Carving out the low surface brightness universe with NoiseChisel

Mohammad Akhlaghi[1,2]

[1]Instituto de Astrofísica de Canarias, C/ Vía Láctea, 38200 La Laguna, Tenerife, Spain.
email: `mohammad@akhlaghi.org`

[2]Facultad de Física, Universidad de La Laguna, Avda. Astrofísico Fco. Sánchez s/n, 38200 La Laguna, Tenerife, Spain.

Abstract. NoiseChisel is a program to detect very low signal-to-noise ratio (S/N) features with minimal assumptions on their morphology. It was introduced in 2015 and released within a collection of data analysis programs and libraries known as GNU Astronomy Utilities (Gnuastro). Over the last ten stable releases of Gnuastro, NoiseChisel has significantly improved: detecting even fainter signal, enabling better user control over its inner workings, and many bug fixes. The most important change may be that NoiseChisel's segmentation features have been moved into a new program called Segment. Another major change is the final growth strategy of its true detections, for example NoiseChisel is able to detect the outer wings of M51 down to S/N of 0.25, or 28.27 mag/arcsec^2 on a single-exposure SDSS image (r-band). Segment is also able to detect the localized HII regions as "clumps" much more successfully. Finally, to orchestrate a controlled analysis, the concept of a "reproducible paper" is discussed: this paper itself is exactly reproducible (snapshot v4-0-g8505cfd).

Keywords. galaxies: halos, galaxies: photometry, galaxies: structure, methods: data analysis, methods: reproducible, techniques: image processing, techniques: photometric

1. Introduction

Signal from the low surface brightness universe is buried deep in the dataset's noise and thus requires accurate detection methods. In Akhlaghi and Ichikawa (2015) (henceforth AI15) a new method was introduced to detect such very low signal-to-noise ratio (S/N) signal from the images in a non-parametric manner. It allows accurate detection of the diffuse outer features of galaxies (that often have a different morphology from the centers). The software implementation of this method (NoiseChisel) is released as part of a larger collection of data analysis software known as GNU Astronomy Utilities† (Gnuastro). It was the first professional astronomical software to be independently refereed by an independent panel (GNU Evaluation committee) and fully conforms with the GNU Coding Standards‡.

Since its release, NoiseChisel has been used in many studies. For example Bacon et al. (2017) used it to identify objects that were missed by Rafelski et al. (2015) (henceforth R15), who used a combination of six SExtractor (Bertin and Arnouts 1996) runs with different configurations to avoid deblending problems, but still missed many sources with significant signal, see Figure 1. Borlaff et al. (2019), Müller et al. (2019), and Trujillo et al. (2019) used it for accurate flat field and Sky subtraction to create deeper co-added images in galaxy fields for optimal detection of the low surface brightness features. Calvi et al. (2019) used it to find Lyman-α emitters in spectra. For future studies, Laine et al.

† https://www.gnu.org/s/gnuastro
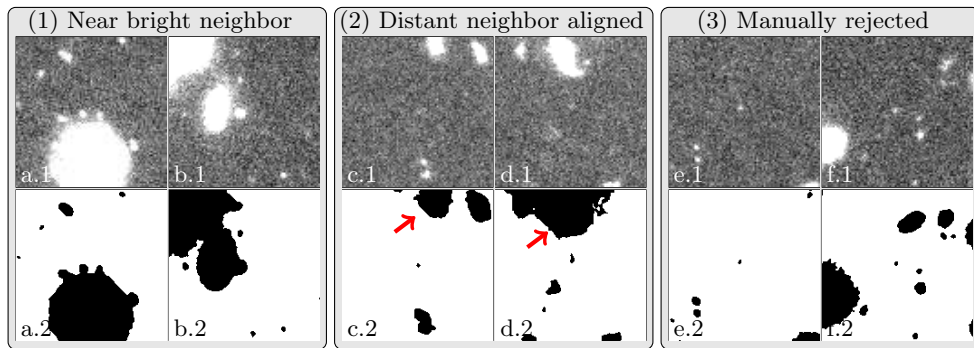‡ https://www.gnu.org/prep/standards

Figure 1. Example targets (central object in top row) found in MUSE datacubes (Inami et al. 2017) (I17) that did not correspond to anything in the Rafelski et al. (2015) (R15) catalog which is based on HST images. They had footprints in the combined SExtractor segmentation map of R15 (bottom row), so they are 'detectable'. However, they were missed due to incorrect labeling: a segmentation/deblending problem. The top row is the HST/ACS F775W image from the XDF survey (Illingworth et al. 2013). From left to right, their IDs in I17 are 6313, 6460, 6331, 6338, 6341, 6431. In case (2), the central detection has the same label as the large detection that the red arrow is pointing to. Note that the R15 segmentation shown is a combination of six SExtractor runs with different deblending parameters. For more, see Section 7.3 of Bacon et al. (2017).

(2018) and Hsieh et al. (2019) have proposed using NoiseChisel to maximize LSST's low surface brightness features (in extra galactic and solar system science outputs).

With ten stable releases until now, NoiseChisel and Gnuastro have greatly evolved. In Sections 2 and 3 a review of the major changes and improvements in Gnuastro's detection features since its first release is discussed and Section 4 demonstrates these new features on an example image of M51. Finally, understanding the systematics (i.e., reproducibility) is particularly important for low surface brightness science, therefore AI15 was also built in a reproducible manner. A short review of the evolved work in this frontier is given in Section 5.

2. NoiseChisel improvements

Over the last ten stable releases of Gnuastro (version 0.10), as with many of Gnuastro's programs, NoiseChisel has also greatly improved: some of the original features have been removed and many new features have been added. In Subsections 2.1 and 2.2 the removed and new features of NoiseChisel are respectively discussed. Note that NoiseChisel will continue to evolve after this paper also, so the best reference to consult is always its own manual/book, in particular the "NoiseChisel changes after publication" section.

The overall direction of Gnuastro's evolution (including NoiseChisel) has been modularity and more user control. In terms of modularity, the most notable change has been that NoiseChisel doesn't do segmentation any more. Segmentation of the signal into its substructure is fundamentally a separate kind of operation than detection (or separating the Sky from the signal).

In terms of more user control NoiseChisel has gradually moved in the direction of not having any number/configuration hard-coded into it. This gives the user a very detailed control over how it operates at every step. As a scientific software, it is critically important for the user to be able to tweak any step and understand the systematics of their final choice. In the latter two complete tutorials of the Gnuastro manual/book the

logic behind customizing them to fit different noise properties is fully described in real analysis scenarios.

2.1. *NoiseChisel removed features*

• Segmentation has been wholly moved to a new program; Segment. From Gnuastro 0.6, NoiseChisel is only in charge on detection. Therefore its output is now just a binary/two-value image (0: Sky, 1: Signal). Identifying the sub-structure over the detected regions (for example true clumps) is a fundamentally different problem. Changes in the original segmentation steps are discussed in Section 3. This spin-off allows much greater modularity/creativity and is in the spirit of Gnuastro's modular design.
For example in some scenarios there may be no Sky (the whole dataset is covered by targets) and detection would thus not be necessary. This happens when imaging parts of dense resolved objects for example globular clusters or the wings of M31. In other cases, segmentation isn't necessary, for example to estimate the Sky or flat-field level in individual exposures during data reduction like Borlaff et al. (2019).

• `--skysubtracted`: This option was used to account for the extra noise that is added if the Sky value has already been subtracted. However, NoiseChisel estimates the Sky standard deviation based on the input data, not exposure maps or other theoretical considerations. Therefore the standard deviation of the undetected pixels also contains the errors due to any previous sky subtraction. This option is therefore no longer present in NoiseChisel.

• `--dilate`: During detection, the outer layers of true detections are eroded. Until version 0.5, after the true detections were found, the were dilated to recover the eroded pixels. However, the simple 8- or 4-connected dilation produced boxy/diamond shapes respectively. They were thus were not realistic and would miss diffuse flux. The final dig into the noise is now done by "grow"ing the true detections. Similar to how true clumps were grown, see the description of `--detgrowquant` below.

2.2. *NoiseChisel new features*

The new features in NoiseChisel are listed below. Note that their order is based on Noise-Chisel's functionality: which step is done before which.

• The quantile difference to identify tiles with no significant signal is measured between the *mean* and median. Until Gnuastro 0.8, it was between the *mode* and median. The quantile of the mean is more sensitive to skewness (the presence of signal), so it is preferable to the quantile of the mode.

• `--widekernel`: By default, NoiseChisel convolves the input once and estimates the proper tiles for quantile estimations on the convolved image. The same convolved image is later used for quantile estimation. A larger kernel has the advantage that it increases the skewness (and thus difference between the mean and median, therefore helps in detecting the presence signal). However, a larger kernel disfigures the shapes/morphology of the objects.
The new `--widekernel` option (and a corresponding `--wkhdu` option to specify its HDU) were added in Gnuastro 0.5 to allow benefiting from the advantage and not being affected by the problem. When its given, the input will be convolved with both the sharp (given through the `--kernel` option) and wide (given through this option) kernels. The mean and median are calculated on the dataset that is convolved with the wider kernel and good tiles are chosen. Finally, quantiles are estimated on the good tiles based on the convolution with the sharper kernel.

• Rejection of outlier tiles used in estimating the quantile threshold: when there are large galaxies or bright stars in the image, their gradient may be on a smaller scale

than the selected tile size. In such cases, those tiles will be identified as tiles with no signal, over subtracting the Sky on the wings of bright sources, and creating artificial dark halos in the most extreme cases. An outlier identification algorithm has been added to NoiseChisel to mitigate this problem since Gnuastro 0.8. It can be configured with the following options: `--outliersigma` and `--outliersclip`.

If we assume there are N elements (useful tiles), the values are first sorted. Searching for the outlier starts on element $N/2$ (integer division). Let's take v_i to be the i-th element of the sorted input (with no blank values) and m and σ as the σ-clipped median and standard deviation from the distances of the previous $N/2 - 1$ elements (not including v_i). If the value given to `--outliersigma` is displayed with s, the i-th element is considered as an outlier when the condition below is true.

$$\frac{(v_i - v_{i-1}) - m}{\sigma} > s$$

Since i begins from the median, the outlier has to be larger than the median. You can use the check images (for example `--checkqthresh`, `--checkdetsky` and `--checksky` options in NoiseChisel) to inspect the steps and see which tiles have been discarded as outliers prior to interpolation.

- `--blankasforeground`: allows blank pixels to be treated as foreground in Noise-Chisel's binary operations: the initial erosion (`--erode`) and opening (`--open`) as well as the filling holes and opening step for defining pseudo-detections (`--dthresh`). Until Gnuastro 0.8, NoiseChisel treated blank pixels as foreground by default. But this could create false positives near blank/masked regions, blank pixels are now considered to be in the background by default. This option will re-create the old behavior.

- `--skyfracnoblank`: To reduce the bias caused by undetected wings of galaxies and stars in the Sky measurements, NoiseChisel only uses tiles that have a sufficiently large fraction of undetected pixels. Until Gnuastro 0.8, the reference for this fraction was the whole tile size. With this option, it is now possible to ask for ignoring blank pixels when calculating the fraction. This is useful when blank/masked pixels are distributed across the image.

- `--dopening`: Number of openings after applying `--dthresh`, introduced in Gnuastro 0.9.

- `--dopeningngb`: Type of connectivity to define neighbors (4-, or 8-connected) in `--dopening`, introduced in Gnuastro 0.9.

- `--holengb`: Type of connectivity to define neighbors (4-, or 8-connected) to define (and thus, fill) holes after applying `--dthresh` to find pseudo-detections. Introduced in Gnuastro 0.8.

- `--pseudoconcomp`: Type of connectivity to define neighbors (4-, or 8-connected) to find individual pseudo-detections. For example if connectivity is defined by 8-connected neighbors, pseudo-detections that are touching on the corner will be identified as one. This was introduced in Gnuastro 0.9.

- `--snthresh`: Manually set the S/N of true pseudo-detections and thus avoid the need to automatically identify this value. Introduced in Gnuastro 0.9.

- `--detgrowquant`: is used to grow the final true detections until a given quantile into the noise since Gnuastro 0.5. It replaces the old `--dilate` option in the paper and older versions of Gnuastro. Dilation is a blind growth method which causes objects to be boxy or diamond shaped when too many layers are added. However, with the growth method that is defined now, we can follow the signal into the noise with any shape. It uses a modified the Watershed algorithm, very similar to the way that clumps are
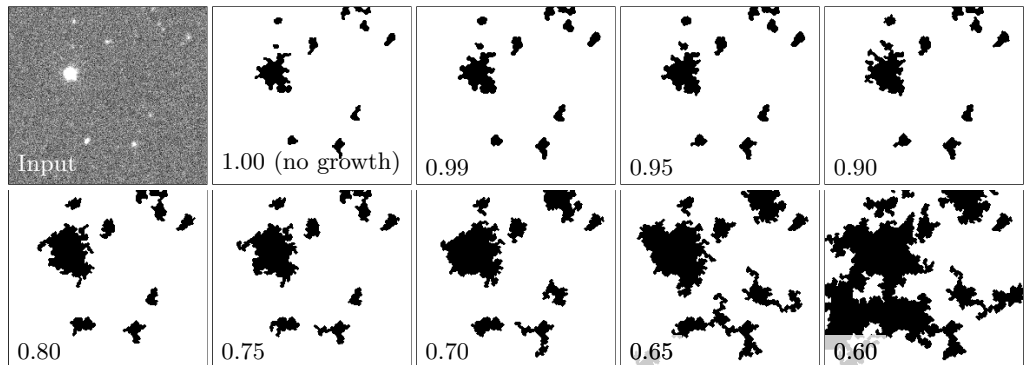
Figure 2. Growing true detections with various values to the `--detgrowquant` option in NoiseChisel. The top-left image is the input image. Each subsequent binary image is the grown detection map after the growth has been completed to the given quantile. All other NoiseChisel parameters are identical.

grown during segmentation (compare columns 2 and 3 in Figure 10 of the paper). The difference with the Watershed algorithm is that when a pixel isn't touching any previously labeled pixel, it is not given a new label, it is just sent to the end of the sorted queue to be checked again in the next round. The best quantile to grow the quantile depends on your dataset's correlated noise properties and how cleanly it was Sky subtracted. The new `--detgrowmaxholesize` can also be used to specify the maximum hole size to fill as part of this growth.

This new growth process can be much more successful in detecting diffuse flux around true detections compared to dilation and give more realistic results. For example see Figures 2 and 3. Figure 2 shows how decreasing this option (going deeper into the noise) effects the final detection map. It can be seen that beyond a certain point, it will just start to follow correlated noise, emanating from the initial detections. See the discussion in Section 4 on how to address this problem while its not too significant.

• `--cleangrowndet`: After dilation, if the signal-to-noise ratio of a detection is less than the derived pseudo-detection S/N limit, that detection will be discarded. In an ideal/clean noise, a true detection's S/N should be larger than its constituent pseudo-detections because its area is larger and it also covers more signal. However, on false detections (especially at lower `--snquant` values), the increase in size can cause a decrease in S/N below that threshold.

When this option is called, if the grown S/N of a detected region is below the pseudo-detection S/N, it is discarded. Because a true detection has flux in its vicinity and dilation will catch more of that flux and increase the S/N. This option will therefore improve purity and not effect the completeness (a true detection will not be discarded).

However, in many real images bad processing creates artifacts that cannot be accurately removed by the Sky subtraction. In such cases, this option will decrease the completeness (will artificially discard true detections). So this feature is not default and should to be explicitly called when you know the noise is clean.

3. Segment program for Segmentation

Prior to Gnuastro version 0.6 (released in June 2018), one program (NoiseChisel) was in charge of detection *and* segmentation. To increase creativity and modularity, NoiseChisel's segmentation features were spun-off into a separate program (Segment). Segment's main

algorithm and working strategy were initially defined and introduced in Section 3.2 of AI15.

• Since the spin-off from NoiseChisel, the default kernel to smooth the input for convolution has a FWHM of 1.5 pixels (still a Gaussian). This is slightly less than NoiseChisel's default kernel (which has a FWHM of 2 pixels). This enables the better detection of sharp/point-like clumps: as the kernel gets wider, the lower S/N (but sharp/small) clumps will be washed away into the noise. Gnuastro's MakeProfiles can be used to build any custom kernel if this is too sharp/wide for a certain purpose.

The ability to use a different convolution kernel for detection and segmentation is one example of how separating detection from segmentation into separate programs can increase productivity. The purpose of detection is separating diffuse and extended emission from noise, but in segmentation, sharp/localized peaks are desired.

• The criteria to select true from false clumps is the peak significance. It is defined to be the difference between the clump's peak value (C_c) and the highest valued river pixel around that clump (R_c). Both are calculated on the convolved image (signified by the c subscript). To avoid absolute values (differing from dataset to dataset), $C_c - R_c$ is then divided by the Sky standard deviation under the river pixel used (σ_r) as shown below (when --minima is given, the nominator becomes $R_c - C_c$):

$$\frac{C_c - R_c}{\sigma_r}$$

The input Sky standard deviation dataset (--std) is assumed to be for the unconvolved image. Therefore a constant factor (related to the convolution kernel) is necessary to convert this into an absolute peak significance†. However, as far as Segment is concerned, the absolute value of this correction factor is irrelevant: a purity level is found based on the ambient noise (undetected regions) to find the numerical threshold of this fraction. The user doesn't directly specify this value by default (unless --clumpsnthresh is given).

A distribution's extremum (maximum or minimum) values, used in the new criteria, are strongly affected by scatter. On the other hand, the convolved image has much less scatter‡. Therefore $C_c - R_c$ is a more reliable (with less scatter) measure to identify signal than $C - R$ (on the unconvolved image).

Initially, the total clump signal-to-noise ratio of each clump was used, see Section 3.2.1 of AI15. Therefore its completeness decreased dramatically when clumps were present on gradients for example see Figure 4. In tests, this measure proved to be more successful in detecting clumps on gradients and on flatter regions simultaneously.

• With the new --minima option, it is now possible to detect inverse clumps (for example absorption features).

---

† To get an estimate of the standard deviation correction factor between the input and convolved images, you can take the following steps: 1) Mask (set to NaN) all detections on the convolved image with the where operator or Gnuastro's Arithmetic program. 2) Calculate the standard deviation of the undetected (non-masked) pixels of the convolved image with the --sky option of Gnuastro's Statistics program (which also calculates the Sky standard deviation). Just make sure the tessellation settings of Statistics and NoiseChisel are the same (you can check with the -P option). 3) Divide the two standard deviation datasets to get the correction factor.

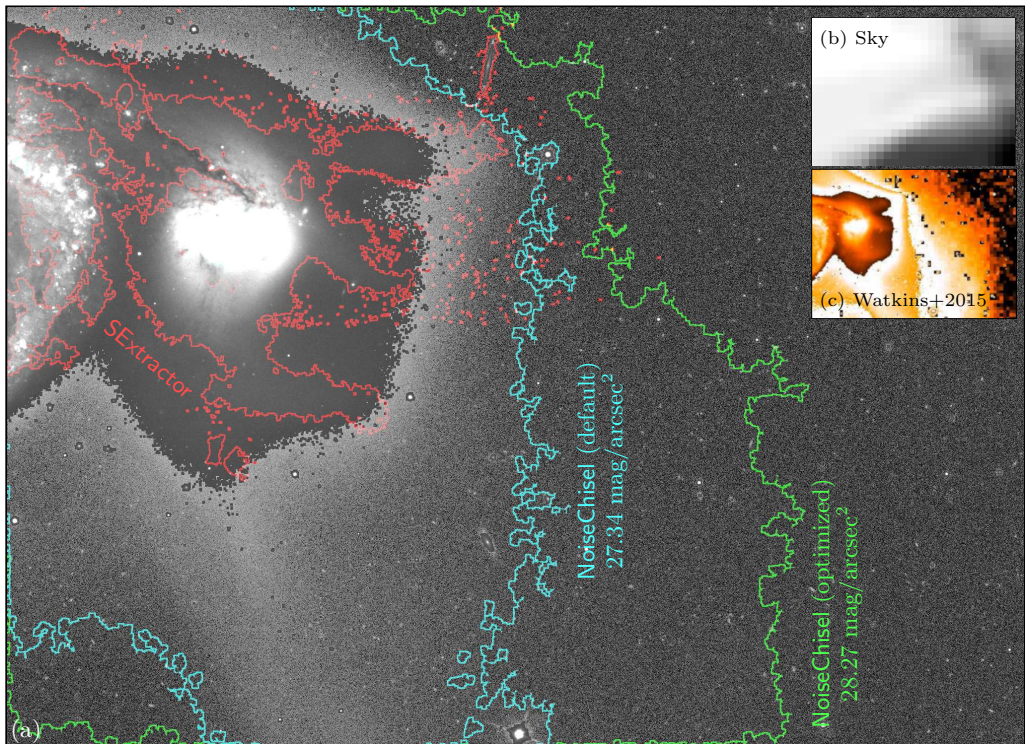‡ For more on the effect of convolution on a distribution, see Section 3.1.1 of AI15

Figure 3. Detection of M51's outer wings on a single-exposure SDSS image. (a) Input image overlayed with the border of detected regions. The lower-S/N regions of the image are shown in a logarithmic scale, the brighter parts are shown in linear scale to simultaneously show the familiar bright structure and outer low surface brightness wings. The red boundary shows **SExtractor**'s detection (default parameters, except thresholds that are set to 0.5σ). The light-blue boundary shows the outer edge of the detection with **NoiseChisel**'s default configuration, and has an average S/N of 0.59 (or 27.34 mag/arcsec^2). The green boundary shows the result after optimizing the configuration for the noise of single-exposure SDSS images. The green boundary reaches an average S/N of 0.25 (or 28.27 mag/arcsec^2). For more on the customization of **SExtractor** and **NoiseChisel**, see Section 4. (b) Optimized **NoiseChisel** Sky image, showing that there is still signal outside its detected regions. (c) Much deeper, processed image from Watkins et al. (2015) showing the outer signal at much higher S/N, reproduced by permission of the AAS.

4. Demonstration: M51 outskirts on a shallow image

As a small demonstration of the detection and segmentation power of Gnuastro, a single exposure (exposure time of 54 seconds), SDSS (Abazajian et al. 2009) image of M51 and NGC5195 in the r-band is used. It is a raw image directly downloaded when searching for NGC5195 in the SDSS "simple field search" online tool.

As a reference for readers who may be more familiar with **SExtractor**, it is run on the image first (version 2.25.0). The borders of **SExtractor**'s detected pixels that overlap with M51 and NGC5195 are marked with red in Figure 3. The default value of all the **SExtractor** parameters were used (generated with the -d option) except for DETECT_THRESH=0.5 and ANALYSIS_THRESH=0.5. The default thresholds of 1.5 (multiple of σ) were far too high and useless in this scenario. Such a low threshold is rarely used because it creates many false positives (see Sections 5.2 and B.1.3 in AI15).

However, even with such a decreased signal, it can only detect the very bright parts of the signal. This is primarily because of poor Sky subtraction in the vicinity of diffuse

signal, see Figure 16 and Section A.6 of AI15. All the small detections also had the same label as the large detections, similar to Figure 1(2). This is a big problem in a general catalog, where all these background galaxies will be missed in the final catalog.

NoiseChisel (from Gnuastro 0.10) is also run on this image. It is first run with its default parameters (no run-time options), producing the light-blue boundary in Figure 3 (S/N=0.59, or 27.34 mag/arcsec^2). However, the default parameters are only for a generic noise pattern. Once NoiseChisel is slightly tweaked for the single-exposure SDSS images (command below), it is able to detect the wings of M51 much more deeply as shown in the green border (S/N=0.25, or 28.27 mag/arcsec^2, almost one magnitude deeper than the default):

```
astnoisechisel m51-r.fits -h0 --tilesize=75,75 --meanmedqdiff=0.001 \
               --interpnumngb=6 --snquant=0.95 --detgrowquant=0.65 \
               --detgrowmaxholesize=10000 --minskyfrac=0.9
```

The displayed boundaries and surface brightness measurements of Figure 3 correspond to the NoiseChisel output after being eroded two times. The reason for this can be seen in Figure 2. As --detgrowquant is decreased, when there is no signal, Noise-Chisel starts following correlated noise that usually manifests as thin connections between smaller detections (like the three small detections that start touching beyond --detgrowquant=0.65 in Figure 2). However, when there is signal (like the outer parts of the central galaxy), it is clear that NoiseChisel is detecting very low S/N signal and such thin branches aren't created. Erosion removes the thin connections while not affecting the large contiguous diffuse detections.

Figure 3(b) shows the Sky image (average of undetected pixels over a grid) for the optimized NoiseChisel run. Even though 28.27 mag/arcsec^2 seems very deep for a short-exposure image, the Sky image still resembles the shape of M51, showing that there is still fainter signal in this image. In the tutorial, the readers are encouraged to optimize NoiseChisel even further to obtain a better result. Finally, Figure 3(c) shows a rotated crop from the much deeper study of M51 by Watkins et al. (2015), confirming the physical reality of this signal with the same morphology.

The logic behind finding the best parameters used in the command above is described in the "Detecting large extended targets" tutorial of the Gnuastro manual. This config-uration is good for any single-exposure SDSS image. When dealing with a new type of dataset (images with different cameras, or reduced differently), for an optimal result, its always good to tweak it a little and optimize it for that particular noise-pattern.

Figure 4 shows the result of Segment (with its default configuration) on the brighter part of the M51 image. It clearly shows the effectiveness of the new criteria to identify true peaks discussed in Section 3. Each one of the HII regions in Segment's clumps can then be cataloged with MakeCatalog to be studied individually, or masked to study the diffuse region.

5. Reproducible data analysis

As shown in Figure 3, when NoiseChisel is optimized for a special noise pattern (in this case, any single-exposure SDSS image), it can operate significantly better. But this isn't peculiar to NoiseChisel, scientific software need to allow their users to modify every step of their analysis. Some software authors try to decrease the number of tweak-able parameters by hard-coding the parameters within the code. But this decreases the ability of the user (a scientist) to understand the details of the analysis and its effect on their scientific interpretation. It is therefore critical for the integrity of the result that the authors report the exact details of all their software's configuration options, what order
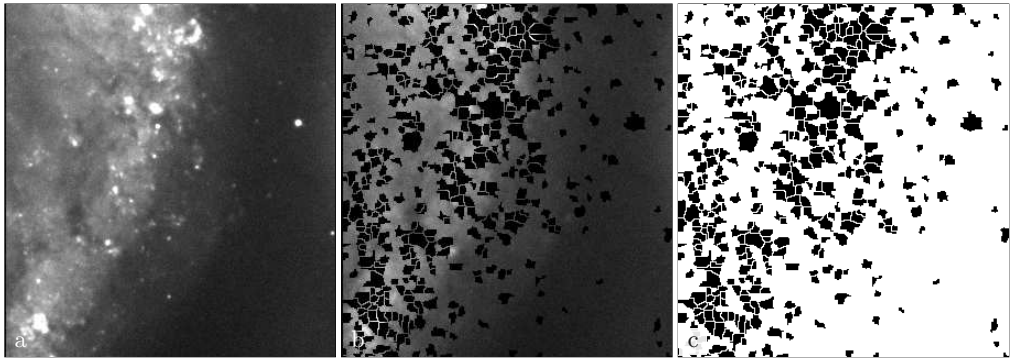
Figure 4. Identification of HII regions over the M51 spiral arm in Figure 3 using Segment. (a) Input image, (b) all detected clumps masked, (c) the detected clumps.

they were run, how they were installed and etc. Simply reporting the name of a software is not enough. But the format of a traditional paper like this also doesn't allow going into too much detail.

Generally, given the large volume of data already in surveys like HSC and DES (which will increase dramatically with LSST), and the many configuration options (parameters to all the software used, software versions, and their operating environment), it is becoming harder to test new configurations, methods or software and thus understand the limits of the final choice in the low S/N regime. This is a major hurdle in using such studies for the low surface brightness universe. Such tests can be facilitated/encouraged in a design where previous results can be exactly reproduced, on different machines, independent of the host operating system.

To address this major problem, AI15 also publicly released all the analysis scripts and configuration files that generated its results with the paper's LaTeX source on arXiv. Since that paper, that solution has grown into a "reproducible paper template"†, which provides a robust framework to start reproducible research projects. It installs fixed versions of all the necessary software from source and cites them where necessary (see Appendix A). Combined with all the scripts that produced every number or figure in this paper, this allows exact reproducibility of this paper by any random reader. This paper has been written using this template and can be exactly reproduced/verified/modified by retrieving version v4-0-g8505cfd of its history‡. All necessary files (reproducible source, its Git history and software tarballs) are also archived on Zenodo (https://doi.org/10.5281/zenodo.3408481). The reproducible source includes links to download the input images from their respective server, as well as their MD5 checksum to confirm that the downloaded file is the same that was used here (has not been updated on the server).

6. Acknowledgements

---

† https://gitlab.com/makhlaghi/reproducible-paper
‡ https://gitlab.com/makhlaghi/iau-symposium-355

References

Abazajian, K. N. et al. (2009). *ApJS*, 182, 543.
Akhlaghi, M. and T. Ichikawa (2015). *ApJS*, 220, 1.
Bacon, R. et al. (2017). *A&A*, 608, A1.
Bertin, E. and S. Arnouts (1996). *A&AS*, 117, 393.
Borlaff, A. et al. (2019). *A&A*, 621, A133.
Calvi, R. et al. (2019). *MNRAS*, 489, 3294.
Frigo, M. and S. G. Johnson (2005). *IEEE Proc.*, 93, 216.
Hsieh, H. H. et al. (2019). *Preprint*, arXiv:1906.11346.
Illingworth, G. D. et al. (2013). *ApJS*, 209, 6.
Inami, H. et al. (2017). *A&A*, 608, A2.
Laine, S. et al. (2018). *Preprint*, arXiv:1812.04897.
Müller, O. et al. (2019). *A&A*, 624, L6.
Rafelski, M. et al. (2015). *AJ*, 150, 31.
Trujillo, I. et al. (2019). *MNRAS*, 486, 1192.
Watkins, A. E. et al. (2015). *ApJL*, 800, L3.

Appendix A. Software acknowledgement