Optimising and comparing source-extraction tools using objective segmentation quality criteria

Haigh, C.; Chamba, N.; Venhola, A.; Peletier, R.; Doorenbos, L.; Wilkinson, M. H. F.

Published in:
Astronomy & astrophysics

DOI:
10.1051/0004-6361/201936561

A&A 645, A107 (2021)
https://doi.org/10.1051/0004-6361/201936561
© ESO 2021

Astronomy & Astrophysics

Optimising and comparing source-extraction tools using objective segmentation quality criteria

Caroline Haigh[1], Nushkia Chamba[2,3,4], Aku Venhola[5,6], Reynier Peletier[5], Lars Doorenbos[1], Matthew Watkins[1], and Michael H. F. Wilkinson[1]

[1] Bernoulli Institute for Mathematics, Computer Science and Artificial Intelligence, University of Groningen, PO Box 407, 9700 AK Groningen, The Netherlands
e-mail: c.haigh@rug.nl, m.h.f.wilkinson@rug.nl
[2] Instituto de Astrofísica de Canarias, Calle Vía Láctea, s/n, 38205 San Cristóbal de La Laguna, Santa Cruz de Tenerife, Spain
e-mail: chamba@iac.es
[3] Departamento de Astrofísica, Universidad de La Laguna, 38205 La Laguna, Tenerife, Spain
[4] Department of Astronomy and Oskar Klein Centre for Cosmoparticle Physics, Stockholm University, AlbaNova University Centre, 10691 Stockholm, Sweden
[5] Kapteyn Astronomical Institute, University of Groningen, PO Box 800, 9700 AV Groningen, The Netherlands
[6] Space Physics and Astronomy Research Unit, University of Oulu, Pentti Kaiteran katu 1, 90014 Oulu, Finland

Received 23 August 2019 / Accepted 11 September 2020

ABSTRACT

Context. With the growth of the scale, depth, and resolution of astronomical imaging surveys, there is increased need for highly accurate automated detection and extraction of astronomical sources from images. This also means there is a need for objective quality criteria, and automated methods to optimise parameter settings for these software tools.

Aims. We present a comparison of several tools developed to perform this task: namely SExtractor, ProFound, NoiseChisel, and MTObjects. In particular, we focus on evaluating performance in situations that present challenges for detection. For example, faint and diffuse galaxies; extended structures, such as streams; and objects close to bright sources. Furthermore, we develop an automated method to optimise the parameters for the above tools.

Methods. We present four different objective segmentation quality measures, based on precision, recall, and a new measure for the correctly identified area of sources. Bayesian optimisation is used to find optimal parameter settings for each of the four tools when applied to simulated data, for which a ground truth is known. After training, the tools are tested on similar simulated data in order to provide a performance baseline. We then qualitatively assess tool performance on real astronomical images from two different surveys.

Results. We determine that when area is disregarded, all four tools are capable of broadly similar levels of detection completeness, while only NoiseChisel and MTObjects are capable of locating the faint outskirts of objects. MTObjects achieves the highest scores on all tests for all four quality measures, whilst SExtractor obtains the highest speeds. No tool has sufficient speed and accuracy to be well suited to large-scale automated segmentation in its current form.

Key words. techniques: image processing – surveys – methods: data analysis

1. Introduction

Segmentation maps, which are images that match specific pixels of an image to a particular source or sources, are used extensively to preprocess observational data for analysis. They are used for masking sources, estimating sky backgrounds, and creating catalogues, amongst other applications. It is therefore essential that the tools used to create these maps are accurate and reliable. Otherwise, the subsequent scientific process may be invalidated by errors in the measurements of sources.

Unfortunately, astronomical images have many properties that cause problems for traditional image-segmentation algorithms. Images may be highly noisy and have an extremely large dynamic range. Objects generally have no clear boundaries, and their outer regions may extend below the level of background noise (see Fig. 1). As many generic segmentation algorithms are edge-based (Pal & Pal 1993; Wilkinson 1998), they are unable to accurately process these images.

In addition, with growth of the scale of astronomical surveys, there is increased need for a fast and accurate tool for segmentation. This is illustrated by current projects such as the Legacy Survey of Space and Time (LSST), which aims to produce around 15 TB of raw data per night (Ivezić et al. 2019). With surveys of this scale, human intervention will no longer be feasible, meaning that the tools should ideally be robust to variations in images without manual tuning.

Because of these unique challenges, a number of tools have been developed for the sole purpose of accurately detecting sources in astronomical images. The most well-known of these for optical data is SExtractor (Bertin & Arnouts 1996). However, in recent years, a number of alternatives have been proposed, including ProFound (Robotham et al. 2018), NoiseChisel (Akhlaghi & Ichikawa 2015), and MTObjects (Teeninga et al. 2013, 2016).

In this paper, we evaluate and compare these segmentation tools in order to study their strengths and weaknesses.

A thorough comparison provides a means for astronomers to choose the algorithm that is best suited for their scientific goals. In addition, several of these tools are still under active development, and such an analysis can help to direct future advancements.

For this comparison, we developed numerical measures for segmentation quality (Sect. 3.3), and propose a method for automatic configuration of tool parameters (Sect. 3.2). This approach to evaluating segmentation maps is designed to provide an objective measure of quality. To test this, we use simulated images with a known ground truth (Sect. 3.1) to provide evaluations that are not dependent on visual bias and preconceptions. We supplement our results by demonstrating the performance of our automatically configured parameters on real survey images (Sect. 5).

Throughout this paper we use the terms 'segmentation', 'source detection', and 'source extraction' interchangeably to refer to the process of identifying unique sources in astronomical images and marking the pixels of the image in which each source is the dominant contributor.

2. Source-extraction methods

2.1. Previous methods

For as long as astronomical images have been produced, it has been necessary for their contents to be catalogued and measured in order that they may be used for scientific applications. As manually locating and outlining objects is a slow and subjective process, particularly when considering the faint outskirts of objects, many attempts have been made at automating this process.

Early automatic tools directly scanned photographic plates to locate sources and produce measurements. A notable example is COSMOS (Pratt 1977), which used a process of repeated thresholding to produce 'coarse measurements' of images, essentially quantising the image over an estimated local background level. It then used 'fine measurements' to produce more accurate measurements of the object profiles. Later additions included improved deblending of adjacent sources (Beard et al. 1990).

Whilst modern tools no longer use digitised photographic plates, instead working directly with data captured by CCDs, the overall process used in recent tools is fundamentally very similar to that used in their predecessors. Almost all tools follow the same four main steps:

1. Identify and measure the background level.
2. Threshold the image relative to the background.
3. Locate (and deblend) sources appearing above the threshold.
4. Produce a catalogue of sources and their measured properties.

SExtractor, described in more detail below, uses a very similar method to COSMOS, namely repeated thresholding. In contrast, several other tools make use of dendrograms, that is, hierarchical representations of images, in which nodes representing local maxima are connected at the highest brightness level where thresholding would show a single, unbroken object. Users may subsequently 'prune' the dendrogram by removing nodes connecting very small or faint regions, and may automatically or manually mark objects meeting some criteria. Dendrograms have been used to visualise and analyse hierarchical structure in both infrared images (Houlahan & Scalo 1992) and radio data cubes (Rosolowsky et al. 2008; Goodman et al. 2009).

Other tools have deviated from a thresholding-based approach. Many of these tools and their methods are described in Masias et al. (2012).

2.2. Deblending

Deblending, the process of separating overlapping or nested sources, is closely linked to source extraction; all of the tools we discuss in this paper make some attempt at deblending. However, for some scientific purposes, the tools do not produce sufficiently accurate separation of sources, leading to problems such as poor photometry (Abbott et al. 2018; Huang et al. 2018), and systematic measurements of physical properties such as redshift (Boucaud et al. 2020) and cluster mass (Simet & Mandelbaum 2015). Consequently, several tools also exist to perform deblending as a separate process. As these tools are predominantly either designed to use the results of another source extraction tool (such as SCARLET Melchior et al. 2018, which uses SExtractor for initial source detection), or are predominantly designed for smaller images with only a few galaxies (such as the machine-learning-based methods proposed in Reiman & Göhre 2019), we chose not to include them in the comparisons in this paper. However, the evaluation process we define in Sect. 3 could equally be used to compare deblending-specific tools.

2.3. Compared tools

We chose to focus our comparison on four tools: SExtractor (Bertin & Arnouts 1996), which is in common use, and three recent alternatives: ProFound (Robotham et al. 2018), NoiseChisel + Segment (Akhlaghi & Ichikawa 2015), and MTObjects (Teeninga et al. 2013, 2016). We chose to exclude several other source-extraction tools from this comparison for various reasons, notably DeepScan (Prole et al. 2018), which is dependent on the use of another tool (such as SExtractor) to produce an initial mask; and AstroDendro (Robitaille et al. 2013), which was prohibitively slow to run on large images.

2.3.1. SExtractor

SExtractor (Bertin & Arnouts 1996) is a widely used tool for the creation of segmentation maps. It was developed with the goal of producing catalogues of astronomical sources from large-scale sky surveys.

The first step in the SExtractor pipeline is the estimation and subtraction of the background. The image is divided into tiles, and a histogram is produced for each. Values more than three standard deviations from the median are removed. Tiles are then classified into crowded and uncrowded fields based on the change in histogram distribution, and a background value is estimated based on the median and mode of each tile.

The image is then thresholded at a fixed number of exponentially spaced levels above a user-defined threshold. This converts the light in the image into trees, with branches representing bright areas within larger, fainter objects. Pixels in branches that contain at least a given proportion of the light of their parent objects are marked as individual objects, whilst branches containing a lower amount of light are regarded as part of the parent object. Pixels in the outskirts of objects are allocated labels based on the probability that a pixel of that value is present at that point, using profiles fitted to the detected sources.

In practice, SExtractor may be used in multiple passes, particularly when detecting extended sources. For example, a hot/cold method may be used, wherein a sensitive pass captures the outskirts of objects, and a less sensitive pass identifies which objects are not false-positive detections (Rix et al. 2004). It may also be used to identify candidate objects, which are then manually verified.

SExtractor version 2.19.5 was used for this comparison, using the default filter: a convolution with a 3 × 3 pyramidal function that approximates Gaussian smoothing. We found in subsequent testing, described in Appendix A, that using a 9 × 9 Gaussian PSF with a full width at half maximum of 5 pixels produced marginally better results, although this difference is not significant, and does not affect the general conclusions of this paper.

2.3.2. ProFound

ProFound (Robotham et al. 2018), like SExtractor, was designed as a general-purpose package for detecting and extracting astronomical sources; however, it is designed to produce a more accurate segmentation, which may be used for galaxy profiling.

Instead of using multiple thresholds, ProFound uses a single threshold after the background estimation stage in order to demarcate pixels containing sources. These pixels are then processed in descending order of brightness, with a watershed process being used to allocate less bright pixels (within some tolerance) neighbouring the object of the brightest pixel in a region, until all pixels bordering the object are either allocated to other objects, are marked as background, or have higher flux than neighbouring pixels within the object.

Following this process, the background is re-estimated, and an iterative process of calculating photometric properties of the segments and repeatedly dilating them is performed, to produce a final segmentation map. ProFound version 1.1.0 was used for this comparison.

2.3.3. NoiseChisel + Segment

NoiseChisel (Akhlaghi & Ichikawa 2015) was designed with the goal of finding ‘nebulous objects’, such as irregular or faint galaxies, accurately. NoiseChisel is intended to be hand-tuned for individual images; the tutorial states that configurations are ‘not generic’ (GNU Astronomy Utilities 2019).

NoiseChisel separates the image into areas containing light from objects, and areas containing only background. To do this, it uses a threshold below the estimated background level, and performs a series of binary morphological operations to create an initial detection map. Further morphological operations are then performed on the ‘objects’ and ‘background’ separately, and area and signal-to-noise thresholds are used to remove false detections. Segment then produces a map of ‘clumps’ by locating connected regions around local maxima in the image with a watershed-like process. It then discards those that do not meet a signal-to-noise threshold, and grows the remaining clumps to create a final segmentation map.

Since the publication of the original paper, the program has been split into two separate tools within the GNU Astronomy Utilities package: NoiseChisel, and Segment. For the purposes of this comparison, the tools are treated as a single pipeline, and evaluated together, and we examine only this final ‘objects’ output. We used the latest version at the start of our comparison, version 0.7.42a. Several new versions have since been released, which may contain different parameters and produce different results.

2.3.4. MTObjects

MTObjects (Teeninga et al. 2013, 2016) takes a similar approach to SExtractor; both operate on the principle that after a background subtraction step, objects can be detected by a thresholding process. However, where SExtractor uses a small number of fixed thresholds, MTObjects uses tree-based morphological operators.

A max-tree (Salembier et al. 1998) is constructed from the smoothed and background-subtracted image. The max-tree is a tree of the image: the leaves represent local maximum pixels, nodes represent increasingly large connected areas of the image, with decreasing minimum pixel values, and the root represents the entire image. This tree is then filtered, using tests to determine which nodes of the tree – or areas of the image – contain an amount of flux, given their area, that is statistically significant relative to their background. If a node has no significant ‘parent’, or its parent has another ‘child’ with greater flux, it is marked as an object. Despite representing all connected components at all grey levels in the image, building the max-tree is typically very efficient ($O(N \log N)$ for floating-point images Carlinet & Géraud 2014).

The max-tree structure used in MTObjects is very similar to the dendrogram used in several astronomical applications as described above (Houlahan & Scalo 1992; Rosolowsky et al. 2008). There are two main differences. Firstly, the dendrogram only contains nodes where areas connect, whereas the max-tree contains a node for every difference in brightness value. Secondly, MTObjects uses a single statistical significance test to detect objects, combining multiple attributes of the node, whilst the dendrogram methods frequently filter small and faint objects at fixed thresholds.

There have been no official software releases of MTObjects. We used a Python and C implementation[1], which we adapted from the software used in the original paper. We used significance test 4 as recommended by Teeninga et al. (2016).

3. Methodology

3.1. Data

In this section we describe the data with which we tested the tools. Simulated data (Sects. 3.1.1 and 3.1.2) allow us to accurately quantify performance on a simplified version of the problem, whilst survey images (Sect. 3.1.3) allow us to qualitatively explore behaviour in a range of different real situations.

3.1.1. Simulated data

Testing source-detection algorithms on real observational data has several limitations. Firstly, the ground truth is not known; even if objects have been manually labelled, it is possible that objects have been missed, or incorrectly measured. In particular, it is difficult to establish the true extent of objects at a low brightness level, as their outer regions may not be clearly visually distinguishable from background.

Secondly, the abundance of many features of interest –such as ultra-diffuse galaxies (Van Dokkum et al. 2015)– is not yet fully understood. This means that it is difficult to establish a statistical measure of how accurately they can be detected. As these objects are also more difficult for algorithms to detect, a larger sample is required to determine the accuracy of the algorithms.

By using simulated data, we gain the ability to test the algorithms on large datasets with a known ground truth. This means that we can make accurate measures of precision and accuracy for faint features, while taking into account the true extent of

[1] https://github.com/CarolineHaigh/mtobjects

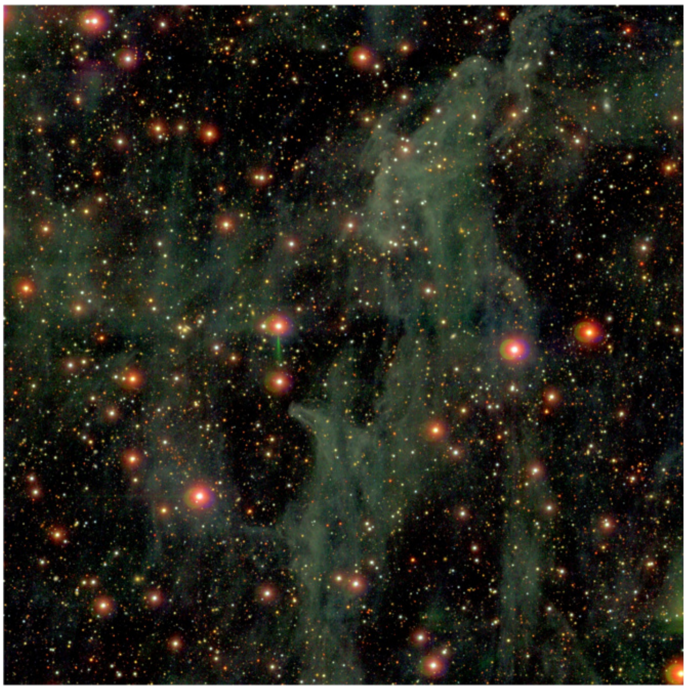


Fig. 1. A *gri*-composite image of IAC Stripe82 field `f0363_g.rec.fits` showing a large structure of Galactic cirri. Such complex, overlapping structures are challenging for source-detection tools.

objects. We can also measure the accuracy of algorithms in different controlled conditions, such as with high noise, background variation, and overlapping sources (see Fig. 2).

We created ten frames of data emulating images in the r'-band of data in the Fornax Deep Survey (FDS). This is a deep, medium-sized ground-based survey of the nearby Fornax cluster, which is located at a distance of 20 Mpc (Iodice et al. 2016; Venhola et al. 2018). Each simulated image contains approximately 1500 'stars', 4000 'cluster galaxies', and 50 'background galaxies'. Stars were simulated as point sources and galaxies as Sérsic models (Sersic 1968). The number and structural parameters of the stars and galaxies were drawn from distributions similar to those found in the FDS. In the simulated images, stars have magnitudes between 10 and 23 mag, and galaxies have mean effective surface brightnesses between 21 mag arcsec^{-2} and 31 mag arcsec^{-2}. Background galaxies have effective radii between 0.5 and 3.5 arcsec and Sérsic indices between 2 and 4. Cluster galaxies have effective radii between 2.5 and 40 arcsec, and Sérsic indices between 0.5 and 2. Axis ratios varied from 0.3 to 1.0. To replicate observation conditions, images were convolved with the r-band point spread function of the OmegaCAM, and Poissonian and Gaussian noise were added (Venhola et al. 2018). For further details of the process, see Venhola (2019, Chapter 5).

3.1.2. Choosing a ground truth

Astronomical sources have no clear boundary; their light merely becomes insignificant in relation to noise and background light at some point in their outskirts. This means that when we create a ground truth for simulated images –a 'correct' segmentation map– we need to choose a threshold, t, below which we judge light from sources to be undetectable. Assuming a flat background, this threshold can be expressed as a sum of the background level, bg, and some multiple, n, of the standard deviation of the noise, σ:

$$t = bg + (n * \sigma). \tag{1}$$

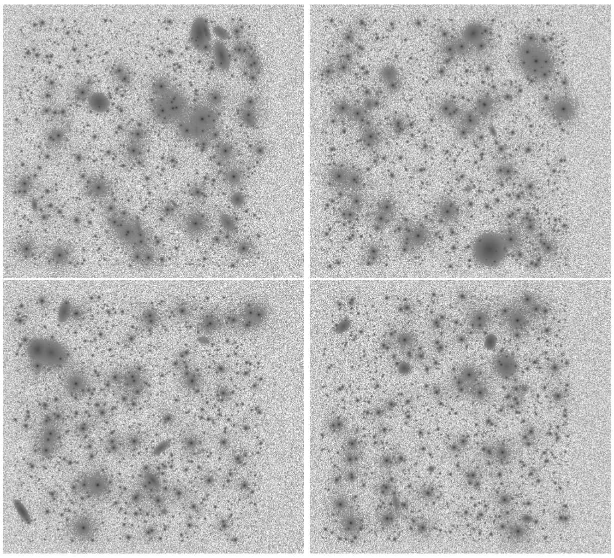


Fig. 2. Simulated survey images.

Sources may also overlap, meaning that each pixel contains light from multiple sources. In segmentation maps, each pixel is allocated to a single source; therefore, it is necessary to determine the source that has the strongest relationship with a given pixel. It should be noted that whilst segmentation maps are the traditional method of demarcating sources within an image, they are limited by their inability to represent the reality that pixels contain light belonging to multiple sources[2]. Consequently, tree-based methods, which inherently model nested objects, are unable to capture this structure within segmentation maps. As such, information contained in the models is lost, and not measured in the evaluation.

We initially considered allocating each pixel to the source which contributed the most flux to it. However, this meant that fainter sources in the vicinity of bright sources were entirely erased, as they had a lower raw flux contribution.

Instead, we chose to allocate labels based on a combination of the importance of the pixel to the source and the importance of the source to the pixel. For a source with total flux F_s, contributing a flux $f_{s,p}$ to a pixel with total flux F_p, the pixel contains $\frac{f_{s,p}}{F_s}$ of the flux of the source. Conversely, the source contributes $\frac{f_{s,p}}{F_p}$ of the light contained within the pixel.

These measures may be combined to give a single measure:

$$\frac{f_{s,p}}{F_s} \times \frac{f_{s,p}}{F_p}. \tag{2}$$

When allocating a pixel to a source, F_p will be constant for all sources contributing to the pixel. Therefore, the pixel may be allocated to the source with the highest value for

$$\frac{(f_{s,p})^2}{F_s} \tag{3}$$

and $f_{s,p} \geq t$.

[2] A new data format would be required to clearly represent this nested data. This could prove to be a challenging problem because of the complexity of allocating multiple labels and proportional brightnesses to each pixel.

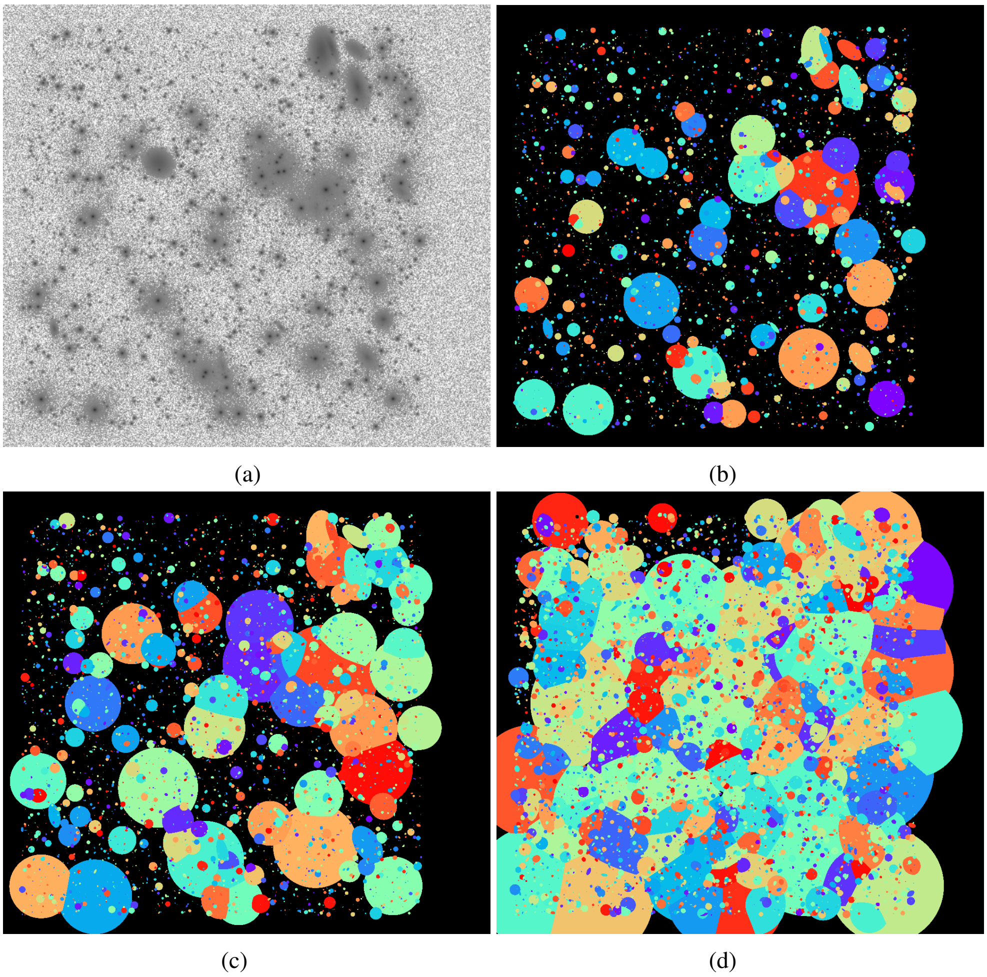


Fig. 3. Ground-truth segmentations of a simulated image, with a varying threshold ($n * \sigma$). The coloured regions label distinct objects, and the black regions make up the background. (*a*) Simulated image. (*b*) Ground truth (1.0σ). (*c*) Ground truth (0.5σ). (*d*) Ground truth (0.1σ).

The value of n has a substantial effect on the areas allocated to objects, as shown in Fig. 3. Consequently, it has a large impact on the evaluation of the segmentation maps produced by the tools.

As we aim to evaluate the performance of the tools at levels of low surface brightness, we chose to use a value of $n = 0.1$ for our ground truths. This pushes the tools to optimise their parameters to capture and correctly allocate as much of the light in the images as possible.

3.1.3. Real-world data

Whilst testing algorithms on real-world data is subject to several limitations, as discussed above, it is nevertheless essential, as it allows us to subjectively evaluate performance on structures and conditions which cannot be easily simulated, such as streams, spiral galaxies, and unusual artefacts.

With this in mind, we selected a number of images in the optical which contained examples of these features. We chose images from the Fornax Deep Survey (FDS; Iodice et al. 2016; Venhola et al. 2018), IAC Stripe 82 Legacy Project[3] (hereafter IAC Stripe82; Fliri & Trujillo 2016; Román & Trujillo 2018) which is a 2.5 degree stripe ($-50° < \mathrm{RA} < 60°$, $-1.25° < \mathrm{Dec} < 1.25°$) with a total area of 275 square degrees in all the five Sloan Digital Sky Survey (SDSS) bands and the *Hubble* Ultra Deep Field (HUDF; Beckwith et al. 2006), a 11 arcmin^2 region in the Southern Sky. As the simulated images were designed to mimic the FDS, using real images from this survey allowed us to test the optimised parameters with similar imaging conditions, where they would be expected to perform well. The additional use of IAC Stripe82 and HUDF images allows us to examine the consistency of parameters on images with very different imaging conditions.

While the FDS and IAC Stripe82 are deep surveys using ground-based telescopes, namely the VLT Survey Telescope (VST) and the SDSS Telescope, respectively, the well-studied HUDF extends our analysis to the higher resolution, space-based data from the *Hubble* Space Telescope. In terms of depth, the HUDF is the deepest with a 5σ point source depth of ~29 mag computed over 0.6″ apertures (see Bouwens et al. 2009, Table 1), which corresponds to a surface brightness limit of $\mu_{V_{606}} \sim 32.5\,\mathrm{mag\,arcsec^{-2}}$ in the V_{606}-band, computed as a 3σ fluctuation with respect to the background of the image in $10 \times 10\,\mathrm{arcsec}^2$ boxes (3σ; $10 \times 10\,\mathrm{arcsec}^2$). The FDS images in the SDSS r-band have a limiting depth of $\mu_r \sim 29.8\,\mathrm{mag\,arcsec^{-2}}$ (3σ; $10 \times 10\,\mathrm{arcsec}^2$) and the IAC Stripe82 survey is ~1 mag shallower than FDS with a limiting surface brightness depth of $\mu_r \sim 28.6\,\mathrm{mag\,arcsec^{-2}}$ (3σ; $10 \times 10\,\mathrm{arcsec}^2$) and $\mu_g \sim 29.1\,\mathrm{mag\,arcsec^{-2}}$ (3σ; $10 \times 10\,\mathrm{arcsec}^2$). In order to select the deepest imaging from all these surveys in the optical regime, we use the V_{606}-band images in the HUDF and the SDSS r- and g-band images from FDS and IAC Stripe82 respectively.

Additionally, the FDS, IAC Stripe82 and HUDF datasets collectively represent deep data with different surface brightness

[3] http://research.iac.es/proyecto/stripe82/

Table 1. Summary of qualitative evaluation.

	MTObjects	NoiseChisel	ProFound	SExtractor
Optimised parameters	2	20	8	6
Language	Python/C	C	R	C
Clean edges of detected objects	–	✓	✓	Sometimes
Detects elongated galaxy (FDS – Fig. 17)	✓	Fragmented	–	Fragmented
Detects galaxy close to star (FDS – Fig. 18)	✓	Fragmented	–	Fragmented
Detects cirrus (Stripe82 – Fig. 19)	✓	✓	–	Sometimes
Isolates spiral substructures (HUDF – Fig. 24)	✓	–	–	–

depths and spatial resolutions: FDS is >1 mag deeper and two times higher in spatial resolution than IAC Stripe82 (0.2 arcsec pixel^{-1} resolution in FDS (rebinned from the 0.21 arcsec pixel^{-1} of the VST) compared to 0.396 arcsec pixel^{-1} in SDSS) and the HUDF is >2 mag deeper than FDS, with the best resolution currently possible from space ~0.05 arcsec pixel^{-1}. Therefore, the optimised parameters of each algorithm are tested on real images with varying depth and resolution. However, in this work we specifically chose images in the optical wavelengths to test the limits of current detection algorithms for upcoming deeper and wider surveys such as LSST. In future work, a similar analysis to that performed here could readily be extended to other wavelengths.

3.2. Parameter optimisation

To produce a fair comparison of the capabilities of the algorithms, they should be tested with parameters that are as close to optimal as possible. Due to the extremely large parameter spaces of some of the tools, it was not feasible to manually optimise the tools, or to test every possible combination of parameters.

We therefore chose to use an automatic method to select good parameters for each tool. We initially considered using a genetic algorithm for this purpose; however, this proved to be prohibitively slow, as a high number of time-consuming runs of each tool was required. Instead, we used Bayesian optimisation.

Bayesian optimisation is a method of black-box optimisation well-suited for functions that take a long time to evaluate (Jones et al. 1998). It operates by creating a model of how the function behaves, identifying the regions in parameter space where it may perform well or where it may not be well-fitted, and choosing points in these regions to evaluate, in order to improve the model.

In the context of source-extraction tools, the input takes the form of a set of relevant parameters, as dictated by each tool's documentation. The parameters are evaluated by running the tool on a training image, comparing the output to a known ground truth, and choosing a metric (as detailed in Sect. 3.3) as the output score to optimise.

We used the GPyOpt optimisation library (The GPyOpt authors 2016) to perform the optimisations. For each metric, each tool was optimised on every image individually, and the found parameters were then applied to all of the remaining images to assess their performance. The tools' default parameters were used as a starting point. Using the local penalisation method, 120 evaluations were performed on each image in batches of four, and the best set of parameters was chosen.

3.3. Metrics

The quality of a segmentation can be measured both in terms of the presence and absence of ground-truth objects and the similarity between the true objects and segmented shapes.

3.3.1. Matching detections

When measuring detection rates, it is necessary to match detected objects with ground-truth objects. It may be the case that a detected object covers the area of multiple true objects, or conversely that multiple detected objects are found within the area of a single true object. Therefore, a one-to-one mapping is required in order to prevent algorithms from being rewarded for failing to correctly distinguish between sources.

We chose to use the brightest pixel in each object as an identifier, and the detected object containing the brightest pixel in a ground-truth object was matched to this identifier. In the event that a detected object contained the brightest pixel of multiple ground-truth objects, the object containing the pixel with the highest flux was chosen as a unique match.

Three measures made use of this matching procedure:

- Detection recall (completeness): the proportion of objects that are detected.
- Detection precision (purity): the proportion of segments that can be matched to real objects.
- F-score: the harmonic mean of precision and recall:

$$\text{F-score} = 2 \times \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (4)$$

3.3.2. Evaluating areas

In order to quantify the accuracy of the areas of segmented objects, we used a modified version of over-merging and under-merging scores (Levine & Nazif 1981). The under-merging score measures the extent to which objects that should be a single segment are broken into multiple pieces by the segmentation tool. The over-merging score measures the opposite, that is, the extent to which multiple objects are incorrectly combined into a single segment by the tool. Combining these scores gives a measure of the overall quality of the segmentation.

In the original method, the ground-truth segmentation is divided into N segments, $R_1 \ldots R_N$, with areas $A_1 \ldots A_N$, and the test segmentation is divided into M segments, $T_1 \ldots T_M$, with areas $a_1 \ldots a_M$. The original metrics are calculated by finding R_k to maximise $T_j \cap R_k$,for each test segment, T_j:

- Under-merging error (UM):

$$\text{UM} = \sum_{j=1}^{M} \frac{(A_k - (T_j \cap R_k))(T_j \cap R_k)}{A_k}. \quad (5)$$

- Original over-merging error (OM_0):

$$\mathrm{OM}_0 = \sum_{j=1}^{M} (a_j - (T_j \cap R_k)). \tag{6}$$

In these original definitions, we found that the over-merging score did not penalise segmentations, which divided large objects into many small pieces. This meant that tools could find enormous numbers of false positives, fragmenting the ‘background’ segment, without penalty. Consequently, we chose to redefine the over-merging score to become symmetric to the under-merging score, which better takes into account the number and size of segments. We also defined an Area score, which combined the two measures to give an overall score.

- Over-merging error (OM): for each reference segment, R_k, find T_j to maximise $T_j \cap R_k$

$$\mathrm{OM} = \sum_{k=1}^{N} \frac{(a_j - (T_j \cap R_k))(T_j \cap R_k)}{a_j}. \tag{7}$$

- Area score:

$$\text{Area score} = 1 - \sqrt{\mathrm{OM}^2 + \mathrm{UM}^2}. \tag{8}$$

As the Area score alone does not take into account precision and recall, we also defined two combined scores. These give us the ability to optimise for a balanced F-score and Area score.

- Combined score A:

$$\sqrt{\text{Area score}^2 + \text{F-score}^2}. \tag{9}$$

- Combined score B:

$$\sqrt[3]{(1 - \mathrm{OM}) \times (1 - \mathrm{UM}) \times \text{F-score}}. \tag{10}$$

We additionally measure speed, that is, the rate at which images can be processed, in megapixels per second.

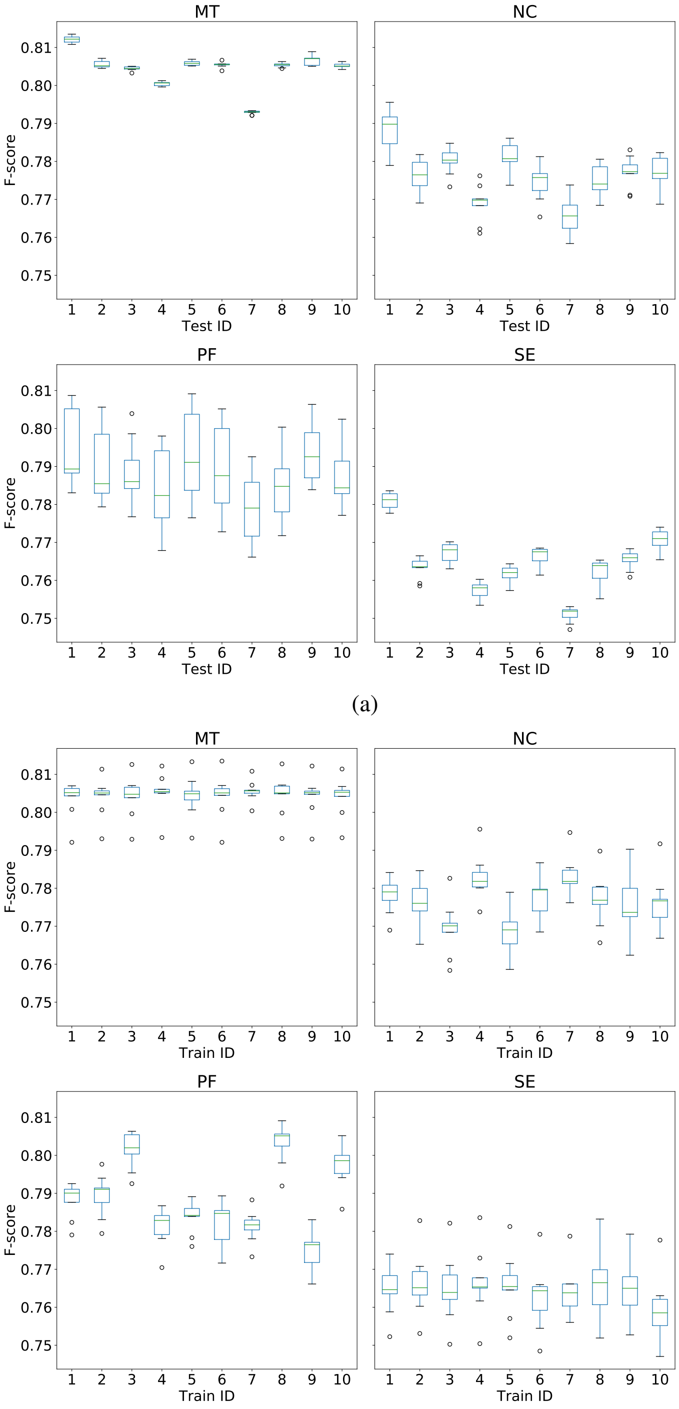
4. Results

Whilst the original intent was to compare all four programs on all metrics, ProFound proved to be very slow to optimise and run, making it impractical for use on large images and surveys. As such, it was optimised only on F-score and Area score. Processing speeds are discussed in more detail in Sect. 4.6.

4.1. Detection accuracy

Figure 4 shows the range of F-scores produced when each tool is optimised for F-score. Two plots are shown for each tool: one in which the scores are grouped by the image being evaluated, and one in which the scores are grouped by the training image used to optimise the parameters. The scores of the training image are excluded from both graphs.

For both MTObjects and SExtractor, it is notable that the scores have smaller interquartile ranges and more varied medians when grouped by test image. This suggests that for these tools, the factor limiting the performance is the structure of each individual test image, rather than the particular parameter set chosen. In contrast, ProFound has a smaller interquartile range when scores are grouped by optimisation image, suggesting that in this case, performance is limited by the image used in the optimisation process.



(a)

(b)

Fig. 4. F-score test distributions. The parameters of each tool were optimised for F-score on each of the ten images, and evaluated on the remaining nine images. Boxes extend from first ($Q1$) to third ($Q3$) quartiles of the results, with median values marked; whiskers extend to the furthest F-score less than $1.5 * (Q3-Q1)$ from each end of the box. (a) F-scores grouped by image evaluated. (b) F-scores grouped by image used to optimise parameters.

Overall, we see the strongest performance from MTObjects, with median scores of over 0.80 for the majority of images. The weakest performance was produced by SExtractor, with scores of under 0.78 in most cases.

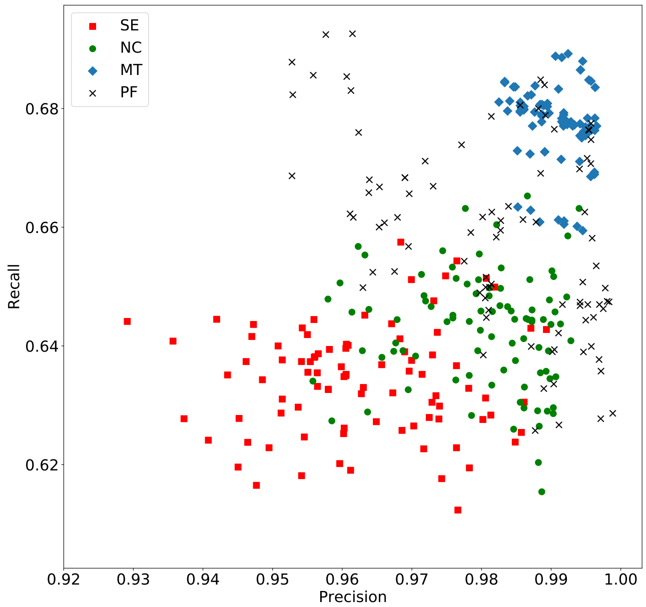


Fig. 5. Precision vs. recall. The parameters of each tool were optimised for F-score on each of the ten images, and evaluated on the remaining nine images.

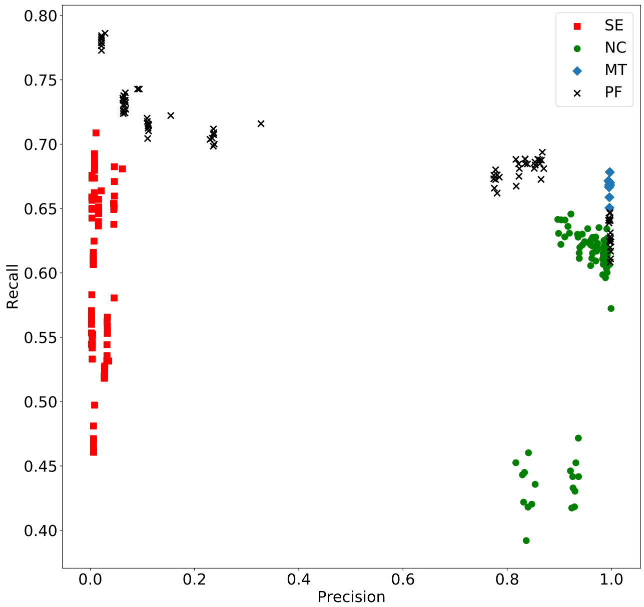


Fig. 6. Precision vs. recall. The parameters of each tool were optimised for Area score on each of the ten images, and evaluated on the remaining nine images.

Examining the precision and recall scores that make up the F-scores shows that all programs are capable of broadly similar performance, with recall between 0.61 and 0.7 and precision greater than 0.93. Whilst the recall scores appear low, many of the faintest objects in the image are not even visible to the human eye, and may in fact be impossible to detect with any tool; these objects are included in order to fully explore the limits of the tools' capabilities. It is therefore useful to regard recall scores primarily as a relative measure, to compare the tools' performances.

Differences between the programs become apparent when the scores are plotted against each other, as shown in Fig. 5. All the tools have a moderate spread of recall scores, which may be caused by differences in difficulty between the individual images.

MTObjects and NoiseChisel both produce generally higher levels of precision than SExtractor; with MTObjects giving a slightly higher maximum value, and a lower spread. ProFound achieves the greatest values for both precision and recall, but has a very wide spread.

When optimised for Area score, SExtractor showed a substantially lower precision; it found an enormous number of false positives, as shown in Fig. 6. Here, we clearly see that optimising for Area score is detrimental to the F-score results. This appears to be the result of a very low threshold being selected in order to maximise the area of large shapes, meaning that a large number of small areas of noise are incorrectly marked as objects.

In contrast, NoiseChisel and MTObjects were capable of increasing their Area scores without substantially compromising their F-scores. ProFound performed inconsistently, covering the full range of precision scores across the ten optimisations.

4.2. Area measures

Unsurprisingly, all tools were capable of reaching higher Area scores when optimised for Area score rather than F-score, as can be seen in Fig. 9.

When optimised for Area score, NoiseChisel and MTObjects both performed well, showing Area scores substantially higher than the other two tools, with MTObjects performing slightly better than NoiseChisel. Both tools also showed lower variation when scores were grouped by test image, as shown in Fig. 7, suggesting that the performance of these tools is being limited by the content of the test images, rather than the parameters found in the optimisation.

In contrast, ProFound showed much greater variability in Area scores when grouped by test image, and indeed, substantial variation between the parameter sets. It also produced the weakest Area scores overall. SExtractor was capable of producing higher Area scores than ProFound, but at substantial cost to precision, as discussed above.

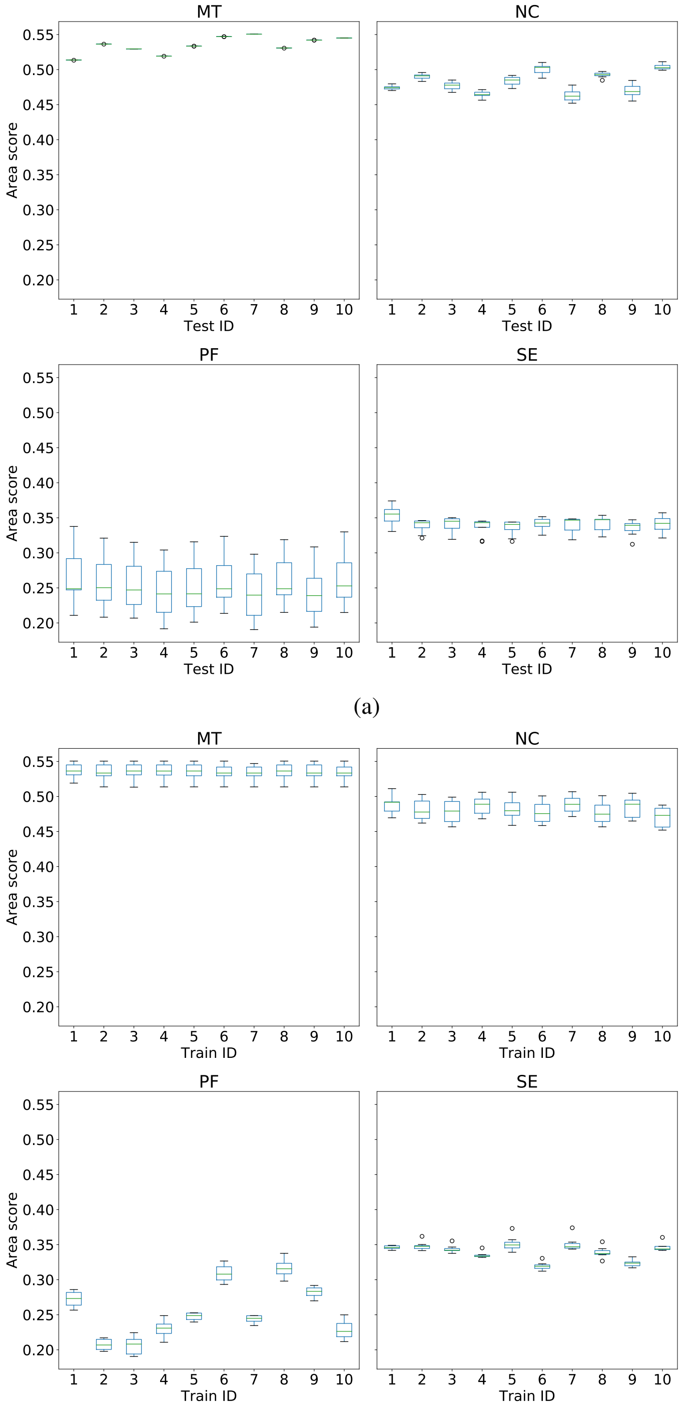
4.3. Combined scores

The two combined metrics offered a way of optimising for both Area and F-score, differing in the balance between the two measures. As such, optimising for these metrics gives an indication of the overall peak performance of the tools.

In practice, both metrics produced broadly similar results in terms of both Area and F-score, as shown in Fig. 8. MTObjects produced the highest values for both F-score and Area score, with NoiseChisel producing slightly lower values in both metrics. SExtractor produced lower F-scores, with a large degree of variability, and substantially lower Area scores, as would be expected from its limited success when optimising purely for area. These results indicate that optimisation for combined scores prevents a large number of spurious detections being found by SExtractor, when compared to Area score alone.

4.4. Overview of optimisation metrics

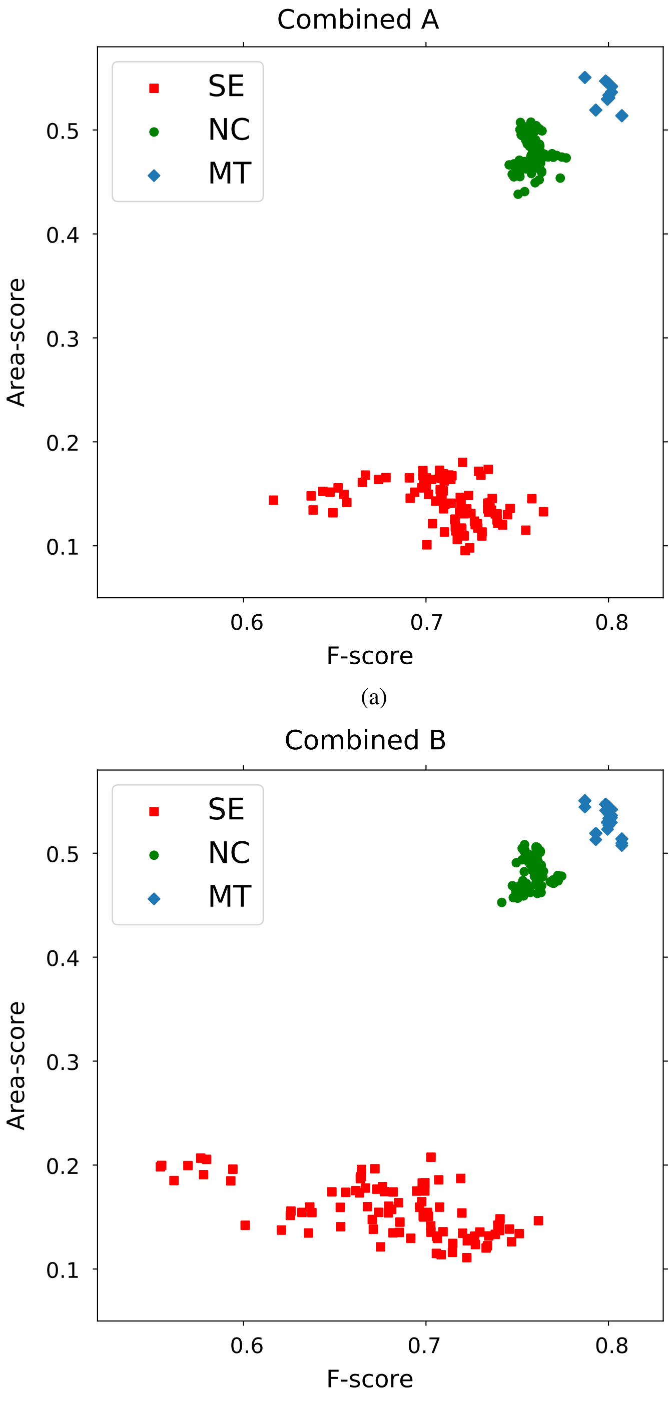
Figure 9 shows an overview of the results of the optimisation in the form of scatter plots of F-score and Area score. Points represent the result of evaluating the performance of the four



(a)

(b)

Fig. 7. Area score test distributions. The parameters of each tool were optimised for Area score on each of the ten images, and evaluated on the remaining nine images. (*a*) Area scores grouped by image evaluated. (*b*) Area scores grouped by image used to optimise parameters.



(a)

(b)

Fig. 8. F-score vs. Area score. The parameters of each tool were optimised for the combined measures on each of the ten images, and evaluated on the remaining nine images. (*a*) Optimised for Combined A. (*b*) Optimised for Combined B.

tools when applied to each image using the parameters found by optimising for each metric on every other image individually. From this, we can make several observations about the tools' performance.

Firstly, the tools designed specifically for locating low-surface-brightness structures (NoiseChisel and MTObjects) are unsurprisingly capable of achieving higher Area scores than the general-purpose tools. Secondly, all the tools must to some degree compromise F-score to obtain a higher Area score, but this trade-off is much greater for the general-purpose tools. Thirdly, MTObjects has less spread than the other tools; indeed, it finds identical parameters and consequently produces identical results for nearly all optimisations over area or combined scores.

Examining Figs. 10 and 11 provides further insight into the behaviour leading to these scores. We see that both NoiseChisel and MTObjects capture regions of light with visually similar

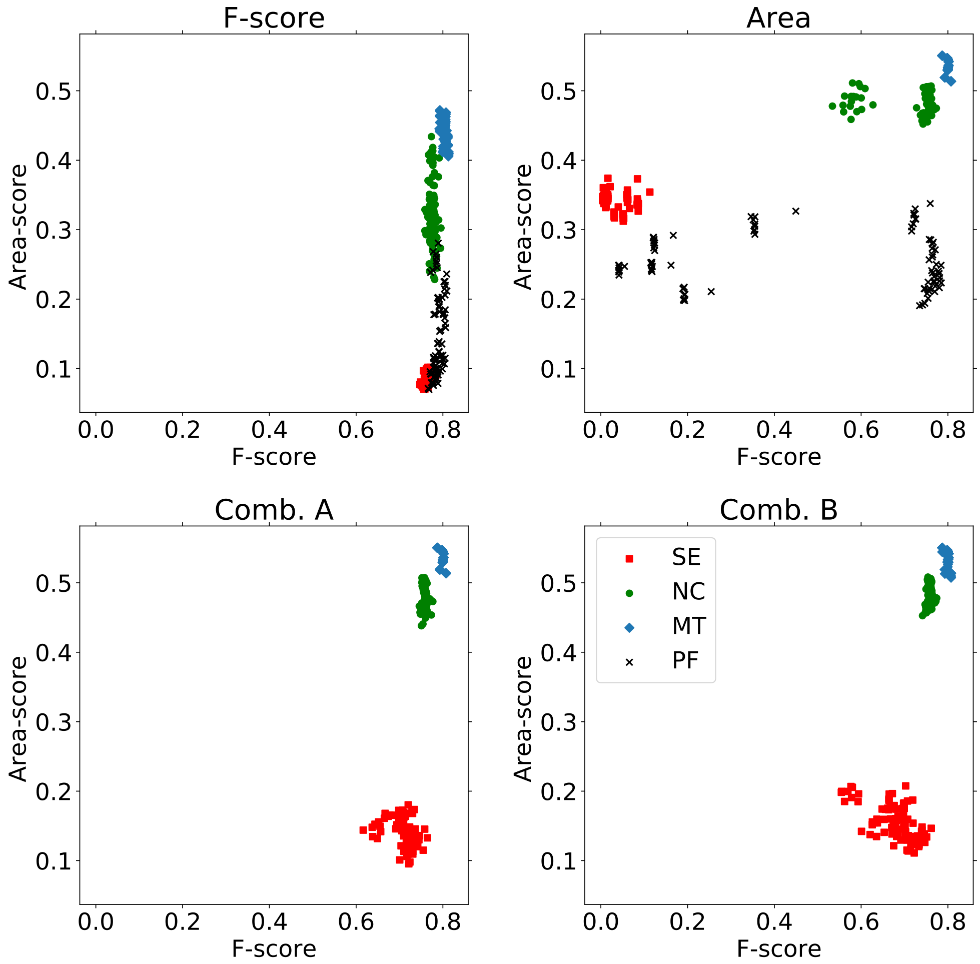


Fig. 9. A summary of test scores for each program using each optimisation method. Each point represents the evaluation of the segmentation of one image using parameters found by optimising on a different image. Each plot shows results for a different optimisation metric. We note that ProFound was only optimised on F-score and Area score.

boundaries, but that MTObjects marks many small, fractured sections in the outer regions as background. Meanwhile, NoiseChisel captures an area of light with fewer holes, but segments it into objects rather arbitrarily. In contrast, SExtractor and ProFound, which both have generally lower Area scores, capture the compact centres of objects and only limited areas of the outskirts.

4.5. Background values

Each program makes internal estimations of background, which may be global or localised. We may also examine the pixels in the image which are not allocated to any segment in the final map. As the simulated images have a flat background with a mean of zero, we can use the mean value of these unallocated pixels as an indication of whether pixels containing no source light are being incorrectly allocated to sources or, conversely, pixels are incorrectly regarded as belonging to sources.

ProFound and SExtractor both consistently overestimated the background, giving values on the order of $10^{-1}\sigma$, where σ is the standard deviation of the background noise (1.1×10^{-12} for the simulated images). This suggests that they are not detecting some parts of the sources; visual inspection of Figs. 10 and 11 confirms that this is the case. There was one exception to this behaviour: SExtractor generally underestimated the background when optimised for area, with values on the order of $-10^{-2}\sigma$. This corresponds to the large number of small false-positive detections made under this optimisation thanks to the low background threshold used (see Table B.1).

MTObjects also underestimated the background, with values of around $-10^{-1}\sigma$ when optimised for metrics including area measures; it underestimated to a lesser degree ($-10^{-2}\sigma$) when optimised for F-score. This behaviour may be a consequence of the holes in the outskirts of objects causing the optimisation process to select parameters that overestimate the size of objects, thereby increasing the solid area within objects but also the number of incorrectly labelled background pixels.

The strongest background estimation performance was produced by NoiseChisel. Whilst optimising for F-score lead to an overestimation in a similar range to SExtractor and ProFound, it produced mean backgrounds in the order $\pm10^{-3}\sigma$ when optimised for a metric including area measures. Not only were the values closer to the goal of zero, but there was also no evidence of systematic over- or underestimation.

4.6. Speed

The speed at which an image can be processed is very important when we consider the size and quantity of images produced by modern surveys.

At its best, SExtractor was the fastest of all the tools by a considerable margin, as shown in Fig. 12. When optimised for area, this advantage vanished completely, potentially due to the vast increase in the number of false positives and large objects. When optimised for combined metrics, processing speed depended heavily on the individual set of parameters, producing a wide spread of speeds.

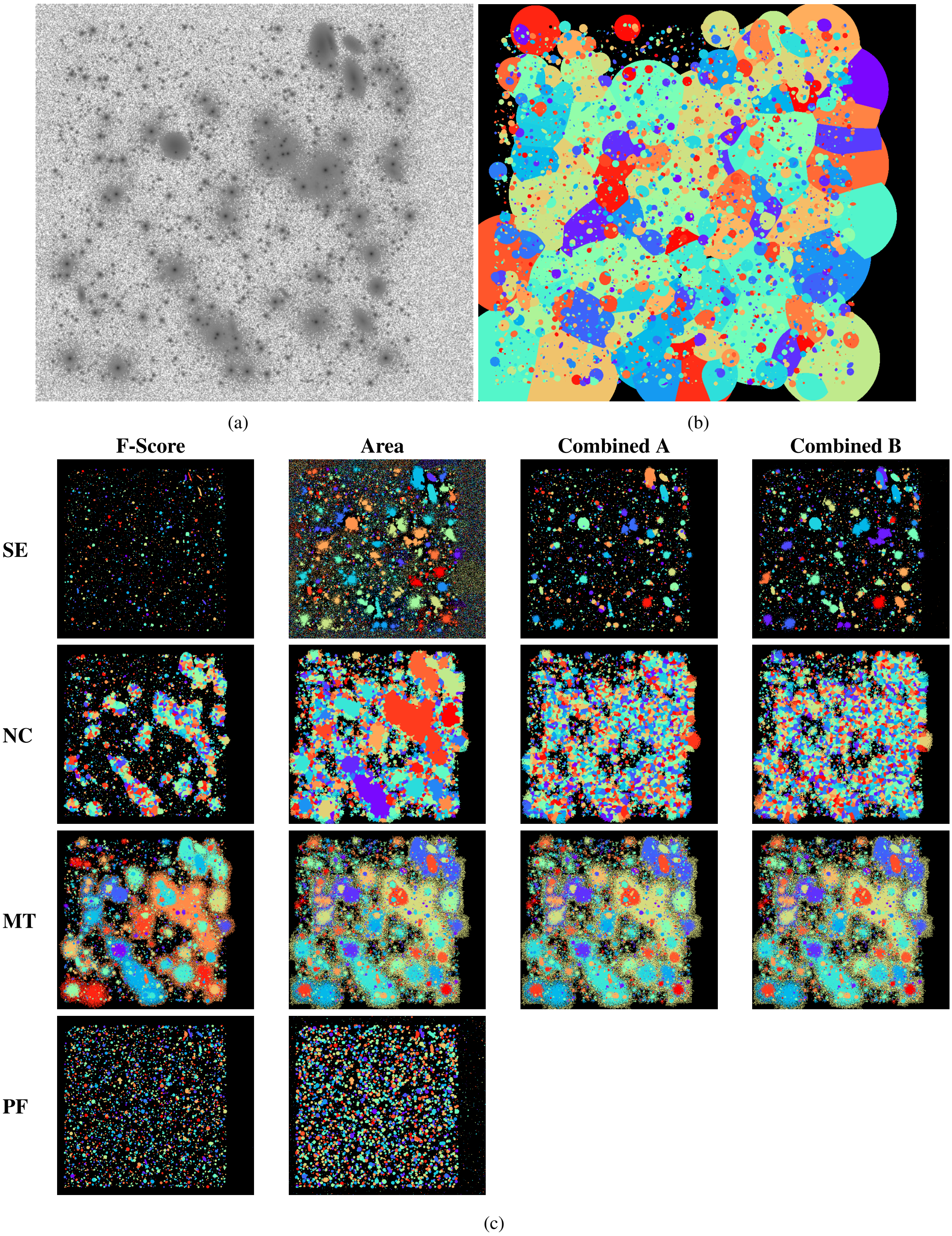
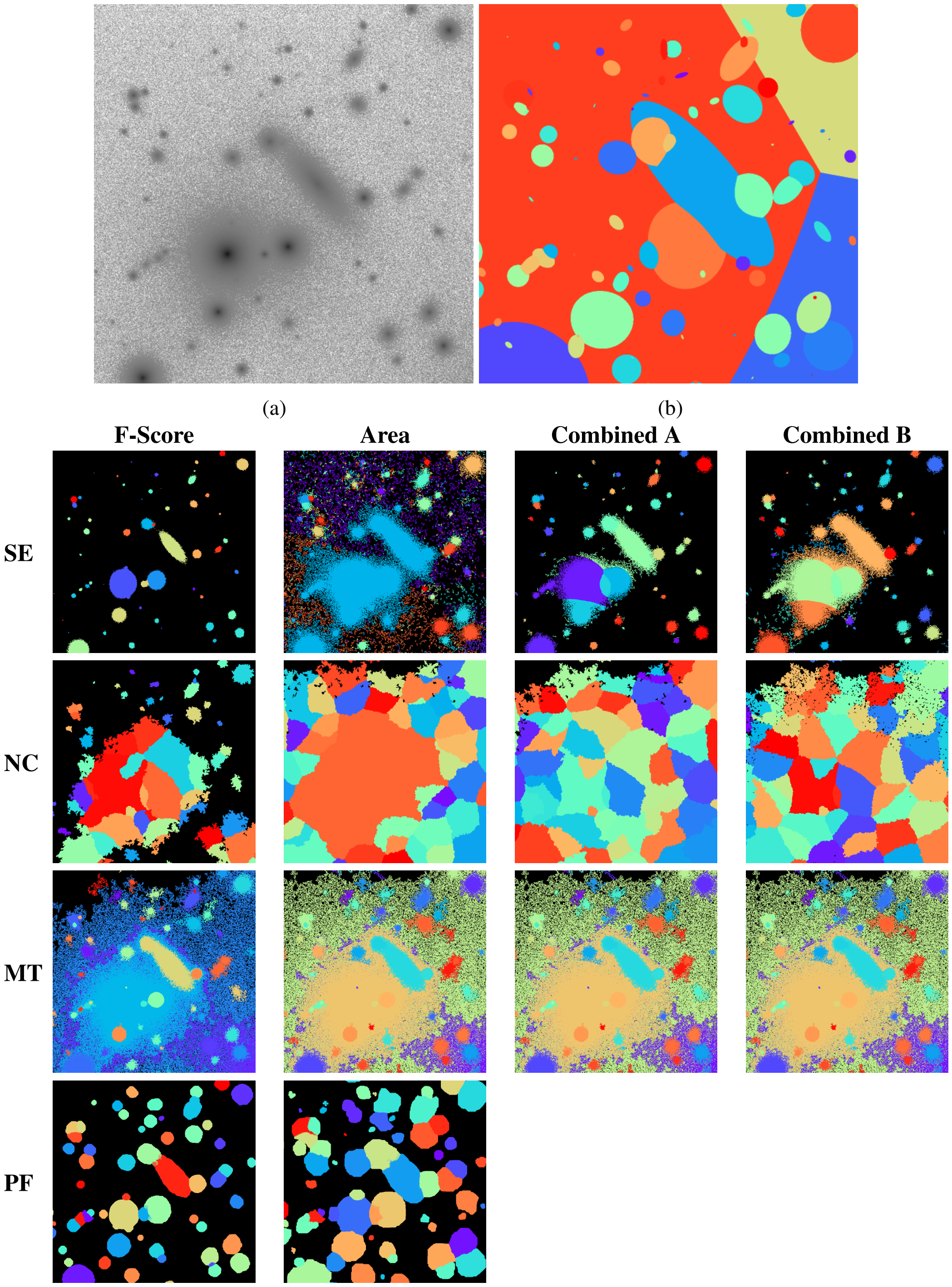


Fig. 10. Segmentations of a full simulated image using the parameters which gave the highest median score for each combination of optimisation measure and tool on the simulated images: SExtractor (SE), NoiseChisel + Segment (NC), MTObjects (MT) and ProFound (PF). The coloured regions label distinct objects, and the black regions make up the background. In the interest of speed, PF was not optimised for Combined A and B. (*a*) Original simulated image. (*b*) Ground truth (0.1σ). (*c*) Segmentation maps.

MTObjects had the most consistent speed across optimisations. Neither SExtractor nor MTObjects used parallel processing, which potentially reduced their speed. It should be noted that the original C implementation of MTObjects is faster than our current Python and C implementation. As reported by Teeninga et al. (2016), SExtractor was only 2.5 times faster than the C version of MTObjects in terms of median performance, and only 1.3 times faster on average. Some code optimisation and using a parallel max-tree algorithm Moschini et al. (2018) should be able to improve the performance in terms of speed.

NoiseChisel showed fast performance when optimised for F-Score alone, but was much slower when Area score was



(a) (b)

(c)

Fig. 11. Segmentations of a section of a simulated image using the parameters which gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10. (*a*) Original simulated image. (*b*) Ground truth (0.1σ). (*c*) Segmentation maps.

included in the optimisation criterion. This appears to be due to a combination of factors; predominantly a lower value for 'detgrowquant', which affects the extent to which objects are grown after detection[4].

As mentioned previously, ProFound consistently had a very long processing time, which greatly reduced its viability as a tool for processing large images from surveys with many images. This is due in part to it writing temporary data to disk, which is discussed in the original ProFound publication (Robotham et al. 2018): ProFound offers a low-memory mode which reduces the amount of data stored, allowing the processing of larger images without a drastic slowdown; however, as noted, the method is fundamentally rather slow. The use of R as the implementation

[4] We find that some non-optimal parameter combinations also cause substantial slowdown, which is due to the program requiring large amounts of memory and consequently writing some data structures to disk.

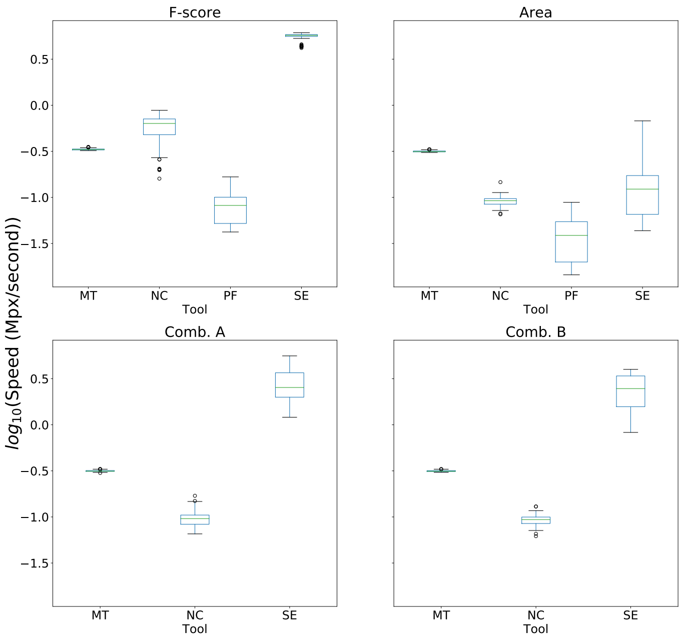


Fig. 12. Distributions of processing speed across all combinations of images and optimised parameters for each tool and optimisation metric.

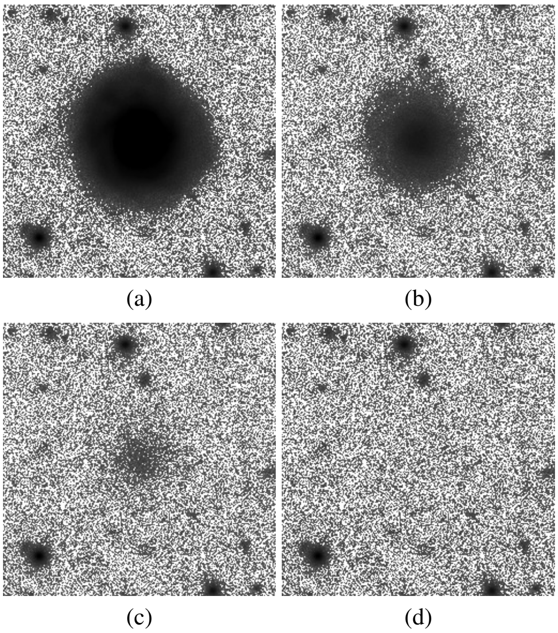


Fig. 13. Galaxy from the EFIGI sample inserted into the FDS frame at the four given brightness scalings. (*a*) 10^{-10}. (*b*) 10^{-11}. (*c*) 10^{-12}. (*d*) 10^{-13}.

language may further reduce the potential speed of the tool. The authors of ProFound are rewriting parts of the code in C++, which should significantly improve its performance.

4.7. Parameter consistency

MTObjects was by far the most consistent of the tools; having only two relevant parameters, it had a much smaller parameter space to explore. While its optimised parameters varied slightly when optimising only over F-score, all other metrics gave the same optimal parameters for all cases but one, as shown in Table B.4.

SExtractor and NoiseChisel, optimised over 6 and 20 parameters respectively, displayed far less consistency in the parameters that were found (Tables B.1–B.3). This could potentially have been reduced by increasing the number of iterations of the optimisation process. However, the similar scores produced using very different parameters suggest that there is no single best choice, and many combinations of settings perform equally well overall, but are better or worse in certain contexts.

4.8. Inserted galaxies

As a final step, we evaluated the performance of the tools on a sample of real galaxies, inserted into a frame of the Fornax Deep Survey (FDS), which the simulated data was designed to emulate. Testing the tools on real galaxies allows us to verify that the behaviour of the tools generalises to galaxies which are not perfect ellipticals.

We selected a sample of 22 galaxies from the EFIGI catalogue (Baillard et al. 2011), which contains images from the fourth data release of the Sloan Digital Sky Survey (SDSS; Adelman-McCarthy et al. 2006). Galaxies were selected with D_{25} (diameter measured at the 25.0 mag arcsec^{-2} isophote, in units of log 0.1 arcmin) between 1.7 and 1.999, a heliocentric velocity <2000 km s^{-1}, and a galactic latitude of between 60° and 70°. This is a representative sample of galaxies in the nearby Universe, with high-quality SDSS images and detailed morphological types. We isolated the galaxy at the centre of each image using k-flat filtering (Ouzounis & Wilkinson 2010), which removed areas of light not connected to the central pixel, whilst preserving the galaxy's internal detail. We then convolved each galaxy with the *r*-band point spread function of the OmegaCAM, and added Poissonian noise.

In order to examine the performance of the algorithms on galaxies of different brightnesses, we scaled the images to four different brightness levels, as shown in Fig. 13. At the brightest level, the brightest pixel in each galaxy had a value on the same order as the brightest pixels in the image, (around 10^{-10}, corresponding to a surface brightness of 21.5 mag arcsec^{-2}). At the faintest, the brightest pixels were barely visible to the human eye (around 10^{-13}, corresponding to a surface brightness of 29 mag arcsec^{-2}). We selected 22 locations in the FDS frame where there were very few objects present in order to minimise interference with the inserted galaxies. We then created four images, with the 22 galaxies inserted into the same locations in the FDS frame using a different brightness scaling for each image. We then ran all four tools on each image with the four sets of optimised parameters obtained on the simulated images.

Whilst using inserted galaxies meant that we had a ground truth for those galaxies, there may still have been other objects present around them in the FDS frame, which would also be detected by the tools. This means that we are unable to rely on the previously defined metrics, as other detected objects would be marked as false positives and raise the under-merging error.

Instead, we use a modified process to determine whether an inserted galaxy has been detected. If the brightest pixel in an object is contained within a non-background segment of the segmentation map and is also the brightest pixel in that segment, we determine that the object has been detected.

Additionally, we classify detections into two types: those where the galaxy has been mostly detected as a single object, and those where the algorithm has substantially fragmented the galaxy. To do this, we check for other detected segments whose brightest pixel is contained within the area of the inserted galaxy,

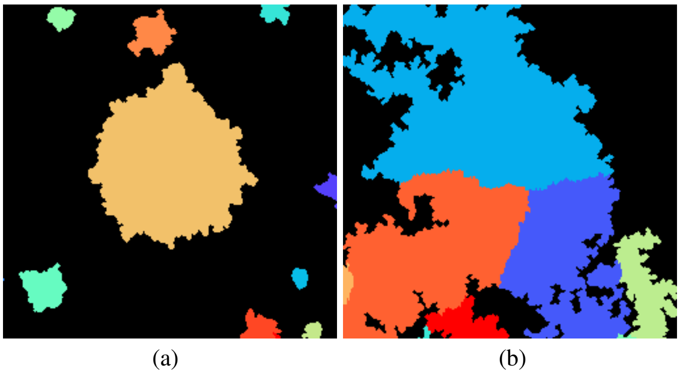


Fig. 14. Segmentation maps showing the two defined types of detection. (*a*) A 'whole' detected galaxy. (*b*) A fragmented galaxy.

suggesting that they are not primarily detecting some other background object. If there are multiple segments which meet this criteria, we check that the segment containing the most light from the inserted galaxy has at least ten times the amount of light contained in the segment containing the second-most light. If it does, we mark the detection as 'whole'; otherwise as fragmented. Whilst lacking the numerical accuracy of the previously defined Area score, this provides an indication of the quality of detections. Examples of the two types of detection are shown in Fig. 14.

The results of this process are summarised in Fig. 15. At higher brightnesses, most tools perform well, with only NoiseChisel failing to detect any objects at the two highest brightness levels.

At fainter levels, the tools show more variation. At the 10^{-12} brightness level, ProFound shows the strongest performance, fully detecting nearly over 90% of the objects under an Area score optimisation. SExtractor shows high levels of fragmentation at this level, consistent with its low Area score found on the simulated data. NoiseChisel maintains a roughly consistent rate of fragmented detections, but with fewer detections overall, whilst MTObjects begins to show some fragmentation and a lower detection rate at this level.

At the faintest brightness, very few of the inserted galaxies are visible to the human eye, and this is reflected in the results. Again, ProFound has a stronger performance than the other tools, with up to 40% of galaxies detected, but a higher rate of fragmentation than at higher brightness levels. SExtractor reaches a similar detection rate under an Area score optimisation, but only produces fragmented detections; visual inspection shows that this is due to the tool finding many tiny objects, as with the simulated images. Both NoiseChisel and MTObjects find very few objects at this low brightness level.

These results are generally consistent with the results shown in the preceding sections: all tools were capable of similar F-scores, and this is reflected in the similar detection rates found on the inserted galaxies. Similarly, variations in Area score roughly correspond to the fraction of the inserted galaxies with substantial fragmentation for each tool, particularly at the 10^{-12} brightness level.

It is notable that when the inserted galaxies are fainter, optimisations for F-score appear to be less effective than optimisations for Area score. This may be due to the higher sensitivity to noise and lower thresholds generally found in area-based optimisations causing the fainter objects to be detected, whilst the F-score-based optimisations ignore these objects in order to minimise false detections.

5. Qualitative evaluation

In this section, we evaluate how the optimised parameters for each tool transfer to different surveys and instruments. We selected three surveys for application of the tools, using the parameters with the highest median test score following the optimisation process: the Fornax Deep Survey (FDS; Iodice et al. 2016; Venhola et al. 2018); the IAC Stripe82 Legacy Project (Fliri & Trujillo 2016; Román & Trujillo 2018); and the HUDF (Beckwith et al. 2006). All of these datasets are deep surveys, with surface brightness limits fainter than $\mu \sim 28\,\mathrm{mag\,arcsec^{-2}}$, and have been used in several studies of galaxies of low surface brightness; for example, Venhola et al. (2017, 2019) and Iodice et al. (2019) for FDS, Román & Trujillo (2017a,b) for IAC Stripe82, and Oesch et al. (2009) and Bouwens et al. (2008) for HUDF. As far as we are aware, all these works used SExtractor for masking sources of light and processing observational data. Therefore, evaluating the quality of segmentation for these deep datasets using the other available source-extraction tools is an added value to ongoing research on faint structures of galaxies.

Moreover, using the source-extraction tools to derive segmentation maps of a completely new dataset with the 'best' optimised parameters allows us to assess whether or not the parameters perform in a consistent manner across different datasets acquired in very different conditions. It is also a test of the practical applicability of each tool to large astronomical surveys of the future, such as those produced by Euclid (Amiaux et al. 2012) and the LSST (Ivezić et al. 2019).

The 'best' parameters derived from our optimisation scheme for each test score that are used for the tools are highlighted with an asterisk in Appendix B.

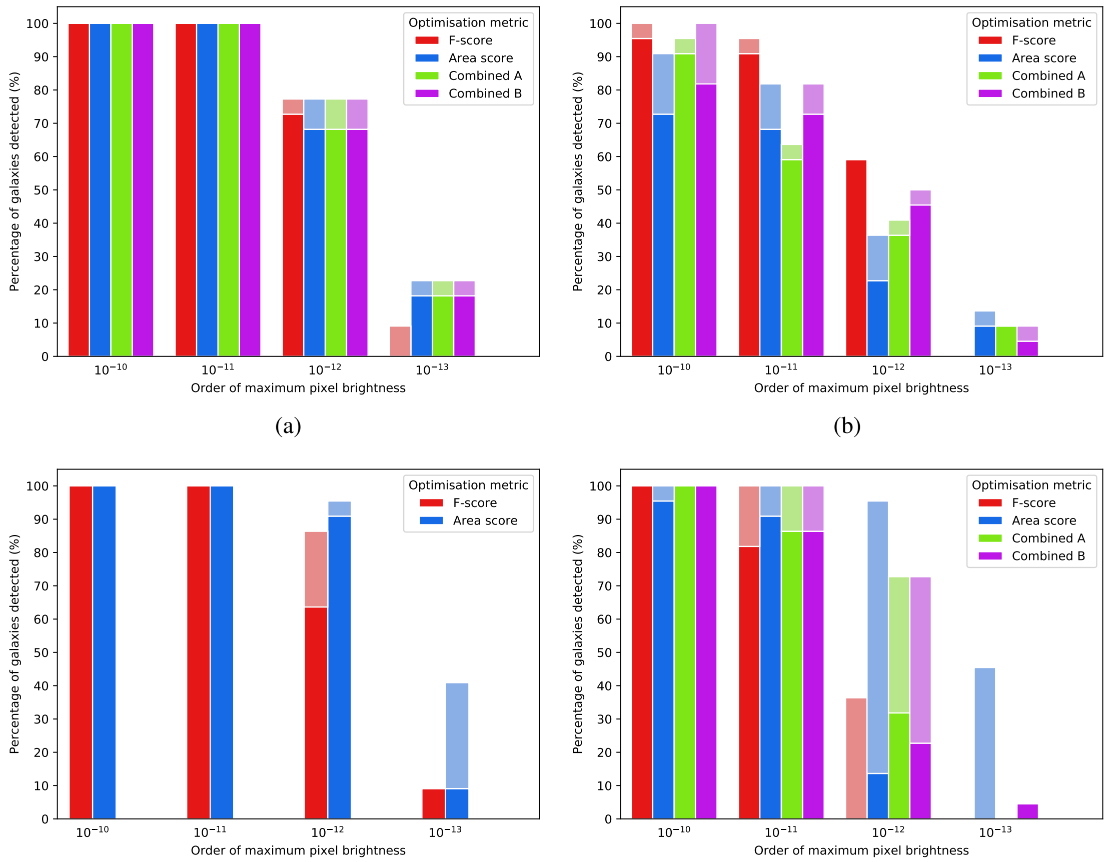
5.1. *Fornax Deep Survey*

As the simulated images were created using the characteristics of the Fornax Deep Survey (FDS), using images from the real survey allows us to check that the parameters found on simulated data perform similarly on data that contain more unusual structures. The limiting surface brightness for *r*-band images of FDS is $29.8\,\mathrm{mag\,arcsec^{-2}}$ (3σ; $10 \times 10\,\mathrm{arcsec^2}$; Venhola et al. 2017).

Here we show a complete frame of the survey, and two smaller areas of the same frame, containing faint and challenging objects. For each combination of training image and optimisation method, the parameters with the highest median test score on the simulated dataset were used.

It is clear from Fig. 16 that the parameters lead to a very similar performance with the real images to with the simulated images. MTObjects and NoiseChisel both capture similar areas of light, but segment them very differently; whilst ProFound and SExtractor capture only the centres of objects.

Examining smaller details of the images gives more insight into behaviour on challenging sources. Figure 17 shows the segmentation of a faint, elongated galaxy. SExtractor only detects a small area of the galaxy when optimised for area, and incorrectly merges it with other surrounding objects; in all other optimisations it fails to detect the galaxy at all, perhaps because of an overly high detection threshold. ProFound detects small blobs covering the area of the galaxy, but does not identify an underlying structure. Similarly, NoiseChisel, whilst locating a larger area of light, breaks it into chunks appearing to correspond to smaller objects, losing the large structure. MTObjects was the only tool to capture the entire structure as one object, but incorrectly labelled it as the same object as the bright source in the



(a)

(b)

(c)

(d)

Fig. 15. Percentage of inserted objects found, grouped by tool, brightness scaling, and optimisation metric, using the parameters which gave the highest median score for each combination of optimisation measure and tool. Lighter, stacked bars represent galaxies that are detected but substantially fragmented. (*a*) MTObjects. (*b*) NoiseChisel. (*c*) ProFound. (*d*) SExtractor.

bottom right corner, and also connected it to the outskirts of the object in the bottom left.

Figure 18 contains another faint structure located near a bright star, which is extremely difficult to visually detect. All four tools struggle to produce ideal results in this situation. As before, ProFound and SExtractor do not detect the faintest parts of objects, which here gives the advantage of allowing both tools to distinguish between smaller sources. SExtractor again produces a high number of false positives when optimised for area, but does begin to detect areas of structure in Combined A and B. In contrast, ProFound produces a blobby segmentation, with less visual similarity to the input image, but again covering a good deal of the smaller structures. NoiseChisel and MTObjects mark almost all of the image section as containing sources, but with a very different segmentation. The area optimisation of NoiseChisel fails to detect any substructures in this part of the image, marking all objects as a single large structure. In the other optimisations, it shows very little visual similarity to the input image. MTObjects correctly detects many of the sources in the area, although it again joins the outskirts of some objects, and produces a ragged appearance.

5.2. IAC Stripe 82 Legacy Project

As an added layer of generalisation, we test the parameters with the highest median test score found for each combination of training image and optimisation method on deep g-band IAC Stripe82 images. The limiting surface brightness is 29.1 mag arcsec^{-2} (3σ, 10×10 arcsec^2; Román & Trujillo 2018).

These images consist of faint and diffuse structures such as Galactic cirri, tidal streams, interacting galaxies, and include scattered light from point sources.

Similarly to the segmentation of the simulated images seen Figs. 10 and 11, we find that SExtractor detects the least amount of light compared to the other tools. In particular, it misses large portions of the Galactic cirrus structure in Fig. 19, even when optimised for the Area score. As in the case of the simulated images, the fact that many smaller objects (including many false positives) are detected in the background when optimised for the Area score is most likely a consequence of the very low threshold used to find larger areas. However, the Galactic cirrus structure is highly extended and diffuse with low- and high-density regions, so the tool is unable to segment the structure as a single object, and fragments it into several pieces. However, this ‘failure of detection’ may be taken advantage of (with some manual intervention) for studying the properties of Galactic cirrus (Román et al. 2020).

Remarkably, the performance of SExtractor on the image of the interacting galaxies connected with a tidal stream in Fig. 20 is much better on the main objects with parameters optimised for Area score and both combined scores, whilst performing poorly in the background. Similar observations can be made in the case of the elliptical galaxy with a large stream in Fig. 21, but this stream is much fainter than in the interacting galaxies case, and

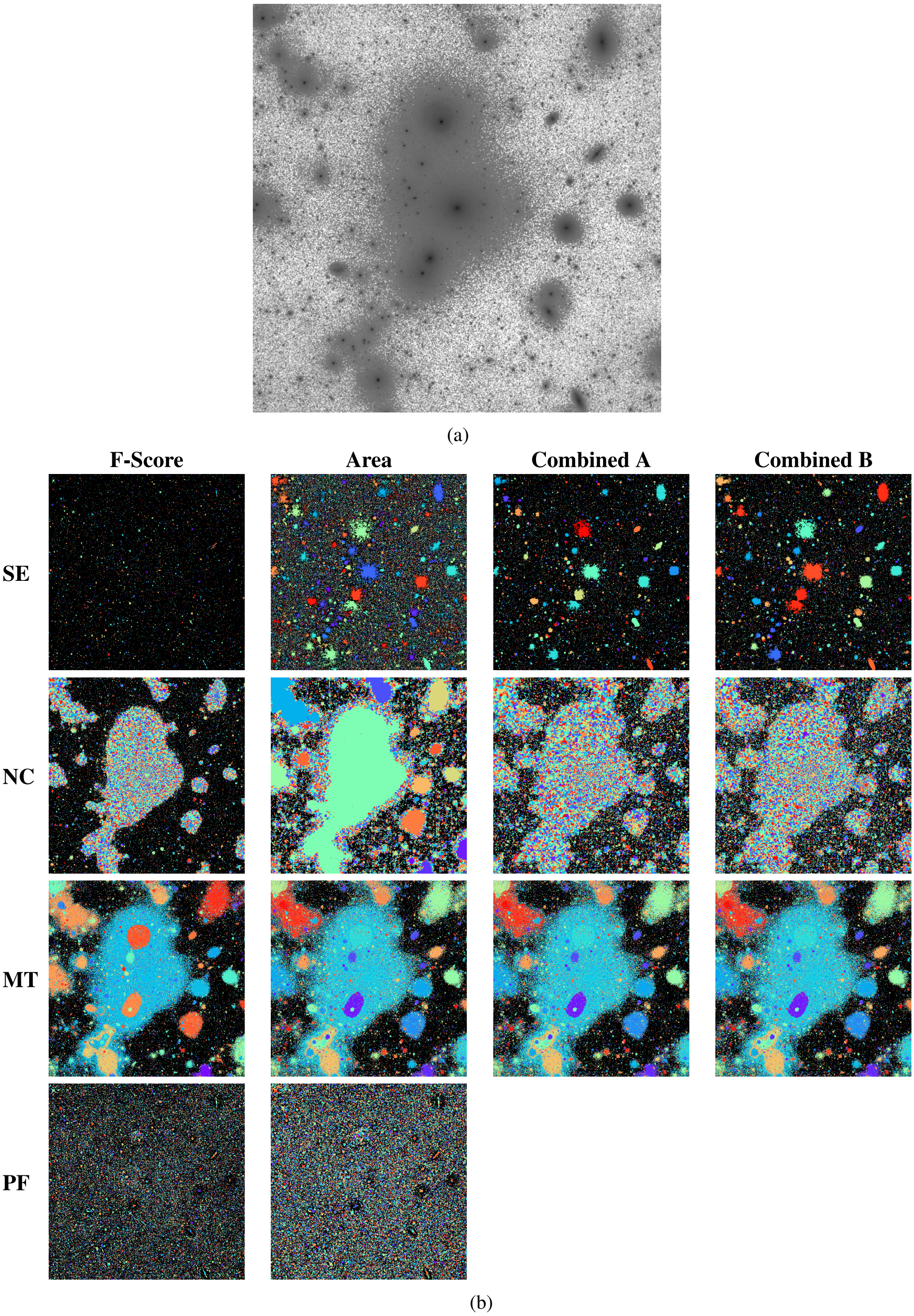
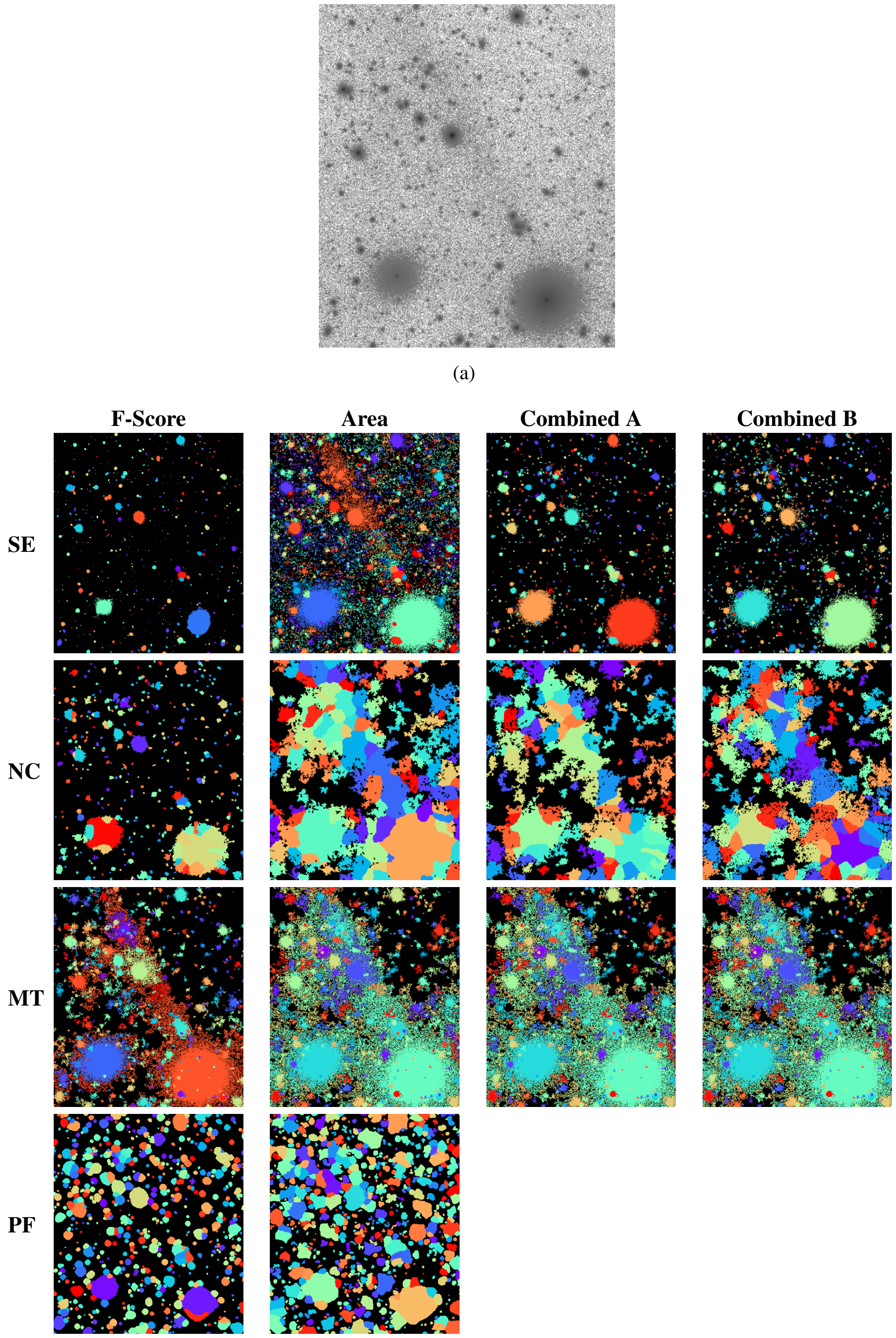


Fig. 16. Segmentations of a complete FDS field (field 11). (*a*) Input image – the *r*-band of field 11 of the FDS. (*b*) Segmentations of the field using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

SExtractor detects the stream in fragments (similar to the Galactic cirrus).

In contrast, for all the IAC Stripe82 images, both NoiseChisel and MTObjects detect the largest amount of light as distinct objects or diffuse regions (reflected in the highest optimisation scores). Visually, the performance of NoiseChisel seems better when optimised for F-Score compared to the other scores, but there is still diffuse light around the objects which have gone undetected. When optimised for Area score or the combined scores, this missing light is recovered, but as mentioned previously, the algorithm seems to segment structures within larger objects rather arbitrarily. When comparing the outputs



(a)

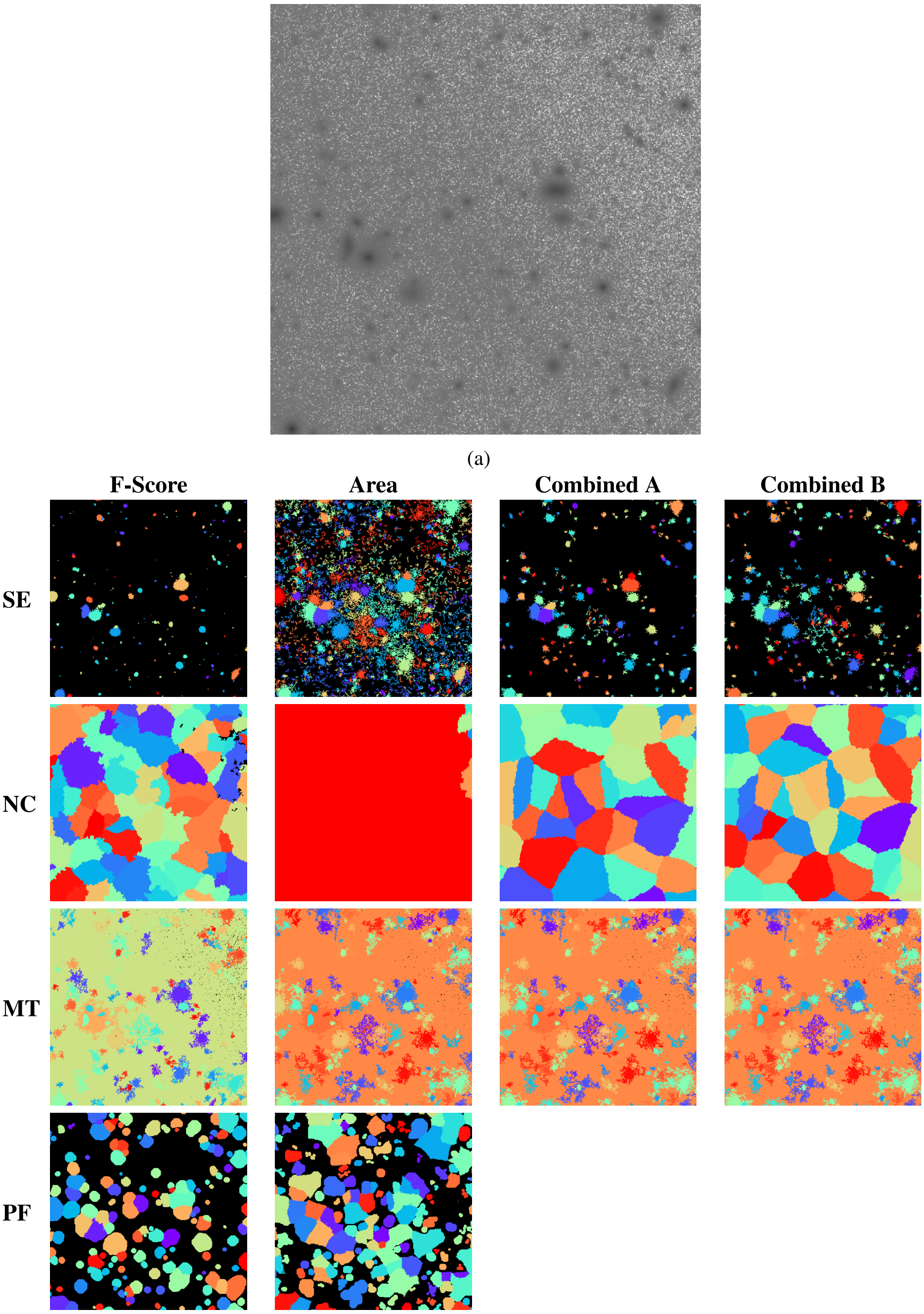
(b)

Fig. 17. Segmentations of a section of an FDS field (field 11), showing a low-surface brightness galaxy. (*a*) Input image – the *r*-band of field 11 of the FDS. (*b*) Segmentations of the field section using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

from each optimisation method, we can see that the substructure is segmented quite differently in all the IAC Stripe82 examples. This is probably a consequence of growing the 'clumps' (as detected in the CLUMPS output of Segment) to cover the full detected area; if the detected area is different, then the growth of the clumps seems to vary. This effect is visible in comparing NoiseChisel's output when optimised for all four measures in all the IAC Stripe 82 examples. The fact that the substructure over the detected regions seems visually arbitrary may not be an issue in some cases, such as when segmentation maps are used for reducing datasets where all pixels with a significant amount of signal above the background needs to be masked for processing (see e.g. Borlaff et al. 2019), or when the user is simply not concerned with the substructure of astronomical sources[5].

However, for studies where more accurate segmentation of tidal streams and nested objects (or substructure) is required for photometric calculations, it is not possible to automatically

[5] The NoiseChisel manual (https://www.gnu.org/software/gnuastro/manual/html_node/NoiseChisel.html) states that the user may choose to run Segment after NoiseChisel depending on whether they want to analyse the substructure of sources.



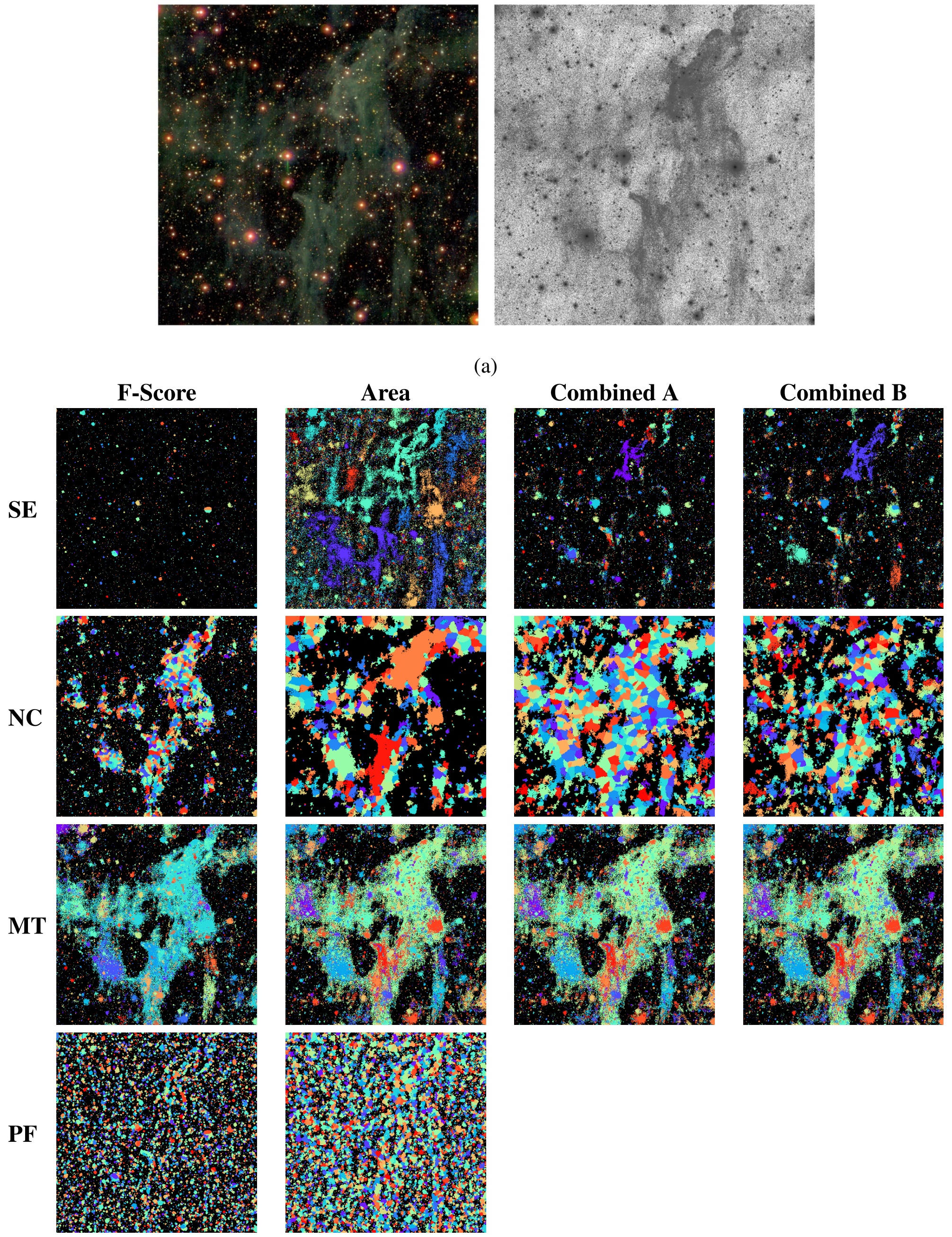
(a)

(b)

Fig. 18. Segmentations of a section of a FDS field (field 11), showing a very faint structure in the lower centre. (*a*) Input image – the *r*-band of field 11 of the FDS. (*b*) Segmentations of the field using the parameters that gave the highest median score for each combination of optimisation measure and tool.

allocate these fragmented regions to their host structure, and the user may need to manually select regions of interest. This is especially visible for the large Galactic cirrus in Fig. 19 and the faint stream in Fig. 21 where the structures are segmented into separate objects of all kinds of shapes.

For the same IAC Stripe82 examples, a similar observation can be made for MTObjects, but the partitioning better follows the visual shape of all objects (background and nested). This behaviour means that the user is able to make a visual mapping between the input image and segmentation map much more easily, if they need to manually select regions of interest. In comparison, the outputs of the tool when optimised for the different scores are very similar; the outputs for the area and combined scores are the same, and the only visible difference



(a)

(b)

Fig. 19. Results for IAC Stripe82 field `f0363_g.rec.fits` showing a large structure of Galactic cirri. (*a*) *Left*: *gri*-composite image. *Right*: *g*-band input image in log scale. (*b*) Segmentation maps using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

with F-Score is the extent to which the edges are fractured outwards. Compared to the other tools, the existence of these highly fractured edges of the segmented regions in MTObjects may not be an appealing characteristic for the user if smoother edges are required; such as for instance to make photometric calculations, such as the total magnitude of objects[6].

Another characteristic of MTObjects can be seen in the field contaminated by a cluster of bright stars to the right of an elliptical galaxy in Fig. 21. MTObjects allocates the diffuse stream and faint halo around the core of the galaxy to the cluster of stars. This is clearly a problem with how the detected regions are represented. MTObjects is finding the diffuse regions in the image (at least those that could be visually identified in this example), but allocating them to the wrong object. This means that the user will need to once again manually select the regions that belong to the galaxy, and this may not always be possible to identify in advance when dealing with deep datasets. Apart from these exceptions, MTObjects performs fairly similarly and consistently across the IAC Stripe82 images tested in this work.

Due to speed, at the time of writing we are only able to complete the optimisation of parameters for ProFound using the

[6] Of the tools, this effect in the segmentation maps can only be controlled in NoiseChisel without compromising the extent to which objects are detected.

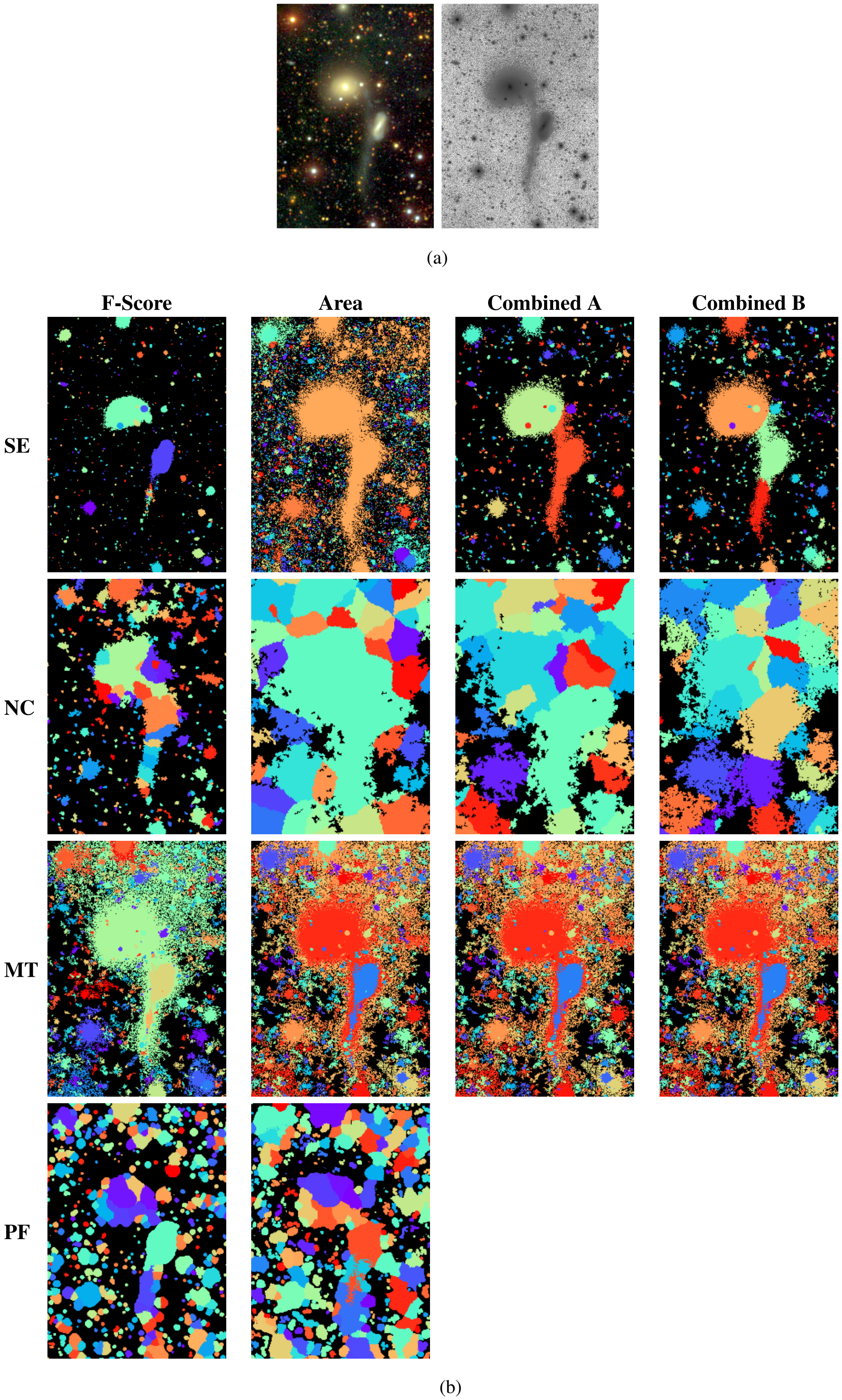
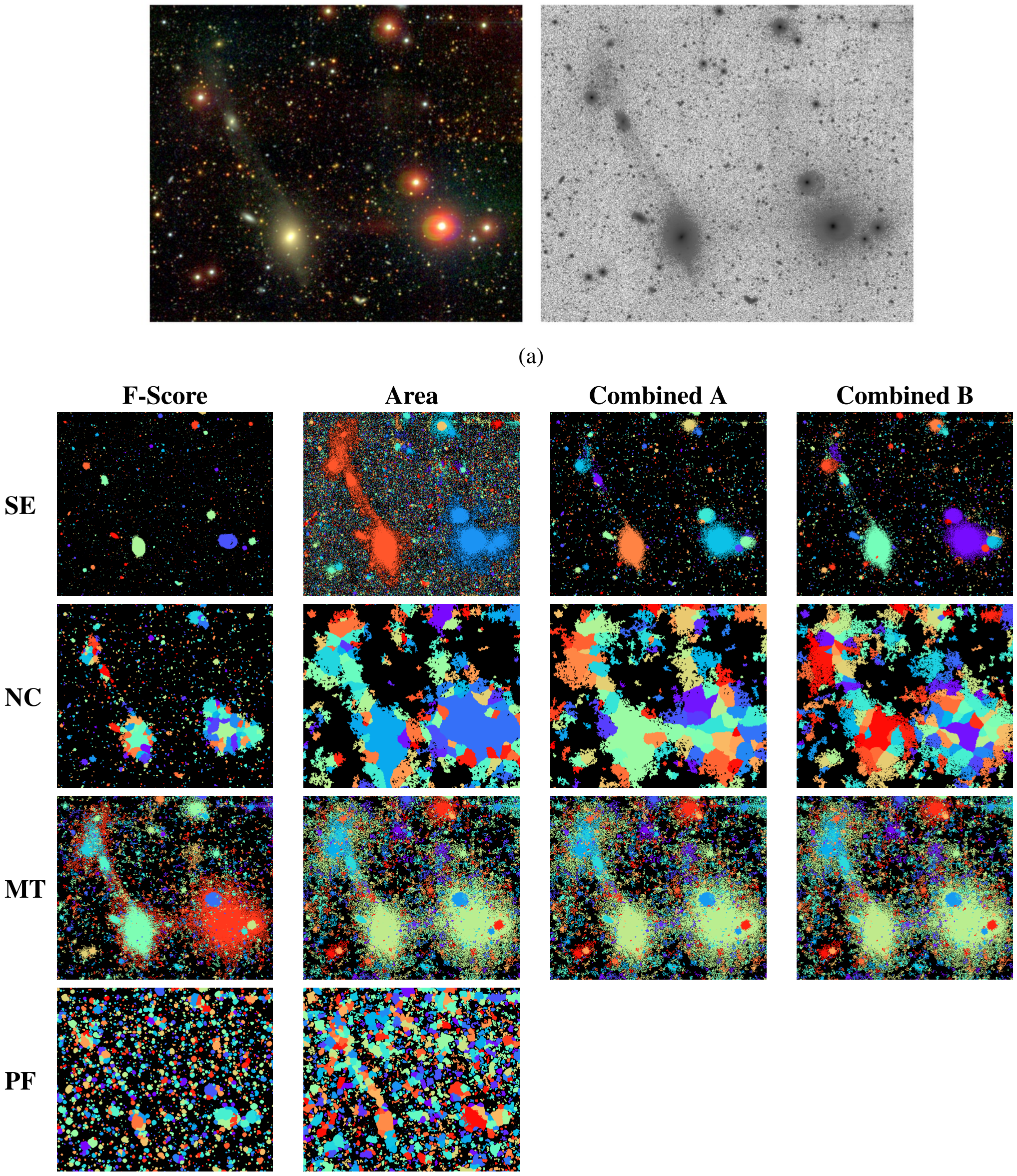


Fig. 20. Results for IAC Stripe82 field cropped to show two interacting galaxies (SDSS J031943.04+003355.64 and SDSS J031947.01+003504.44). (*a*) *Left*: *gri*-composite image. *Right*: *g*-band input image in log scale. (*b*) Segmentation maps using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.



(a)

(b)

Fig. 21. Results for IAC Stripe82 field zoomed in on an elliptical galaxy with an extended, very faint tidal stream (SDSS `J235618.80-001820.17`) and a bright collection of stars with a significant amount of scattered light contaminating the galaxy from the right. (*a*) *Left*: *gri*-composite image. *Right*: *g*-band input image in log scale. (*b*) Segmentation maps using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

F-Score and Area score. In this work, we find that when optimised for these scores, only in the merging galaxies case in Fig. 20 does the tool segment the galaxy shape and its companion (though also fragmented into several arbitrary pieces, as in NoiseChisel's segmentation). In all of the other images, the large galaxies or structures are barely visible, and only because our eye is able to connect the smaller fragments into one connected region.

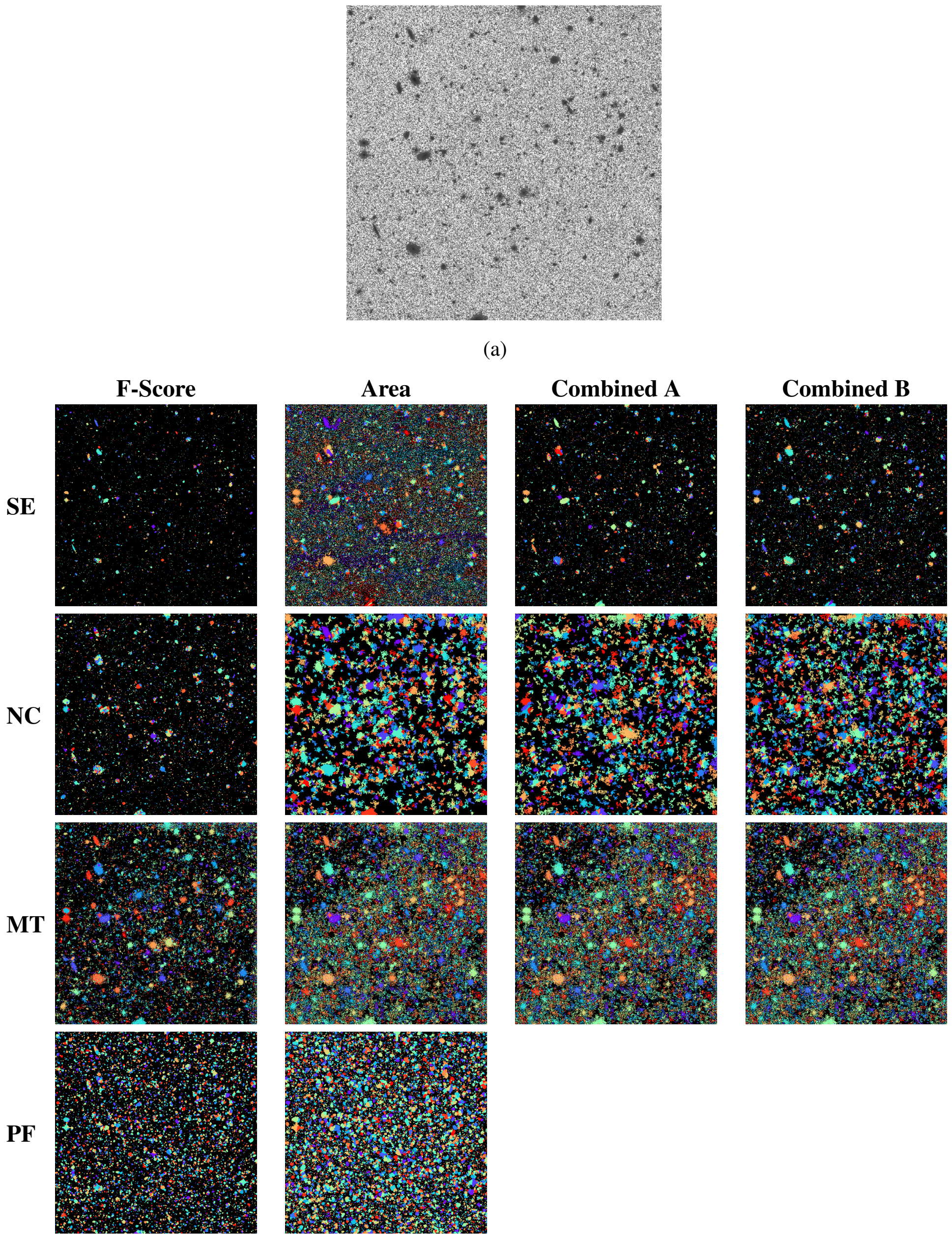
5.3. The Hubble Ultra Deep Field

In order to examine the behaviour of the tools on space-based observations, we ran the tools on the V_{606}-band of the HUDF. As mentioned in Sect. 3.1, the HUDF is the deepest data used in this work, with a point source depth of 29.3 mag (Beckwith et al. 2006) which is equivalent to a limiting surface brightness depth of $\mu_{V_{606}} \sim 32.5\,\mathrm{mag\,arcsec}^{-2}$ (3σ; $10 \times 10\,\mathrm{arcsec}^2$).

As the original drizzled image contained wide, zero-valued borders, we rotated and cropped it to contain as much of the field as possible, while excluding the borders. We then ran the tools on the image, using the same optimised parameters as in the previous sections. We show here the complete image (see Fig. 22) and two smaller areas of interest containing a type of feature not common in the other surveys: face-on spiral galaxies with visible substructures.

Besides these artefacts, the behaviour of all four tools on the HUDF image appears to be generally similar to their behaviour on the images from other surveys, despite the higher depth and the different telescope type.

This is further corroborated by the results shown in Fig. 23, which shows a face-on spiral galaxy, as well as



(a)

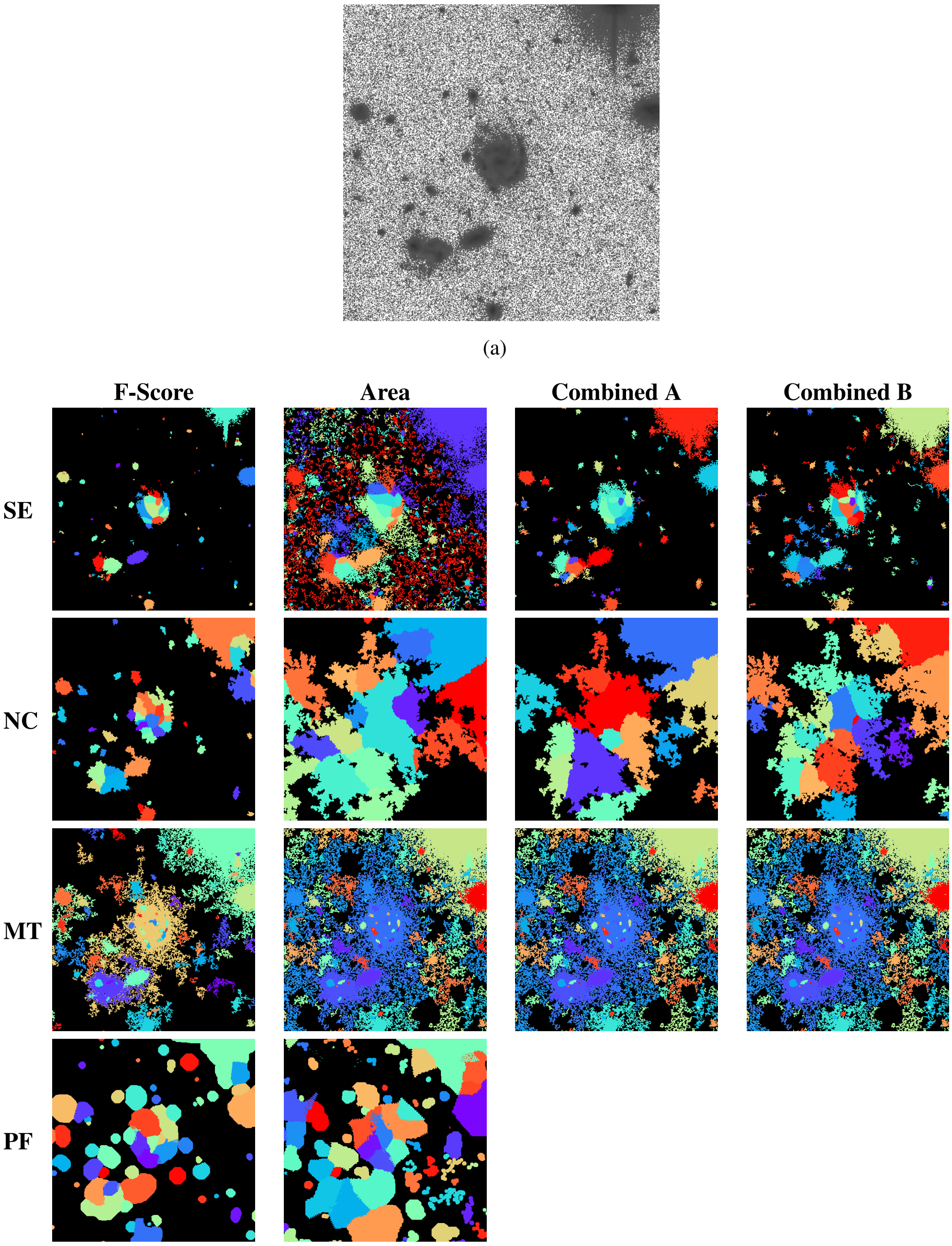
(b)

Fig. 22. Segmentations of the rotated and cropped *Hubble* Ultra Deep Field. (*a*) Input image – the V_{606}-band of the field. (*b*) Segmentations of the field, using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

several smaller elliptical galaxies. As before, SExtractor finds only the bright centres of objects, except when optimised for area; however, it noticeably divides the spiral galaxy into chunks where there is substructure. Somewhat arbitrary division of the galaxy is also visible in the results of NoiseChisel and ProFound; with NoiseChisel capturing more of the outskirts, as before. MTObjects appears to be the most successful at segmenting the spiral, with the majority of the galaxy captured as a single object, with smaller structures nested within it; although, as in previous instances, the outskirts are fractured.

In Fig. 24, which shows a larger spiral galaxy displayed at the same scale, the tools have even greater difficulty segmenting the galaxy in a meaningful way. As before, MTObjects has the most success in separating nested structures without fragmenting the overall structure of the object. NoiseChisel is also consistent with previous behaviour. In contrast, ProFound produces quite different segmentations, with a far less blobby appearance. SExtractor produces quite poor segmentations when area is included in the optimisation; with elongated ovals being found in both of the combined score images.

It must be borne in mind that the parameters were optimised for images in quite different conditions, and so it is difficult to quantify the extent to which these inaccurate segmentations are caused by parameters ill-suited to this context. However, as the behaviour is very similar to that shown in the images from different surveys, it is reasonable to expect that it is largely caused by inherent limitations of the tools.



(a)

(b)

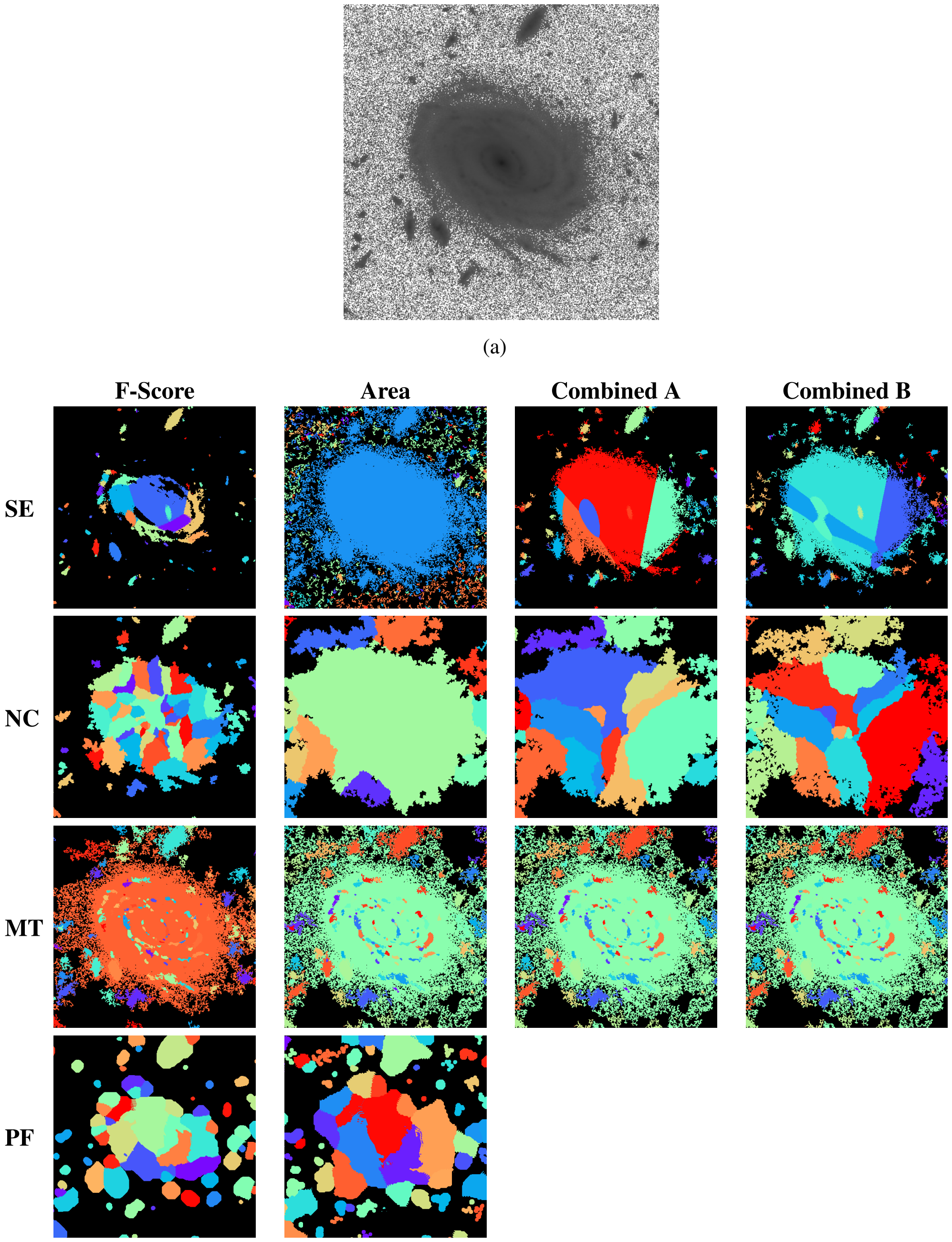
Fig. 23. Segmentations of a section of the *Hubble* Ultra Deep Field. (*a*) Input image – the V_{606}-band of the field. (*b*) Segmentations of the field using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

5.4. Usability

As shown in the parameter tables in Appendix B, only MTObjects reliably found the same set of 'optimal' parameters over multiple optimisations. All of the other tools appeared to have multiple locally optimum parameter combinations. This has a negative impact on ease of use; users manually configuring a tool through trial and error may fail to find globally optimum parameters, and be unaware of this fact. The best parameters may also be dependent on the image used for optimisation, that is, the parameters found for one image or survey may not produce optimal results when applied to others.

All four programs define parameters in terms of the individual steps of the method (e.g. use n thresholds), rather than in terms of how they affect the overall detection (e.g. detect objects to a given degree of certainty). Without using an optimisation framework, users have no choice but to manually select settings that visually produce a good result, but which do not necessarily have any scientific justification for being chosen. This is further exacerbated by large parameter spaces in the cases of NoiseChisel and ProFound allowing the user to infinitely adjust the behaviour of the tools without the implications of their choices being clear. The ability to define performance in terms of the result rather than the process would greatly improve the ease of use of the tools, and would reduce the opacity of their behaviour.

These are no major problems in the case of a user processing a small number of images, but problems arise when large surveys requiring automatic segmentation for many images are


(a)

(b)

Fig. 24. Segmentations of a face-on spiral in the *Hubble* Ultra Deep Field. (*a*) Input image – the V_{606}-band of the field. (*b*) Segmentations of the field, using the parameters that gave the highest median score for each combination of optimisation measure and tool. For more information, see Fig. 10.

considered. The user must select a set of parameters that produces good results for all images in their survey, an impossible task if the tool requires manual tuning on individual images.

6. Conclusions

All the compared tools were capable of a reasonable level of object detection, as measured by F-score. However, ProFound and SExtractor were incapable of detecting the outskirts of objects with any degree of accuracy. NoiseChisel and MTObjects were both much more efficient at finding these fainter regions, but both had other difficulties: the ‘Segment’ tool used in NoiseChisel divided detected light into apparently arbitrary regions, whilst MTObjects produced extremely ragged edges and had a tendency to over-allocate faint regions to the brightest objects. NoiseChisel also produced the most accurate background values.

We found that there appears to be a trade-off between speed and accurate detection of the outskirts of objects. SExtractor was capable of the highest speeds by a substantial margin, but was unable to accurately detect faint regions. MTObjects and NoiseChisel were both able to detect these regions but at the cost of processing speed. There may potentially be improvements to be made on both tools by increased parallelisation and optimisation of the code.

A common weakness in the tools was in accurately deblending nested objects. The MTObjects approach, using tree-based connected morphological filters (Salembier & Wilkinson 2009) deals relatively well when small, faint objects are nested within larger, brighter ones, but performs poorly when more similar objects overlap. In the latter case, the other methods, which are generally based on a form of watershed segmentation (Beucher 1982; Roerdink & Meijster 2000) might give a better result. This is a non-trivial problem, which merits further investigation.

MTObjects was the only tool to find stable parameters across multiple optimisations, suggesting that it requires the least adjustment for individual images, and may be the best-suited for use in automatic pipelines. Furthermore, in the test on simulated data, it consistently outperformed the other methods, regardless of the quality measure used. The likelihood of MTObjects ranking in first place out of four in the case of F-score and Area score in ten tests is about 10^{-6} under the null hypothesis that all tools have equal performance. Despite the modest performance margin with respect to the others, the result is statistically significant.

We find that the optimisation criteria must be chosen carefully in order to produce useful parameters. In particular, we find that optimising for area alone causes a substantial drop in accuracy for SExtractor and ProFound, whereas combining multiple criteria yields more meaningful results.

As discussed in the introduction, the growth of the scale of modern surveys means that there is a need for segmentation tools which are fast, automatic, and accurate. We find that of the tools tested, MTObjects is capable of the highest scores on both area and detection measures, and has the most consistent parameters, whilst SExtractor obtains the highest speeds, but with much lower accuracy. As noted earlier, a faster implementation of MTObjects already exists, and the developers of ProFound are rewriting parts of their tool to improve its speed.

In addition, we present a framework for automated parameter setting and evaluation of astronomical source-detection tools, which is generic, and can be used with any other quality measure or model ground truth. This procedure could be used to analyse improvements to existing tools, as well as to evaluate the capabilities of future techniques.

Acknowledgements. We acknowledge financial support from the European Union's Horizon 2020 research and innovation programme under Marie Skłodowska-Curie grant agreement No 721463 to the SUNDIAL ITN network. NC acknowledges support from the State Research Agency (AEI) of the Spanish Ministry of Science and Innovation and the European Regional Development Fund (FEDER) under the grant with reference PID2019-105602GB-I00, and from IAC project P/300724, financed by the Ministry of Science and Innovation, through the State Budget and by the Canary Islands Department of Economy, Knowledge and Employment, through the Regional Budget of the Autonomous Community. AV acknowledges financial support from the Emil Aaltonen Foundation. The Dell R815 Opteron server was obtained through funding from the Netherlands Organisation for Scientific Research (NWO) under project number 612.001.110. This research made use of Astropy, (`http://www.astropy.org`) a community-developed core Python package for Astronomy (Astropy Collaboration 2013, 2018). This work was partly done using GNU Astronomy Utilities (Gnuastro, ascl.net/1801.009) version 0.7.42-22d2. Gnuastro is a generic package for astronomical data manipulation and analysis which was initially created and developed for research funded by the Monbukagakusho (Japanese government) scholarship and European Research Council (ERC) advanced grant 339659-MUSICOS.

References

Abbott, T., Abdalla, F., Allam, S., et al. 2018, ApJS, 239, 18

Adelman-McCarthy, J. K., Agüeros, M. A., Allam, S. S., et al. 2006, ApJS, 162, 38

Akhlaghi, M., & Ichikawa, T. 2015, ApJS, 220, 1

Amiaux, J., Scaramella, R., Mellier, Y., et al. 2012, in Space Telescopes and Instrumentation 2012: Optical, Infrared, and Millimeter Wave, Int. Soc. Opt. Photon., 8442, 84420Z

Astropy Collaboration (Robitaille, T. P., et al.) 2013, A&A, 558, A33

Astropy Collaboration (Price-Whelan, A. M., et al.) 2018, AJ, 156, 123

Baillard, A., Bertin, E., De Lapparent, V., et al. 2011, A&A, 532, A74

Beard, S., MacGillivray, H., & Thanisch, P. 1990, MNRAS, 247, 311

Beckwith, S. V., Stiavelli, M., Koekemoer, A. M., et al. 2006, AJ, 132, 1729

Bertin, E. 2006, Automatic Astrometric and Photometric Calibration with SCAMP (San Francisco: Astronomical Society of the Pacific)

Bertin, E., & Arnouts, S. 1996, A&AS, 117, 393

Beucher, S. 1982, ICASSP '82. IEEE International Conference on Acoustics, Speech, and Signal Processing, 7

Borlaff, A., Trujillo, I., Román, J., et al. 2019, A&A, 621, A133

Boucaud, A., Heneka, C., Ishida, E. E., et al. 2020, MNRAS, 491, 2481

Bouwens, R. J., Illingworth, G. D., Franx, M., & Ford, H. 2008, ApJ, 686, 230

Bouwens, R., Illingworth, G., Franx, M., et al. 2009, ApJ, 705, 936

Carlinet, E., & Géraud, T. 2014, IEEE Trans. Image Process., 23, 3885

Fliri, J., & Trujillo, I. 2016, MNRAS, 456, 1359

GNU Astronomy Utilities 2019, NoiseChisel Optimization, `https://www.gnu.org/software/gnuastro/manual/html_node/NoiseChisel-optimization.html`

Goodman, A. A., Rosolowsky, E. W., Borkin, M. A., et al. 2009, Nature, 457, 63

Houlahan, P., & Scalo, J. 1992, ApJ, 393, 172

Huang, S., Leauthaud, A., Murata, R., et al. 2018, PASP, 70, S6

Iodice, E., Capaccioli, M., Grado, A., et al. 2016, ApJ, 820, 42

Iodice, E., Spavone, M., Capaccioli, M., et al. 2019, A&A, 623, A1

Ivezić, Z., Tyson, J., Abel, B., et al. 2019, ApJ, 873, 111

Jones, D. R., Schonlau, M., & Welch, W. J. 1998, J. Global Optim., 13, 455

Levine, M. D., & Nazif, A. 1981, An Experimental Rule-based System for Testing Low Level Segmentation Strategies (McGill University)

Masias, M., Freixenet, J., Lladó, X., & Peracaula, M. 2012, MNRAS, 422, 1674

Melchior, P., Moolekamp, F., Jerdee, M., et al. 2018, Astron. Comput., 24, 129

Moschini, U., Meijster, A., & Wilkinson, M. H. F. 2018, IEEE Trans. Pattern Anal. Mach. Intell., 40, 513

Oesch, P., Bouwens, R. J., Carollo, C. M., et al. 2009, ApJ, 709, L21

Ouzounis, G. K., & Wilkinson, M. H. 2010, IEEE Trans. Pattern Anal. Mach. Intell., 33, 224

Pal, N. R., & Pal, S. K. 1993, Pattern Recognit., 26, 1277

Pratt, N. 1977, Vistas Astron., 21, 1

Prole, D. J., Davies, J. I., Keenan, O. C., & Davies, L. J. 2018, MNRAS, 478, 667

Reiman, D. M., & Göhre, B. E. 2019, MNRAS, 485, 2617

Rix, H.-W., Barden, M., Beckwith, S. V., et al. 2004, ApJS, 152, 163

Robitaille, T., Beaumont, C., McDonald, B., & Rosolowsky, E. 2013, Astrodendro, A Python Package to Compute Dendrograms of Astronomical Data, `http://www.dendrograms.org`

Robotham, A., Davies, L., Driver, S., et al. 2018, MNRAS, 476, 3137

Roerdink, J. B. T. M., & Meijster, A. 2000, Fundam. Inf., 41, 187

Román, J., & Trujillo, I. 2017a, MNRAS, 468, 703

Román, J., & Trujillo, I. 2017b, MNRAS, 468, 4039

Román, J., & Trujillo, I. 2018, Res. Notes Am. Astron. Soc., 2, 144

Román, J., Trujillo, I., & Montes, M. 2020, A&A, 644, A42

Rosolowsky, E., Pineda, J., Kauffmann, J., & Goodman, A. 2008, ApJ, 679, 1338

Salembier, P., & Wilkinson, M. H. F. 2009, IEEE Signal Process. Mag., 26, 136

Salembier, P., Oliveras, A., & Garrido, L. 1998, IEEE Trans. Image Process., 7, 555

Sersic, J. L. 1968, Atlas de Galaxias Australes (Cordoba, Argentina: Observatorio Astronomico)

Simet, M., & Mandelbaum, R. 2015, MNRAS, 449, 1259

Teeninga, P., Moschini, U., Trager, S. C., & Wilkinson, M. H. F. 2013, 11th International Conference "Pattern Recognition and Image Analysis: New Information Technologies" (PRIA-11-2013), IPSI RAS, 746

Teeninga, P., Moschini, U., Trager, S. C., & Wilkinson, M. H. F. 2016, Mathematical Morphology – Theory and Applications, 1, 100

The GPyOpt authors 1968, GPyOpt: A Bayesian Optimization Framework in Python, `http://github.com/SheffieldML/GPyOpt`

Van Dokkum, P. G., Abraham, R., Merritt, A., et al. 2015, ApJ, 798, L45

Venhola, A. 2019, PhD Thesis, University of Groningen

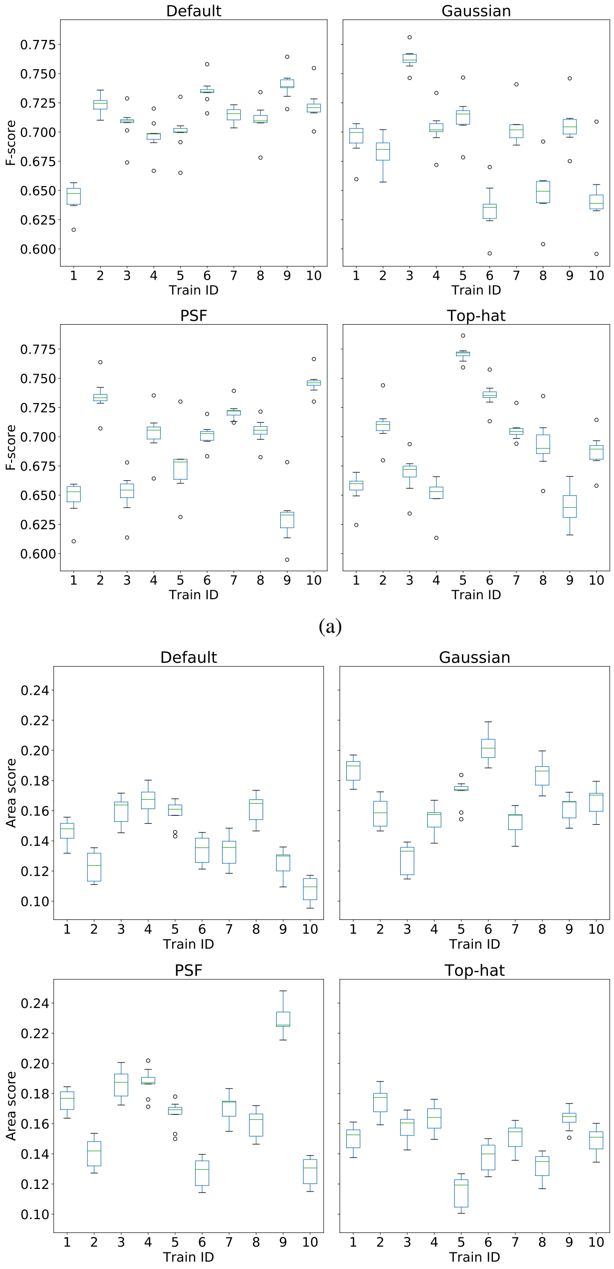
Venhola, A., Peletier, R., Laurikainen, E., et al. 2017, A&A, 608, A142

Venhola, A., Peletier, R., Laurikainen, E., et al. 2018, A&A, 620, A165

Venhola, A., Peletier, R., Laurikainen, E., et al. 2019, A&A, 625, A143

Wilkinson, M. H. F. 1998, in Digital Image Analysis of Microbes, eds. M. H. F. Wilkinson, & F. Schut (Chichester, UK: John Wiley and Sons, Ltd), 135

Appendix A: SExtractor filters



(a)

(b)

Fig. A.1. Optimised test distributions. The parameters for each tool were optimised for Combined A score on each of the ten images, and evaluated on the remaining nine images. Boxes extend from first ($Q1$) to third ($Q3$) quartiles of the results, with median values marked; whiskers extend to the furthest F-score less than $1.5 * (Q3-Q1)$ from each end of the box. (*a*) F-scores grouped by image used to optimise parameters. (*b*) Area scores grouped by image used to optimise parameters.

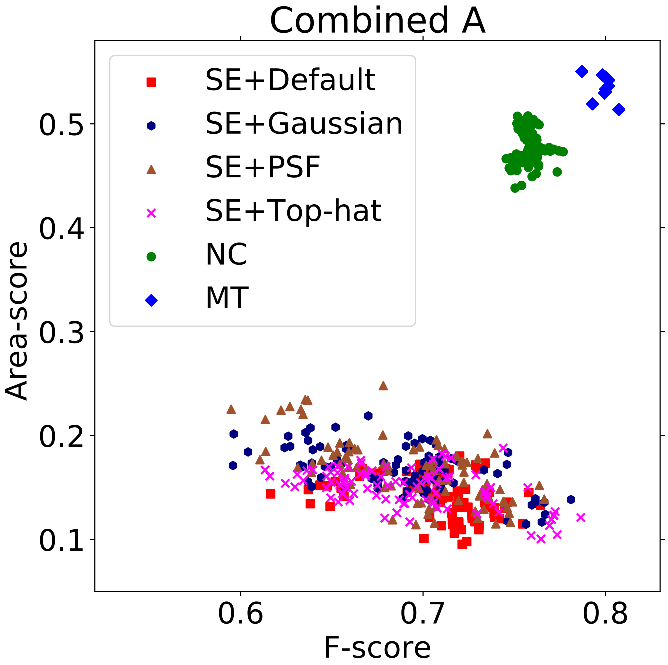


Fig. A.2. F-score vs. Area score. The parameters for each tool were optimised for Combined A score on each of the ten images, and evaluated on the remaining nine images.

SExtractor uses a filter to pre-process the input image. A number of filters are provided with the tool, but custom filters may also be used. The SExtractor manual suggests that the symmetrical PSF of the data is an optimal filter for detecting stars (Bertin 2006), whilst documentation provided with the filters suggests that Gaussian or top-hat filters are effective in detecting extended, low-surface-brightness objects.

As the range of valid filters is infinite, it would not be feasible to optimise the filter in addition to the other parameters. Accordingly, we used the default filter throughout the main experiments of the paper. We subsequently tested a subset of the available filters to determine whether or not they had a significant effect on the tool's performance:

- Default – 3 × 3 pyramidal function (approximating gaussian smoothing).
- Gaussian – 9 × 9 gaussian PSF with a full width at half maximum of 5 pixels.
- PSF – 9 × 9 symmetrical window of the PSF of the simulated images.
- Top-hat – 5 × 5 top-hat PSF.

We optimised SExtractor's parameters for Combined A score as described in Sect. 3.2. Figure A.1 shows the distribution of F-scores and Area scores for each of the four filters.

We find that the different filters have very little effect on F-score, but that there is a slightly higher Area score on average when using the Gaussian filter as compared to the default. Whilst the Gaussian filter could therefore be recommended in this situation, the choice of filter has no effect on the overall conclusions. As shown in Fig. 8, both MTObjects and NoiseChisel achieved substantially higher Area scores of 0.4–0.6 compared to SExtractor's scores of 0.1–0.25. Plotting the four SExtractor filters on the same axes as Fig. 8 shows the relative similarity of the scores, as in Fig. A.2.

Appendix B: Optimised parameter tables

Parameter sets marked in bold and with an asterisk produce the highest median test score for their optimisation metric and tool.

Table B.1. Optimised parameters – Source Extractor.

Mode	Image	BACK SIZE	BACK FILTERSIZE	DEBLEND MINCONT	DEBLEND NTHRESH	DETECT MINAREA	DETECT THRESH
0	1	65	2	1.00E−03	33	6	1.45E+00
0	2	64	2	1.00E−03	34	9	1.35E+00
0	3	65	2	1.00E−03	35	5	1.59E+00
0	4	64	3	1.00E−03	33	5	1.48E+00
0	5	64	2	1.00E−03	31	5	1.51E+00
0	6	37	5	1.00E−03	23	7	1.46E+00
0	7	65	2	1.00E−03	33	5	1.78E+00
0*	**8**	**63**	**3**	**1.00E−03**	**30**	**5**	**1.43E+00**
0	9	63	3	1.00E−03	32	6	1.69E+00
0	10	17	7	1.73E−02	62	8	1.28E+00
1	1	88	5	1.00E−03	21	8	1.00E−01
1	2	36	9	1.00E−03	8	17	1.00E−01
1	3	63	5	9.82E−02	32	4	1.00E−01
1	4	22	7	9.81E−02	3	6	1.00E−01
1*	**5**	**107**	**5**	**1.00E−03**	**41**	**30**	**1.00E−01**
1	6	24	11	1.00E−01	8	49	1.00E−01
1	7	111	6	2.12E−02	6	14	1.05E−01
1	8	108	3	1.00E−03	26	34	1.00E−01
1	9	35	5	4.40E−02	48	28	1.13E−01
1	10	80	4	3.67E−02	23	2	1.06E−01
2	1	84	7	6.96E−02	44	36	6.56E−01
2	2	29	7	1.82E−02	3	49	5.96E−01
2	3	105	6	1.00E−03	63	37	6.03E−01
2	4	110	11	1.00E−03	60	32	7.28E−01
2	5	98	9	1.00E−03	40	28	7.07E−01
2	6	33	8	1.00E−03	47	29	6.19E−01
2	7	124	10	1.00E−03	50	29	1.24E+00
2*	**8**	**98**	**7**	**1.00E−03**	**29**	**46**	**5.70E−01**
2	9	85	4	1.00E−03	44	22	8.94E−01
2	10	118	3	6.87E−03	41	9	1.07E+00
3	1	123	7	1.00E−03	60	43	3.93E−01
3	2	71	5	2.53E−02	23	41	4.91E−01
3	3	119	2	2.33E−02	23	44	4.94E−01
3	4	72	11	1.00E−03	55	47	5.23E−01
3	5	61	7	9.62E−02	53	43	5.35E−01
3	6	101	3	1.00E−03	38	41	5.30E−01
3*	**7**	**119**	**6**	**1.00E−03**	**20**	**49**	**4.37E−01**
3	8	104	7	1.00E−03	49	37	5.11E−01
3	9	25	8	1.07E−02	39	25	6.56E−01
3	10	88	4	1.00E−03	49	16	8.23E−01

Table B.2. Optimised parameters – Noise Chisel (Noise Chisel).

Mode	Image	tilesize	qthresh	snquant	detgrow-quant	dthresh	erode	opening	detgrow-maxholesize	meanmed-qdiff	erode-ngb	opening-ngb	minsky-frac	noerode-quant
0	1	27	4.99E−01	9.99E−01	9.99E−01	0.00E+00	3	4	100	5.00E−03	4	4	8.00E−01	7.00E−01
0	2	90	4.99E−01	9.71E−01	9.99E−01	6.34E−01	4	3	52	2.00E−02	4	4	7.54E−01	1.00E+00
0	3	50	4.00E−01	9.27E−01	9.98E−01	8.26E−01	5	4	19	2.00E−02	4	4	6.62E−01	9.73E−01
0*	**4**	**26**	**4.99E−01**	**9.99E−01**	**9.86E−01**	**4.03E−01**	**3**	**1**	**59**	**2.00E−02**	**4**	**8**	**8.00E−01**	**1.00E+00**
0	5	49	4.99E−01	6.12E−01	9.46E−01	6.87E−01	1	5	77	1.98E−02	8	8	8.00E−01	7.02E−01
0	6	40	4.99E−01	9.99E−01	9.99E−01	3.19E−01	10	1	86	2.00E−02	4	4	8.00E−01	1.00E+00
0	7	87	4.33E−01	9.71E−01	9.99E−01	5.20E−01	6	2	14	6.30E−03	8	8	8.00E−01	9.79E−01
0	8	26	2.00E−01	9.99E−01	9.99E−01	0.00E+00	2	5	48	2.00E−02	4	4	8.00E−01	9.48E−01
0	9	98	4.99E−01	6.00E−01	9.99E−01	5.39E−01	8	4	88	5.00E−03	8	8	4.00E−01	7.00E−01
0	10	87	4.99E−01	9.99E−01	8.28E−01	1.67E−01	5	2	26	2.00E−02	4	8	6.03E−01	7.00E−01
1*	**1**	**75**	**3.69E−01**	**9.99E−01**	**6.00E−01**	**1.00E+00**	**6**	**5**	**79**	**5.00E−03**	**4**	**8**	**4.00E−01**	**7.75E−01**
1	2	40	4.99E−01	9.99E−01	6.00E−01	1.00E+00	1	5	25	2.00E−02	4	8	4.00E−01	1.00E+00
1	3	27	4.99E−01	6.00E−01	6.00E−01	0.00E+00	1	5	85	1.60E−02	8	8	4.56E−01	1.00E+00
1	4	86	4.99E−01	9.99E−01	6.00E−01	1.00E+00	6	3	32	2.00E−02	4	8	8.00E−01	1.00E+00
1	5	34	4.99E−01	6.00E−01	6.00E−01	0.00E+00	5	2	97	8.07E−03	4	4	4.00E−01	1.00E+00
1	6	28	2.00E−01	6.00E−01	6.19E−01	6.86E−01	2	1	94	1.97E−02	4	8	4.00E−01	1.00E+00
1	7	57	2.00E−01	9.99E−01	6.00E−01	0.00E+00	10	5	58	5.00E−03	8	4	4.00E−01	7.00E−01
1	8	33	4.99E−01	6.15E−01	6.15E−01	1.69E-13	5	1	65	2.00E−02	8	4	8.00E−01	1.00E+00
1	9	96	3.64E−01	9.28E−01	6.03E−01	7.15E−01	1	4	63	1.03E−02	8	8	6.87E−01	9.27E−01
1	10	43	3.74E−01	8.28E−01	6.54E−01	8.98E−01	1	5	16	1.14E−02	4	8	4.98E−01	1.00E+00
2*	**1**	**68**	**3.26E−01**	**6.00E−01**	**6.11E−01**	**6.85E−01**	**8**	**5**	**68**	**2.00E−02**	**4**	**8**	**5.16E−01**	**8.05E−01**
2	2	30	4.99E−01	6.00E−01	6.00E−01	6.36E−01	2	2	67	1.06E−02	8	8	8.00E−01	1.00E+00
2	3	27	3.46E−01	9.99E−01	6.09E−01	0.00E+00	9	2	55	1.98E−02	4	4	4.94E−01	8.84E−01
2	4	28	4.99E−01	6.00E−01	6.16E−01	0.00E+00	4	3	80	2.00E−02	8	4	4.00E−01	9.03E−01
2	5	89	2.00E−01	6.58E−01	6.18E−01	7.90E−01	5	2	8	2.00E−02	8	8	8.00E−01	7.09E−01
2	6	29	2.91E−01	9.99E−01	6.10E−01	1.19E−02	3	1	68	2.00E−02	4	4	5.01E−01	1.00E+00
2	7	30	3.50E−01	6.00E−01	6.00E−01	0.00E+00	8	5	28	7.24E−03	8	8	6.97E−01	7.00E−01
2	8	26	2.00E−01	6.00E−01	6.00E−01	2.73E−02	9	5	23	2.00E−02	8	8	5.20E−01	7.00E−01
2	9	45	2.00E−01	8.07E−01	7.21E−01	7.51E−01	7	2	76	8.42E−03	8	8	7.46E−01	7.88E−01
2	10	82	4.42E−01	6.87E−01	6.82E−01	5.27E−01	6	2	38	2.00E−02	4	8	7.44E−01	9.14E−01
3	1	27	2.04E−01	9.08E−01	6.03E−01	0.00E+00	4	4	23	2.00E−02	8	4	7.20E−01	7.68E−01
3	2	64	4.99E−01	6.00E−01	6.00E−01	7.87E−01	4	2	41	7.97E−03	8	8	4.00E−01	7.00E−01
3	3	22	2.00E−01	9.98E−01	6.02E−01	1.57E−03	1	5	48	2.00E−02	8	8	4.02E−01	7.01E−01
3	4	20	2.00E−01	9.99E−01	6.00E−01	0.00E+00	1	5	76	5.00E−03	8	8	4.00E−01	7.00E−01
3	5	41	4.99E−01	9.50E−01	6.00E−01	2.16E−01	8	1	90	5.00E−03	4	8	4.00E−01	1.00E+00
3	6	82	4.67E−01	9.96E−01	6.37E−01	1.07E−02	8	1	38	1.09E−02	8	4	8.00E−01	9.56E−01
3	7	63	4.82E−01	6.93E−01	6.38E−01	5.90E−01	7	2	85	1.89E−02	8	8	4.80E−01	8.70E−01
3	8	31	4.99E−01	7.88E−01	6.29E−01	5.51E−01	9	5	29	2.00E−02	4	4	5.33E−01	9.86E−01
3*	**9**	**71**	**2.00E−01**	**9.99E−01**	**6.00E−01**	**0.00E+00**	**1**	**5**	**11**	**2.00E−02**	**4**	**8**	**4.00E−01**	**7.00E−01**
3	10	100	3.44E−01	9.99E−01	6.00E−01	0.00E+00	9	5	5	2.00E−02	4	4	4.00E−01	7.28E−01

Table B.3. Optimised parameters – Noise Chisel (Segment).

Mode	Image	tilesize	snquant	gthresh	snminarea	minriverlength	objbordersn	minskyfrac
0	1	72	9.99E−01	1.00E+00	25	23	1.28E+01	4.00E−01
0	2	90	9.99E−01	1.00E+00	18	12	1.86E+01	5.75E−01
0	3	93	9.94E−01	8.65E−01	23	20	3.18E+01	6.67E−01
0*	**4**	**60**	**9.99E−01**	**6.69E−01**	**25**	**9**	**2.56E+01**	**7.52E−01**
0	5	20	9.99E−01	2.44E−01	20	40	2.00E+01	8.00E−01
0	6	20	9.99E−01	1.00E+00	19	5	1.12E+01	8.00E−01
0	7	29	9.99E−01	1.00E+00	21	37	1.49E+01	7.28E−01
0	8	72	9.99E−01	1.00E+00	20	40	3.16E+01	8.00E−01
0	9	45	9.99E−01	0.00E+00	20	14	2.26E+01	8.00E−01
0	10	91	9.99E−01	3.77E−01	25	25	1.51E+01	4.00E−01
1*	**1**	**65**	**9.99E−01**	**5.41E−01**	**10**	**34**	**5.00E−01**	**4.00E−01**
1	2	85	9.99E−01	1.00E+00	25	35	1.73E+01	8.00E−01
1	3	35	9.99E−01	0.00E+00	15	21	1.88E+01	4.00E−01
1	4	65	9.99E−01	1.00E+00	23	8	1.80E+01	7.43E−01
1	5	32	9.99E−01	6.34E−01	15	14	1.34E+00	8.00E−01
1	6	41	9.99E−01	4.41E−01	21	36	1.15E+01	5.99E−01
1	7	52	9.99E−01	0.00E+00	25	40	1.81E+01	4.00E−01
1	8	46	9.99E−01	2.85E−02	25	40	2.12E+01	8.00E−01
1	9	82	9.97E−01	5.72E−01	20	23	3.45E+01	7.81E−01
1	10	61	9.99E−01	2.14E−01	17	31	8.88E+00	5.64E−01
2*	**1**	**25**	**9.99E−01**	**3.28E−01**	**25**	**24**	**3.11E+01**	**4.90E−01**
2	2	45	9.99E−01	4.89E−01	24	21	1.77E+01	4.00E−01
2	3	80	9.99E−01	6.60E−03	22	30	5.88E+00	4.09E−01
2	4	81	9.99E−01	0.00E+00	22	8	1.03E+01	8.00E−01
2	5	31	9.99E−01	2.48E−01	22	16	7.32E+00	4.00E−01
2	6	74	9.99E−01	9.37E−03	25	22	6.71E+00	4.10E−01
2	7	38	9.99E−01	2.11E−01	20	31	1.29E+01	4.00E−01
2	8	20	9.99E−01	0.00E+00	25	33	7.35E+00	4.00E−01
2	9	90	9.99E−01	8.11E−01	14	27	3.96E+01	4.00E−01
2	10	48	9.99E−01	6.99E−01	18	15	2.60E+01	6.55E−01
3	1	72	9.99E−01	5.36E-16	24	13	1.29E+01	4.03E−01
3	2	94	9.99E−01	8.28E−01	25	30	2.42E+01	4.00E−01
3	3	31	9.97E−01	3.17E−02	25	5	2.21E+01	4.02E−01
3	4	20	9.99E−01	1.00E+00	25	11	1.63E+01	4.00E−01
3	5	32	9.99E−01	0.00E+00	25	25	1.81E+01	8.00E−01
3	6	20	9.99E−01	3.93E−01	22	26	9.46E+00	6.79E−01
3	7	78	9.99E−01	5.50E−02	24	21	2.89E+01	4.52E−01
3	8	47	9.99E−01	6.88E−01	16	16	3.86E+01	8.00E−01
3*	**9**	**74**	**9.99E−01**	**0.00E+00**	**25**	**11**	**2.38E+01**	**8.00E−01**
3	10	32	9.99E−01	2.18E−01	23	18	1.11E+01	5.88E−01

Table B.4. Optimised parameters – MTObjects.

Mode	Image	Move_factor	Min_dist
0	1	4.96E−02	1.49E−01
0	2	6.14E−02	1.15E−01
0	3	5.73E−02	1.57E−01
0	4	8.24E−03	1.32E−01
0	5	1.21E−01	1.45E−01
0	6	0	1.50E−01
0*	**7**	**0**	**1.13E−01**
0	8	9.34E−02	1.55E−01
0	9	1.13E−01	1.33E−01
0	10	3.07E−02	1.17E−01
1*	**1**	**0**	**0**
1	2	0	0
1	3	0	0
1	4	0	0
1	5	0	0
1	6	0	0
1	7	0	0
1	8	0	0
1	9	0	0
1	10	0	0
2*	**1**	**0**	**0**
2	2	0	0
2	3	0	0
2	4	0	0
2	5	0	0
2	6	0	0
2	7	0	0
2	8	0	0
2	9	0	0
2	10	0	0
3*	**1**	**0**	**0**
3	2	0	0
3	3	0	0
3	4	0	0
3	5	0	0
3	6	0	0
3	7	0	0
3	8	9.36E−03	0
3	9	0	0
3	10	0	0

Table B.5. Optimised parameters – ProFound.

Mode	Image	Skycut	Tolerance	Ext	Sigma	Pixcut	Size	Iters	Threshold
0	1	7.22E−01	3.71E+00	1.92E+00	2.22E+00	4	5	7	7.50E−01
0	2	9.34E−01	2.06E+00	2.29E+00	1.35E+00	10	7	2	8.26E−01
0	3	1.00E+00	1.00E+00	7.03E+00	1.19E+00	7	7	4	7.50E−01
0	4	8.23E−01	1.58E+00	8.03E+00	2.23E+00	6	7	3	1.20E+00
0	5	1.22E+00	3.52E+00	3.89E+00	9.17E−01	6	5	6	1.78E+00
0	6	6.54E−01	1.00E+00	3.78E+00	3.00E+00	15	7	9	7.50E−01
0	7	9.34E−01	1.00E+00	4.55E+00	1.43E+00	16	5	0	1.96E+00
0*	**8**	**5.56E−01**	**2.03E+00**	**3.28E+00**	**2.11E+00**	**15**	**7**	**6**	**9.49E−01**
0	9	1.79E+00	3.90E+00	3.29E+00	8.76E−01	4	5	7	1.35E+00
0	10	6.25E−01	2.93E+00	4.33E+00	1.36E+00	14	9	8	2.00E+00
1	1	5.26E−01	5.29E+00	2.22E+00	1.98E+00	7	9	7	8.68E−01
1	2	2.38E−01	2.62E+00	9.01E+00	2.21E+00	11	7	6	1.12E+00
1	3	7.65E−01	5.71E+00	2.70E+00	1.45E+00	5	5	8	8.42E−01
1	4	6.13E−01	4.44E+00	4.77E+00	2.83E+00	5	7	7	8.00E−01
1	5	1.57E−01	3.72E+00	4.23E+00	2.74E+00	10	9	6	1.24E+00
1	6	2.77E−01	2.49E+00	8.57E+00	2.26E+00	15	7	9	8.31E−01
1	7	6.23E−01	3.68E+00	5.18E+00	1.01E+00	1	5	4	1.04E+00
1*	**8**	**6.45E−01**	**6.00E+00**	**8.97E+00**	**1.13E+00**	**14**	**9**	**9**	**9.10E−01**
1	9	2.82E−01	4.48E+00	6.28E+00	2.07E+00	3	7	4	7.50E−01
1	10	1.33E+00	3.23E+00	3.63E+00	1.82E+00	9	7	9	8.23E−01