussian sources at different peak values and FWHM sizes. Top; using a 450μm image as the original. Bottom; using an 850μm image as the original.



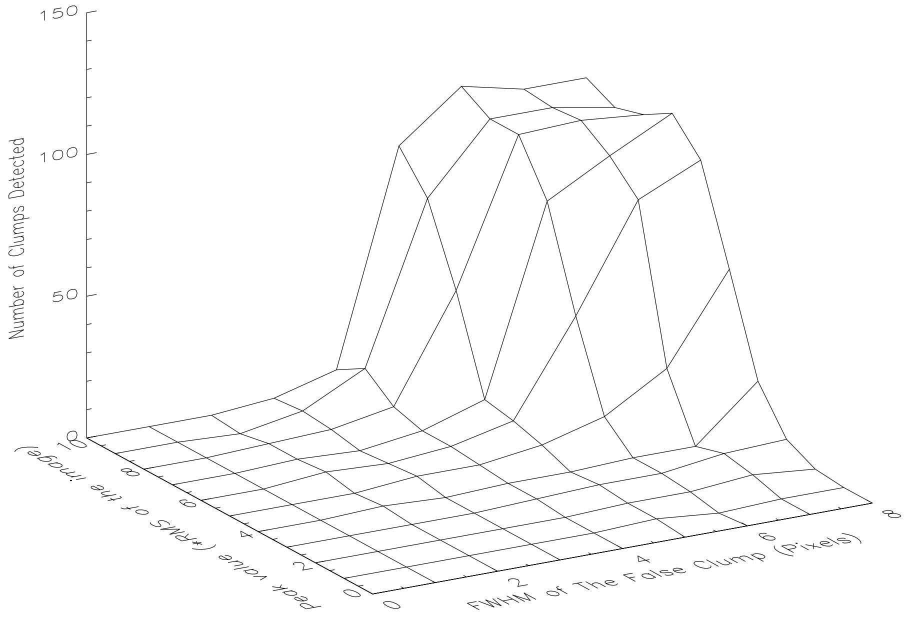


Figure 5.5: Three-dimensional plots to show how well Reinhold detects artificial Gaussian sources at different peak values and FWHM sizes. Top; using a 450μm image as the original. Bottom; using an 850μm image as the original.

parameters have this minimum value set at three times the RMS value.

There is a parameter that determines the minimum pixel size of the clumps (MinPix). This was set due to the beam size of the telescope used, as such only sources larger than this value are likely to be real, sources smaller than the telescope beam are more likely to be artefacts. The FWHM of the artificial clump determines the size of the clump produced. If FWHM causes a clump which at no point is larger than MinPix then the clump would not be detected, also if FWHM caused a clump to have a size larger than MinPix but only below the minimum peak value then the clump would also not be detected. Just as the clump size could be too small for detection it could also be too large to be realistic. A large flatter Gaussian would be more susceptible to noise spikes and misinterpretation by the algorithms as multiple clumps. Since MinPix was 9 for the 450μm data and 25 for the 850μm data the FWHM needed to have a minimum value of the square root of these values, therefore at least 3 pixels for the 450μm artificial clumps and 5 pixels for the 850μm artificial clumps.

The peak value and size of the clump are clearly linked, if a clump is too small it would never be detected no matter how high the peak value is and if the peak value is too small then the clump would never be detected no matter on its size, therefore for both parameters there must be a minimum value and in the case of FWHM also maximum value.

From testing different values of both of these parameters between one and eight pixels for FWHM and one and ten times the RMS of the image for the peak value, Figures 5.2 - 5.5 show that the peak value needs to be greater than three times the RMS value, and the FWHM value needs to be greater than the square root of the value of the MinPix parameter. Even at large clump sizes the artificial clump was not detected unless the peak value was at least four or five time the RMS value. With these peak values the detection may still only be a portion of the clumps, therefore the minimum peak value of the artificial clumps should be no less than six times the value of the RMS noise of the image to ensure a high probability of detection. At high peak values the clumps were not detected until a FWHM value of at least the square root of the MinPix value though these detections dropped as the peak value decreased. For further investigations we used a minimum peak value of six, therefore there needed to be a detection of the majority of the clumps at this value. This was only achieved once the FWHM value was at least five pixels for the 450μm data and six pixels for the 850μm data. To try to keep the artificial clumps similar for both data sets the minimum value was the same for both, as such a minimum FWHM value was six pixels. After examination of the images once the artificial clump had been inserted, and the clump detection process had been run we determined that a clump with a FWHM value of ten pixels should be the maximum value for the clumps. Clumps larger than this became unrealistic and would also be highly susceptible to being broken up by the algorithms into multiple clumps. Thus the FWHM parameter values were chosen to be between six and ten pixels.

5.3.3 Peak Depth

During the investigation of the peak values it was noticed that ClumpFind was unable to make detections around five times the RMS value whereas the other methods were able to make a number of detections of clumps at this peak level. Although the parameter values for each algorithm determined that each one has the same minimum value for the clump peak the results show that ClumpFind is unable to detect clumps with a peak value closer to this minimum value than the other algorithms. Figures 5.6 and 5.7 show plots of clumps detected with different peak values, the dashed line is at four and a half times the RMS value of the image and it can be seen that FellWalker, GaussClumps, and Reinhold detect clumps all the way down to this point though with a decreasing number of detections as the peak value decreases. ClumpFind peak value detections stop abruptly at an approximate value of six times the RMS value of the image, from Figure 5.2 it can be seen that ClumpFind does make detections at around five times the RMS value of the image yet the output values did not show this. Also the abrupt line where the data points seem to stop suddenly does not match with the other algorithms and their gradual decline in the number of detected clumps at lowering peak values. Though from Figures 5.2 - 5.5 it can be seen that after detections start to occur, the number of detections increases more rapidly for ClumpFind than the other algorithms as the peak value of the clump increases. This would explain a quicker decline in the detections for ClumpFind over the other algorithms in Figures 5.6 and 5.7 but does not account for such an abrupt halt to the detection as is observed in these Figures. One explanation as to why ClumpFind does not make detections of clumps less than six times the RMS value is due to the parameters DeltaT and TLow, these dictate the minimum value at which to investigate for clumps and the parameter level which is used to select the contour levels independently, if no level values are input, as was the case, then the lowest contour level is equivalent to the sum of DeltaT and TLow which would equal six times the RMS value (both DeltaT and TLow having values of three times the RMS value). This explains why the detections only start when the peak value is at least six times the RMS value as shown in Figures 5.6 and 5.7 but does not explain why it does not match with Figure 5.2. If the artificial source were to lay on an area of high noise then its peak value would be greater than what was input as such an inserted source with a peak of five times the RMS could appear to have a peak value high enough for ClumpFind to detect it, that could explain why some sources were detected that had an input peak value of five times the RMS yet the results show no detections with a peak value of less than six times the RMS.

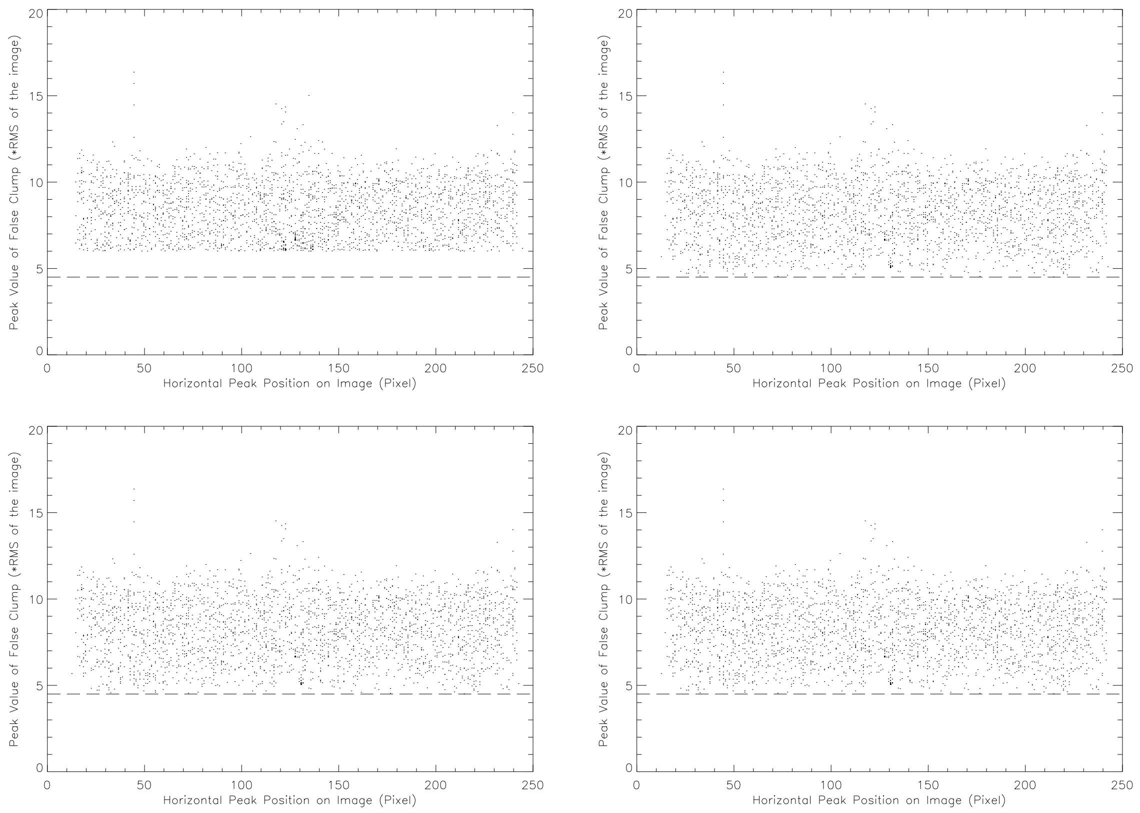


Figure 5.6: Plots showing the result for the clump peak value against its position in the image for the 450μm data. The dotted line is at a value of four and a half times the RMS value of the image for all the plots. Top, from left; ClumpFind, FellWalker. Bottom, from left; GaussClumps, Reinhold.

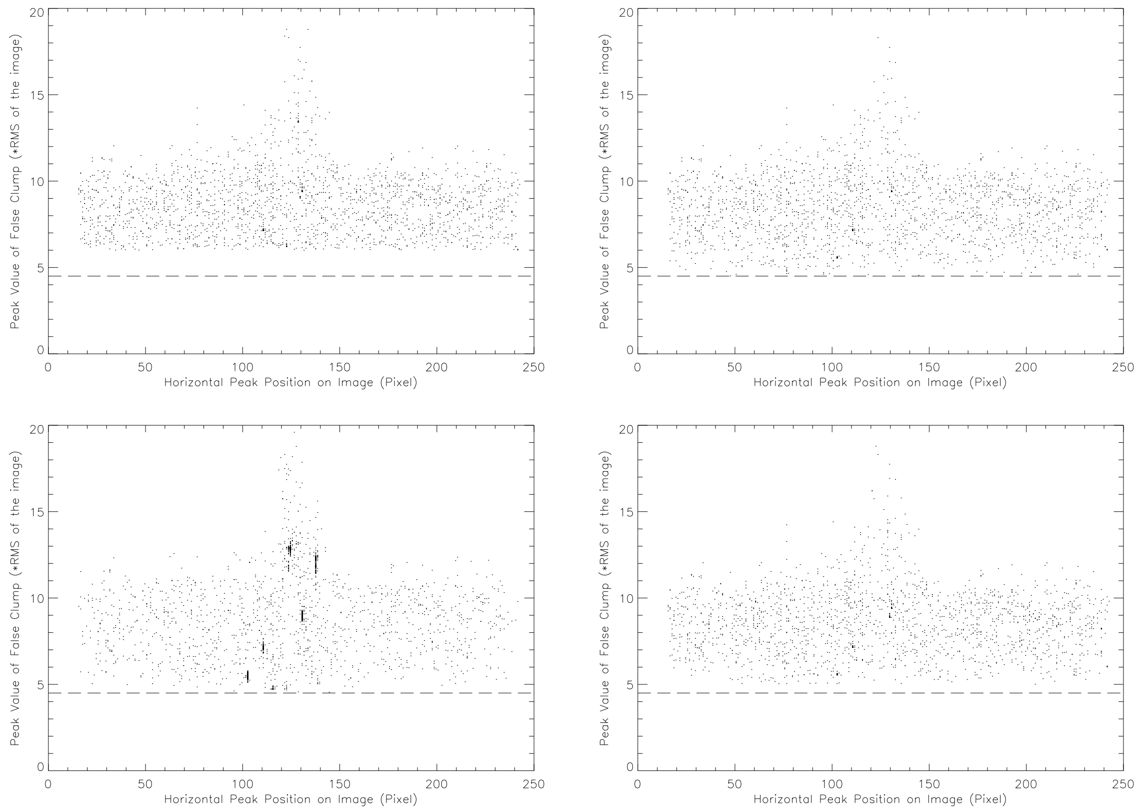


Figure 5.7: Plots showing the result for the clump peak value against its position in the image for the 850μm data. The dotted line is at a value of four and a half times the RMS value of the image for all the plots. Top, from left; ClumpFind, FellWalker. Bottom, from left; GaussClumps, Reinhold.

5.3.4 Clump Location

Although the data image is a 256×256 pixel array, the way in which SCUBA takes the image does not give data in each one of those pixels, causing the array image to have ragged edges, which the artificial clump would not be able to be added onto. The Gaussian array needs to be the same size as the image therefore the only way to control the location of where the artificial clumps may be placed it to manipulate the RandomU value such that it avoids certain areas. To avoid the boundaries the manipulation is simple, since RandomU only produces a value between 0 and 1, the value needs to be multiplied by 256 so the value goes between 0 and 256. Thus to restrain the boundary it was a simple matter of reducing the range allowed for the RandomU value. After implementing a boundary which just excluded the poor boundary areas we then had to extend it so that not only the central location of the clump but none of the clump resided within the boundary areas. After testing this new boundary it was noted that the noise spikes around the edges (and within the present boundary) were great and frequent enough to cause repeated errors in the clump detection due to a large clump being broken up into smaller clumps. The boundaries were further restrained to avoid these noisy areas as well.

We were only interested in the artificial clump and not the original clumps, to this end we did not allow an artificial clump to be located in the same place as one of the original clumps in the image. To prevent the artificial clumps being present in the central region where the original clumps are located this area also had to be excluded, this was achieved using a loop such that RandomU would create a location and if it was not within the accepted area it would then discard that location and create a new one. With both of these restrictions in place the acceptable area for the clumps to be located within is shown in Figure 5.9

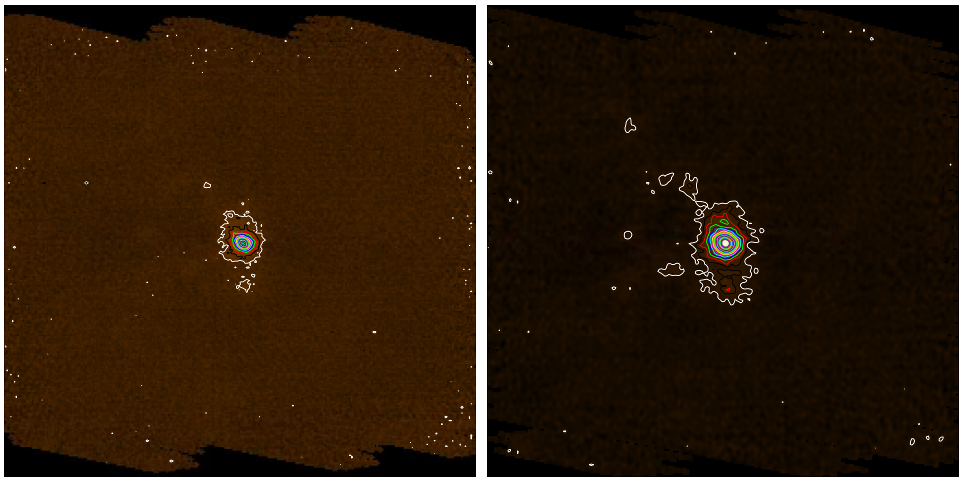


Figure 5.8: From left; g1084-259_SCAMPS_sho_flat.sdf, g1084-259_SCAMPS_lon_flat.sdf, both with contour levels starting from three times the RMS value with spacing between the contours of three times the RMS value.

Once the artificial clump had been added and the image was ready for CUPID to perform the clumpfinding process the original sources had to be removed. This was achieved using the

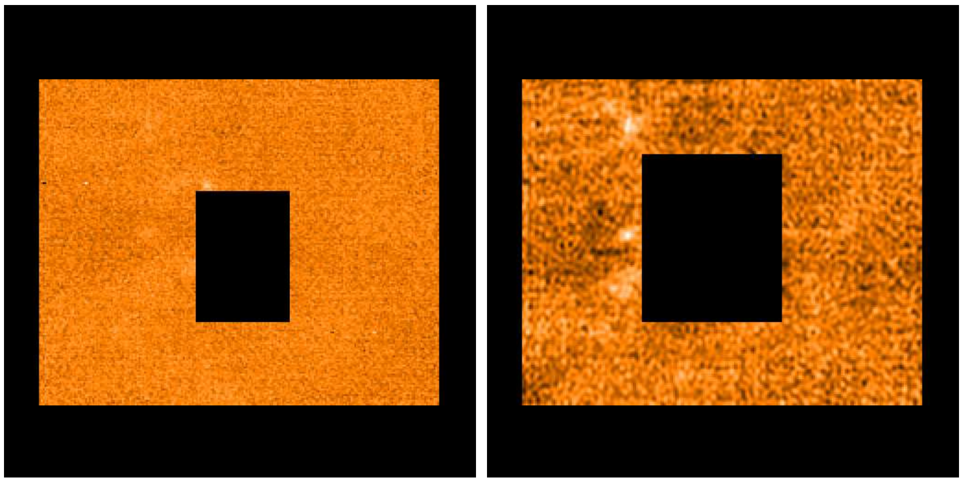


Figure 5.9: Image of boundary area for the insertion of artificial clumps, the black areas show the regions where the centre of the artificial clump is not allowed to be located. Left; 450μm image, right; 850μm image.

mask command in KAPPA, each pixel in the central region was changed to a bad pixel in a similar manner to the edge of the image. The clumpfinding process would not detect any sources that were originally present in this masked area. Since this area was considered the same as the image boundary then any detected clump touching this area would be discarded due to the AllowEdge parameter. With this area removed the only detectable source within the image was the artificial clump.

In the 850μm image in Figure 5.9 areas that are marginally higher than the background emission can be seen, one clear area is just above the top, left corner of the central boundary area. This area can also be seen in the 850μm image in Figure 5.8. These areas were not detected by any of the clumpfinding algorithms in Chapters 3 or 4 since they were not bright enough to be considered a real source. Therefore they were overlooked during the creation of the boundary area. If an artificial clump were placed on top of one of these areas the properties of the clump would be altered, i.e. its size and integrated flux would increase.

5.4 Monte-Carlo Simulations

Monte-Carlo simulations involve using random numbers as inputs for a computer model to provide a statistically large number of results such that a valid conclusion can be made. Random inputs are used to test the thoroughness of a model since the expected output can be compared against the actual output for a large range of input values. This investigation involved the insertion of an artificial source into one of the SCAMPS images (see §5.2.1) and repeating the process a number of times. The location, size and peak value of the artificial clump were chosen to be random each time (between the fixed limits discussed earlier in the chapter). The detection of each artificial source was independent of other artificial sources, this was achieved by not only having a single artificial source on each image but having many of the same image each with its own randomly positioned artificial source. Since each artificial source does not affect another source there is no confusion within the images.

Previous investigations have used Monte Carlo simulations for testing these algorithms, (Enoch et al., 2006) where the simulation was used to test the completeness of ClumpFind, and also from the ClumpFind results determine the mass versus size distribution of clumps within the Perseus Molecular Cloud. Coppin et al. (2006) undertook an investigation where by artificial point sources were inserted and different reduction processes were performed to determine the completeness of each process for SHADES (SCUBA Half Degree Extragalactic Survey). In my investigation artificial sources were inserted into a SCAMPS image to test the completeness of each algorithm and whether the output flux distribution of the clumps adequately reflected the input flux distribution.

5.4.1 Recovery of an Input Flux Distribution

The final part of the investigation required the insertion of 1000 artificial clumps, each with a specific and constant integrated flux value. Only one artificial source would be added to a single image so to avoid confusion and blended emission, again the image used was g1084-259_SCAMPS as discussed in §5.2.1. The purpose of inserting clumps with the same integrated flux value was to test what the algorithms truly detected. If they are not biased to detecting clumps of a particular flux then they would be able to give results that match the input values exactly. The end result allowed us to determine if the algorithms can be used to correctly determine the clump mass spectrum. Previous investigations such as Pineda et al. (2009) have used ClumpFind to determine the clump mass spectrum. Mookerjea et al. (2004) and Muñoz et al. (2007) used both ClumpFind and GaussClumps independently to determine the clump mass spectrum and found the results for both algorithms to be similar. To this end the expectation would be that both algorithms determine similar flux values, §4.2.3 shows that there was some mild variation in the flux values output from each of these algorithms, what must be determined is whether these flux values accurately reflect the ”true” values.

The artificial clumps all had a Gaussian profile. This is not exactly the same as real clumps since there are other factors that affect the profile of a source, but it is similar to real sources and has the advantage of being easy to generate with only two parameters. The integrated flux of a two-dimensional Gaussian is proportional to the FWHM and peak value by the following equation:

$$Integrated\ flux = 1.133 \times PeakValue \times FWHM^2$$

The integrated flux value was chosen to remain constant for each set of 1000 images, but the investigation would be performed at five different flux values for each wavelength; 23.4Jy, 30.6Jy, 37.9Jy, 45.1Jy, and 52.3Jy for the 450μm investigation and 1.23Jy, 2.02Jy, 2.82Jy, 3.61Jy, and 4.41Jy for the 850μm investigation. From the earlier tests we had a range of acceptable FWHM values, a minimum peak value and an area where the placement of the artificial clump would be acceptable. All these details came together for the final part of the investigation. The FWHM value was randomly chosen using RandomU to a value between six and ten pixels, therefore to maintain a constant integrated flux the peak must have a value that coincided with the FWHM value. The ranges of the values chosen are shown in Tables 5.2 and 5.3, since the lowest peak value will be with the largest clump and the highest peak with the smallest clump only the peak values for the smallest clump size (FWHM = 6 pixels) and largest clump size (FWHM = 10 Pixels) are displayed.

Table 5.2: Range of Peak values used with the corresponding input flux values for the 450μm investigation.

Input Flux Value (Jansky)	Peak Value when FWHM=6 (Jansky/Beam)	Peak Value when FWHM=10 (Jansky/Beam)
23.4044	5.85	2.11 (6*RMS)
30.6322	7.66	2.76
37.8601	9.47	3.41
45.0879	11.27	4.05
52.3158	13.08 (37.3*RMS)	4.71

Table 5.3: Range of Peak values used with the corresponding input flux values for the 850μm investigation.

Input Flux Value (Jansky)	Peak Value when FWHM=6 (Jansky/Beam)	Peak Value when FWHM=10 (Jansky/Beam)
1.22798	0.853	0.307 (6*RMS)
2.02256	1.405	0.506
2.81714	1.956	0.704
3.61171	2.508	0.903
4.40629	3.060 (60*RMS)	1.102

These values are the limit of the investigation, as such any analysis past these values is highly speculative and any assumed trends are based purely on the known data.

5.4.2 Results

Each of the Figures (5.10 - 5.17) show the peak value against the total integrated flux of the 1000 artificial clumps. The dashed line shows the input flux value of the clumps, each dot represents a clump detected by one of the algorithms and the solid line represents the best fit curve as made by curvefit[8]. This fit does not reflect any physical model, and is only to better display the observed trend.

With all the 850μm artificial clump plots (Figures: 5.11, 5.13, 5.15, and 5.17) there are a small number of clumps with a detected integrated flux value greater than the input flux value. Looking at Figure 5.8 and 5.9 it can be seen that just outside the masked area there is an area of higher emission, though this area itself is not above three times the RMS value so it was not detected earlier in Chapters 3 or 4. If the artificial clump is located in this area it could have caused the detection of clumps with flux values higher than the input flux value.

ClumpFind

The first noticeable thing with the plots in Figures 5.10 and 5.11 is that nearly all the clumps have a detected flux less than the flux of the inserted clumps. This difference is greater for clumps with a low peak value and therefore a large size. This causes an asymptotic trend for the clump flux towards the input flux value, though the point where it asymptotes to is less than the input flux value. As the integrated flux value of the clumps increases the difference between the input and output flux decreases, this slowly produces a flatter function for the results.

For all the ClumpFind 450μm plots in Figure 5.10 there is a sharp cut off point causing a flattened surface where there are many clumps within a small range of peak values but a larger range of flux values. This cut off occurs around a peak value of ten times the RMS value. For the plots with ranges below ten times the RMS value, the clumps also follow the observed trend to lower values. The plots of higher flux clumps show many clumps that have large peak values but flux values much lower than the fit would expect. These are more than likely due to the breaking up of larger clumps into smaller clumps, therefore clumps are detected with a high peak value but a portion of the clump is detected as an additional clump resulting in a smaller size and therefore lower flux. The 850μm clumps (Figure 5.11) do not display this sharp cut off nor an obvious display of the larger clumps being broken up, perhaps due to the lower noise in the 850μm images.

[8]Curvefit is an IDL program used to assess the fit for a mathematical function on a set of data points, this gives values for any variable used in the function. full details can be found at http://www.astro.virginia.edu/class/oconnell/astr511/idl_5.1_html/idl50.htm (documentation for IDL 5.1, accessed March 2009).

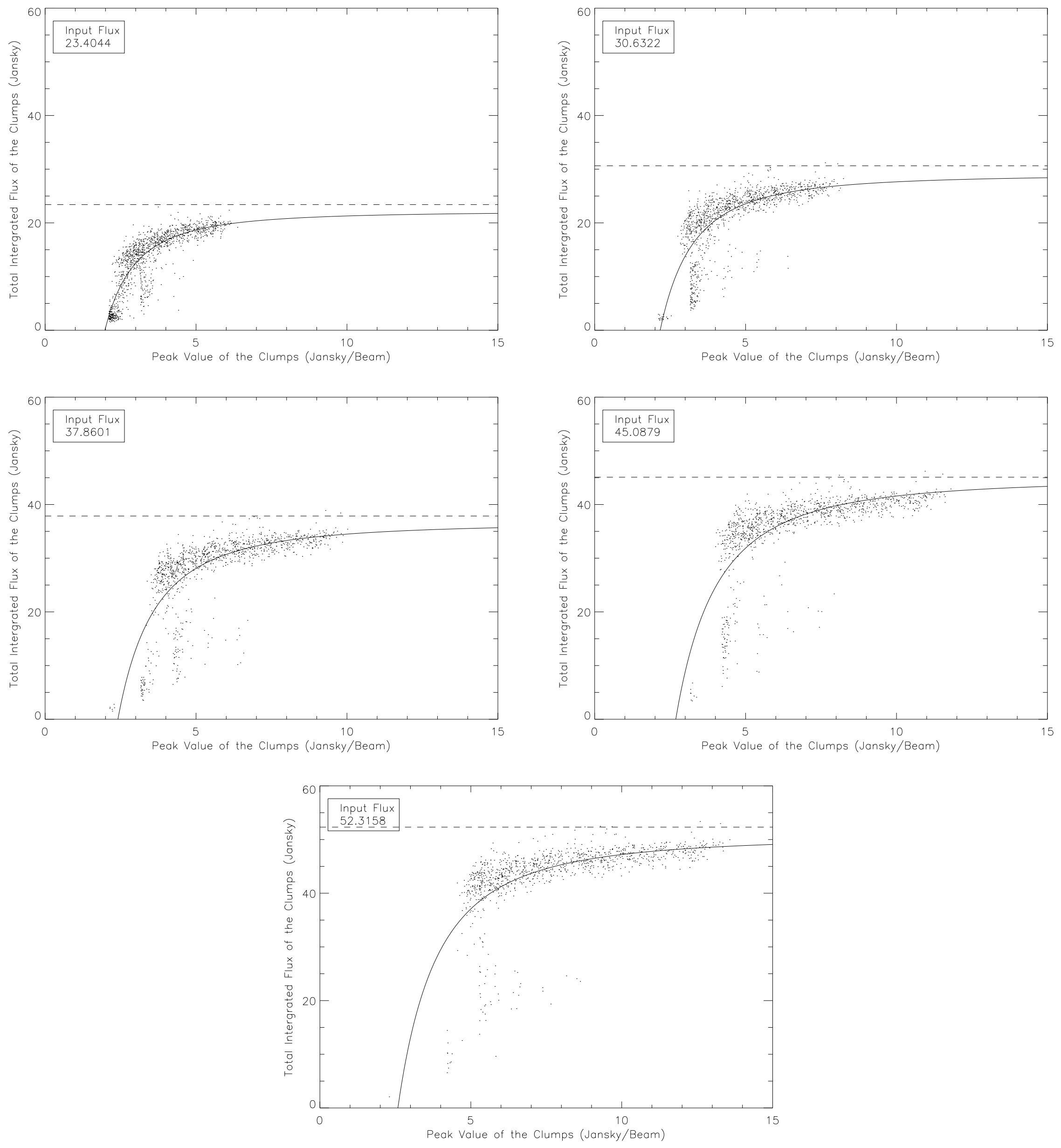


Figure 5.10: Plots showing what ClumpFind determined as the integrated flux values of the artificial clumps using the 450μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

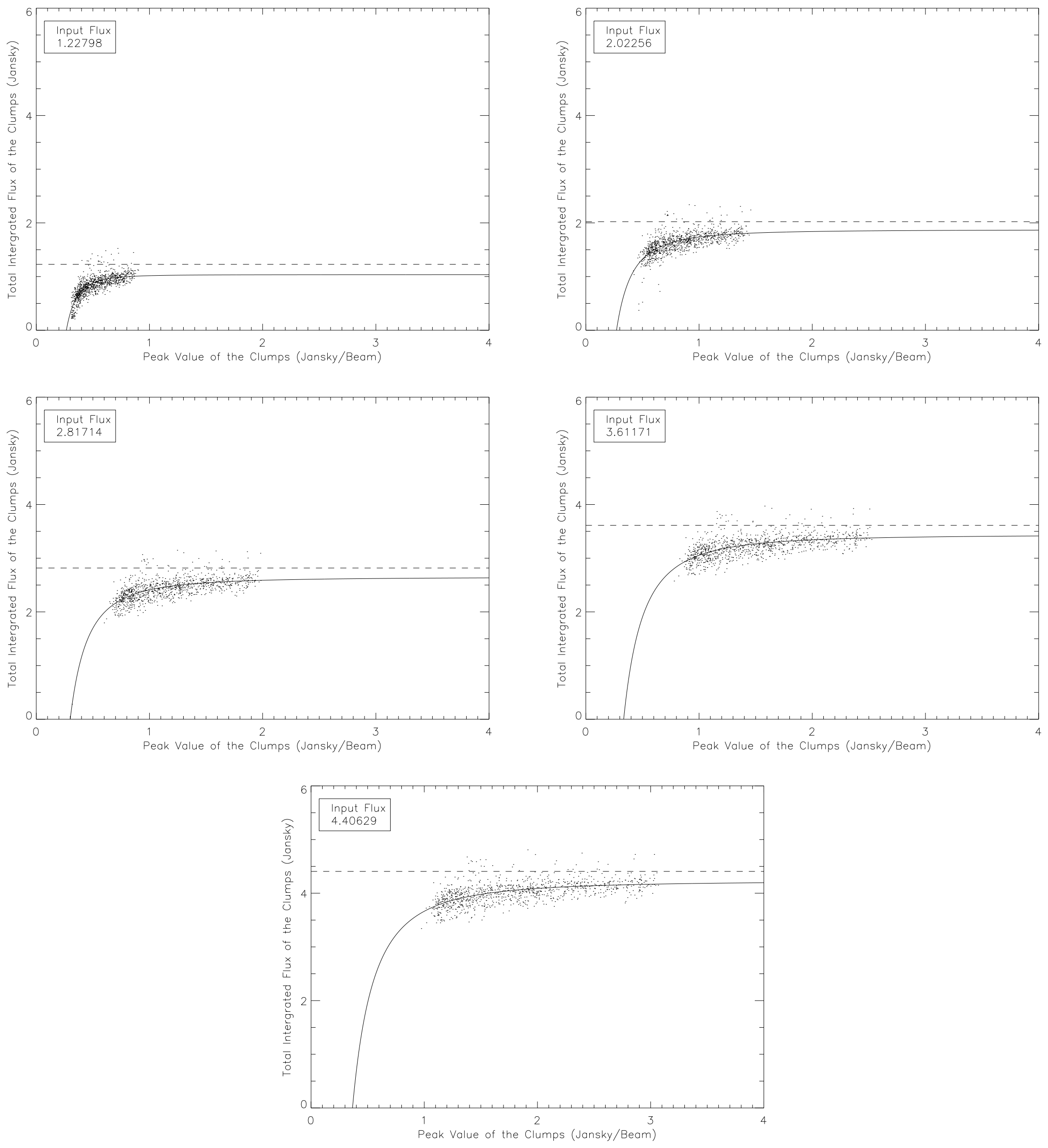


Figure 5.11: Plots showing what ClumpFind determined as the integrated flux values of the artificial clumps using the 850μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

FellWalker

In the plots of Figures 5.12 and 5.13 nearly all of the clumps have a detected flux less than the inserted clump flux. This difference is greater for clumps with a low peak value and therefore a large size. This causes an asymptotic trend for the clump flux towards the input flux value, though the point where it asymptotes to is often less than the input flux value. This trend is similar to the one observed for ClumpFind

As the flux value increases the difference between the output and input fluxes remain greater for the large clumps than the small, as shown by the curve in the data. With the other algorithms as the input flux value increases a flatter function is produced whereas the curve remains present with FellWalker even at higher flux levels.

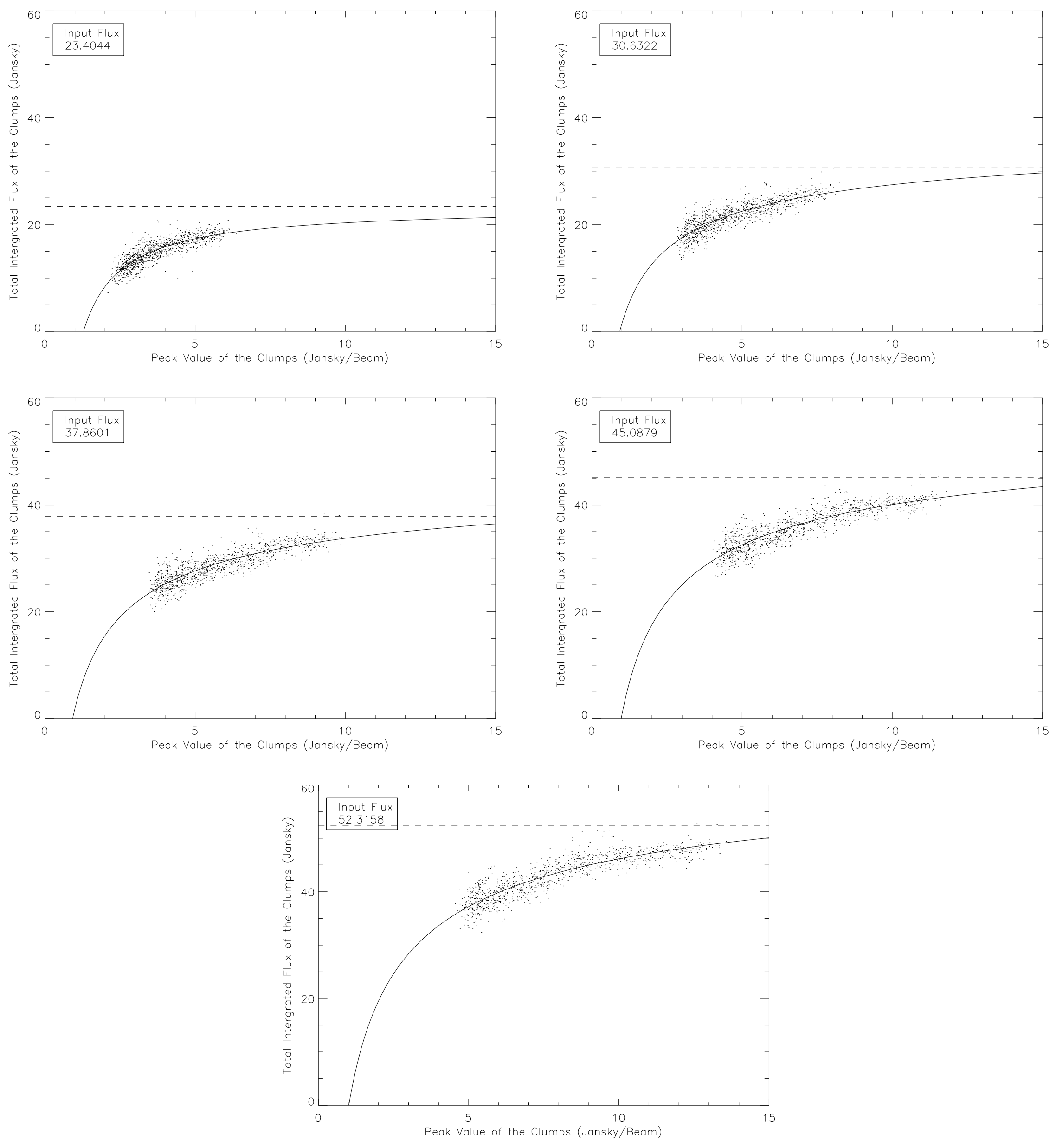


Figure 5.12: Plots showing what FellWalker determined as the integrated flux values of the artificial clumps using the 450μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

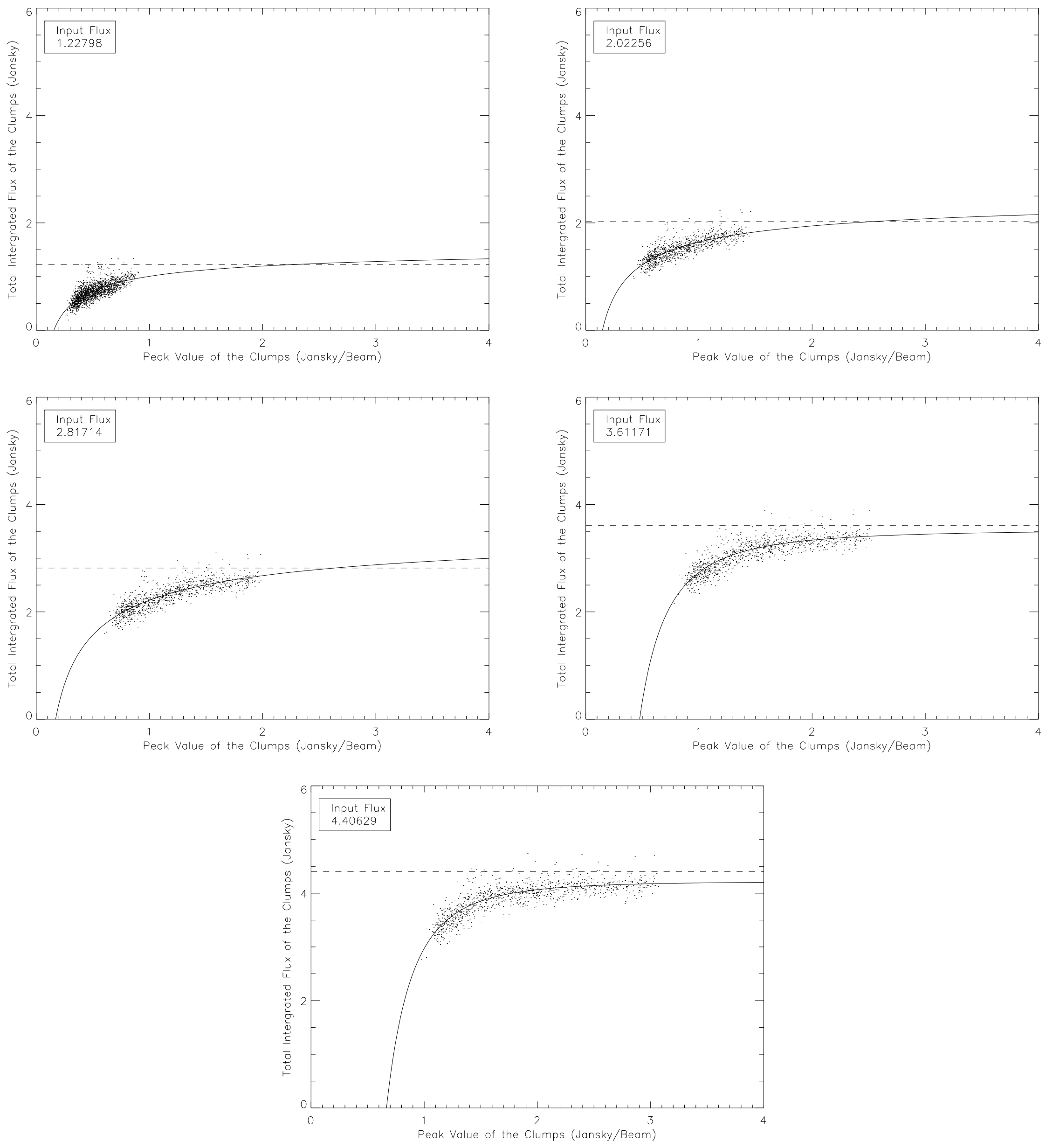


Figure 5.13: Plots showing what FellWalker determined as the integrated flux values of the artificial clumps using the 850μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

GaussClumps

Figures 5.14 and 5.15 show that GaussClumps recovers the input flux distribution much more closely than the other algorithms. There is still a tendency to underestimate the flux at low peak values which is particularly evident in the 450μm data. There is still an asymptotic trend for the clump flux towards the input flux value same as the other algorithms, though the point where it asymptotes to is nearly always approximate to the input flux value. The 850μm data curve approximates to a value closer to the input flux value than the 450μm data where the best fit lines are often less than the input flux value

The plots show a few clumps that have large peak values but flux values much lower than the fit would expect, these are more than likely due to the breaking up of larger clumps into smaller clumps. The 850μm artificial clumps do not show the larger clumps being broken up as much as the 450μm artificial clumps, which is probably due to the lower noise in these images

The 450μm artificial clumps appear shifted to the left along the asymptotic curve compared to the 850μm clumps, such that many appear on the turning point of the curve rather than the flatter portion afterward. This effect causes large, low peak value clumps to give a flux value lower than what the small, high peak value clumps do. With the data being shifted to the left the 450μm data clumps give a larger difference between the input and output integrated flux values than the 850μm for the larger, flatter clumps. small, high peak clumps in the 850μm images give values that are the same as the input values, within an acceptable degree of variation. Though with the 450μm images the clump output fluxes remain consistently lower than the input value.

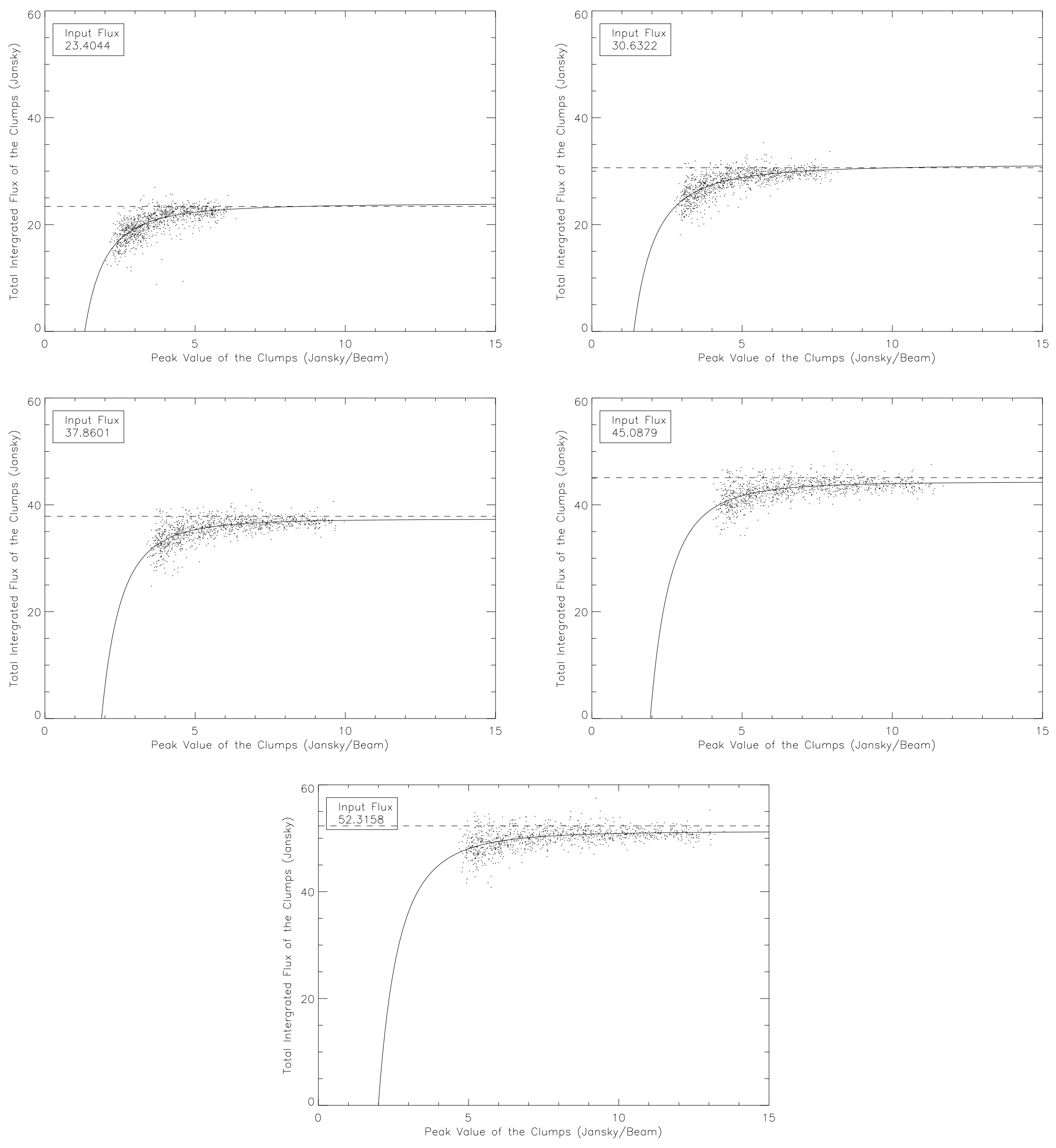


Figure 5.14: Plots showing what GaussClumps determined as the integrated flux values of the artificial clumps using the 450μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

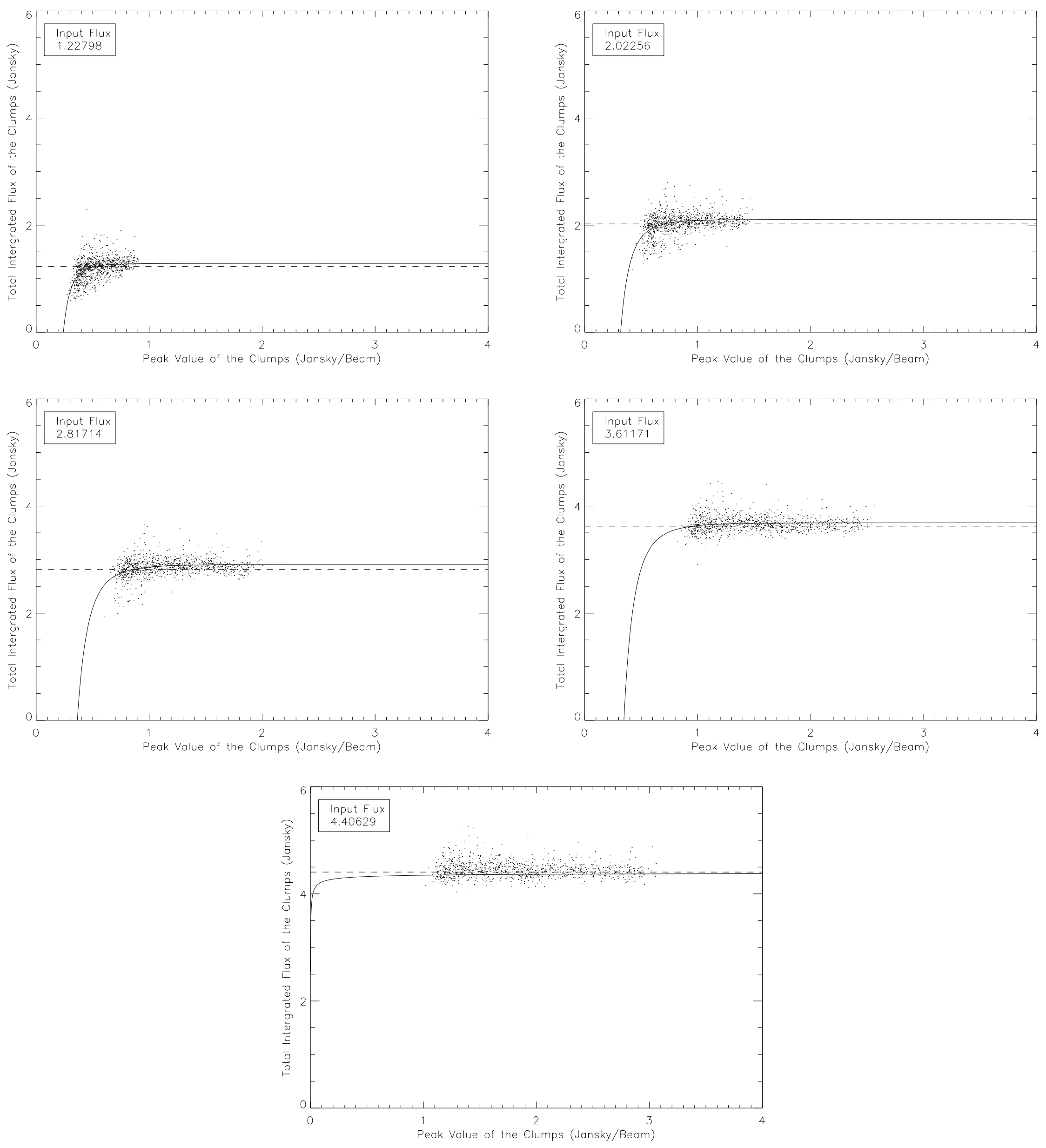


Figure 5.15: Plots showing what GaussClumps determined as the integrated flux values of the artificial clumps using the 850μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

Reinhold

In the plots in Figures 5.16 and 5.17 nearly all of the clumps have a detected flux less than the flux of the inserted clumps. This difference is greater for clumps with a low peak value and therefore a large size. This causes an asymptotic trend for the clump flux towards the input flux value, though the point where it asymptotes to is nearly always less than the input flux value. As the flux value increases the difference between the input and output flux for the larger clumps decreases, this slowly produces a flatter function for the results. Again, this is broadly similar to the results of ClumpFind and FellWalker.

The plots show many clumps that have large peak values but flux values much lower than the fit would expect, these are due to the process breaking up the larger clumps into smaller clumps, therefore clumps are detected that have a high peak value but a portion of the clump is detected as another clump resulting in a smaller size and therefore lower flux. The 850μm artificial clumps do not display the larger clumps being broken up as much as the 450μm artificial clumps.

Compared to the other algorithms Reinholds best fitting line often asymptotes to a value less than the other algorithms, the best fit line does not run through the middle of the main curve of the data but slightly below it which may account for it asymptoting to a lower value. The reason for the line not running through the centre is due to it being off-set by the large number of high peak, low flux clumps caused by the breaking up. Since the 850μm artificial clumps do not experience being broken up as much as the 450μm clumps, then their line of best fit runs more centrally through the main data points.

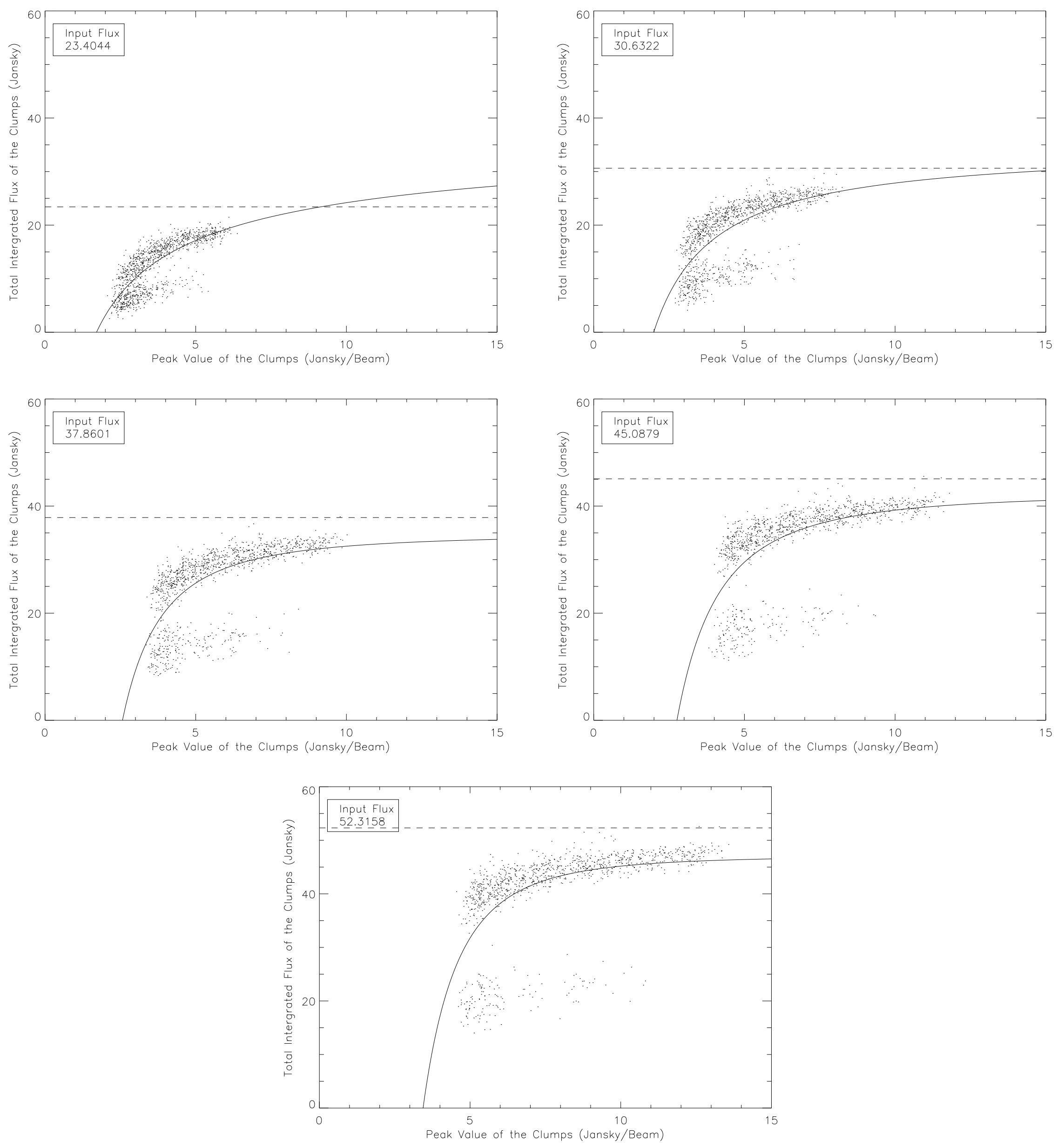


Figure 5.16: Plots showing what Reinhold determined as the integrated flux values of the artificial clumps using the 450μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

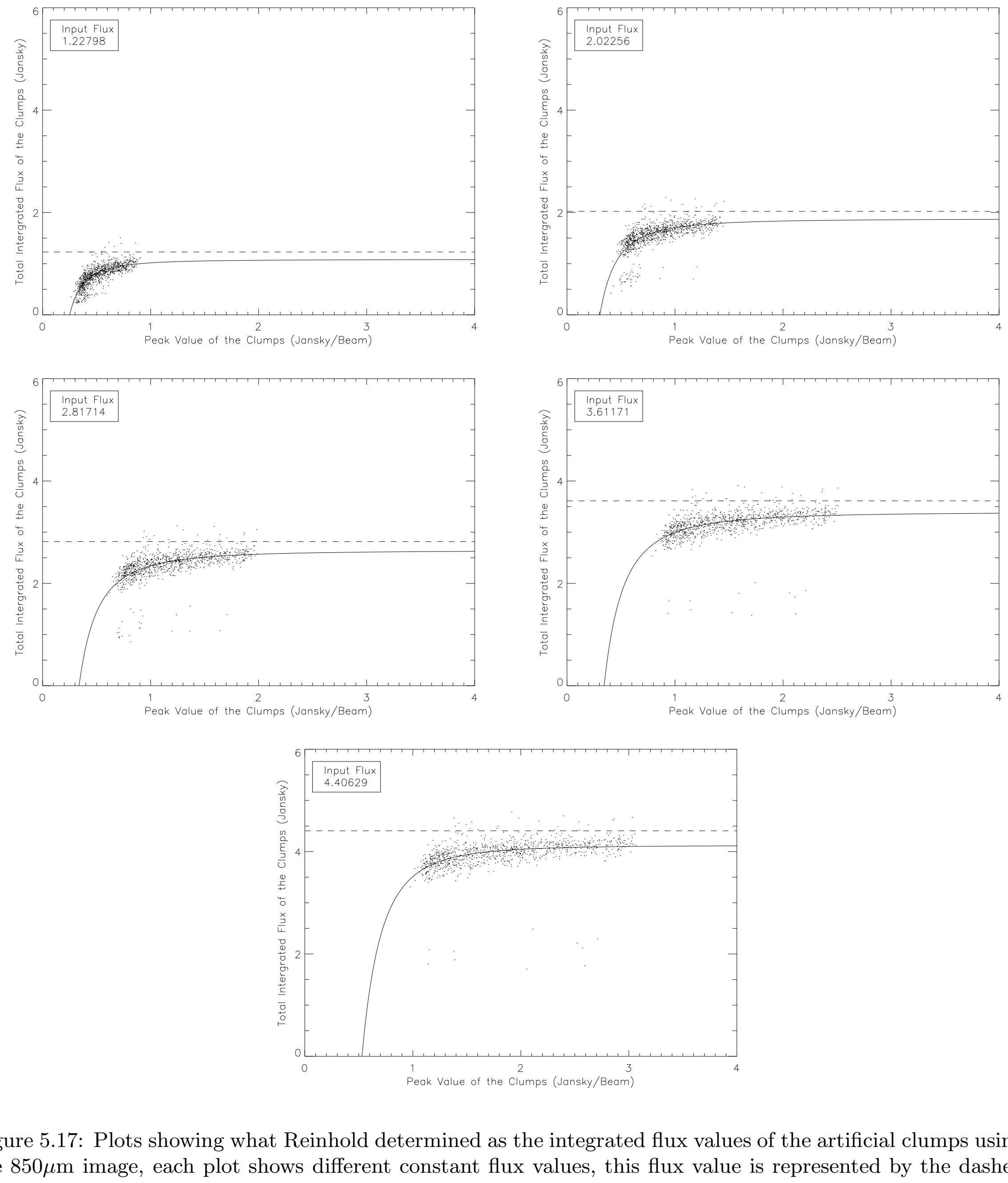


Figure 5.17: Plots showing what Reinhold determined as the integrated flux values of the artificial clumps using the 850μm image, each plot shows different constant flux values, this flux value is represented by the dashed line and equals the value in Janskys as the value in the box in the top left of each plot.

Fraction of Missing Flux

Figures 5.10 - 5.17 give the output flux in comparison to the input flux, it is clearly visible that the output flux is always less than the input (with the exception of GaussClumps. Figures 5.18 and 5.19 show the difference between the input value and the fit for the output values as a percentage of the original input. This allows the comparison between the algorithms for the same input flux values. The dashed lines show the range of peak values for each input flux values, this range was predetermined by the range of the simulations and is shown in tables 5.2 and 5.3. The curves beyond these lines are an extrapolation and do not necessarily represent an accurate trend

The plots in Figure 5.18 and 5.19 show a clear trend that clumps with a lower peak value and therefore larger size have a large error in their measured flux, and that as the peak value of the clump increases the error from the input flux decreases. The measured flux at all peak values is an underestimate of the true clump flux. GaussClumps has a consistently smaller error in the measured flux than the other algorithms. although the other algorithms underestimate the flux to a higher degree than GaussClumps they each provide similar results to each other such that at many points the fits overlap each other. This correspondence is much more evident in the 850μm results.

The 450μm plots in Figure 5.18 show the flux underestimation to range from 90% of the inserted flux being missed for the lowest peak values to approximately 10% or less of the flux being missed as the peak value increases. For the 850μm plots in Figure 5.19 the underestimate ranges between 70% and less than 10% of missed flux. The 850μm results show a lower error value for the flux detection over the 450μm results. The 850μm data extends to a higher peak value relative to the background noise and thus shows a greater range than the 450μm data, indicating what happens with even higher peak values. Taking this into account, the 450μm and 850μm results give similar trends for peak values of the same multiplication of the RMS value on the image.

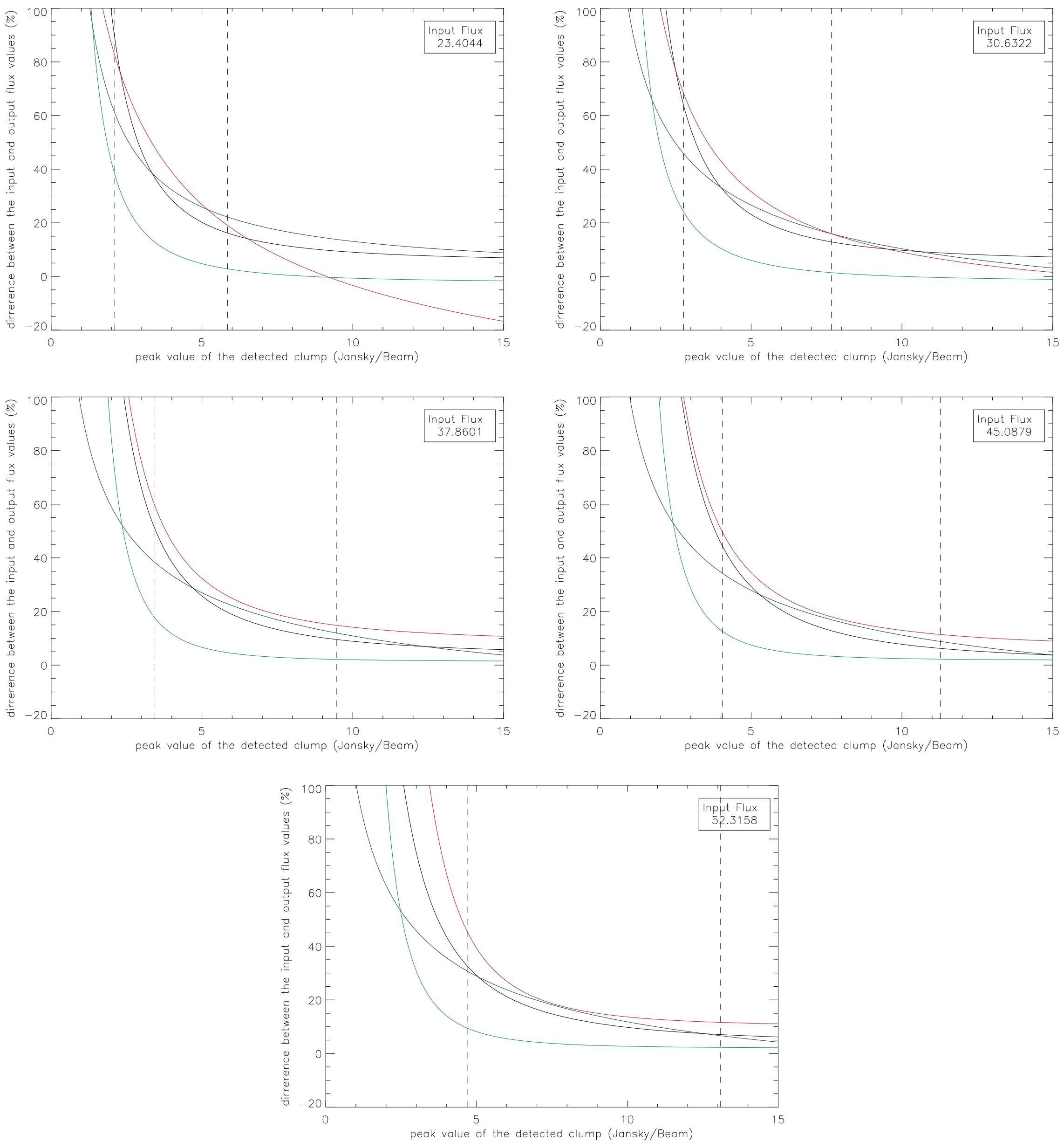


Figure 5.18: Plots showing the percentage error between the constant input flux and the flux output by the algorithms for the 450μm data. Each plot showing different input flux values (shown in Janskys at the top right corner) for all the algorithms; black being ClumpFind, blue being FellWalker, green being GaussClumps and red being Reinhold, with the dashed lines showing the range of peak values for that flux value as shown in table 5.2.

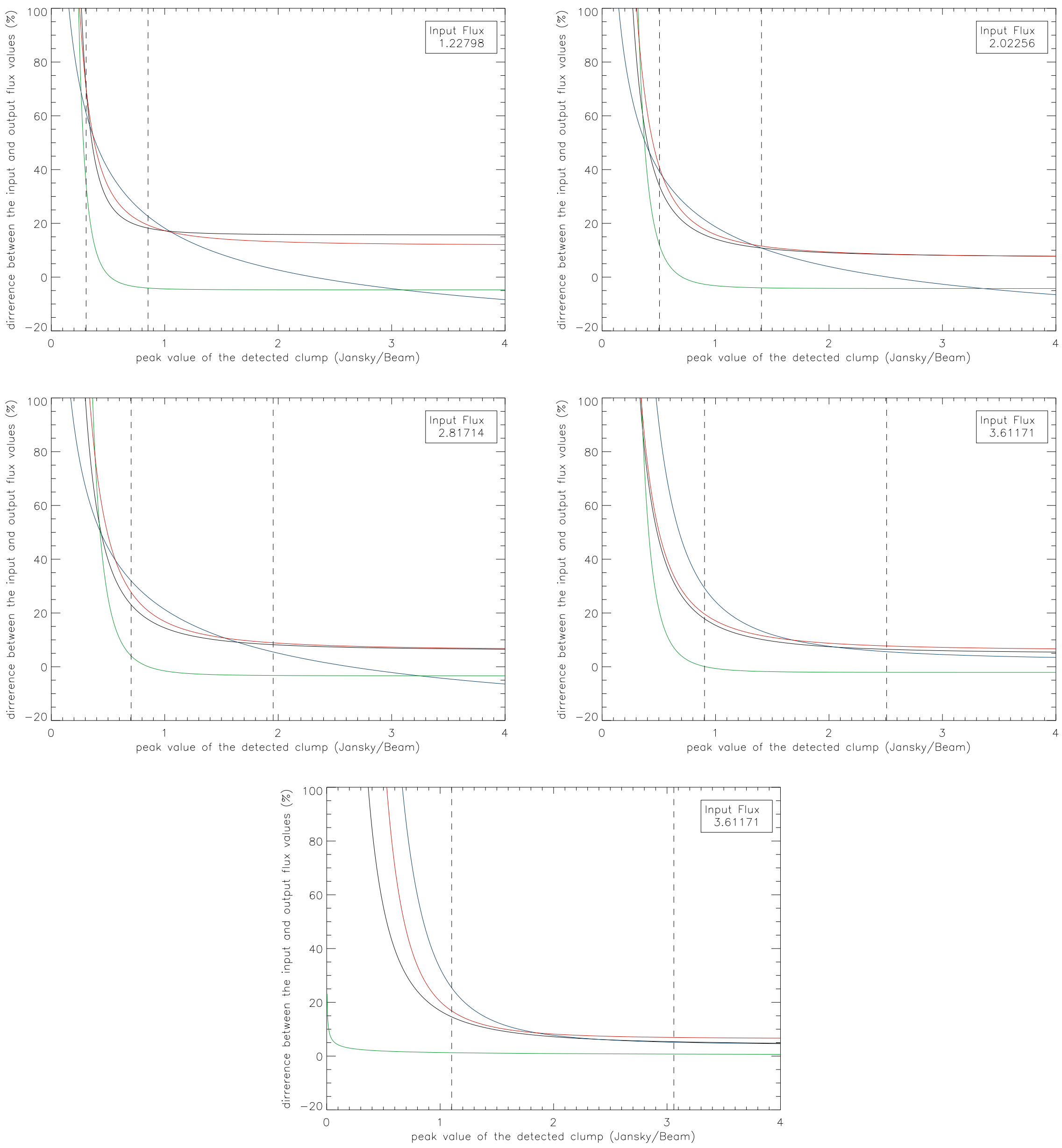


Figure 5.19: Plots showing the percentage error between the constant input flux and the flux output by the algorithms for the 850μm data. Each plot showing different input flux values (shown in Janskys at the top right corner) for all the algorithms; black being ClumpFind, blue being FellWalker, green being GaussClumps and red being Reinhold, with the dashed lines showing the range of peak values for that flux value as shown in table 5.3.

5.5 Conclusion

ClumpFind, FellWalker, and Reinhold almost always gave a flux value less than the input value. I believe the possible reason for this is shown by Figure 5.20. The Figure shows two Gaussian artificial clumps, both of equal integrated flux value, clump 'A' is smaller therefore has a higher peak value than clump 'B'. Since the parameters for all the algorithms have a minimum pixel value for the clumps of three times the RMS value (shown in the Figure by the solid horizontal line) then any pixel with a value less than this minimum would not be considered part of the clump and the hashed areas would be discarded. For small clumps the discarded area would be small. Whereas for larger, fainter clumps the algorithms might ignore a large proportion of the clump. GaussClumps on the other hand continues to fit the Gaussian profile below the RMS cut-off values, hence detects more flux than the other clumpfinding algorithms. For GaussClumps, the cut off at three times the RMS value only restricts the detection of new Gaussian sources. The available fitting region decreases for broad, low peak Gaussians (as shown in Figure 5.20) it is likely that these Gaussians give an underestimate to the width and therefore flux. This explains the falloff in flux values for GaussClumps as the peak value decreases. The effect is greatly reduced as the peak value increases or the clump size decreases.

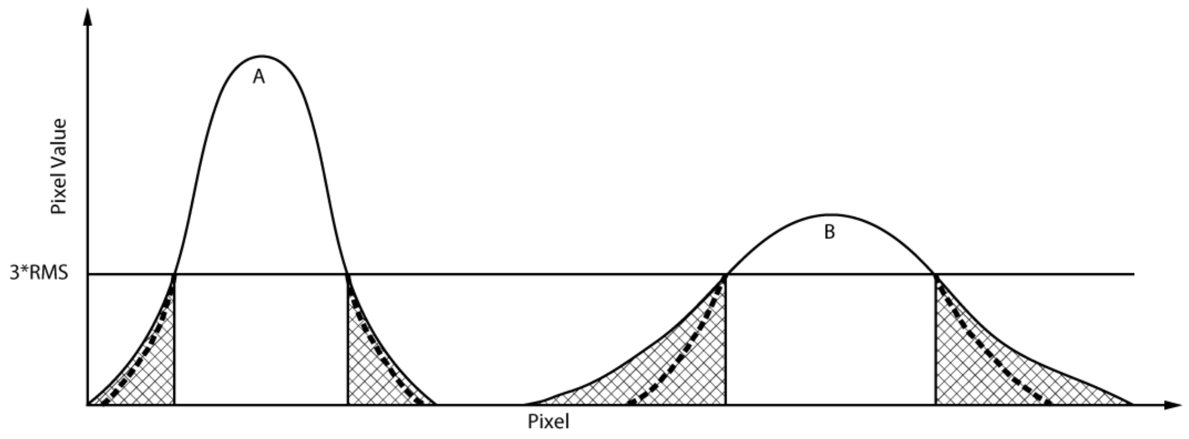


Figure 5.20: Graphical representation of the clump detections with the algorithms. Each of the two solid line Gaussians; A and B, have the same integrated flux value but different FWHM and peak values. The line at three times the RMS value shows the algorithm cut-off point and the hashed areas show the potential area of the clump discarded by ClumpFind, FellWalker, and Reinhold. The dashed lines show how GaussClumps makes a clump profile below this cut off point.

This explanation not only gives a reason why ClumpFind, FellWalker, and Reinhold output flux values that are much lower than what the image but also explains why each plot (Figures 5.10 - 5.17) demonstrates a greater separation between the input and output flux values for clumps with a lower peak value. In each case the clumps with lower peak values would be larger and therefore a greater proportion of the clump is discarded. For high constant flux values the effect is less than with lower constant flux as the size of the clumps remain within the same range. With higher flux clumps the peak value is greater causing a larger proportion of the clump to be above the three times RMS cut off point. One thing this explanation does not account for is why each of the algorithms give different results and trend for each flux level. If all the algorithms except GaussClumps are subject to this cut-off point then it would be expected that the results from ClumpFind, FellWalker, and Reinhold should be much more similar then they appear to be. The effect of this could be reduced by altering the algorithm parameters for ClumpFind, FellWalker, and Reinhold which dictate the minimum peak size and the lowest value the clump should extend to, but as earlier investigations have shown (§3.2.1) altering these parameters can greatly effect the number of clumps detected. Indeed, lowering the minimum threshold value below three times the RMS value can result in the detection of a large number of spurious clumps.

The FellWalker plots in Figures 5.12 and 5.13 show that even at high flux levels the flux for larger clumps has a greater underestimation than the smaller clumps. This could be due to the FlatSlope parameter. FlatSlope dictates the edge of a clump to have a specific gradient over four pixels, its default value is one times the RMS value but to make the algorithms more comparable this was changed to three times the RMS value. Lowering this parameter value would reduce the effect of larger clumps having a lower flux than smaller clumps. It would therefore cause the plots to give a flatter function, especially for the higher flux value plots.

GaussClumps is not as subject to either of these problems since it fits its own Gaussian curves over the clumps and from the profile it is then able to take the detection below the three times RMS cut-off. This explains why GaussClumps results are more similar to the input values than the other algorithms. Although the 450μm images provided results that were consistently lower than the 850μm images. A possible reason may be due to the FwhmStart parameter, as shown in §3.2.4 altering this parameter can affect the size of the clumps detected which in turn effects the integrated flux value for the clumps. §2.2.3 states how the FwhmStart parameter works, it also states that the value for this parameter varies, the algorithm takes a guess for the value based on the local profile around the peak pixel. This may be giving a different value for the 450μm which causes the fit of the Gaussian to be incorrect.

It is not surprising that the GaussClumps algorithm works well since it is fitting an ideal Gaussian profile over an ideal Gaussian clump. Further investigations would require either the insertion of more realistic clumps or at least non Gaussian profiles (e.g. Bonnor-Ebert spheres (Bonnor, 1956)) to see how accurately GaussClumps would be able to extract them. ClumpFind, FellWalker and Reinhold segment the data such that a pixel can only belong to one clump, the intention is that this process balances out such that although the clumps are determined to be smaller the edges of them would have higher pixel values and the integrated flux would not be effected. Since only a single artificial source was present in each image this segmentation would not have had any effect on the results though it is uncertain what the effect would be in a more crowded image.

As previously mentioned in §3.3 the Reinhold algorithm rarely had clumps that would touch each other in a crowded image, this would mean the algorithm detects the clumps to be smaller than they are and therefore the flux would be lower. The artificial clump process did not use a crowded image and this effect would not be noticeable from looking at the output clump index image from CUPID. As with the other algorithms Reinholds flux values (Figures 5.16 and 5.17) gave an asymptotic curve that asymptotes to a value lower than the input flux value. In the majority of cases the value where the curve asymptotes to is lower than the value for the other algorithms, this shows that Reinhold detects the clumps to be smaller than the other algorithms, this gives a consistently lower flux value for the Reinhold results.

A constant integrated flux value was given to each of the input clumps. Not only did the results output a lower flux value but it was not constant. Therefore any further analysis undertaken based upon the results from these algorithms could be incorrect. These further analyses could be to find the clump mass function, which depends upon the integrated flux of each clump. With lower integrated flux values the clump mass function plot would appear different and possibly give a different value for it. Mookerjea et al. (2004) and Schneider & Brooks (2004) calculated the clump mass function using results from ClumpFind and Gauss-Clumps, while their results gave different values for the number of detected clumps and the range of fluxes the obtained clump mass functions were similar. From my constant flux plots, Figures 5.10 - 5.17, this appears not to be the case. Though my results of the algorithms may be different from each other, there is a strong similarity between them. In my investigation the clumps were all Gaussian in profile yet in Mookerjea et al. (2004) and Schneider & Brooks (2004) real data was used showing that the Gaussians are not a perfect representation of the real clumps. Figure 5.21 shows how the clump mass function plot could be affected due to the clump flux value being detected at a value lower than its true one as this investigation has shown. The low flux clumps would be calculated to be low mass clumps and high flux clumps, high mass. The low flux clumps are produced by clumps that are either large with a low peak or small with a high peak. The large flat clumps would be greatly affected by the clump cut-off and their determined flux would be much lower than the true value, the small high peak clump would be affected in the same way but less so. The high flux clumps are either large with a high peak or small with a very high peak, the large clumps would be affected by the cut off but not as much as than with the low flux clumps, the small clumps would not be affected much by the cut-off. The low flux clumps would be greatly affected by this and deemed to have a flux and therefore mass lower than true, the high flux clumps would be only marginally affected. This is shown in Figure 5.21 where the red line represents the true values and the green line the detected values. This could produce a false turnover in the mass function, but more investigation with more realistic spectrum of clumps would be required to prove this hypothesis. My results although giving a different value for the detected flux over the input flux do so constantly and therefore the error is always present, though be it the amount of error can vary dependant on the size and peak value of the clump. From my results an error of a factor of two can be seen, this is well within the margin of error of the clump mass function which has an error in the order of a factor of ten. Therefore the error within the algorithms should not have an effect on the mass function.

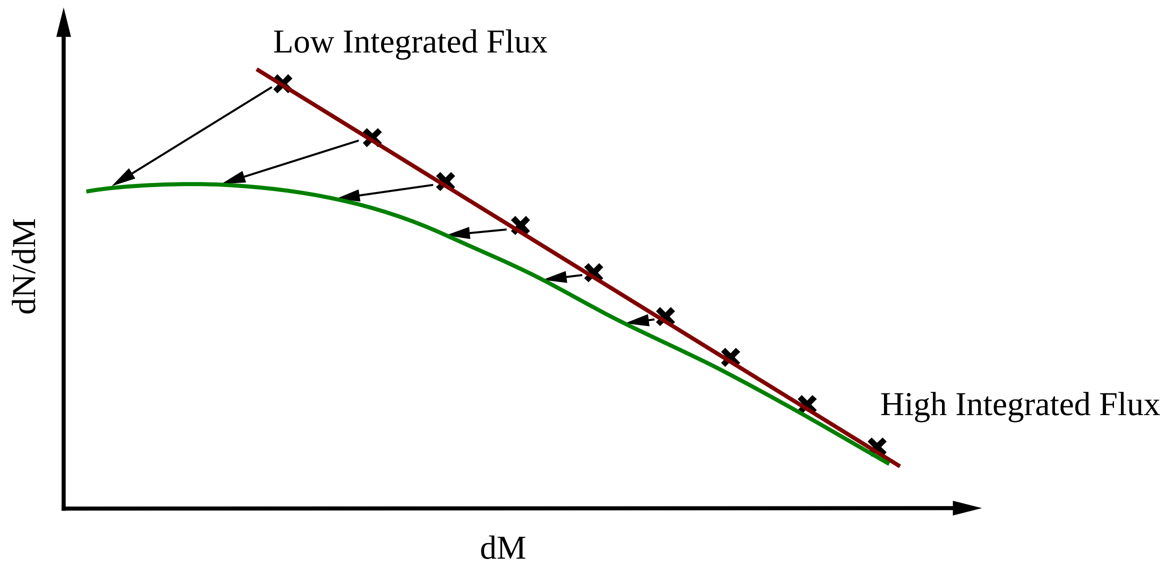


Figure 5.21: Graphical representation of effect on the clump mass function due to lower detected flux values. The red line shows the true mass function, the green line shows what we believe a lower flux detection would give. Both axis have a logarithmic scale.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

The investigation examined the clumpfinding algorithms of the CUPID package in order to understand how each one worked, compare each algorithm to the others, and from there determine any potential flaws or systematic bias that each algorithm may have.

The parameters for the clump detection algorithms determine what the algorithms look for when analysing an image. Changing many of these can have a large effect on the number of clumps detected and on their shape and size. To this end a user must carefully consider as to what the parameter values should be before CUPID is run on the images as the wrong value could give incorrect results. There is no value for these parameters that are optimum for every input image but the default values or similar appear appropriate for most images. From this investigation I found that some of the algorithms parameters can have a large effect on the number and profile of clumps detected, some of these are expected, such as; DeltaT, TLow, and MinDip where as the parameter value increases the number of clumps decreases as does their size. With some parameters there were large changes if the value was altered from the default such as in MaxJump and CAIterations. These values should be left at their default values to prevent irregular results. A few of the parameters caused little or no change in the detection of the clumps, this could have only been the case with the data and range of parameter values used, these parameters were; FlatSlope, NPad, Thresh, and FixClumpIterations. Since changing these had minimal effect then these can be changed to suit the data if required but there is no necessity for them to be changed from the default values.

For each of the algorithms, the detection of the peak pixel and its value was similar when compared to the other algorithms, though with minor discrepancies for GaussClumps. GaussClumps peak values were consistently lower than the other algorithms. The size and integrated flux of the detected clumps also show a large variation between the algorithms with GaussClumps detecting many of the clumps to be larger than the other algorithms. This is likely to be due to GaussClumps ability to associate a single pixel with multiple clumps, as it fits overlapping Gaussians to the image rather than segmenting the pixel values. Despite this

FellWalker often detected a higher integrated flux value than the other algorithms.

ClumpFind has been seen to break large clumps up into multiple clumps. This breakup was sometimes either into multiple equally sized clumps, or more often a single large clump with many smaller satellite clumps around the edge of the larger clump. The detected location of the peak of the main clumps is accurate though the size of the clumps can be smaller due to the breaking procedure and therefore lower fluxes are detected than are actually present in the image. FellWalker appears to give an accurate number of detected clumps and profile of clumps though detecting the integrated flux to be lower than it should be. GaussClumps also breaks up some larger clumps in a similar manner to ClumpFind. CUPID does not output an image showing the clump indexing when GaussClumps is used making it more difficult to see the clumps that the algorithm has detected. There is a way for the results to be displayed as a number of ellipses instead, making a clump profile more distinct from the surrounding clumps, but that was not possible at the time of this investigation. Reinhold clumps can be erratically shaped and are regularly smaller than when detected by other algorithms causing a large variation in the flux values of the detected clumps. The detected location of the clumps with ClumpFind and FellWalker do match with that of Reinhold. ClumpFind, FellWalker, and Reinhold all detect lower clump flux values due to the trailing edges of the clumps being cut off and ignored. GaussClumps does not do this as obviously though the 450μm images do give a lower flux than the 850μm images, GaussClumps may not be as correct with non Gaussian clump profiles.

The completeness of the algorithms has been shown to follow the expected trends with low peak clumps not being detected and all high peak clumps being detected. Providing a large enough source size the algorithms start to make detections at as low as four times the RMS value of the image and 100% completeness being obtained at as low as six times the RMS value of the image. The most ideal result trend is given by FellWalker since after making 100% completeness the detections flatten off and no additional detections are made. ClumpFind and Reinhold giving near ideal trend, though reaching a point higher than 100% completeness therefore clearly making additional, spurious detections. GaussClumps though generally following the trend also gave erratic results with the occasional detection of further additional sources.

If constant flux value clumps are input into the algorithms the flux values are not correctly recovered. This is more so for faint, large clumps in which there is an increasing underestimation of their integrated flux as the clump becomes fainter and larger. These results could affect any further analysis done based upon them, such as the determination of the clump mass function. Though as previously determined, the effect of this error is well within the margin of error for the determination of the mass function. Therefore the incorrect recovery of the input fluxes should only have a minimal effect.

The results from this thesis indicate that previous investigations using either ClumpFind or GaussClumps may have a number of inaccuracies in either the detection of the clump or the proper assessment as to the clump profile. An accurate detection of clumps is defined as detecting all the clumps within the image but not detecting any additional clumps, also correctly detecting their peak and central locations along with an accurate determination of the clump edges.

ClumpFind and GaussClumps have been used for the detection of clumps in a number of investigations without any determination as to the accuracy of the results output from the algorithm. Fellwalker and Reinhold have remained unused due to their relative newness and lack of supporting literature. I have shown that ClumpFind and GaussClumps may not work as effectively as has been previously assumed, that FellWalker is more than able to compete with the other algorithms, and that the Reinhold algorithm although able to detect the clump peak accurately is less able to determine a correct profile for the clump.

This investigation does not give a definitive quantitative value by which the output results for each algorithm can be corrected by to obtain correct integrated flux values. It does show that these algorithms require further investigation as to their ability to correctly detect clumps. In addition it has shown that GaussClumps is able to determine the integrated flux of a clump more correctly than the other algorithms. However this is only the case with uncrowded Gaussian sources and on occasions the detection of a clump can be erratic. FellWalker although giving a lower value for the integrated flux was more consistent with the detection of the clumps to the end where the inserted clumps were detected and not broken up. Due to this it is my belief that FellWalker to be a better Clumpfinding algorithm over ClumpFind or Reinhold (since ClumpFind and Reinhold also give low flux values but are more erratic in the clump detection process). If the issue of the lower flux detection in FellWalker were to be resolved then it may also prove a better algorithm than GaussClumps for the process of clump detection.

If a clumpfinding algorithm was required for the process of clump detection this investigation shows that GaussClumps would be able to determine the integrated flux of the clumps more correctly than the other algorithms. This is only known to be true for isolated and perfectly formed Gaussian sources. Data such as that in SCAMPS is not necessarily isolated and would not be such a perfectly formed Gaussian. After GaussClumps the next recommended algorithm would be FellWalker. Although FellWalker outputs an integrated flux similar to that of ClumpFind and Reinhold it was more robust during the parameter analysis described in Chapter 3. Altering the parameters for FellWalker, although having an effect on changing the number of the clumps detected (and their size and flux) the changes were found to be less severe and unpredictable than with the other algorithms. After FellWalker would be ClumpFind, ClumpFind was by far the easiest to use with only three parameters but that then allowed little manipulation of the algorithm to compensate for irregularities such as noise spikes, ClumpFind was also found to detect possibly erroneous clumps due to the breaking up of larger clumps as also mentioned in Enoch et al. (2006). Reinhold comes last in recommendation in this investigation due to its clump profiles being irregular on a number of occasions and the consistent breaking up of larger clumps.

6.2 Future Work

This research has provided results that warrant the further investigation into these algorithms. These investigations can build on the procedures used here and from that be able to fully test the CUPID clumpfinding process, determine all the effects of changing the parameters and exactly what each algorithm is able to detect, or more importantly what it is unable to detect. In this investigation any detection was counted, though it is clear that some detections were inaccurate. Further investigations would require a strict definition of a successful detection and also a way of confirming if each detection was completely successful. The ambiguous area is with the breaking up of a single clump into multiple parts. The clump is detected but the number of clumps detected is inaccurate, as is both the size and integrated fluxes of all those clumps.

When investigating the parameters for each of the algorithms the SCAMPS data was used. The exact details of the clumps within these images are not known so it was not possible to say if an algorithm wasn't able to detect all the clumps or if it detected an excess of clumps, and whether the detections accurately determine the size and integrated flux of the clumps when the parameter values are altered. To this end the parameter analysis should be performed on a set of created artificial sources. This would allow the completeness of each parameter to be determined and thus which parameter values are best suited for different types of data or clump.

For the constant input flux investigation a Gaussian clump was used, this gave GaussClumps a huge advantage over the other algorithms. Though real sources may be Gaussian in shape it is doubtful they would be as perfectly formed as the artificial clumps used here. To remove this bias different, more realistic clump profiles should be used e.g. elliptical Gaussians (circular ones were used in this investigation), a Gaussian with an additional flatter Gaussian placed on top (this is what would be observed due to the error beam of the telescope used), or sources with a Bonnor-Ebert (Bonnor, 1956) profile (e.g. Enoch et al. (2006)). These more realistic sources would provide better understanding as to the limitations of each of the algorithms as to the minimum peak value and clump size required for successful detection.

Once the algorithms are understood and their limitations known for the detection of single clumps the next progressive step would be to provide the algorithms with an image containing multiple sources, real or artificial, and observe how correctly they determine the clumps. Having multiple clumps in the image provides confusion, if the clumps are independent of each other there is little difference from the algorithms having to detect a single clump. If the clumps are positioned close enough so that they either touch or overlap then the algorithms would be required to determine as to where the edges of each of these clumps were. If a high and low peak clump are positioned next to each other it is possible that the algorithms may detect the clumps as only one, or it may determine one of the clumps to be larger that it is and thus the other to be smaller.

The SCAMPS images are only a small number of sub-millimetre images that are already available. Once SCUBA2 and Herschel are fully operational the number of such images will increase drastically. This provides a large amount of real data from various instruments to allow continued investigation into the algorithms. The background noise using the newer instruments ought to be different from those in the SCAMPS images (e.g. SCUBA-2 will take data without chopping), so certain effects experienced when investigating the SCAMPS data might be reduced or exaggerated. SCAMPS data are two-dimensional, CUPID is able to handle three-dimensional data, such as that from HARP (Heterodyne Array Receiver Programme). Any differences on how successfully the algorithms would work between these two types of data is not widely known, this clearly warrants further investigation once the algorithms are fully understood using two-dimensional data

All these further investigations will help to fully understand how each algorithm works and when it would be best to use one algorithm instead of the others. If one algorithm is clearly superior to its counterparts then that algorithm could possibly be made the standard algorithm for the detection of clumps. With only one algorithm in use, all investigations requiring clump detection would hopefully obtain results that could be concordant with other investigations of the same data.

References

Berry, D. 2009, CUPID users manual, `http://starlink.jach.hawaii.edu/docs/sun255.htx/sun255.html`

Berry, D. S., Reinhold, K., Jenness, T., & Economou, F. 2007, in Astronomical Society of the Pacific Conference Series, Vol. 376, Astronomical Data Analysis Software and Systems XVI, ed. R. A. Shaw, F. Hill, & D. J. Bell, 425–+

Blitz, L. & Shu, F. H. 1980, ApJ, 238, 148

Blitz, L. & Stark, A. A. 1986, ApJ, 300, L89

Bonnor, W. B. 1956, MNRAS, 116, 351

Coppin, K., Chapin, E. L., Mortier, A. M. J., et al. 2006, MNRAS, 372, 1621

Dent, W. R. F., Hovey, G. J., Dewdney, P. E., et al. 2009, MNRAS, 395, 1805

Emerson, D. T. 1995, in Astronomical Society of the Pacific Conference Series, Vol. 75, Multi-Feed Systems for Radio Telescopes, ed. D. T. Emerson & J. M. Payne, 309–+

Enoch, M. L., Young, K. E., Glenn, J., et al. 2006, ApJ, 638, 293

Friesen, R. K., Di Francesco, J., Shirley, Y. L., & Myers, P. C. 2009, ApJ, 697, 1457

Holland, W. S., Robson, E. I., Gear, W. K., et al. 1999, MNRAS, 303, 659

Jenness, T. & Lightfoot, J. F. 2003, SURF - SCUBA User Reduction Facility, `http://starlink.jach.hawaii.edu/docs/sun216.htx/sun216.html`

Johnstone, D., Wilson, C. D., Moriarty-Schieven, G., et al. 2000, ApJ, 545, 327

Kainulainen, J., Lada, C. J., Rathborne, J. M., & Alves, J. F. 2009, A&A, 497, 399

Lo, N., Cunningham, M. R., Jones, P. A., et al. 2009, MNRAS, 395, 1021

Massi, F., de Luca, M., Elia, D., et al. 2007, A&A, 466, 1013

Mookerjea, B., Kramer, C., Nielbock, M., & Nyman, L.-Å. 2004, A&A, 426, 119

Muñoz, D. J., Mardones, D., Garay, G., et al. 2007, ApJ, 668, 906

Pence, W. 1999, in Astronomical Society of the Pacific Conference Series, Vol. 172, Astronomical Data Analysis Software and Systems VIII, ed. D. M. Mehringer, R. L. Plante, & D. A. Roberts, 487–+

Pineda, J. E., Rosolowsky, E. W., & Goodman, A. A. 2009, ApJ, 699, L134

Rathborne, J. M., Johnson, A. M., Jackson, J. M., Shah, R. Y., & Simon, R. 2009, ApJS, 182, 131

Reid, M. A. & Wilson, C. D. 2005, ApJ, 625, 891

Schneider, N. & Brooks, K. 2004, Publications of the Astronomical Society of Australia, 21, 290

Schneider, N., Stutzki, J., Winnewisser, G., & Block, D. 1998, A&A, 335, 1049

Stutzki, J. & Guesten, R. 1990, ApJ, 356, 513

Taylor, M. B. 2005, in Astronomical Society of the Pacific Conference Series, Vol. 347, Astronomical Data Analysis Software and Systems XIV, ed. P. Shopbell, M. Britton, & R. Ebert, 29–+

Thompson, M. A., Gibb, A. G., Hatchell, J. H., Wyrowski, F., & Pillai, T. 2005, in ESA Special Publication, Vol. 577, ESA Special Publication, ed. A. Wilson, 425–426

Williams, J. P., de Geus, E. J., & Blitz, L. 1994, ApJ, 428, 693

Appendix A

RMS Values for SCAMPS Images

Table A.1: RMS values for the 450μm images as calculated by CUPID, and myself using GAIA.

Image	GAIA RMS	CUPID RMS
g1015-034_SCAMPS_sho_flat.sdf	0.5861974	0.415907094
g1030-015_SCAMPS_sho_flat.sdf	5.08397	2.8236824
g1062-038_SCAMPS_sho_flat.sdf	0.7116784	0.38931021
g1084-259_SCAMPS_sho_flat.sdf	0.3000043	0.25309130
g10987+211_SCAMPS_sho_flat.sdf	0.5840346	0.54225520
g1319+004_SCAMPS_sho_flat.sdf	0.4604184	0.3929368
g1504-068_SCAMPS_sho_flat.sdf	1.180784	0.7709265
g1815-028_SCAMPS_sho_flat.sdf	0.4577619	0.34125437
g1907-027_SCAMPS_sho_flat.sdf	0.6188368	0.5383848
g2346-020_SCAMPS_sho_flat.sdf	0.3327582	0.2240455
g2387-020_SCAMPS_sho_flat.sdf	0.6335199	0.56274576
g2396+015_SCAMPS_sho_flat.sdf	0.5395406	0.47758670
g2538-018_SCAMPS_sho_flat.sdf	0.4352394	0.34644206
g2572+005_SCAMPS_sho_flat.sdf	0.682731	0.4496461
g2728+015_SCAMPS_sho_flat.sdf	0.4141675	0.3804550
g2820-005_SCAMPS_sho_flat.sdf	1.529278	1.13609127
g2880+017_SCAMPS_sho_flat.sdf	1.422018	1.08804514
g2996-002_SCAMPS_sho_flat.sdf	2.24251	1.2923852
g3078-002_SCAMPS_sho_flat.sdf	2.826561	1.22018832
g3128+006_SCAMPS_sho_flat.sdf	1.109922	0.77014498
g3502+035_SCAMPS_sho_flat.sdf	0.7713627	0.71325620
g3558-003_SCAMPS_sho_flat.sdf	1.46233	1.13983379
g6088-013_SCAMPS_sho_flat.sdf	2.677988	2.41155440
g7578+034_SCAMPS_sho_flat.sdf	2.488308	2.0856390
g7844+023_SCAMPS_sho_flat.sdf	0.8184776	0.74991368
g7930+028_SCAMPS_sho_flat.sdf	1.144496	1.03649501
g8087+042_SCAMPS_sho_flat.sdf	0.3863362	0.34130359
g814+023_SCAMPS_sho_flat.sdf	1.599066	1.2774736
g8168+054N_SCAMPS_sho_flat.sdf	0.5825882	0.43488051
g8168+054S_SCAMPS_sho_flat.sdf	1.594306	1.45412212
g867-036_SCAMPS_sho_flat.sdf	0.7731709	0.56888940

Table A.2: RMS values for the 850μm images as calculated by CUPID, and myself using GAIA.

Image	GAIA RMS	CUPID RMS
g1015-034_SCAMPS_lon_flat.sdf	0.07063728	0.02182198
g1030-015_SCAMPS_lon_flat.sdf	0.1337341	0.01444824
g1062-038_SCAMPS_lon_flat.sdf	0.2124847	0.016912319
g1084-259_SCAMPS_lon_flat.sdf	0.04596421	0.01540634
g10987+211_SCAMPS_lon_flat.sdf	0.08888455	0.02518203
g1319+004_SCAMPS_lon_flat.sdf	0.06583121	0.02070719
g1504-068_SCAMPS_lon_flat.sdf	0.1905133	0.0262378
g1815-028_SCAMPS_lon_flat.sdf	0.0881284	0.2000474
g1907-027_SCAMPS_lon_flat.sdf	0.08248799	0.02219707
g2346-020_SCAMPS_lon_flat.sdf	0.078476	0.01534876
g2387-020_SCAMPS_lon_flat.sdf	0.5713682	0.01189613
g2396+015_SCAMPS_lon_flat.sdf	0.1085911	0.02016414
g2538-018_SCAMPS_lon_flat.sdf	0.04614707	0.012163180
g2572+005_SCAMPS_lon_flat.sdf	0.05745599	0.01208666
g2728+015_SCAMPS_lon_flat.sdf	0.0522497	0.01648540
g2820-005_SCAMPS_lon_flat.sdf	0.09170613	0.01770435
g2880+017_SCAMPS_lon_flat.sdf	0.06345583	0.01455768
g2996-002_SCAMPS_lon_flat.sdf	0.1078535	0.0127641
g3078-002_SCAMPS_lon_flat.sdf	0.1497014	0.01794412
g3128+006_SCAMPS_lon_flat.sdf	0.06359545	0.01335445
g3502+035_SCAMPS_lon_flat.sdf	0.0724537	0.02308213
g3558-003_SCAMPS_lon_flat.sdf	0.06673279	0.01457241
g6088-013_SCAMPS_lon_flat.sdf	0.06368237	0.01739713
g7578+034_SCAMPS_lon_flat.sdf	0.07278302	0.00909111
g7844+023_SCAMPS_lon_flat.sdf	0.07114823	0.02170256
g7930+028_SCAMPS_lon_flat.sdf	0.045052	0.01293240
g8087+042_SCAMPS_lon_flat.sdf	0.05948586	0.15959118
g814+023_SCAMPS_lon_flat.sdf	0.1052001	0.03125025
g8168+054N_SCAMPS_lon_flat.sdf	0.0903954	0.01339822
g8168+054S_SCAMPS_lon_flat.sdf	0.1289309	0.02606263
g867-036_SCAMPS_lon_flat.sdf	0.08594918	0.0229782

Appendix B

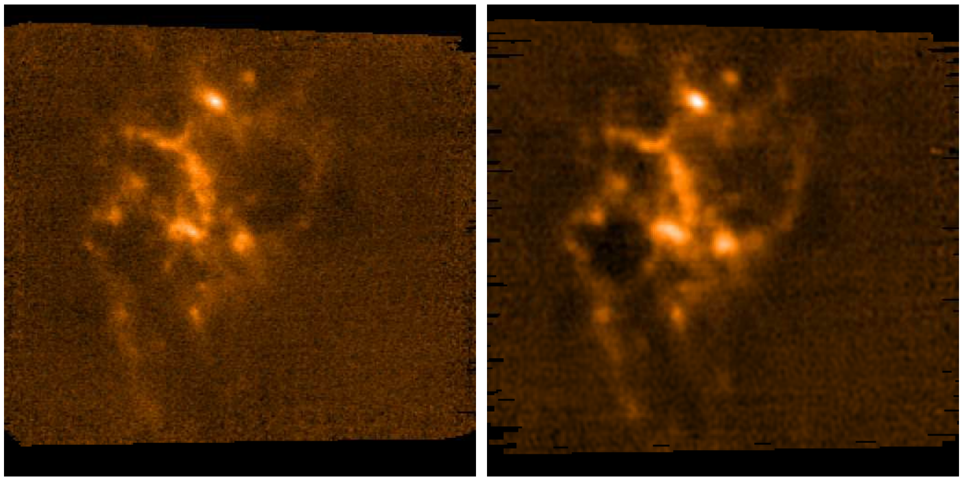
Artificial Clump IDL Code

```
PRO false_clump1000

imagesho = readfits('g1084-259_SCAMPS_sho_flat.fits', NaNVALUE=-100);
sky, imagesho,av,std;
std1=std;

imagelon = readfits('g1084-259_SCAMPS_lon_flat.fits', NaNVALUE=-100);
sky, imagelon,av,std;
std2=std;

openw, lun, 'false_clump_locations.txt', /get_lun;

c=680;
d=680;

printf, lun, 'run FWHM RMS Horizontal Vertical 450' ;

for l=1,1000,1 do begin;
lAsString= strtrim(l,2);

FWHM=RANDOMU(seed);
v=(FWHM* 4)+6;
vAsString= strtrim(v,2);
x=(c/(1.133*(v^2)));
xAsString= strtrim(x,2);
repeat begin;
horizontal1=RANDOMU(seed);
horizontal2=(horizontal1* 216)+20;
vertical1=RANDOMU(seed);
vertical2=(vertical1*176)+40;
endrep until [horizontal2 lt 155 and vertical2 lt 85] or [horizontal2
lt 105 and vertical2 gt 85] or [horizontal2 gt 105 and vertical2 gt 155]
or [horizontal2 gt 155 and vertical2 lt 155];
printf, lun, l, v, x, horizontal2, vertical2;
array = psf_Gaussian(Npixel=256, FWHM=v, NDIMEN=2,
Centroid=[horizontal2, vertical2]);
```

```
a = std1 * x;
array2 = array * a;
outputsho = imagesho + array2;

writefits, '/data3/mwatson/FalseClumps/450' + '/g1084259_sho_run'
+ strtrim(l,2)+ '_rms' + strtrim(x,2) + '_FWHM' + strtrim(v,2)
+ '_flat.fits', outputsho;

endfor;


printf, lun, '#comparison run run FWHM RMS Horizontal Vertical 850';

for m=1,1000,1 do begin;
lAsString= strtrim(m,2);

FWHM2=RANDOMU(seed);
w=(FWHM2* 4)+6;
wAsString= strtrim(w,2);
y=(d/(1.133*(w^2)));
yAsString= strtrim(y,2);
repeat begin;
horizontal3=RANDOMU(seed);
horizontal4=(horizontal3* 216)+20;
vertical3=RANDOMU(seed);
vertical4=(vertical3*176)+40;
endrep until [horizontal4 lt 160 and vertical4 lt 85] or [horizontal4
lt 85 and vertical4 gt 85] or [horizontal4 gt 85 and vertical4 gt 175]
or [horizontal4 gt 160 and vertical4 lt 175];
printf, lun, m, w, y, horizontal4, vertical4;
array = psf_Gaussian(Npixel=256, FWHM=w, NDIMEN=2,
Centroid=[horizontal4, vertical4]);

b = std2 * y;
array2 = array * b;
outputlon = imagelon + array2;

writefits, '/data3/mwatson/FalseClumps/850' + '/g1084259_lon_run'
+ strtrim(m,2) + '_rms' + strtrim(y,2) + '_FWHM' + strtrim(w,2)
+ '_flat.fits', outputlon;

endfor;

free_lun, lun;

end;
```

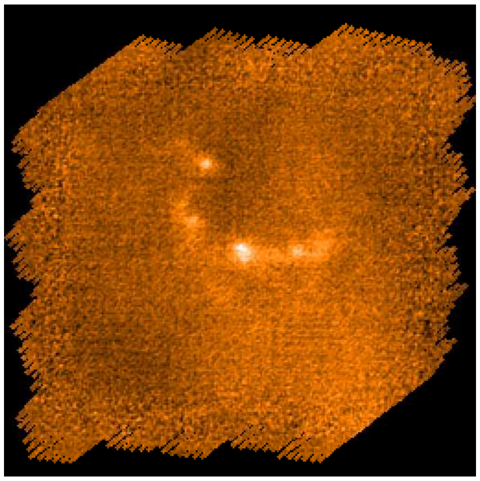
Appendix C

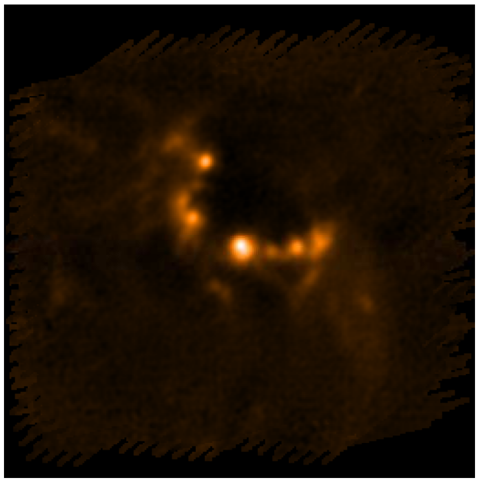
SCAMPS Image Catalogue

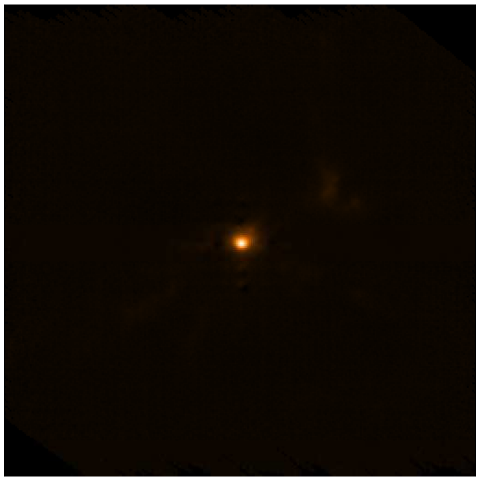


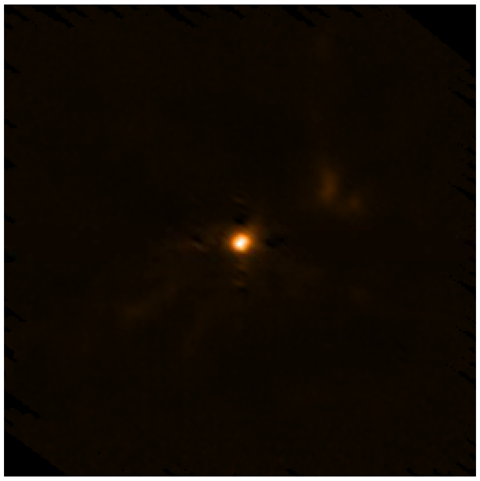
g1015-034_SCAMPS_sho_flat g1015-034_SCAMPS_lon_flat

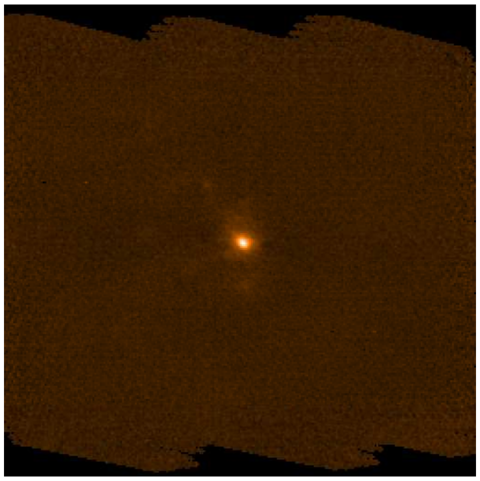
Figure C.1: Catalogue of the SCAMPS images used in this investigation.


g1030-015_SCAMPS_sho_flat


g1030-015_SCAMPS_lon_flat


g1062-038_SCAMPS_sho_flat


g1062-038_SCAMPS_lon_flat


g1084-259_SCAMPS_sho_flat

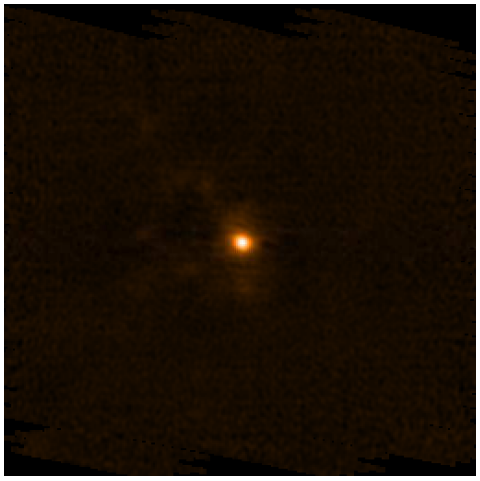
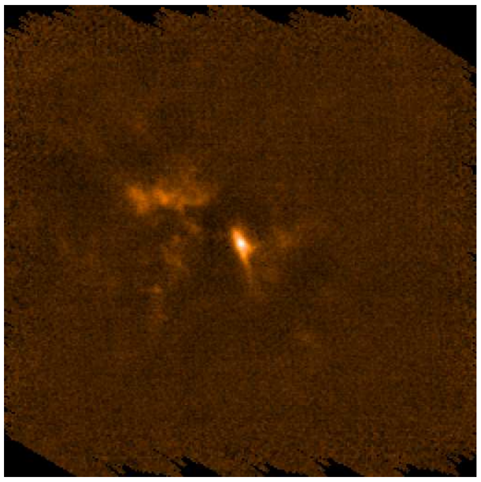
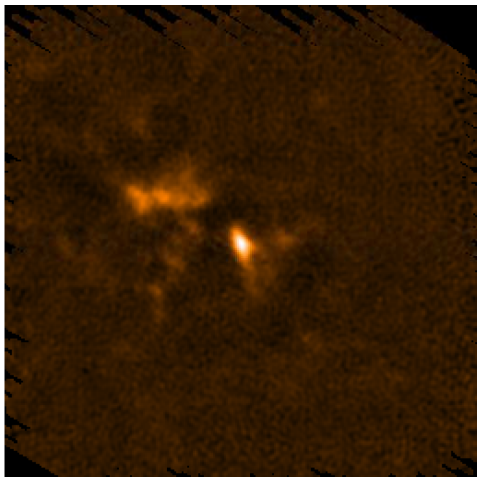
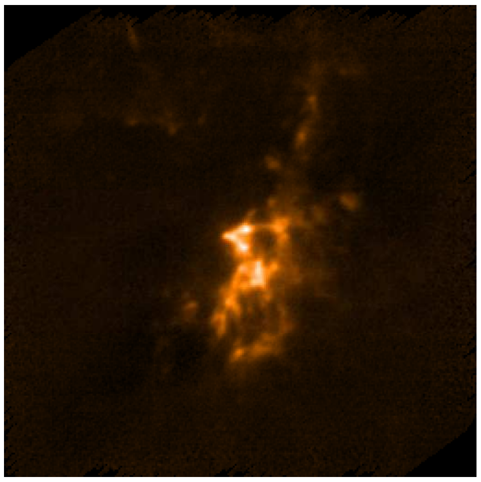
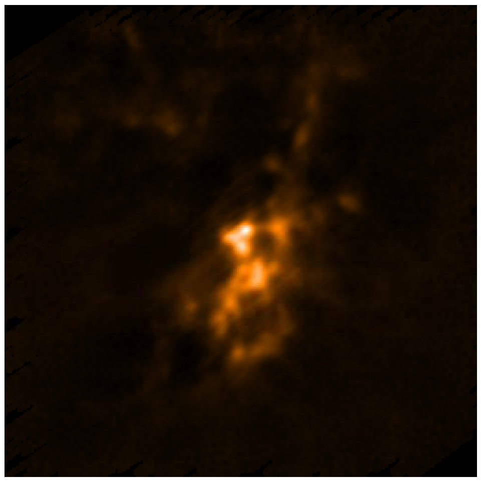

g1084-259_SCAMPS_lon_flat

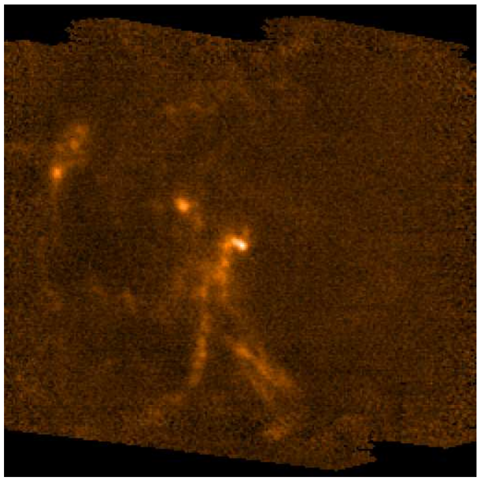
Figure C.1: Catalogue of the SCAMPS images continued.


g1319+004_SCAMPS_sho_flat


g1319+004_SCAMPS_lon_flat


g1504-068_SCAMPS_sho_flat


g1504-068_SCAMPS_lon_flat


g1815-028_SCAMPS_sho_flat

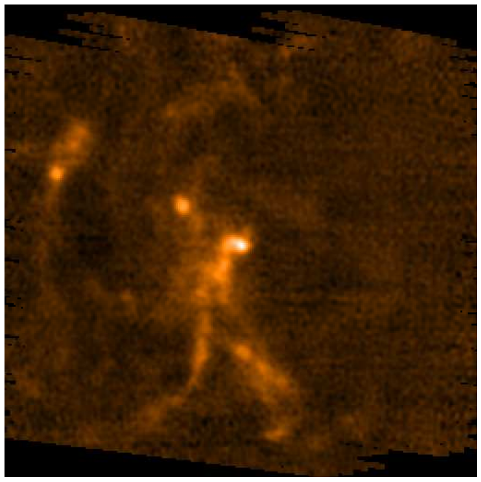
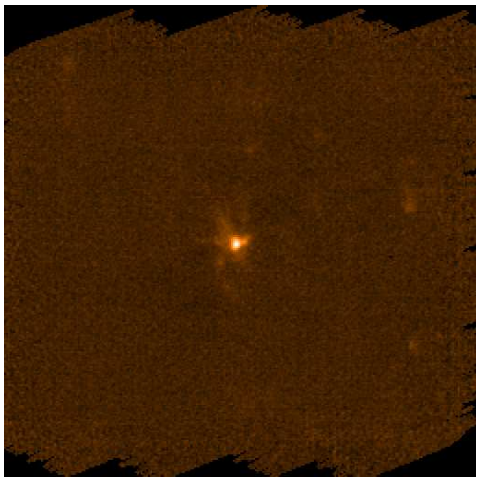
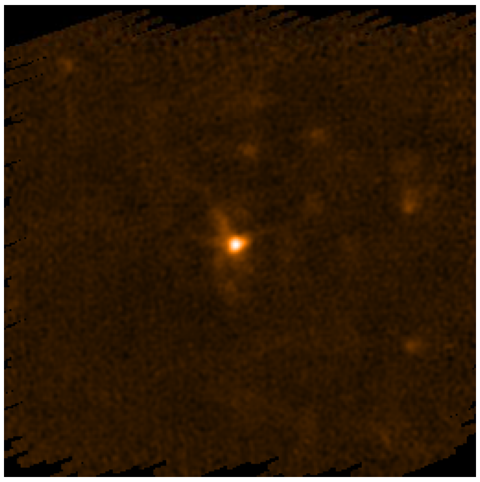

g1815-028_SCAMPS_lon_flat

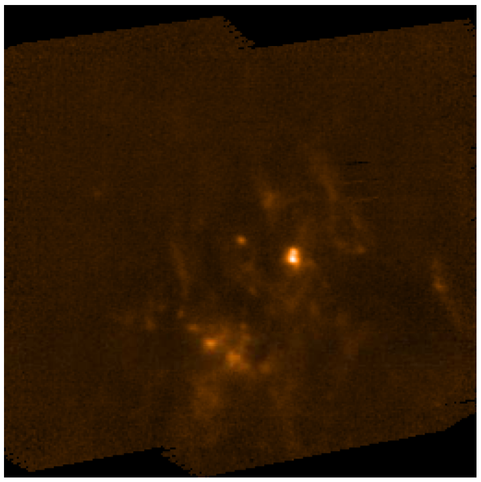
Figure C.1: Catalogue of the SCAMPS images continued.



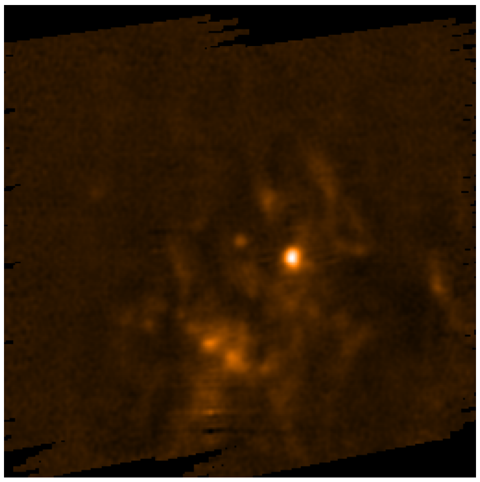
g1907-027_SCAMPS_sho_flat



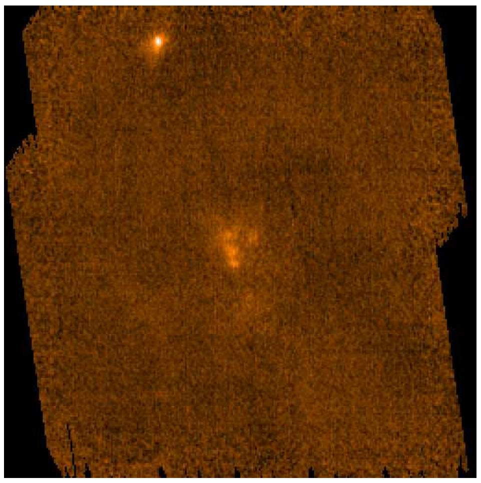
g1907-027_SCAMPS_lon_flat



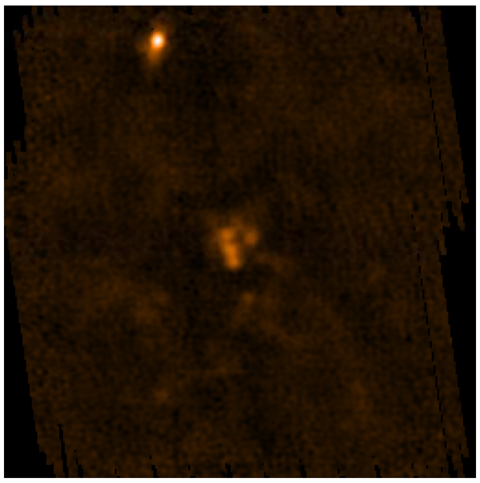
g2346-020_SCAMPS_sho_flat



g2346-020_SCAMPS_lon_flat

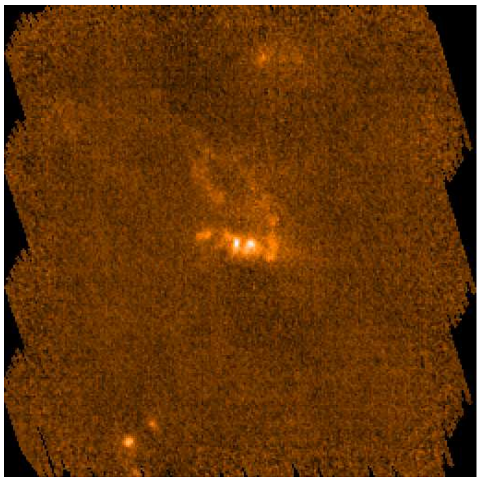


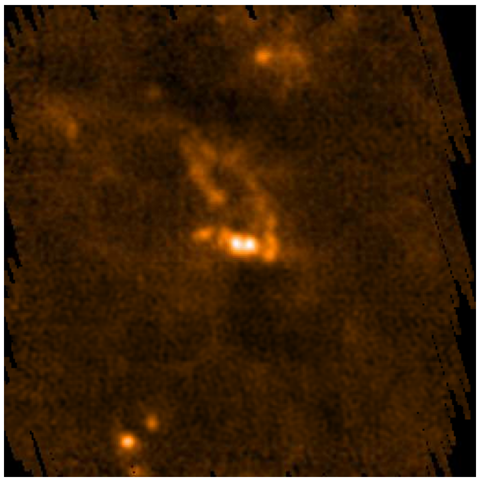
g2387-012_SCAMPS_sho_flat

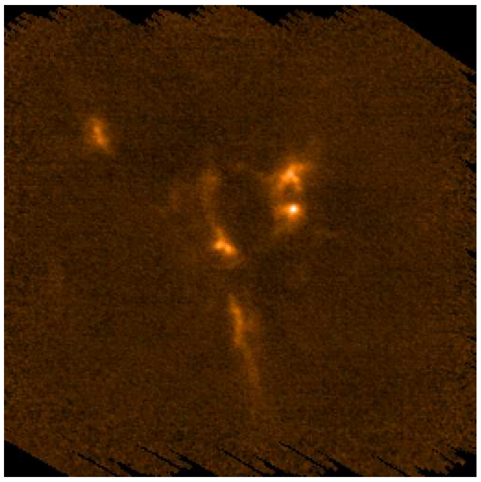


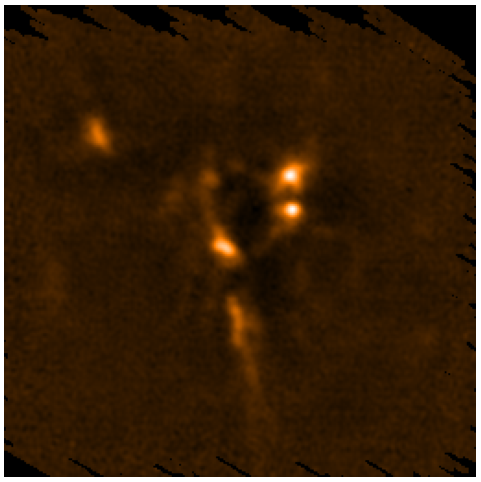
g2387-012_SCAMPS_lon_flat

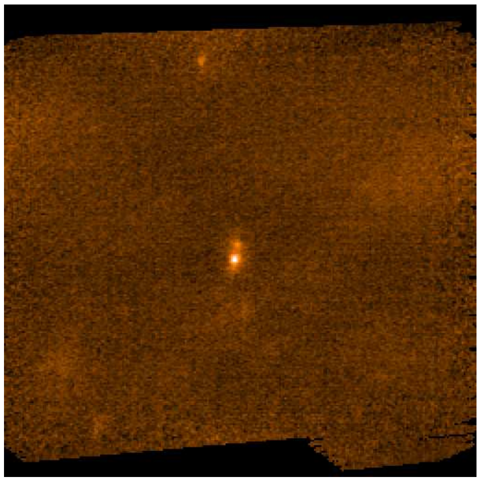
Figure C.1: Catalogue of the SCAMPS images continued.


g2396+015_SCAMPS_sho_flat


g2396+015_SCAMPS_lon_flat


g2538-018_SCAMPS_sho_flat


g2538-018_SCAMPS_lon_flat


g2572+005_SCAMPS_sho_flat

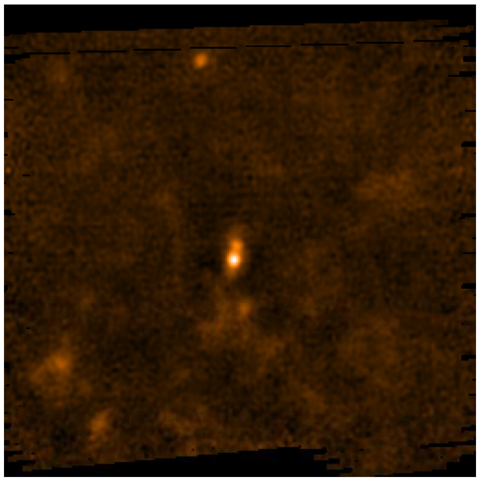
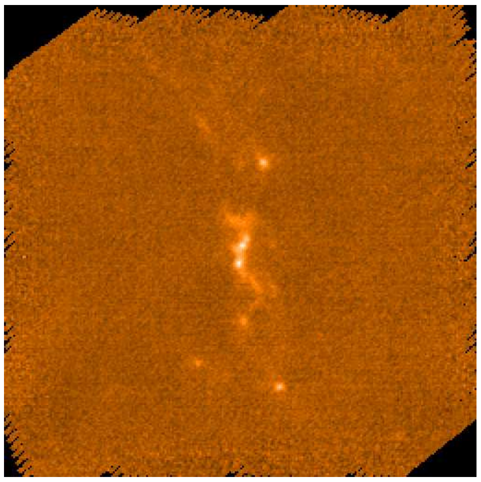
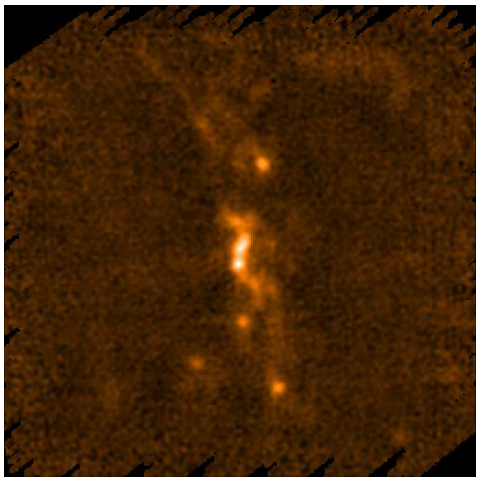
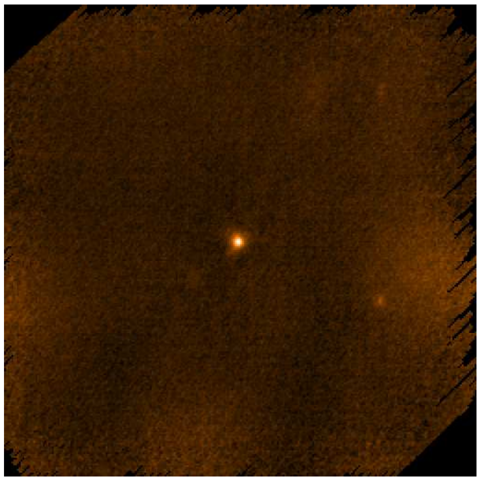
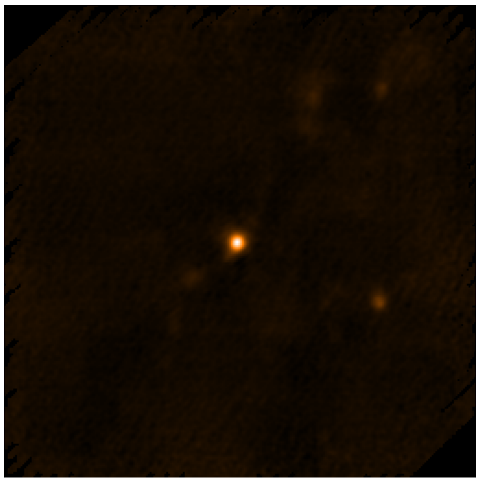

g2572+005_SCAMPS_lon_flat

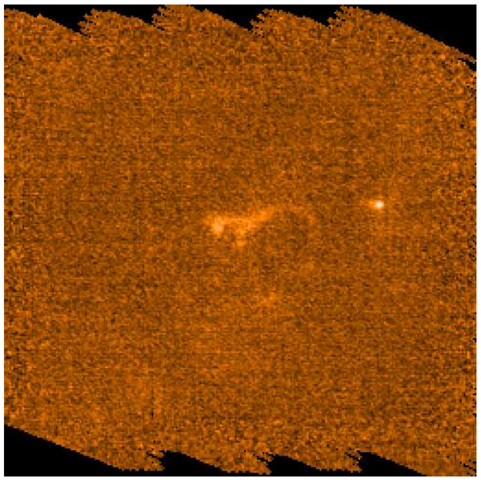
Figure C.1: Catalogue of the SCAMPS images continued.


g2728+015_SCAMPS_sho_flat


g2728+015_SCAMPS_lon_flat


g2820-005_SCAMPS_sho_flat


g2820-005_SCAMPS_lon_flat


g2880+017_SCAMPS_sho_flat

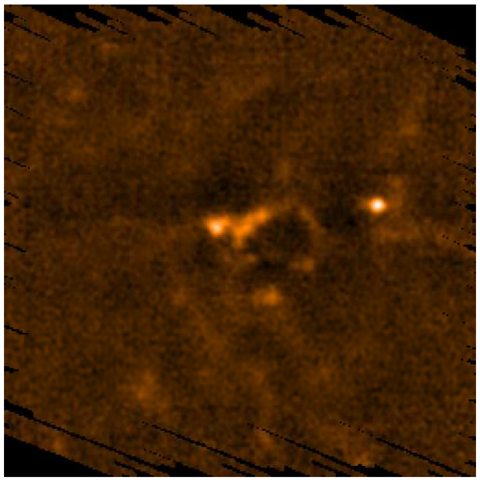
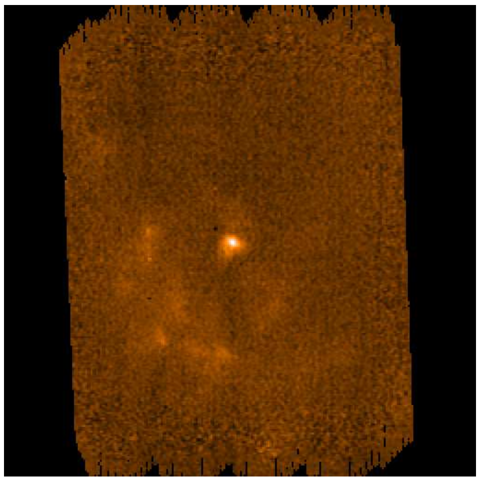
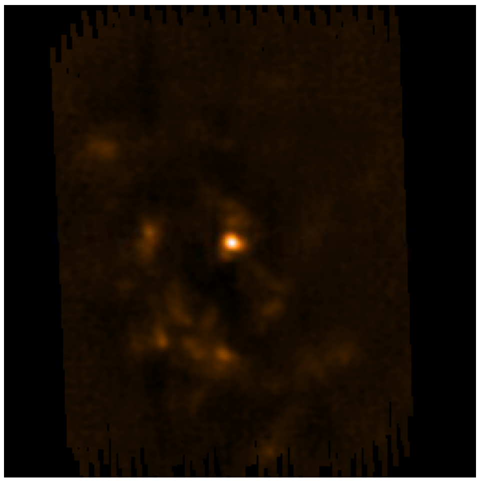
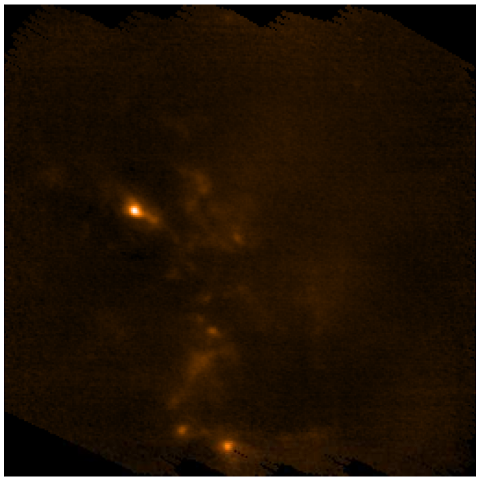
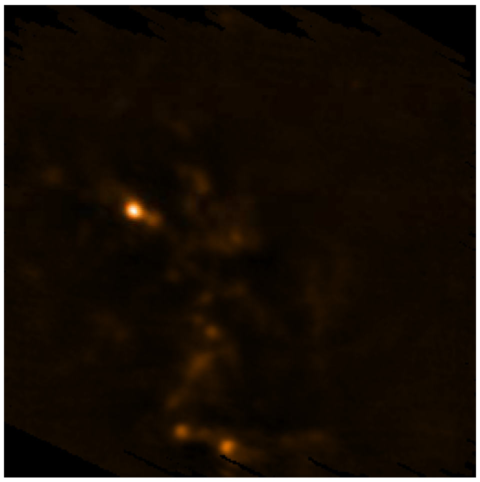

g2880+017_SCAMPS_lon_flat

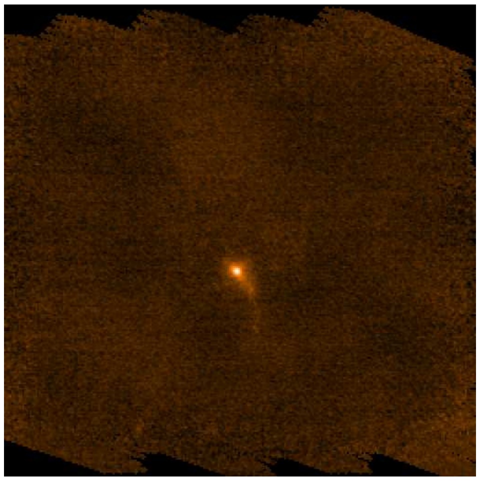
Figure C.1: Catalogue of the SCAMPS images continued.


g2996-002_SCAMPS_sho_flat


g2996-002_SCAMPS_lon_flat


g3078-002_SCAMPS_sho_flat


g3078-002_SCAMPS_lon_flat


g3128+006_SCAMPS_sho_flat

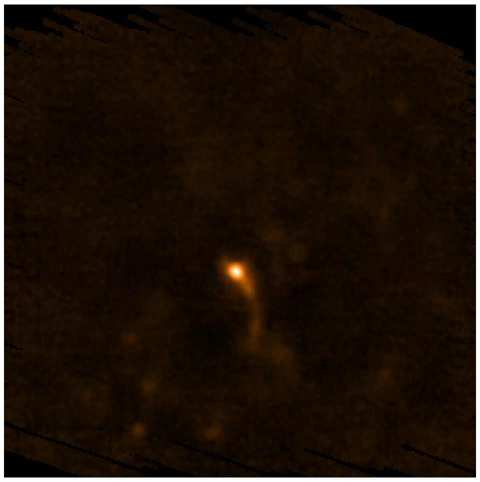
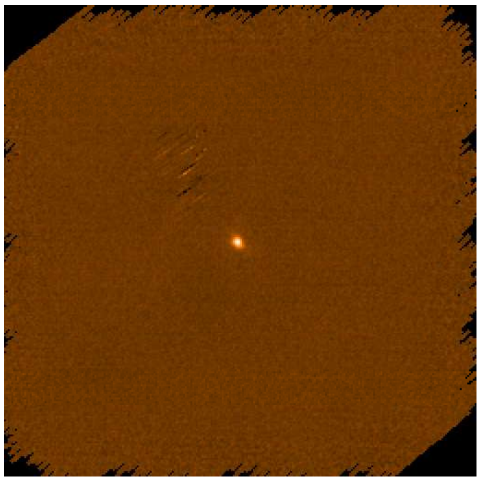
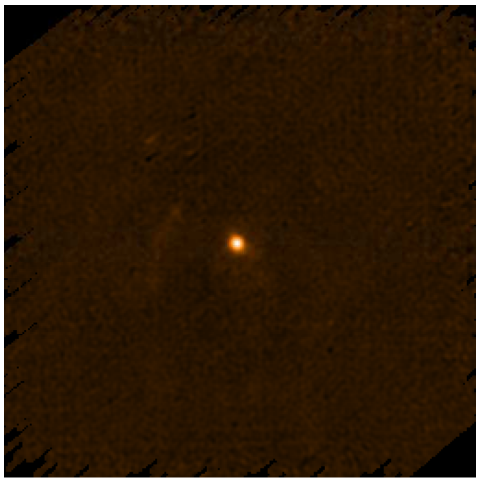
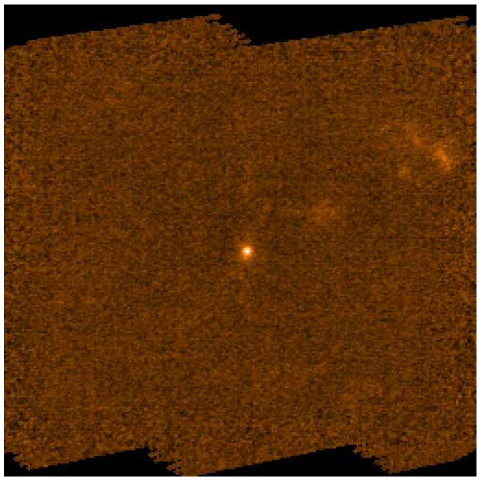
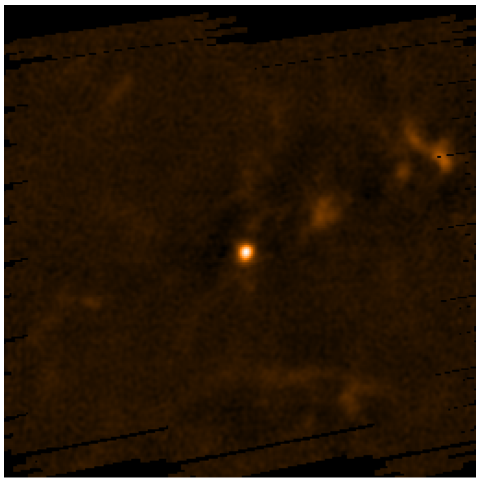

g3128+006_SCAMPS_lon_flat

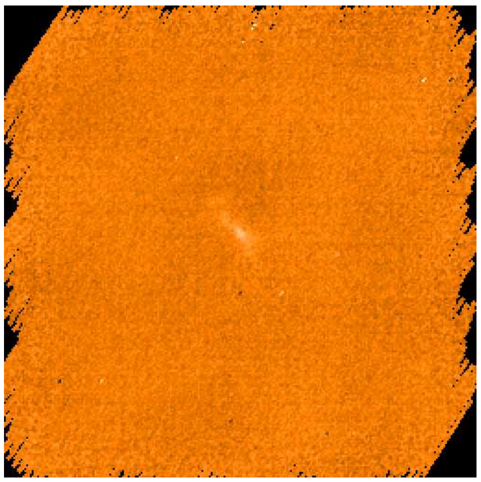
Figure C.1: Catalogue of the SCAMPS images continued.


g3502+035_SCAMPS_sho_flat


g3502+035_SCAMPS_lon_flat


g3558-003_SCAMPS_sho_flat


g3558-003_SCAMPS_lon_flat


g6088-013_SCAMPS_sho_flat

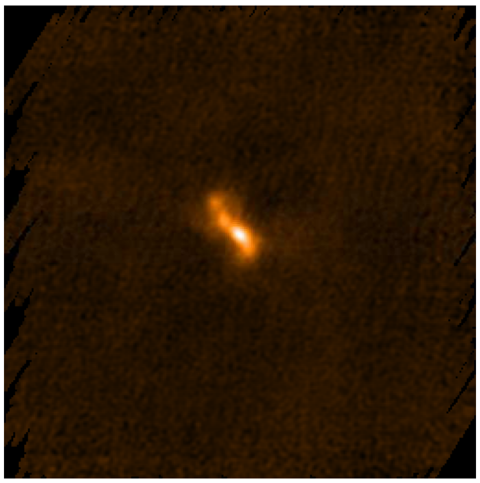
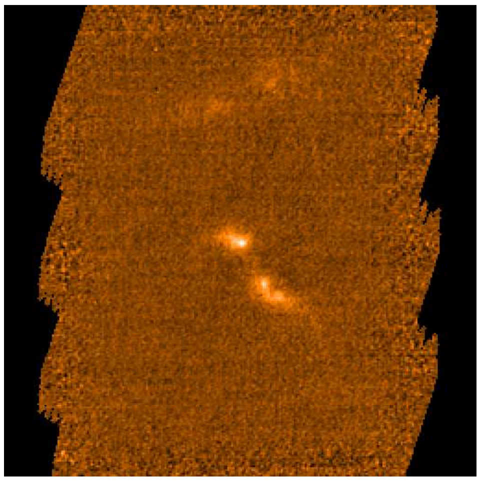
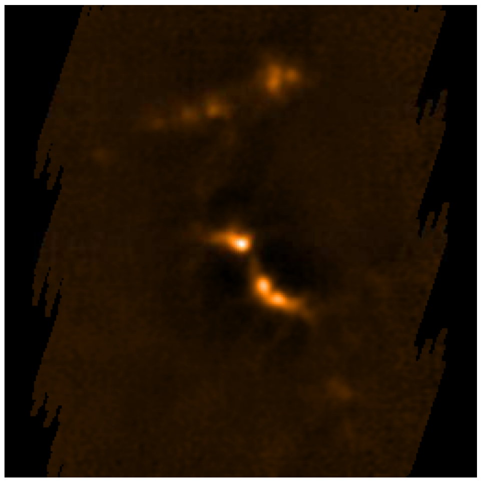
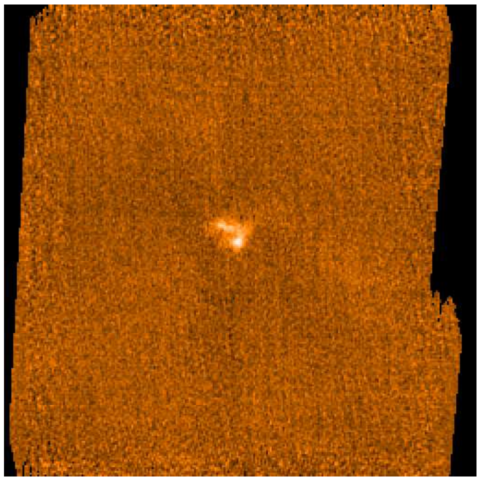
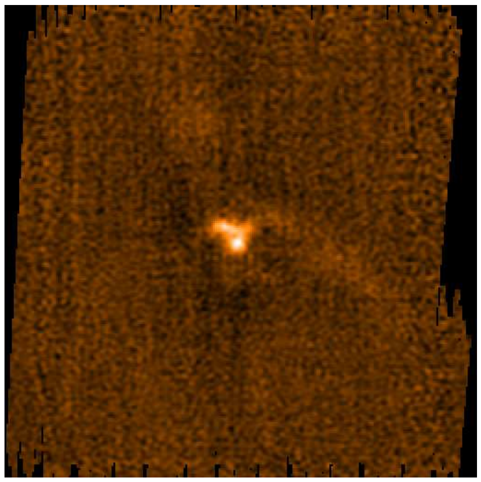

g6088-013_SCAMPS_lon_flat

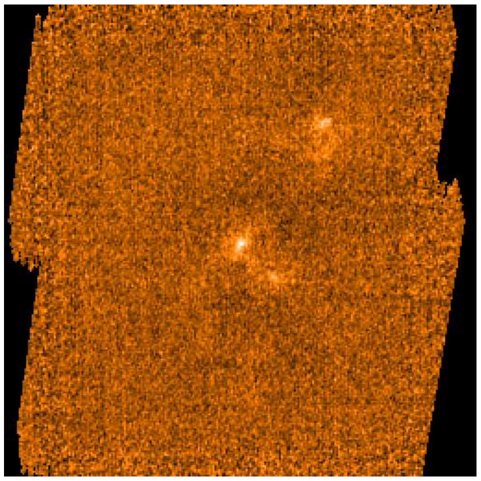
Figure C.1: Catalogue of the SCAMPS images continued.


g7578+034_SCAMPS_sho_flat


g7578+034_SCAMPS_lon_flat


g7844+023_SCAMPS_sho_flat


g7844+023_SCAMPS_lon_flat


g7930+028_SCAMPS_sho_flat

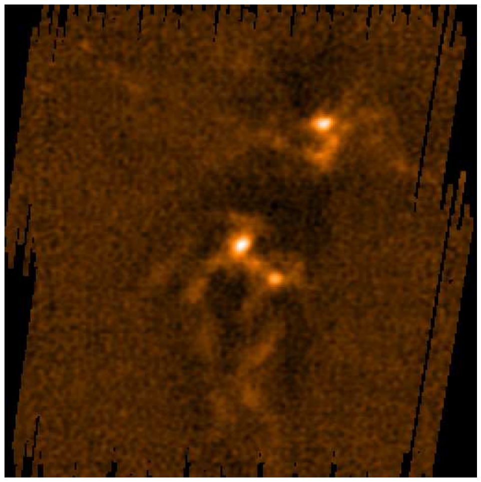
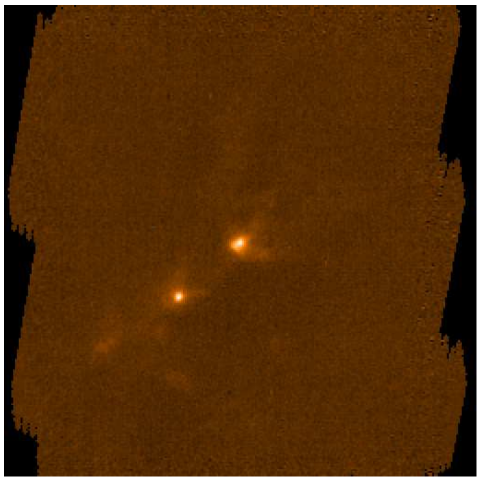
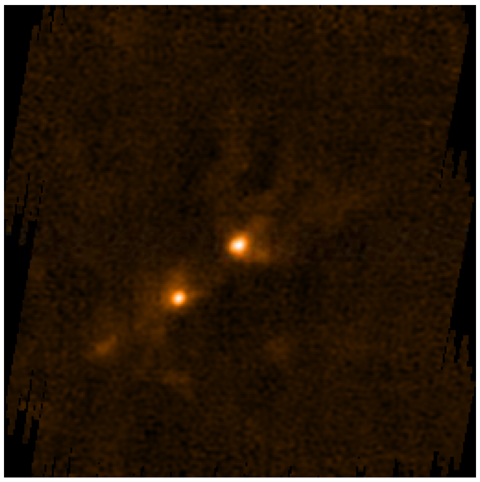
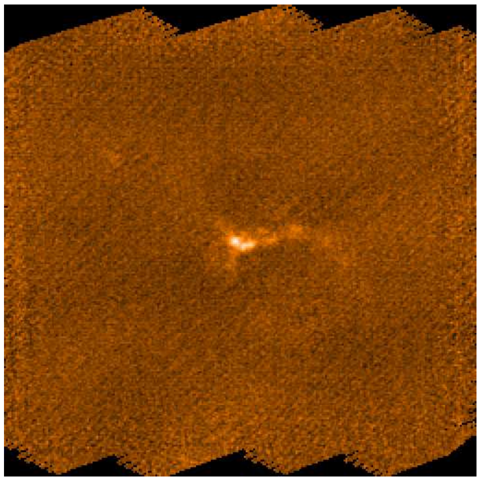
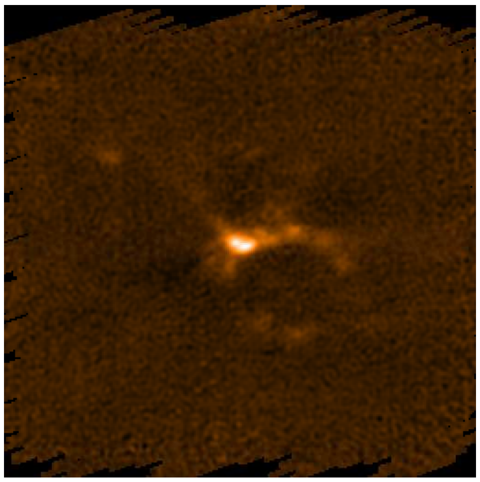

g7930+028_SCAMPS_lon_flat

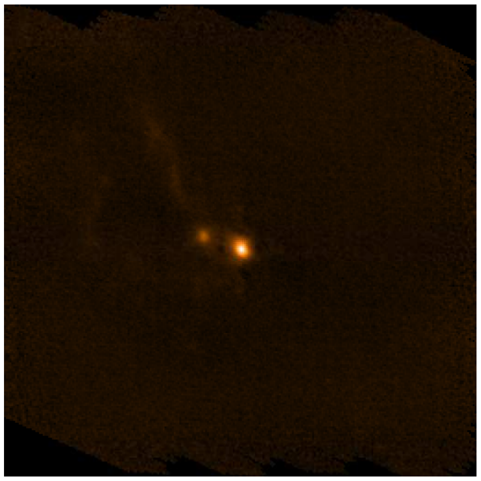
Figure C.1: Catalogue of the SCAMPS images continued.


g8087+042_SCAMPS_sho_flat


g8087+042_SCAMPS_lon_flat


g814+023_SCAMPS_sho_flat


g814+023_SCAMPS_lon_flat


g867-036_SCAMPS_sho_flat

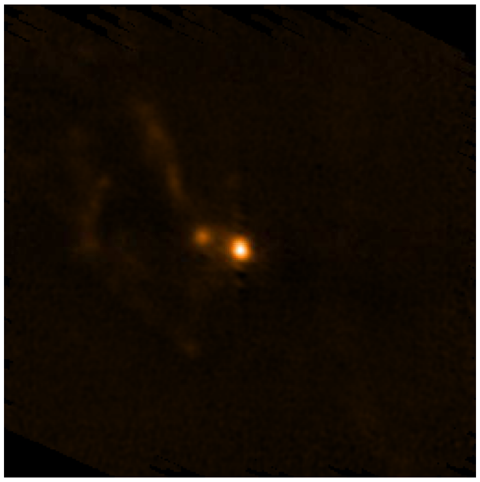
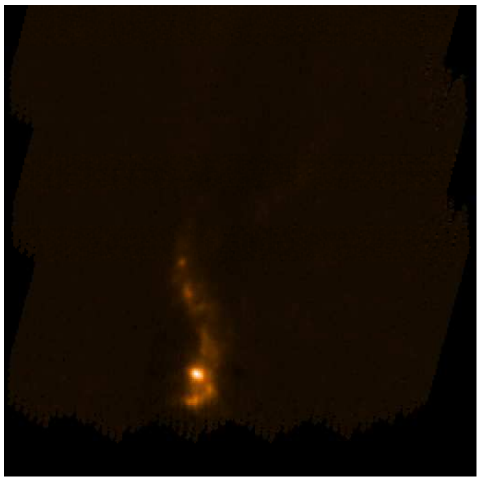
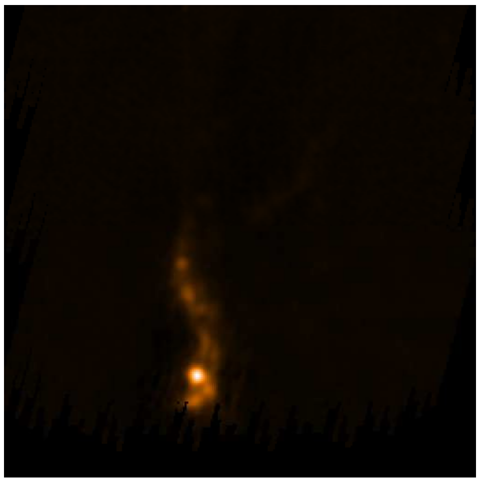
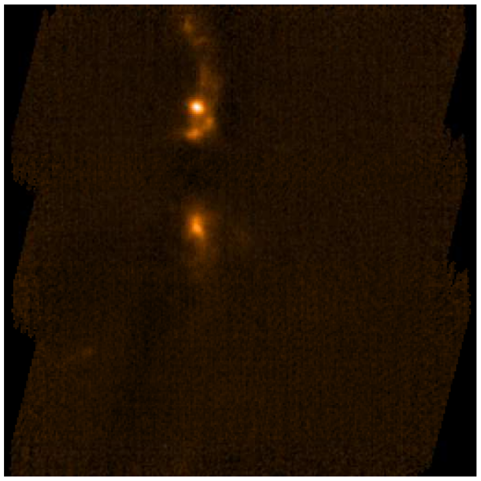
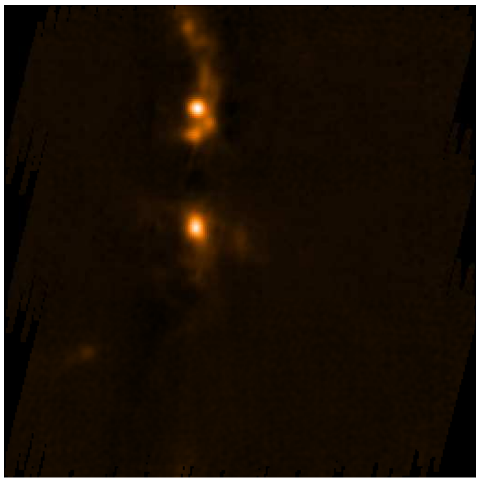

g867-036_SCAMPS_lon_flat

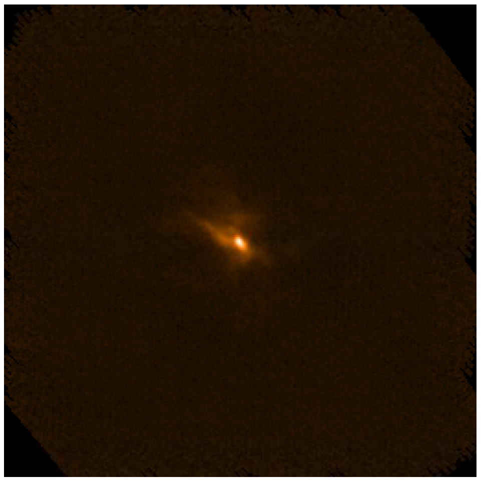
Figure C.1: Catalogue of the SCAMPS images continued.


g8168+054N_SCAMPS_sho_flat


g8168+054N_SCAMPS_lon_flat


g8168+054S_SCAMPS_sho_flat


g8168+054S_SCAMPS_lon_flat


g10987+211_SCAMPS_sho_flat

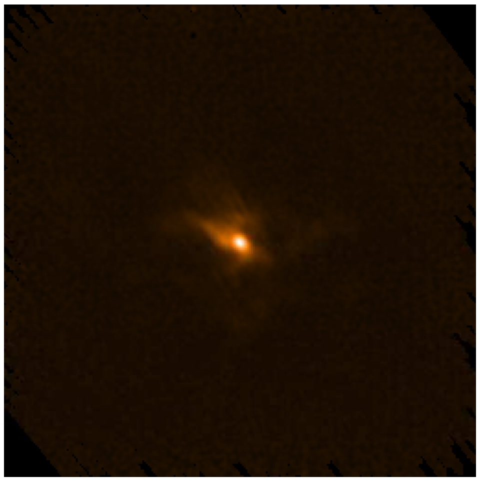

g10987+211_SCAMPS_lon_flat

Figure C.1: Catalogue of the SCAMPS images continued.