

User's Guide

Extracting Sources and Filaments with *getsf*

Alexander Men'shchikov

AIM, CEA, CNRS, Université Paris-Saclay, Université Paris Diderot, Sorbonne Paris Cité
F-91191 Gif-sur-Yvette, France

24th March 2023

The *getsf* method was published by [Men'shchikov \(2021b\)](#)

<http://irfu.cea.fr/Pisp/alexander.menshchikov/>
<https://ascl.net/2012.001>

Contents

1	Introduction	3
2	The <i>getsf</i> software suite	3
2.1	Prerequisites	3
2.2	Installation of <i>getsf</i>	4
2.3	The utilities and scripts	5
2.4	Notable improvements or new features	5
3	Preparing images for extraction	6
3.1	Handling multiple extensions and data cubes	6
3.2	Modifying FITS headers	6
3.3	Reprojecting images using a reference image	7
3.4	Reprojecting images with <i>prepobs</i>	8
3.5	Verifying alignment of all images	9
3.6	Creating observational masks	9
4	Deriving high-resolution surface densities	10
4.1	The <i>hires</i> configuration file	10
4.2	The configuration parameters	11
4.3	Convolving images to lower resolutions	11
4.4	Fitting spectral shapes of pixels with <i>fitfluxes</i>	11
4.5	Computing high-resolution surface densities	12
4.6	Observational mask for surface densities	13
5	Configuring source and filament extraction	13
5.1	The <i>getsf</i> configuration file	13
5.2	The configuration parameters	14
6	Extracting sources and filaments	17
6.1	Running <i>getsf</i> sequentially for all images	18
6.2	Running <i>getsf</i> in parallel for each image	19
7	Understanding extraction output files	20
7.1	Directory structure created by <i>getsf</i> runs	20
7.2	Directory for <i>getsf</i> extraction runs	21
7.3	Directory for monochromatic processing	22
7.3.1	Directory for separation of sources or filaments	22
7.3.2	Directory for flattening of sources or filaments	23
7.3.3	Directory for completion of sources or filaments	23
7.4	Directory for separated structural components	23
7.4.1	Directory for partially reconstructed images	23
7.5	Directory for output images and catalogs	23
7.5.1	Directory for improved structural components	23
7.5.2	Directory for catalogs of extracted sources	24
7.5.3	Directory for measurement iterations of sources	24
7.5.4	Directory for visualization of extracted sources	24
7.5.5	Directory for derived skeletons of filaments	25
7.5.6	Directory for skeletons, normals, and footprints	25
7.5.7	Directory for catalogs and profiles of filaments	25
7.5.8	Directory for visualization of extracted filaments	26
8	User agreement	26
9	Citations	26

1 Introduction

The term *source* denotes almost round intensity emission peaks in the observed images, significantly stronger than the local surrounding fluctuations, indicating the presence of one or several physical *objects* in space that produced the emission. An implicit assumption that an unresolved far-infrared source on a complex fluctuating background contains emission of just one single object is invalid in general: the unresolved peak of emission may well be a blend of several unrelated components, produced by different physical entities. The term *filament* refers to significantly elongated structures of various shapes. The term *background* refers to the approximately isotropic astrophysical backgrounds, such as molecular clouds, emission nebulae, or HII regions. The term *noise* refers to the instrumental noise with possible contributions from any other signals that are not astrophysical in nature, i.e., unrelated to the emission of the areas in space being observed.

Like other extraction algorithms, *getsf* does not know anything about the physical nature of the objects that produce the detected sources of emission. It extracts *sources*, separating them from background, filaments, and noise. It is a subject of the post-extraction investigation to identify and sub-select from the catalogs those sources of infrared emission that are produced by the physical objects of a specific nature.

Note. The *getsf* method has no free parameters to vary. It has been carefully tested and fine-tuned to produce the best results with the default values of the configuration parameters and the common user should not change them. There is only one parameter (per image) that a user must specify before any extraction: the maximum size (FWHM) of the sources or filaments of interest (see Sect. 3.1.3 in [Men'shchikov 2021b](#)). This can easily be done by opening an image in *ds9* (or another image viewer) and placing a region fully encircling the *entire* emission peak of the largest structure; radius of the region (in arcsec) would be the maximum size.

2 The *getsf* software suite

The tool has been developed as a *bash* script *getsf* that employs several utilities (in FORTRAN) to perform all numerical work. The Linux or macOS operating systems have been used to install and run the code.

2.1 Prerequisites

The *getsf* software can be installed on any modern laptop, desktop, or server. For large images, it requires many gigabytes of memory and space, hence the more the better. It is necessary to verify that a FORTRAN compiler, either *ifort*¹ or *gfortran*², is available in the operating system:

```
> ifort --version
> gfortran --version
```

If neither is found, one of them must be installed. For reading and writing FITS images, *getsf* uses the *cfitsio* library ([Pence 1999](#)), hence *libcfitsio.a* must exist in your operating system. For resampling and reprojecting images, *getsf* calls *swarp* ([Bertin et al. 2002](#)), hence it is necessary to verify its availability:

```
> swarp --version
```

If *swarp* is not found, it needs to be installed (<http://www.astromatic.net/software/swarp/>). For determining the world coordinates of sources, *getsf* employs *xy2sky* from *wcstools* ([Mink 2002](#)):

```
> xy2sky -version
```

If *xy2sky* is not found, it needs to be installed (<http://tdc-www.harvard.edu/wcstools/>) to ensure that the source catalogs have world coordinates. Without this (optional) utility, the source catalogs would have coordinates in pixels only. Finally, to make the *getsf* screen output colored, it is recommended to have the *highlight* utility:

```
> highlight --version
```

This utility is optional (<http://www.andre-simon.de/>), it just improves visibility of the screen output.

¹<https://software.intel.com/content/www/us/en/develop/tools/oneapi/components/fortran-compiler.html>

²<https://github.com/fxcoudert/gfortran-for-macos/releases>

2.2 Installation of *getsf*

The software is compressed in a *zip* archive `getsf.vYYMMDD.zip`, whose name contains the latest *getsf* version (e.g., `getsf.v230320.zip`). It is recommended that the downloaded archive is moved to a dedicated directory (for example, `/Users/yourname/+GETSF`), where it must be unpacked:

```
> unzip getsf.v230320.zip
```

The unpacked code is created in `/Users/yourname/+GETSF/v230320`. It is necessary to add the environment variable `GETSF_HOME` to the `~/ .bashrc` file:

```
export GETSF_HOME=/Users/yourname/+GETSF
```

It is also necessary to create the binary directory, where the executable code will be found, and add the environment variable `GETSF_BIN` to the `~/ .bashrc` file:

```
> mkdir -p /Users/yourname/bin/+GETSF
export GETSF_BIN=/Users/yourname/bin/+GETSF
```

The installation script *installg* will put the compiled binaries and scripts in `GETSF_BIN`. Additionally, this current version will also be installed in a dedicated directory `/Users/yourname/bin/+GETSF/v230320`, which allows running both the default and specific versions of the code, when necessary. The `PATH` environment variable in the `~/ .bashrc` file must be modified to include the binary directory, for example:

```
export PATH=$GETSF_BIN:$PATH
```

The *cfitsio* library `libcfitsio.a` must be copied from its default location in your operating system to the *getsf* home directory, for example:

```
> cp /path/to/cfitsio/library/in/your/system/libcfitsio.a "$GETSF_HOME"
```

The *getsf* installation must be started from its directory `$GETSF_HOME/v230320`:

```
> ./installg
```

The installation script compiles all utilities and copies their binaries and other scripts in the two directories:

```
> $GETSF_BIN ~> binary directory of the default version
> $GETSF_BIN/v230320 ~> binary directory of the specific version
```

To make sure that the installed *getsf* is indeed available, its utilities may be run, for example:

```
> modfits
> operate
```

The default version of *getsf* can be started by typing its name:

```
> getsf
```

The specific version of *getsf* can be started by typing its name and the version number:

```
> getsf 230320
```

Upon each *getsf* run, several log files are created and/or appended:

```
> +log.AGREE ~> user agreement and suggested citations
> +log.HISTORY ~> history of recent changes in getsf and its utilities
> +log.VERSION ~> brief information on the version of the code executed
> +log.USAGE ~> brief getsf usage information
> +log.GETSF ~> main log file with information on the extraction process
> +log.GETSF.iter ~> iterations log for the single-scale image processing
> +log.FOOTS.iter ~> iterations log for the source footprints in measurements
```

Note. The software may become corrected and improved in the future and only the most recent version of *getsf* is supported. If apparently abnormal results are found or the code aborts for any reason, the user is asked to install the latest version of the code from its website and restart the failed action to see if the problem persists. If it does, please refer to <http://irfu.cea.fr/Pisp/alexander.menshchikov/#support> and request assistance from the author.

2.3 The utilities and scripts

The following list of the *getsf* utilities and scripts explains their purpose and functions. They are quite useful for command-line image manipulations, even if there is no need in the source or filament extraction. The utilities are sorted in a decreasing sequence of their general usability outside *getsf*.

<i>modfits</i>	↪ modify an image or its header in various ways: math transformations; profiling an image along a line; image segmentation; filament skeletonization; removal of connected clusters of pixels; addition or removal of border areas; correction of saturated or bad pixel areas; conversion of intensity units; changes of header keywords; <i>etc.</i>
<i>operate</i>	↪ operate on two input images: addition, subtraction, multiplication, division; relative differencing; minimization or maximization; extension or expansion of masks; copying of an image header; computation of surface densities, temperatures, or intensities; <i>etc.</i>
<i>imgstat</i>	↪ display and/or save image statistical quantities; produce mode-, mean-, or median-filtered images; compute images of standard deviations, skewness, kurtosis; <i>etc.</i>
<i>fftconv</i>	↪ fast Fourier transform or convolve image with few predefined kernels or an external kernel image
<i>fitfluxes</i>	↪ fit spectral shapes of source fluxes or image pixel intensities to derive masses or surface densities
<i>convolve</i>	↪ convolve an image to a desired lower resolution
<i>resample</i>	↪ resample and re-project an image with optional rotation
<i>hires</i>	↪ derive high-resolution densities and temperatures
<i>prepobs</i>	↪ convert observed images to the same pixel grid
<i>installg</i>	↪ install <i>getsf</i> in an operating system (macOS, Linux)
<i>iospeed</i>	↪ test I/O speed of a hard drive for a specific image
<i>readhead</i>	↪ display an image header or save selected keywords
<i>cleanbg</i>	↪ interpolate background under footprints of sources
<i>ellipses</i>	↪ overlay an image with ellipses of extracted sources
<i>sfinder</i>	↪ detect sources in combined single-scale images
<i>smeasure</i>	↪ measure and catalog properties of detected sources
<i>fmeasure</i>	↪ measure and catalog properties of detected filaments
<i>finalcat</i>	↪ produce final catalogs of detected sources
<i>expanda</i>	↪ expand masked areas of an image to its edges
<i>extractx</i>	↪ extract all FITS extensions in separate images
<i>splitcube</i>	↪ split a FITS data cube in separate images
<i>makecube</i>	↪ make a FITS data cube from separate images

Note. With *getsf* utilities, it is not necessary to supply the `.fits` extension of the image name on the command line. When a utility is run without any parameter, its usage information is displayed.

2.4 Notable improvements or new features

- Revision v220610 (new feature). Starting with this version, *getsf* creates *ds9* region files for both sources (half-maximum and footprint ellipses) and filaments (global and scale-dependent skeletons) to facilitate their visualization and analyses. The region files `*.reg` are found in relevant subdirectories (Sects. 7.5.2, 7.5.4, 7.5.7, and 7.5.8).
- Revision v210915 (new feature). Starting with this version, *getsf* provides also the measurements of filaments for the global skeletons the same way the measurements are done for the set of scale-dependent skeletons. The images and catalogs `*.fil.ske.global.*` are placed in the usual subdirectories (Sect. 7.5.7).
- Revision v210907 (new feature). Starting with this version, *getsf* adds the *global*, scale-independent skeletons to its output. Several users experienced difficulties analyzing the complex nature of observed filaments with the default set of scale-dependent skeletons produced by *getsf*. Responding to their needs, the updated *getsf* produces also images of the global skeletons tracing the filamentary structures on all scales, independently of the filament width. The additional images of the global skeletons `*ske.global.fits`, are located in the subdirectories `+skeletons` and `+filaments/+visuals` (Sects. 7.5.5 and 7.5.8). They are created from the scale-dependent skeletons within the range of scales of interest for the user, that is up to the maximum width of the filaments specified in the *getsf* configuration file (Sect. 5.2).

3 Preparing images for extraction

The first step is to create the extraction directories (anywhere in the file system, may be named freely). In practice, it makes sense to use current date in the extraction directory name (in the YYYYMMDD format) and the name of the observed region, for example:

```
mkdir -p images ~> preparation directory (name may be arbitrary)
mkdir -p 230322/runs ~> extraction directory (names may be arbitrary)
```

This document uses the names from the above example. It is assumed that the observed data set is found in the directory `images`, which is made the current directory in a terminal window. If the data set consists of more than a single image, the images usually have different pixel sizes and coverage areas; for use with `getsf`, all images need to be reprojected to the same grid of pixels.

3.1 Handling multiple extensions and data cubes

Observed images may come in the FITS files with multiple extensions (e.g., the *Herschel* HIPE output) or in the 3D data cubes. To use the images with `getsf`, it is necessary to extract them from the files as plain 2D FITS images. This can easily be done with the `getsf` utilities `extractx` and `splitcube`:

```
> extractx hipec.file.fits
> splitcube data.cube.fits 3
```

An optional parameter `1|2|3` of `splitcube` defines the data axis, along which the cube must be split into the normal 2D images, with the default value of 3. The resulting output images are named as

```
hipec.file.01.fits, hipec.file.02.fits, hipec.file.03.fits, ...
data.cube.0001.fits, data.cube.0002.fits, data.cube.0003.fits, ...
```

In the HIPE files, observed images are usually found in the second FITS extension. The observed position-velocity data cubes usually contain too many slices, from the practical point of view. To use them with `getsf` (that handles up to 99 images), the slices can be averaged over carefully chosen intervals of velocities to reduce the number of images well below 100. For their first test extractions, the `getsf` users are advised to have two or three images.

3.2 Modifying FITS headers

It is not unusual that the headers of FITS images are either incorrect or incomplete. The headers can be displayed and modified using the `readhead` and `modfits` utilities, respectively. The standard minimal subset of the FITS header keywords, supported by `getsf` and its utilities, is shown by the following example:

```
> readhead universe_250

READHEAD • Display FITS Headers • Nov 05 2020 17:25
Alexander Men'shchikov, DAp IRFU CEA Saclay, France.
Using CFITSIO library version 3.260 by William D Pence.

  Reading header of 'universe_250.fits'...

SIMPLE =                               T / file does conform to FITS standard
BITPIX =                               -32 / number of bits per data pixel
NAXIS  =                               2 / number of data axes
NAXIS1 =                               2600 / length of data axis 1
NAXIS2 =                               3500 / length of data axis 2
COMMENT FITS (Flexible Image Transport System) format is defined in 'Astronomy
COMMENT and Astrophysics', volume 376, page 359; bibcode: 2001A&A...376..359H
CREATOR = 'MODFITS ' / Alexander Menshchikov, DAp IRFU CEA Saclay
DATE    = '2020-08-05T17:30:38' / creation date and time
BZERO   = 0.0000000000000000E+00 / zero point in scaling equation
BSCALE  = 1.0000000000000000E+00 / linear factor in scaling equation
DATAMAX = 8.4043195312500E+04 / maximum data value
DATAMIN = -8.9359838867188E+02 / minimum data value
```

```

BUNIT = 'MJy/sr ' / physical units of the array values
CTYPE1 = 'RA---TAN' / name of the coordinate axis
CTYPE2 = 'DEC--TAN' / name of the coordinate axis
CRPIX1 = 1.30050000000000E+03 / coordinate system reference pixel
CRPIX2 = 1.75050000000000E+03 / coordinate system reference pixel
CROTA1 = 5.1246834126890E-04 / coordinate system rotation angle
CROTA2 = 5.1246834126890E-04 / coordinate system rotation angle
CD1_1 = -8.3333333330000E-04 / linear projection matrix
CD1_2 = 0.0000000000000E+00 / linear projection matrix
CD2_1 = 0.0000000000000E+00 / linear projection matrix
CD2_2 = 8.3333333330000E-04 / linear projection matrix
CRVAL1 = 3.5999968770000E+02 / coordinate value at reference pixel
CRVAL2 = 3.6225910130000E-04 / coordinate value at reference pixel
CDELTA1 = -8.3333333333333E-04 / coordinate increment along axis
CDELTA2 = 8.3333333333333E-04 / coordinate increment along axis
OBJECT = 'Observed region ID' / object identifier
RA = 3.5999968770000E+02 / right ascension
DEC = 3.6225910130000E-04 / declination
EQUINOX = 2.0000000000000E+03 / equinox
WAVE = 2.5000000000000E+02 / wavelength (microns)
END
No more extensions.

```

Pixel sizes: -3.000E+00 3.000E+00 arcsec

Done.

The easiest way of adapting the FITS headers of the new images to the *getsf* headers is to multiply them by 1:

```
> modfits multiply 1 universe_250 -o universe_250.m
```

A different name (`universe_250.m`) was specified for the output file, in order to keep the original image and header unaltered. The keywords `CDELTA1`, `CDELTA2`, and `BUNIT` are among those that need to be changed most likely. The supported keywords can be added or modified, one at a time, for example:

```
> modfits key CDELTA1 -2 universe_250.m -o universe_250.m
> modfits key CDELTA2 2 universe_250.m -o universe_250.m
```

Attention must be paid to the presence of `CDELTA1`, `CDELTA2`, `CD1_1`, `CD1_2`, `CD2_1`, `CD2_2`, and `BUNIT`. If the linear projection matrix is absent, then it can be added using *modfits*, assuming a zero rotation angle: `CD1_1=CDELTA1`, `CD1_2=0`, `CD2_1=0`, `CD2_2=CDELTA2`. When the rotation matrix is present, but the `CDELTA1` and `CDELTA2` keywords are not found or their values are wrong, they must be specified using *modfits*. Only two physical units are supported by *getsf*: `MJy/sr` and `H2/cm2`. If the image units are neither of the two, *modfits* can be used for their conversion or to add the intensity or surface density units, for example:

```
> modfits Jy/pixel~MJy/sr universe_250.m -o universe_250.m
> modfits Jy/beam~MJy/sr 18.2 universe_250.m -o universe_250.m
> modfits mJy/beam~MJy/sr 18.2 universe_250.m -o universe_250.m
> modfits key BUNIT MJy/sr universe_250.m -o universe_250.m
> modfits key BUNIT H2/cm^2 universe_250.m -o universe_250.m
```

Note. When the observed data set contains just a single image, the reader can skip the next three sections and fast forward to Sect. 3.6.

3.3 Reprojecting images using a reference image

This is the simplest approach to the preparation of the observed images for *getsf*. All images can be resampled (reprojected) by *resample* to the same grid of pixels, using the header information from one of the images, considered as a reference image. This method can be applied, if the parameters of one of the images (pixel size, numbers of pixels, coordinates, etc.) are deemed suitable also for all other images. Assuming that the data set contains three images, `universe_250`, `universe_350`, and `universe_500`, and that one of them (`universe_250`) can be used as a reference image, the commands look like this:

```
> resample universe_350 universe_250
> resample universe_500 universe_250
```

The resulting images, named `universe_350.resamp` and `universe_500.resamp`, are resampled to the pixel grid of `universe_250`, with consistent FITS headers. Before using `universe_250` as the reference image, it may be necessary to optimize (reduce) its size by cutting off image borders that do not contain useful information, for example:

```
> modfits border -100 universe_250 -o universe_250.m
> modfits border -100,-200,-50,-20 universe_250 -o universe_250.m
```

In the first of the two examples, the widths of the borders to cut (in pixels) are the same (100) on all sides, whereas the second example shows that they may be different on each side (left: 100, right: 200, bottom: 50, top: 20). For very large images, the `getsf` computation time (also the storage space) may be an issue, hence it is very important to minimize the image size, in order to reduce its processing time as much as possible.

3.4 Reprojecting images with *prepobs*

This approach to the preparation of the observed images for `getsf` is incorporated in *prepobs* that internally uses *resample* for the actual reprojection of the images. This method is more versatile, with a much larger number of optional parameters, than when using the reference image (Sect. 3.3). It may be recommended to prepare images in their sequence from the short to long wavelengths. It may be necessary to rename the input files before running *prepobs*, inserting the wavelength in the file name (separated by '_'), to make sure that the wavelength is understood by the script.

Both plain FITS images and HIPE files with several extensions may be given on input to *prepobs* (cf. Sect. 3.1). The common usage is illustrated below, assuming that the data set consists of six *Herschel* images:

```
> prepobs pacs 20p -scanamorphos Universe universe_070 2 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs pacs 20 -scanamorphos Universe universe_100 2 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs pacs 20p -scanamorphos Universe universe_160 2 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs spire 60p -naive Universe universe_250 2 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs spire 60p -naive Universe universe_350 2 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs spire 60p -naive Universe universe_500 2 1000 1000 12:35:58.0 +27:57:38.8 0
```

where `pacs` and `spire` refer to the *Herschel* instruments, `20`, `20p` and `60p` are the scanning speeds of the imaging (`20` and `60''s-1`, `p` indicates the parallel mode), `scanamorphos` and `naive` are the map-making methods, used to produce the images ('-' indicates the plain FITS image), `Universe` is the name of the observed region, `universe_xxx` are the names of the image files (with the wavelength `xxx` attached), `2` is the desired pixel size (arcsec), `1000 1000` are the desired image sizes (pixels), `12:35:58.0 +27:57:38.8` are the world coordinates of the field center, and the last `0` is the rotation angle (east of north) of the final prepared images.

The following commands prepare the *Spitzer* images:

```
> prepobs irac 1 -other Universe universe_irac1 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs irac 2 -other Universe universe_irac2 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs irac 3 -other Universe universe_irac3 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs irac 4 -other Universe universe_irac4 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs mips 24 -other Universe universe_mips24 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs mips 70 -other Universe universe_mips70 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs mips 160 -other Universe universe_mips160 0.5 1000 1000 12:35:58.0 +27:57:38.8 0
```

where `irac` and `mips` refer to the *Spitzer* instruments, `1`, `2`, `3`, `4`, `24`, `70`, and `160` identifying the bands, `-other` indicates that the map-making method is unknown or unimportant, and `0.5` is the desired pixel size (arcsec).

The following command prepares the ALMA images in the 1 and 3 mm bands:

```
> prepobs other 0.5 -other Universe universe_1mm 0.1 1000 1000 12:35:58.0 +27:57:38.8 0
> prepobs other 0.5 -other Universe universe_3mm 0.1 1000 1000 12:35:58.0 +27:57:38.8 0
```

where `other` refers to an instrument (unknown to `getsf`), `0.5` is the angular resolution (arcsec), `-other` refers to an map-making method (unknown to `getsf`), and `0.1` is the desired pixel size (arcsec).

Several runs of *prepareobs* with different center coordinates, dimensions, and rotation angles are sufficient to optimize the layout of the observed coverages and to reduce the final size of the prepared images; it is also possible

to create a smaller sub-field from the original large image and to cut useless borders (an alternative to the *modfits* technique, Sect. 3.3). For very large images, the *getsf* computation time (also the storage space) may be an issue, hence it is very important to minimize the image size, in order to reduce its processing time as much as possible.

3.5 Verifying alignment of all images

In the multi-wavelength extractions with *getsf*, it is very important to verify alignment of all images. In general, the prepared data set contains images, observed with different telescopes and/or instruments and created with different map-making methods. Such differences may easily lead to misalignments of some images. Any substantial shift of the intensity peaks in some images with respect to the other could lead to the appearance of many spurious sources. Therefore, all prepared images must be carefully compared with each other, to make sure that relatively bright unresolved peaks in various places across the images stay at the same pixels. To reveal possible misalignments, it is sufficient to open each pair of the prepared images in *ds9*, one over the other, and quickly alternate between the two frames, going from the highest-resolution to the lowest-resolution pairs. If some images are misaligned, such blinking would reveal obvious jumps between the two frames. Misalignments can be corrected by shifting the problematic images, for example:

```
> modfits border -1,1,-1,1 universe_350 -o universe_350.shift
```

In the above example, the image is shifted by 1 pixel in the diagonal direction to the lower-right. After the attempt to shift one image, the visual inspection must be repeated, until all images become well aligned.

3.6 Creating observational masks

Most astronomical observations have irregularly shaped coverages and limited usable areas that are smaller than the image. To include in the image processing only the usable areas of good quality, it is necessary to create the observational masks – special images with the pixel values of only 1 and 0. Using the masks, *getsf* processes only the good areas of the original images, with the mask value of 1. To facilitate the image preparation with *prepobs* (Sect. 3.4), the script always creates the simplest default masks (with all pixels of 1). The masks can also be made by running *modfits*, for example:

```
> modfits multiply 0 universe_250 -o universe_250.omask
> modfits add 1 universe_250.omask -o universe_250.omask
```

Such simple masks may only be used, if *all pixels* of the images are good (usable). However, for most observations the masks must be prepared manually, for each image independently.

To create the masks manually, it is necessary to use an image manipulation program, such as *imagej* (Abràmoff et al. 2004) or *gimp* (<http://www.gimp.org/>), that allows users to create a polygon over an image, convert the polygon into a mask, and save it in the FITS format. For example, assuming that *imagej* is used, here are the steps: (1) Open one of the prepared images: Menu ▷ File ▷ Open ▷ *universe_250.fits*. (2) Mirror the image vertically (*imagej* opens it upside down): Menu ▷ Image ▷ Transform ▷ Flip vertically. (3) Adjust brightness and contrast: Menu ▷ Image ▷ Adjust ▷ Brightness/contrast. (4) Press the button 'Polygon selections'. (5) Select a polygon, surrounding the usable area of the image. (6) Convert the polygon in a mask: Menu ▷ Edit ▷ Selection ▷ Create mask. (7) Save the mask: Menu ▷ File ▷ Save as ▷ FITS ▷ *mask_250.fits*.

The resulting mask image, created by *imagej* has a poor FITS header and inappropriate values (255, instead of 1), therefore it must be corrected for use with *getsf*. Assuming that *mask.fits* was saved in the *images* directory with the prepared *universe_250.fits*, the following two commands correct the mask:

```
> operate universe_250 h mask_250 -o universe_250.omask
> modfits divide 255 universe_250.omask -o universe_250.omask
```

As the result, the following pair of files (with identical headers) must exist in the *images* directory:

```
universe_250.fits
universe_250.omask.fits
```

The same steps must be repeated for all prepared images. Consistency of each pair must be verified by opening the prepared image in together with its observational mask in *ds9* and blinking them to make sure that the mask indeed covers only the good area of the images, for which the user plans to run a *getsf* extraction. It is also necessary

to check that the images do not have wide empty areas around the border, with zero values in the observational masks. If such empty borders exist, it is strongly recommended to reduce the image size by cutting the borders, for example:

```
> modfits border -50,-30,-40,-60 universe_250 -o universe_250.m
> modfits border -50,-30,-40,-60 universe_250.omask -o universe_250.omask.m
```

It is important to emphasize that the observational masks must have topologically simple shapes, without any isolated 'holes' (zero areas) inside the masked field. All zero pixels of the mask image must be connected to the image edges.

4 Deriving high-resolution surface densities

Note. If the prepared data set contains only one or two images or it does not allow derivation of surface densities (by pixel fitting) for other reasons, the reader can skip this section and fast forward to Sect. 5.

A high-resolution surface density image can be derived, using *hires*, from the multi-wavelength far-infrared and submillimeter continuum images, such as those obtained in the *Herschel* observations. The density image may be as sharp as the shortest-wavelength image of a sufficient quality, with the angular resolution reaching $5.6''$, when using the *Herschel* $70\ \mu\text{m}$ image. If the image is too noisy or unusable for other reasons, the highest resolution would be defined by the 100 or $160\ \mu\text{m}$ image (6.8 – $11.3''$). In the multi-wavelength extractions with *getsf*, it may be recommended to use the (optional) high-resolution surface density for better detections and deblending of dense structures. It would be advantageous to have it complement the original data set, as an additional image, “observed” in a fictitious waveband. For more details of the algorithm, refer to [Men’shchikov \(2021b\)](#).

4.1 The *hires* configuration file

To prepare for the derivation of surface density images, it is necessary to create the default configuration file by executing *hires* in a dedicated subdirectory, preferably within the *images* directory:

```
> mkdir -p hires
> cd hires
> hires +hires.cfg
```

This command creates the configuration file named *+hires.cfg*. When opening files with '+' in their names in the text editors interpreting the '+' character as a special symbol, it is necessary to append the './' string to the file name, to tell the editor that the character is a part of the file name, for example:

```
> nedit ./+hires.cfg
```

The contents of the default configuration file:

```
#
# HIRES: CONFIGURATION PARAMETERS
#
# PM WAVE BEAM OFFSET IMAGE MASKIMAGE
# +/- (um) (" MJy/sr file file
#
- 070 8.4 0.00 image_070 image_070.omask
- 100 9.4 0.00 image_100 image_100.omask
+ 160 13.5 0.00 image_160 image_160.omask
+ 250 18.2 0.00 image_250 image_250.omask
+ 350 24.9 0.00 image_350 image_350.omask
+ 500 36.3 0.00 image_500 image_500.omask
```

The files with images and observational masks are those created in the *images* directory (Sect. 3), specified with either relative or absolute path. If some of the wavebands are not present in the set of prepared images (e.g., the $100\ \mu\text{m}$ image), the corresponding lines must be deleted.

4.2 The configuration parameters

The meaning of the first line of parameters in `+ hires.cfg` is explained below.

```
- 070 8.4 0.00 image_070 image_070.omask
```

The 1st parameter ('-') indicates that the $70\ \mu\text{m}$ image will not be used for the fitting of the spectral shape of each pixel. Otherwise, the short-wavelength image would lead to the overestimated temperatures, hence underestimated surface densities. In general, images at the wavelengths shorter than $160\ \mu\text{m}$ should not be used for the fitting. The 2nd parameter (070) is the wavelength of the corresponding image, with a leading zero. The 3rd parameter (8.4) is the angular resolution, i.e., the observational beam size (FWHM, in arcsecs). The 4th parameter (0.00) is the offset (in MJy/sr) for the image, correcting the image absolute calibration. In general, the default zero must be replaced with an estimated non-zero value, which is likely to be a separate non-trivial problem. However, the value may be critically important for the acceptably good quality of the resulting surface density image. The 5th parameter (image_070) is the prepared $70\ \mu\text{m}$ image (images/universe_070) and the 6th parameter its corresponding observational mask (Sect. 3). When prepared, the configuration file may look like:

```
#
# HIRES: CONFIGURATION PARAMETERS
#
# PM WAVE BEAM OFFSET IMAGE MASKIMAGE
# +/- (um) (") MJy/sr file file
#
- 070 8.4 400 images/universe_070 images/universe_070.omask
+ 160 13.5 850 images/universe_160 images/universe_160.omask
+ 250 18.2 0 images/universe_250 images/universe_250.omask
+ 350 24.9 50 images/universe_350 images/universe_350.omask
+ 500 36.3 160 images/universe_500 images/universe_500.omask
```

4.3 Convoluting images to lower resolutions

The following command instructs `hires` to convolve all input images to all lower angular resolutions found in `+ hires.cfg` and to create the configuration files `+ hires.r250.cfg`, `+ hires.r350.cfg`, and `+ hires.r500.cfg`, necessary for the next step of fitting the spectral shapes of images pixels:

```
> hires +hires.cfg prepare
```

As a result, the following set of convolved images is created in the `hires` directory:

```
universe_070.r070 ~> 070  $\mu\text{m}$  image at the resolution of the 070  $\mu\text{m}$  image
universe_070.r160 ~> 070  $\mu\text{m}$  image at the resolution of the 160  $\mu\text{m}$  image
universe_070.r250 ~> 070  $\mu\text{m}$  image at the resolution of the 250  $\mu\text{m}$  image
universe_070.r350 ~> 070  $\mu\text{m}$  image at the resolution of the 350  $\mu\text{m}$  image
universe_070.r500 ~> 070  $\mu\text{m}$  image at the resolution of the 500  $\mu\text{m}$  image
universe_160.r160 ~> 160  $\mu\text{m}$  image at the resolution of the 160  $\mu\text{m}$  image
universe_160.r250 ~> 160  $\mu\text{m}$  image at the resolution of the 250  $\mu\text{m}$  image
universe_160.r350 ~> 160  $\mu\text{m}$  image at the resolution of the 350  $\mu\text{m}$  image
universe_160.r500 ~> 160  $\mu\text{m}$  image at the resolution of the 500  $\mu\text{m}$  image
universe_250.r250 ~> 250  $\mu\text{m}$  image at the resolution of the 250  $\mu\text{m}$  image
universe_250.r350 ~> 250  $\mu\text{m}$  image at the resolution of the 350  $\mu\text{m}$  image
universe_250.r500 ~> 250  $\mu\text{m}$  image at the resolution of the 500  $\mu\text{m}$  image
universe_350.r350 ~> 350  $\mu\text{m}$  image at the resolution of the 350  $\mu\text{m}$  image
universe_350.r500 ~> 350  $\mu\text{m}$  image at the resolution of the 500  $\mu\text{m}$  image
universe_500.r500 ~> 500  $\mu\text{m}$  image at the resolution of the 500  $\mu\text{m}$  image
```

4.4 Fitting spectral shapes of pixels with *fitfluxes*

The fitting method used by *fitfluxes* is described in [Men'shchikov \(2016\)](#); its usage information is shown, when the utility is run without parameters. The following command lists all *fitfluxes* runs and parameters, necessary to produce all variants of the surface density and temperature images with different resolutions:

```
> hires +hires.cfg fitfluxes
```

As a result, the user screen displays the following information on the parameters to supply to *fitfluxes* and on the correct sequence of the commands:

Here are the commands fitting the spectral shapes of pixels in observed images to compute the high-resolution surface density and temperature images using the HIRES algorithm, described by Men'shchikov (2021) A&A, 649, A89:

FITTING PIXEL SPECTRAL SHAPES IN IMAGES AT 500 μm RESOLUTION:

```
fitfluxes 0 0 0 -1:070,-2:160,-3:250,4:350,5:500 1:0.20,1:0.15,1:0.10,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r500.cfg -verb0  
fitfluxes 0 0 0 -1:070,-2:160,3:250,4:350,5:500 1:0.20,1:0.15,1:0.10,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r500.cfg -verb0  
fitfluxes 0 0 0 -1:070,2:160,3:250,4:350,5:500 1:0.20,1:0.15,1:0.10,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r500.cfg -verb0
```

FITTING PIXEL SPECTRAL SHAPES IN IMAGES AT 350 μm RESOLUTION:

```
fitfluxes 0 0 0 -1:070,-2:160,3:250,4:350 1:0.20,1:0.15,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r350.cfg -verb0  
fitfluxes 0 0 0 -1:070,2:160,3:250,4:350 1:0.20,1:0.15,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r350.cfg -verb0
```

FITTING PIXEL SPECTRAL SHAPES IN IMAGES AT 250 μm RESOLUTION:

```
fitfluxes 0 0 0 -1:070,2:160,3:250 1:0.20,1:0.15,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r250.cfg -verb0
```

FITTING PIXEL SPECTRAL SHAPES IN IMAGES AT 500 μm RESOLUTION (OFFSETS CONSISTENCY CHECKS):

```
fitfluxes 0 0 0 -1:160,2:250,3:350,-4:500 1:0.15,1:0.10,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r500.cfg -verb0  
fitfluxes 0 0 0 1:160,2:250,-3:350,-4:500 1:0.15,1:0.10,1:0.10,1:0.10 \  
thinbody 0:1,0:1,2:0 1 10 +hires.r500.cfg -verb0
```

Start the commands (in the sequence shown above) on different processors, in parallel.

The *fitfluxes* runs are not executed by *hires*, because it is much more efficient for the user to run the jobs in parallel in different terminal sessions (on different processors), to greatly accelerate the processing. The parallel jobs must be started *in the sequence shown, one after the other*, because the numbering of the output files depends on that. As a result, the following sets of the surface densities and temperatures are created in the *hires* directory:

```
thin.+hires.rWWW.cfg.0N.chisqu.fits  $\rightsquigarrow$   $\chi^2$  values for the resulting fit  
thin.+hires.rWWW.cfg.0N.log  $\rightsquigarrow$  log file for the run  
thin.+hires.rWWW.cfg.0N.surfden.fits  $\rightsquigarrow$  surface density at the resolution of the WWW  $\mu\text{m}$  image  
thin.+hires.rWWW.cfg.0N.surferr.fits  $\rightsquigarrow$  relative errors in surface densities  
thin.+hires.rWWW.cfg.0N.temperr.fits  $\rightsquigarrow$  relative errors in temperatures  
thin.+hires.rWWW.cfg.0N.temppers.fits  $\rightsquigarrow$  temperature at the resolution of the WWW  $\mu\text{m}$  image
```

In the above names, WWW is the wavelength and 0N is the sequential number (00, 01, 02, 03, 04) of the *fitfluxes* run. When the jobs are cancelled before they have finished, all log files with the names *.cfg.?.log must be deleted and the sequence of the *fitfluxes* jobs must be started again.

4.5 Computing high-resolution surface densities

The derived set of surface densities with three angular resolutions is used to differentially increase their resolution to any value among the resolutions present in *+hires.cfg*:

```
> hires +hires.cfg complete
```

The command creates the high-resolution surface densities and temperatures (in `+hires/+results`), employing the algorithm published by [Men'shchikov \(2021b\)](#), a substantial improvement over the previous algorithm ([Palmeirim et al. 2013](#)):

```
hi.c.surface.density.r8p4.fits  ~> high-resolution high-contrast density at 8.4'' resolution
hi.c.surface.density.r9p4.fits  ~> high-resolution high-contrast density at 9.4'' resolution
hi.c.surface.density.r13p5.fits ~> high-resolution high-contrast density at 13.5'' resolution
hi.c.surface.density.r18p2.fits ~> high-resolution high-contrast density at 18.2'' resolution
hi.c.surface.density.r24p9.fits ~> high-resolution high-contrast density at 24.9'' resolution
hi.c.surface.density.r36p3.fits ~> high-resolution high-contrast density at 36.3'' resolution
hi.surface.density.r8p4.fits    ~> high-resolution density at 8.4'' resolution
hi.surface.density.r9p4.fits    ~> high-resolution density at 9.4'' resolution
hi.surface.density.r13p5.fits   ~> high-resolution density at 13.5'' resolution
hi.surface.density.r18p2.fits   ~> high-resolution density at 18.2'' resolution
hi.surface.density.r24p9.fits   ~> high-resolution density at 24.9'' resolution
hi.surface.density.r36p3.fits   ~> high-resolution density at 36.3'' resolution
```

The high-contrast variants of the high-resolution images (`hi.c.surface.density.*.fits`) with an artificially enhanced contrast may be useful for the analysis of the fine unresolved structures, present in the images, but they may not be suitable for the `getsf` extractions. It is recommended to use only the regular versions of the images (`hi.surface.density.*.fits`) in the source and filament extractions with `getsf`. When choosing the best angular resolution of the images, the user should carefully verify their quality. This is especially critical for the images with the highest resolutions of 8.4'' and 9.4'', whose quality may be unacceptably low due to excessively high noise or artifacts.

4.6 Observational mask for surface densities

The `+hires/+results` contains an observational mask, automatically created for only the highest-resolution surface density image, i.e., for `hi.surface.density.r8p4`. If another image is planned for an extraction (e.g., `hi.surface.density.r13p5` or `hi.surface.density.r18p2`), then the mask must either be copied or re-named, for example:

```
> cp +results/hi.surface.density.r8p4.omask.fits +results/hi.surface.density.r13p5.omask.fits
> mv +results/hi.surface.density.r8p4.omask.fits +results/hi.surface.density.r18p2.omask.fits
```

The surface density mask is created as the intersection of the masks of the individual images (Sect. 3.6) specified in the configuration file `+hires.cfg` (Sect. 4.1), therefore it does not require any adjustments.

5 Configuring source and filament extraction

In order to complete the preparation for the `getsf` extraction, it is necessary to create the default configuration file `+getsf.cfg` by executing `getsf` in a directory that does not contain that file:

```
> cd ../230322/runs
> getsf
```

5.1 The `getsf` configuration file

The content of the default configuration file `+getsf.cfg` facilitates preparations for the 6 *Herschel* wavebands. However, the information can be easily modified for use with any other set of images. When opening files with '+' in their names in the text editors interpreting the '+' character as a special symbol, it is necessary to append the './' string to the file name, to tell the editor that the character is a part of the file name, for example:

```
> nedit ./+getsf.cfg
```

The contents of the default configuration file:

```

#-----
#
# GETSF • Multi-Scale Multi-Wavelength Source & Filament Extraction • Version 230320 • Alexander
#-----
#
# VALUES          | PARAMETERS          | COMMENT NB: default beams are for
#-----|-----|-----
#
fieldname 100      | prefix distance ..... ! prefix for naming files (never use
6                | nwmax .....{ nwmax < 100 } ! maximum number of wavelengths spec
6 1 2 3 4 5 6     | nwaves nw[i] ..... ! number of wavelengths and their i
070 8.4 s070 f070 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
/path/to/image ../070 | ..... ! paths to image and monochromatic d
image_070         | ..... ! prepared image name
100 9.4 s100 f100 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
/path/to/image ../100 | ..... ! paths to image and monochromatic d
image_100        | ..... ! prepared image name
160 13.5 s160 f160 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
/path/to/image ../160 | ..... ! paths to image and monochromatic d
image_160        | ..... ! prepared image name
250 18.2 s250 f250 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
/path/to/image ../250 | ..... ! paths to image and monochromatic d
image_250        | ..... ! prepared image name
350 24.9 s350 f350 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
/path/to/image ../350 | ..... ! paths to image and monochromatic d
image_350        | ..... ! prepared image name
500 36.3 s500 f500 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
/path/to/image ../500 | ..... ! paths to image and monochromatic d
image_500        | ..... ! prepared image name
y y              | sources filaments ..... ! extract sources and/or filaments {
y 1 1 50         | separate its1 itf1 niter ..... ! separate components and background
y 1 1           | flatten itsd1 itfd1 ..... ! flatten background-subtracted comp
y               | complete ..... ! complete separation and flattening
y               | combine ..... ! combine decomposed structures over
y 2             | detect skelsig ..... ! detect structures; skeleton signif
y 1 1 50 n      | measure goodmin itn itx sproima .... ! measure structures; minimum goodne
y               | visualize ..... ! visualize sources
5               | nsigmacombos (!expert) ..... ! number of sigmas for combining sou
2               | nsigmacombof (!expert) ..... ! number of sigmas for combining fil
1               | nwavesdetect ..... ! required minimum number of wavelen
2 ?            | savespace delete ..... ! space saving level {0|1|2}: clean
0               | verbosity ..... ! verbosity level {0|1|2} for the sc
#-----

```

5.2 The configuration parameters

The meaning of each line of parameters in `+getsf.cfg` is explained below.

```
fieldname 100      | prefix distance ..... !
```

The extraction must be given a name that will serve as the prefix for naming the output files. It makes sense to replace the placeholder `fieldname` with the name of the observed region, for example, `universe`. The number `100` is the distance (in parsecs) to the region, used to compute the masses and linear densities of filaments. If the distance is unknown or unimportant, the default value may be kept.

```
6                | nwmax .....{ nwmax < 100 } !
```

This number (`nwmax`) is the total number of images (wavebands) specified in this configuration file. This value is suitable for the default set of 6 *Herschel* images (the upper limit is 99).

```
6 1 2 3 4 5 6     | nwaves nw[i] ..... !
```

The 1st number (`nwaves`) is the maximum number of images (wavebands) to be used in the current run of `getsf`. The other six numbers refer to the corresponding images used in the current run. This default line of parameters means that the extraction will be done sequentially, processing the next image only after the previous one has been completed (Sect. 6.1). This is the easiest way of running `getsf`, but it makes sense for only relatively small images. For large images, it is recommended to always process the images in parallel independent `getsf` runs (Sect. 6.2) to strongly reduce the total computation time, approximately by the factor `nwmax`. Hence, for the six parallel jobs, this line of parameters takes the following values:

```
1 1          | nwaves nw[i] ..... ! ~ for the 1st parallel run
1 2          | nwaves nw[i] ..... ! ~ for the 2nd parallel run
1 3          | nwaves nw[i] ..... ! ~ for the 3rd parallel run
1 4          | nwaves nw[i] ..... ! ~ for the 4th parallel run
1 5          | nwaves nw[i] ..... ! ~ for the 5th parallel run
1 6          | nwaves nw[i] ..... ! ~ for the 6th parallel run
```

More information on the source and filament extraction with `getsf` is found in Sect. 6.

```
070 8.4 s070 f070 n col | ..... !
```

The 1st number (`070`) is the wavelength (in microns) of the corresponding image with a leading zero. The number of digits must be the same for all wavelengths, hence the leading zero. The 2nd number (`8.4`) is the angular resolution, i.e., the observational beam size (FWHM, in arcsecs). All default beams correspond to the *Herschel* fast parallel (PACS/SPIRE) scanning mode ($60''\text{s}^{-1}$). The 3rd parameter (`s070`) is the placeholder for the maximum size (FWHM, arcsec) of the sources to extract in this image. The 4th parameter (`f070`) is the placeholder for the maximum width (FWHM, arcsec) of the filaments to extract in this image. If a user is interested in only unresolved sources and filaments, both `s070` and `f070` must be set to the angular resolution. If there are no filaments or there is no interest in filaments, then `f070` should be set to the angular resolution *and* processing of filaments must be disabled by setting `filaments` to `'n'` (see below). For more information on the maximum size (width) of sources and filaments, refer to Sect. 3.1.3 in [Men'shchikov \(2021b\)](#). This single parameter of `getsf` can be conveniently estimated by opening an image in `ds9` and creating a region, fully encircling the *entire* emission peak of the largest source or filament of interest; the *radius* of the circular region (in arcsec) is the value that must replace the respective placeholder. The maximum size may also be entered as negative values to stipulate that the image must not be used for the detection of sources (`-s070`) or filaments (`-f070`). Then the image will only be used for the measurements of the sources or filaments, detected in the other images. The 5th parameter (`'n'`) disables the *Herschel* aperture corrections, therefore it must be enabled (changed to `'y'`) for the *Herschel* images. The last *optional* parameter (`col`) is the placeholder for the desired color of sources and skeletons in the `ds9` region files `*.reg`, created by `getsf` to aid users in visualizations and analyses. The `getsf` colors are abbreviated as `'gre'`, `'blu'`, `'ora'`, `'pur'`, `'cya'`, `'mag'`, `'lim'`, `'pin'`, `'tea'`, `'lav'`, `'bro'`, `'bei'`, `'mar'`, `'min'`, `'oli'`, `'apr'`, `'gra'`, `'red'`, `'yel'`, `'whi'`, and `'bla'`. If the optional parameter is not specified, then the default colors are used.

```
/path/to/image ../070 | ..... !
```

The 1st parameter (`/path/to/image`) is the absolute or relative path to the prepared image, i.e., to the `images` directory (Sect. 3). The 2nd parameter (`../070`) is the relative path to the monochromatic directory, created by `getsf`, where all files related to the processing of the image are created. The two paths must be separated by space, the monochromatic directory must be separated by space from the vertical line (`|`) that follows it, and *the trailing slash (/) in the paths must never be used*.

```
image_070      | ..... !
```

The name of the observed image, without the `.fits` extension, that was prepared in the `images` directory, i.e., `universe_070`. The observational mask is not specified, because it has the same name as the image, with `.omask` attached (Sect. 3.6).

```
100 9.4 s100 f100 n col | ..... !
/path/to/image ../100 | ..... !
image_100           | ..... !
160 13.5 s160 f160 n col | ..... !
/path/to/image ../160 | ..... !
image_160           | ..... !
```

```

250 18.2 s250 f250 n col | ..... !
/path/to/image ../250 | ..... !
image_250 | ..... !
350 24.9 s350 f350 n col | ..... !
/path/to/image ../350 | ..... !
image_350 | ..... !
500 36.3 s500 f500 n col | ..... !
/path/to/image ../500 | ..... !
image_500 | ..... !

```

All other wavebands are edited the same way as the first one. If `nwmax` is smaller than 6, the unused entries for the extra wavebands of the default configuration file must be deleted.

```

y y | sources filaments ..... !

```

The 1st parameter (`sources`) enables the extraction of sources and the 2nd parameter (`filaments`) enables the extraction of filaments. Changing either of them to `n` disables the processing of the respective structural component in `getsf`. If the user is interested only in the source extraction, the 2nd parameter may be changed to `'n'`. However, if the user is interested only in the filament extraction, it is not recommended to disable the source extraction, because the extracted filaments would remain contaminated by the sources. Therefore, if the user is interested in the filament extraction, both parameters should remain `'y'`.

```

y 1 1 50 | separate its1 itf1 niter ..... !

```

The 1st parameter (`separate`) enables the separation step of the structural components; this parameter may be changed to `'n'`, when `getsf` must be restarted to process only the other steps. When the separation step is enabled, but the image processing must be restarted for any reason, the 2nd parameter (`its1`) defines the number of the iteration to restart with for the component of sources and the 3rd parameter (`itf1`) defines the same for the component of filaments. The last parameter (`niter`) defines the maximum number of iterations (50) in the separation of the components.

```

y 1 1 | flatten itsd1 itfd1 ..... !

```

The 1st parameter (`flatten`) enables the flattening step of the structural components; this parameter may be changed to `'n'`, when `getsf` must be restarted to process only the other steps. When the flattening step is enabled, but the image processing must be restarted for any reason, the 2nd parameter (`itsd1`) defines the number of the iteration to restart with for the component of sources and the 3rd parameter (`itfd1`) defines the same for the component of filaments. The maximum number of iterations in the flattening of the components is defined by `niter` from the separation step.

```

y | complete ..... !

```

This parameter (`complete`) enables the completion step after the separation and flattening of the structural components; this parameter may be changed to `'n'`, when `getsf` must be restarted to process only the other steps.

```

y | combine ..... !

```

This parameter (`combine`) enables the combination of the detection images of sources and filaments over wavebands; this parameter may be changed to `'n'`, when `getsf` must be restarted to process only the other steps.

```

y 2 | detect skelsig ..... !

```

The 1st parameter (`detect`) enables the detection of sources and filaments; this parameter may be changed to `'n'`, when `getsf` must be restarted to process only the other steps. The 2nd parameter (`skelsig`) is the desired significance of the skeletons (crests of filaments); the default value may be changed, although this is not recommended.

```

y 1 1 50 n | measure goodmin itn itx sproima .... !

```

The 1st parameter (`measure`) enables the measurements of sources and filaments; this parameter may be changed to `'n'`, when `getsf` must be restarted to process only the other steps. The 2nd parameter (`goodmin`) defines the minimum goodness of sources that are cataloged. When the measurements are enabled, but they must be restarted for any reason, the 3rd parameter (`itn`) defines the number of the iteration to restart with for the component of sources. The 4th parameter (`itx`) defines the maximum number of the measurement iterations (50). The last parameter (`sproima`), defaulting to `'n'`, may be changed to `'y'` to enable creation of small individual images and intensity profiles of each (background-subtracted) cataloged source.


```
y | visualize ..... !
```

This parameter (`visualize`) enables the visualization of extracted sources and filaments; this parameter may be changed to ‘n’, when `getsf` must be restarted to process only the other steps.

```
5 | nsigmacombos (!expert) ..... !
```

This parameter (`nsigmacombos`) defines the default number of standard deviations in thresholding the component of sources, when combining them over wavelengths. The default value (5) is optimal and it should never be changed by non-experts.

```
2 | nsigmacombof (!expert) ..... !
```

This parameter (`nsigmacombof`) defines the default number of standard deviations in thresholding the component of filaments, when combining them over wavelengths. The default value (2) is optimal and it should never be changed by non-experts.

```
1 | nwavesdetect ..... !
```

This parameter (`nwavesdetect`) defines the required minimum number of images (wavebands), in which a source is detected and significant, to be included in the final catalog of sources. This value may be increased to 2, in order to reduce the number of potentially spurious sources in the catalogs. However, higher values of this parameter may also lead to the elimination of the real sources.

```
2 ? | savespace delete ..... !
```

The 1st parameter (`savespace`) controls removal of the files created by `getsf` (when it has finished), defining the hard drive cleaning level: 0 (none), 1 (medium), or 2 (maximum). In the 2nd parameter (`delete`), the user can specify, whether `getsf` must wait for a confirmation before deleting the files (‘?’) or assume that the user would confirm (‘y’) or disapprove (‘n’) the actual deletion of files, without asking any question.

```
0 | verbosity ..... !
```

This parameter (`verbosity`) defines three levels for the output produced by `getsf` on the user screen: ‘0’ (minimum), ‘1’ (medium), or ‘2’ (maximum).

6 Extracting sources and filaments

This section presents examples of the `getsf` extraction runs, illustrating how `getsf` runs in sequential and parallel modes. It is assumed that the prepared data set consists of the five *Herschel* images, observed in the PACS/SPIRE fast parallel mode, and an additional high-resolution surface density image. It is also assumed that the user has supplied all required information about prepared images and the maximum sizes of interest in the upper section of `+getsf.cfg`, for example:

```
#-----
#
# GETSF • Multi-Scale Multi-Wavelength Source & Filament Extraction • Version 230320 • Alexander
#-----
#
# VALUES | PARAMETERS | COMMENT NB: default beams are for
#-----|-----|-----
#
universe 1000 | prefix distance ..... ! prefix for naming files (never use
6 | nwmax .....{ nwmax < 100 } ! maximum number of wavelengths spec
6 1 2 3 4 5 6 | nwaves nw[i] ..... ! number of waveblengths and their i
070 8.4 17 17 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
../images ../070 | ..... ! paths to image and monochromatic d
universe_070 | ..... ! prepared image name
160 13.5 27 27 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
../images ../160 | ..... ! paths to image and monochromatic d
universe_160 | ..... ! prepared image name
```

```

165 13.5 27 27 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
../images/hires/+results ../165 | ..... ! paths to image and monochromatic d
hi.surface.density.r13p5 | ..... ! prepared image name
250 18.2 36 36 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
../images ../250 | ..... ! paths to image and monochromatic d
universe_250 | ..... ! prepared image name
350 24.9 50 50 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
../images ../350 | ..... ! paths to image and monochromatic d
universe_350 | ..... ! prepared image name
500 36.3 72 72 n col | ..... ! wave, beam, [-]srcmaxsize, [-]film
../images ../500 | ..... ! paths to image and monochromatic d
universe_500 | ..... ! prepared image name

```

6.1 Running *getsf* sequentially for all images

The bottom section of the configuration file tells *getsf* how to proceed with the extraction:

```

y y          | sources filaments ..... ! extract sources and/or filaments {
y 1 1 50     | separate its1 itf1 niter ..... ! separate components and background
y 1 1       | flatten itsd1 itfd1 ..... ! flatten background-subtracted comp
y          | complete ..... ! complete separation and flattening
y          | combine ..... ! combine decomposed structures over
y 2         | detect skelsig ..... ! detect structures; skeleton signif
y 1 1 50 n   | measure goodmin itn itx sproima .... ! measure structures; minimum goodne
y          | visualize ..... ! visualize sources
5          | nsigmacombos (!expert) ..... ! number of sigmas for combining sou
2          | nsigmacombof (!expert) ..... ! number of sigmas for combining fil
1          | nwavesdetect ..... ! required minimum number of wavelen
2 ?       | savespace delete ..... ! space saving level {0|1|2}: clean
0         | verbosity ..... ! verbosity level {0|1|2} for the sc
#-----

```

The first line tells *getsf* that both sources and filaments must be extracted.

Note. The correct approach to *getsf* extractions is first to inspect the observed images for presence of filaments. If no filament-like structures are visible or there is no interest in the filaments, then processing of filaments must be disabled by setting `filaments` to ‘n’. Processing of sources must *always* be enabled: the source-like peaks must be separated from filaments or background fluctuations. Only in rare cases, when *getsf* has to be restarted from intermediate processing steps, sources may be disabled, depending on the circumstances.

If there is interest only in extracting sources, the *getsf* execution time can be reduced by roughly a factor of two, by setting `filaments` to ‘n’:

```

y n          | sources filaments ..... ! extract sources and/or filaments {

```

The next block of parameters controls execution of the corresponding processing steps of *getsf*:

```

y 1 1 50     | separate its1 itf1 niter ..... ! separate components and background
y 1 1       | flatten itsd1 itfd1 ..... ! flatten background-subtracted comp
y          | complete ..... ! complete separation and flattening
y          | combine ..... ! combine decomposed structures over
y 2         | detect skelsig ..... ! detect structures; skeleton signif
y 1 1 50 n   | measure goodmin itn itx sproima .... ! measure structures; minimum goodne
y          | visualize ..... ! visualize sources

```

By default, they instruct *getsf* to process all images in the sequence of their appearance in the upper section of the configuration file. This is the simplest approach to running *getsf*: the user makes the extraction directory (runs) the current one and starts the code within a new *screen* session:

```

> cd 230322/runs
> screen -h 1000 -LS get
> getsf

```

Having started the code, the user may forget about it for some time, until it finishes (the time may be very long for many large images). This simplest approach to running *getsf* is quite inefficient for the multi-wavelength extractions, because most of the time the code spends in the first two steps (*separate* and *flatten*), processing the images completely independently of each other. A natural and easy way to accelerate the extraction is to parallelize the image processing in independent jobs, one *getsf* run per wavelength.

6.2 Running *getsf* in parallel for each image

The parallel processing mode of images makes sense only for the steps of the separation of structural components (*separate*), flattening of background-subtracted images (*flatten*), and completion of the two steps (*complete*). This means that the parameters *combine*, *detect*, *measure*, and *visualize* must be changed from ‘y’ to ‘n’, for example:

```

y y          | sources filaments ..... ! extract sources and/or filaments {
y 1 1 50     | separate its1 itf1 niter ..... ! separate components and background
y 1 1       | flatten itsd1 itfd1 ..... ! flatten background-subtracted comp
y          | complete ..... ! complete separation and flattening
n          | combine ..... ! combine decomposed structures over
n 2        | detect skelsig ..... ! detect structures; skeleton signif
n 1 1 50 n  | measure goodmin itn itx sproima .... ! measure structures; minimum goodne
n          | visualize ..... ! visualize sources

```

Before running *getsf*, it is necessary to modify the line that defines the wavelengths to proceed. Edit the configuration file by specifying the index of the 1st image:

```

1 1          | nwaves nw[i] ..... ! number of wavelengths and their i

```

Open a new *screen* session and start *getsf* for the 1st image:

```

> screen -h 1000 -LS get1
> getsf

```

Edit the configuration file by specifying the index of the 2nd image:

```

1 2          | nwaves nw[i] ..... ! number of wavelengths and their i

```

Open a new *screen* session and start *getsf* for the 2nd image:

```

> screen -h 1000 -LS get2
> getsf

```

Edit the configuration file by specifying the index of the 3rd image:

```

1 3          | nwaves nw[i] ..... ! number of wavelengths and their i

```

Open a new *screen* session and start *getsf* for the 3rd image:

```

> screen -h 1000 -LS get3
> getsf

```

Edit the configuration file by specifying the index of the 4th image:

```

1 4          | nwaves nw[i] ..... ! number of wavelengths and their i

```

Open a new *screen* session and start *getsf* for the 4th image:

```

> screen -h 1000 -LS get4
> getsf

```

Edit the configuration file by specifying the index of the 5th image:

```

1 5          | nwaves nw[i] ..... ! number of wavelengths and their i

```

Open a new *screen* session and start *getsf* for the 5th image:

```
> screen -h 1000 -LS get5
> getsf
```

Edit the configuration file by specifying the index of the 6th image:

```
1 6          | nwaves nw[i] ..... ! number of wavelengths and their i
```

Open a new *screen* session and start *getsf* for the 6th image:

```
> screen -h 1000 -LS get6
> getsf
```

Having started the parallel jobs, the user may forget about them for some time, until they all finish (the time may be very long for very large images). Then it is necessary to invert the controls in the configuration file ('n' ↔ 'y') to tell *getsf* to complete all remaining steps:

```
y y          | sources filaments ..... ! extract sources and/or filaments {
n 1 1 50     | separate its1 itf1 niter ..... ! separate components and background
n 1 1        | flatten itsd1 itfd1 ..... ! flatten background-subtracted comp
n            | complete ..... ! complete separation and flattening
y            | combine ..... ! combine decomposed structures over
y 2         | detect skelsig ..... ! detect structures; skeleton signif
y 1 1 50 n   | measure goodmin itn itx sproima .... ! measure structures; minimum goodne
y            | visualize ..... ! visualize sources
```

The final *getsf* run completes the extraction (in a much shorter time than that required by the parallel steps):

```
> screen -h 1000 -LS get
> getsf
```

7 Understanding extraction output files

Names of some output files contain numbers: WWW is the waveband (μm), NN the waveband number, JJ the scale number, OO the angular resolution ("), TT the iteration number, SSSSSS the decomposition scale ("), FFFF the maximum scale (") of the partially reconstructed filament, GG the skeleton significance, XX or XXX the maximum size (") of sources, YY or YYY the maximum size (") of filaments, KKKK the skeleton scale ("), and NNNN the filament number. The images with *obs* and *det* in their names are related to the original images and detection images, respectively.

7.1 Directory structure created by *getsf* runs

When *getsf* is run in the extraction directory (230322/runs) with the configuration example described in Sect. 6, the following structure of sub-directories is created:

```
230322/070 ~> monochromatic processing (70  $\mu\text{m}$  image)
230322/160 ~> monochromatic processing (160  $\mu\text{m}$  image)
230322/165 ~> monochromatic processing (surface density image with the 160  $\mu\text{m}$  image resolution)
230322/250 ~> monochromatic processing (250  $\mu\text{m}$  image)
230322/350 ~> monochromatic processing (350  $\mu\text{m}$  image)
230322/500 ~> monochromatic processing (500  $\mu\text{m}$  image)
230322/components ~> separated structural components (all wavebands)
230322/components/+obsimage_partial ~> partially reconstructed images of original images
230322/components/+filaments_partial ~> partially reconstructed images of filaments
230322/runs/results ~> detection and measurement results
230322/runs/results/+components ~> improved structural components (all wavebands)
230322/runs/results/+sources/+catalogs ~> catalogs of extracted sources
230322/runs/results/+sources/+images ~> individual images of extracted sources
230322/runs/results/+sources/+iterations ~> measurement iterations of sources (technical)
230322/runs/results/+sources/+profiles ~> individual profiles of extracted sources
230322/runs/results/+sources/+visuals ~> visualization of extracted sources
```

230322/runs/results/+skeletons ~ derived skeletons of filaments
 230322/runs/results/+filaments/+images/070 ~ measurements of filaments (images)
 230322/runs/results/+filaments/+images/160 ~ measurements of filaments (images)
 230322/runs/results/+filaments/+images/165 ~ measurements of filaments (images)
 230322/runs/results/+filaments/+images/250 ~ measurements of filaments (images)
 230322/runs/results/+filaments/+images/350 ~ measurements of filaments (images)
 230322/runs/results/+filaments/+images/500 ~ measurements of filaments (images)
 230322/runs/results/+filaments/+profiles/070 ~ measurements of filaments (catalogs)
 230322/runs/results/+filaments/+profiles/160 ~ measurements of filaments (catalogs)
 230322/runs/results/+filaments/+profiles/165 ~ measurements of filaments (catalogs)
 230322/runs/results/+filaments/+profiles/250 ~ measurements of filaments (catalogs)
 230322/runs/results/+filaments/+profiles/350 ~ measurements of filaments (catalogs)
 230322/runs/results/+filaments/+profiles/500 ~ measurements of filaments (catalogs)
 230322/runs/results/+filaments/+visuals ~ visualization of extracted filaments

Note. When an extraction has finished and the user has no plans to restart the same extraction in the future, it is possible to save considerable disk space by removing unnecessary files. In the case that the extraction is indeed deemed final, the user needs just the `components` and `results` directories in the future. Therefore, it may be recommended that the user removes all other directories: `070`, `160`, `165`, `250`, `350`, `500`, `+iterations`, and all *files* from the extraction directory `runs`, *with an exception* of the `getsf` configuration and log files (`+getsf.cfg`, `+log.*`) that must be preserved as a documentation. Another possibility is to compress the directories using an archiver that produces highly compressed solid archives, e.g., `rar` (<https://www.rarlab.com/download.htm>).

7.2 Directory for `getsf` extraction runs

The directory `runs` contains the following files:

<fieldname>.WWW.obs.sources.n.fits ~ regularized component of sources
 <fieldname>.WWW.obs.sources.n.cb.fits ~ interpolated background of regularized component of sources
 <fieldname>.WWW.obs.filaments.n.fits ~ regularized component of filaments
 <fieldname>.WWW.det.sflatfactor.fits ~ flattening image for the component of sources
 <fieldname>.WWW.det.fflatfactor.fits ~ flattening image for the component of filaments
 <fieldname>.WWW.det.fits ~ detection image for the component of sources or filaments
 <fieldname>.WWW.obs.fits ~ original observed image
 <fieldname>.WWW.omask.fits ~ observational mask
 <fieldname>.WWW.bs.omask.fits ~ observational mask for background subtraction
 <fieldname>.f{w|N}.det.JJ.cSSSSSas.an.fits ~ detection images for filaments combined over waves
 <fieldname>.s{w|N}.det.JJ.cSSSSSas.an.fits ~ detection images for sources combined over waves
 <fieldname>.s{w|N}.det.JJ.cSSSSSas.as.fits ~ detection images for sources combined over waves
 <fieldname>.f{w|N}.coadded.omask.fits ~ observational mask accumulated over wavebands for filaments
 <fieldname>.s{w|N}.coadded.omask.fits ~ observational mask accumulated over wavebands for sources
 <fieldname>.combining.WWW.omask.fits ~ special mask (optional) to exclude bad pixels in combining
 sm.<fieldname>.WWW.obs.subsfp.fits ~ footprints of sub-structured sources (if any)
 sm.<fieldname>.s{w|N}.det.footprints.fits ~ footprints of all detected sources
 sm.<fieldname>.s{w|N}.det.unrotated.fits ~ positions of all detected sources
 sm.<fieldname>.s{w|N}.det.wcs ~ conversion table from pixels to world coordinates of sources
 sm.<fieldname>.s{w|N}.det.cat ~ detection catalog of sources
 sm.<fieldname>.WWW.obs.smf.cat ~ catalog of the detected sources updated in measurement iterations
 sm.seg.<fieldname>.s{w|N}.det.JJ.cSSSSSas.fits ~ segmentation of sources at scale SSSSSS
 sm.seg.<fieldname>.s{w|N}.det.JJ.cSSSSSas.fp.fits ~ footprints of detected sources at scale SSSSSS
 sm.seg.<fieldname>.s{w|N}.det.JJ.cSSSSSas.cat ~ detection catalog of sources at scale SSSSSS
 bigfoot.WWW.fits ~ annuli around the source footprints (used for flux errors estimation)
 flatfoot.WWW.fits ~ converged footprints of all acceptable sources
 <fieldname>.mask.s~max.wWWW.sum.hi.fits ~ masks of the structures accumulated over spatial scales
 <fieldname>.mask.s~max.wWWW~WWW.sum.hi.fits ~ masks accumulated over scales and wavebands
 <fieldname>.s{w|N}.det.combine ~ technical info on the detection images for sources being combined

<fieldname>.f{w|N}.det.combine ~ technical info on the detection images for filaments being combined
 <fieldname>.s{w|N}.det.combined ~ technical info on the combined detection images for sources
 <fieldname>.f{w|N}.det.combined ~ technical info on the combined detection images for filaments
 <fieldname>.WWW.srcmaxsize ~ technical info on the maximum size of sources
 <fieldname>.WWW.det.lastbright ~ technical info on the last bright spatial scale
 +wave.WWW.done ~ technical info on convergence of the source measurement iterations
 +wave.WWW.foot ~ technical info on footprint expansion or shrinkage in measurement iterations

7.3 Directory for monochromatic processing

The directories runs/./WWW contain the following files:

<fieldname>.WWW.obs.sources.fits ~ separated component of sources
 <fieldname>.WWW.obs.filaments.fits ~ separated component of filaments
 <fieldname>.WWW.obs.sources.n.fits ~ regularized component of sources
 <fieldname>.WWW.obs.filaments.n.fits ~ regularized component of filaments
 <fieldname>.WWW.det.sources.fits ~ detection image of the separated component of sources
 <fieldname>.WWW.det.filaments.fits ~ detection image of the separated component of filaments
 <fieldname>.WWW.obs.fbackground.fits ~ background of the separated component of filaments
 <fieldname>.WWW.obs.sbackground.fits ~ background of the separated component of sources
 <fieldname>.WWW.det.fits ~ updated background image of the separated component
 <fieldname>.WWW.omask.fits ~ observational mask image
 <fieldname>.WWW.det.decompose ~ technical info on the decomposed single scale images
 <fieldname>.WWW.det.lastbright ~ technical info on last bright spatial scale
 <fieldname>.WWW.det.separateso ~ technical info on the decomposed images of sources
 <fieldname>.WWW.det.separatefi ~ technical info on the decomposed images of filaments
 +log.GETSF.iter.NN ~ technical info on the threshold iterations for decomposed images

7.3.1 Directory for separation of sources or filaments

The directories runs/./WWW/+components/{sources|filaments} contain the following files:

<fieldname>.WWW.det.bg.TT.fits ~ background of the component in iterations
 <fieldname>.WWW.det.co.TT.fits ~ separated component in iterations
 <fieldname>.WWW.det.bg.TT.m.fits ~ background after subtraction of a constant
 <fieldname>.WWW.det.bg.TT.r.fits ~ relative correction to the background in iterations
 <fieldname>.WWW.det.bg.fits ~ background of the component at the latest iteration
 <fieldname>.WWW.det.co.fits ~ separated component at the latest iteration
 <fieldname>.WWW.det.fits ~ initial image for the component separation
 <fieldname>.WWW.det.fm.01.fits ~ cleaning masks accumulated over scales at the first iteration
 <fieldname>.WWW.det.b.fits ~ approximately reconstructed background of the component
 <fieldname>.WWW.obs.bg.sXXas.fits ~ final separated background of the component
 <fieldname>.WWW.obs.bg.sXXas.n.fits ~ final regularized background of the component
 <fieldname>.WWW.obs.bgs.sXXas.fits ~ final separated component
 <fieldname>.WWW.obs.bgs.sXXas.std.fits ~ standard deviations in the component
 <fieldname>.WWW.obs.bgs.sXXas.flat.fits ~ flattened separated component
 <fieldname>.WWW.obs.bgs.sXXas.flat.std.fits ~ standard deviations in the flat component
 <fieldname>.WWW.obs.bgs.sXXas.n.fits ~ regularized component
 <fieldname>.WWW.obs.bgs.sXXas.n.flat.fits ~ regularized flattened component
 <fieldname>.WWW.obs.bgs.sXXas.n.flat.std.fits ~ standard deviations in the regularized component
 <fieldname>.WWW.obs.fits ~ original observed image
 <fieldname>.WWW.det.sigma ~ technical info on the standard deviation over all scales
 <fieldname>.WWW.det.decompose ~ technical info on the decomposed images
 <fieldname>.WWW.det.lastbright ~ technical info on the last bright spatial scale
 <fieldname>.WWW.det.separatebg ~ technical info on the decomposed component
 <fieldname>.WWW.filmaxsize ~ technical info on the maximum size of filaments

7.3.2 Directory for flattening of sources or filaments

The directories `runs/./WWW/+components/+{srcfactor|filfactor}` contain the following files:

<fieldname>.WWW.det.bg.TT.fits ~ background of the standard deviations in iterations
<fieldname>.WWW.det.co.TT.fits ~ separated component in iterations
<fieldname>.WWW.det.bg.TT.m.fits ~ background after subtraction of a constant
<fieldname>.WWW.det.bg.TT.r.fits ~ relative correction to the background in iterations
<fieldname>.WWW.det.bg.fits ~ background of standard deviations at the latest iteration
<fieldname>.WWW.det.co.fits ~ separated components at the latest iteration
<fieldname>.WWW.det.fits ~ initial image for the component separation
<fieldname>.WWW.det.fm.01.fits ~ cleaning masks accumulated over scales at the first iteration
<fieldname>.WWW.det.b.fits ~ approximately reconstructed background of the standard deviations
<fieldname>.WWW.obs.bgs.sXXas.std.bg.s00as.fits ~ separated background (flattening image)
<fieldname>.WWW.obs.fits ~ initial image of standard deviations
<fieldname>.WWW.det.sigma ~ technical info on the standard deviation over all scales
<fieldname>.WWW.det.decompose ~ technical info on the decomposed images
<fieldname>.WWW.det.lastbright ~ technical info on the last bright spatial scale
<fieldname>.WWW.det.separateff ~ technical info on the decomposed component
<fieldname>.WWW.filmaxsize ~ technical info on the maximum size of filaments

7.3.3 Directory for completion of sources or filaments

These directories (`runs/./WWW/+components/+{scomplete|fcomplete}`) contain the following files:

<fieldname>.WWW.det.fits ~ flattened detection image for the structural component
<fieldname>.WWW.det.sigma ~ technical info on the standard deviation over all scales

7.4 Directory for separated structural components

This directory `runs/./components` contains the following files:

<fieldname>.WWW.det.sources.fits ~ separated component of sources (flattened)
<fieldname>.WWW.det.filaments.fits ~ separated component of filaments (flattened)
<fieldname>.WWW.det.sbackground.fits ~ separated background for the component of sources
<fieldname>.WWW.det.fbackground.fits ~ separated background for the component of filaments
<fieldname>.WWW.det.sflatfactor.fits ~ flattening factor image for the component of sources
<fieldname>.WWW.det.fflatfactor.fits ~ flattening factor image for the component of filaments
<fieldname>.WWW.obs.sources.fits ~ separated component of sources (observed)
<fieldname>.WWW.obs.sources.n.fits ~ regularized component of sources (observed)
<fieldname>.WWW.obs.filaments.fits ~ separated component of filaments (observed)
<fieldname>.WWW.obs.filaments.n.fits ~ regularized component of filaments (observed)
<fieldname>.WWW.obs.sbackground+sources.fits ~ background of sources plus sources
<fieldname>.WWW.obs.fbackground+filaments.fits ~ background of filaments plus filaments

7.4.1 Directory for partially reconstructed images

The directories `runs/./WWW/+components/+{obsimage|filaments}_partial` contain the following files:

<fieldname>.WWW.det.FFFFas.fits ~ observed image or the filaments reconstructed up to scale FFFF

7.5 Directory for output images and catalogs

The directory `runs/results` contains several subdirectories.

7.5.1 Directory for improved structural components

The directory `runs/results/+components` contains the following files:

<fieldname>.WWW.obs.fits \leadsto original observed image
 <fieldname>.s{w|N}.WWW.obs.sources.fits \leadsto sources obtained by subtraction of their interpolated background
 <fieldname>.s{w|N}.WWW.obs.sbackground.fits \leadsto background of sources interpolated under their footprints
 <fieldname>.WWW.obs.fbackground.fits \leadsto regularized background of separated component of filaments
 <fieldname>.WWW.obs.filaments.fits \leadsto observed filaments recomputed as sbackground-fbackground
 <fieldname>.WWW.det.filaments.fits \leadsto flattened image of observed recomputed filaments

7.5.2 Directory for catalogs of extracted sources

The directory runs/results/+sources/+catalogs contains the following files:

<fieldname>.s{w|N}.WWW.sources.cat \leadsto final catalog of extracted sources (numbering from detection)
 <fieldname>.s{w|N}.WWW.sources.ok.cat \leadsto final catalog of extracted sources (resorted and renumbered)
 <fieldname>.s{w|N}.WWW.sources.ok.cat.ell.fk5.reg \leadsto half-max ellipses saved in a ds9 region file
 <fieldname>.s{w|N}.WWW.sources.ok.cat.ell.phy.reg \leadsto half-max ellipses saved in a ds9 region file
 <fieldname>.s{w|N}.WWW.sources.ok.cat.foo.fk5.reg \leadsto footprint ellipses saved in a ds9 region file
 <fieldname>.s{w|N}.WWW.sources.ok.cat.foo.phy.reg \leadsto footprint ellipses saved in a ds9 region file
 <fieldname>.s{w|N}.WWW.sources.ok.add.cat \leadsto additional catalog for sources (resorted and renumbered)
 <fieldname>.s{w|N}.WWW.obs.chisq.dat \leadsto parameters of elliptical Gaussians fits to extracted sources
 sm.<fieldname>.s{w|N}.det.cat \leadsto catalog of detected sources

7.5.3 Directory for measurement iterations of sources

The directory runs/results/+sources/+iterations contains the following files:

<fieldname>.s{w|N}.sources.TT.cat \leadsto final catalog of extracted sources in measurement iterations
 <fieldname>.s{w|N}.sources.ok.TT.cat \leadsto final catalog of extracted sources (renumbered) in iterations
 <fieldname>.WWW.det.NN.cSSSSSas.fpeak.deblending.dat \leadsto info on single-scale source peak deblending
 sm.s{w|N}.WWW.foot.expansion.TT.dat \leadsto footprint expansion and shrinkage in measurement iterations
 sm.s{w|N}.WWW.fpeak.deblending.TT.dat \leadsto deblending source peak intensity in measurement iterations
 sm.<fieldname>.WWW.obs.foots.TT.fits \leadsto footprints of sources in measurement iterations
 sm.s{w|N}.<fieldname>.WWW.obs.smf.TT.cat \leadsto catalog of detected sources updated in measurement iterations
 sm.<fieldname>.WWW.obs.smf.cat \leadsto catalog of detected sources at the last measurement iteration
 sm.<fieldname>.WWW.obs.smf.previous.cat \leadsto catalog of detected sources at previous to last iteration

7.5.4 Directory for visualization of extracted sources

The directory runs/results/+sources/+visuals contains the following files:

<fieldname>.s{w|N}.WWW.fitshapes.bground.fits \leadsto background of sources by subtraction of Gaussian shapes
 <fieldname>.s{w|N}.WWW.fitshapes.res.mean.fits \leadsto mean difference of sources from Gaussian shapes
 <fieldname>.s{w|N}.WWW.fitshapes.res.true.fits \leadsto actual difference of sources from Gaussian shapes
 <fieldname>.s{w|N}.WWW.fitshapes.sources.fits \leadsto elliptical Gaussian shapes fitted to extracted sources
 <fieldname>.s{w|N}.WWW.obs.cb.fits \leadsto background of sources obtained by interpolation under footprints
 <fieldname>.s{w|N}.WWW.obs.bs.fits \leadsto sources obtained by subtraction of interpolated background
 <fieldname>.s{w|N}.WWW.obs.cb.smo.fits \leadsto interpolated background smoothed to the lowest resolution
 <fieldname>.s{w|N}.WWW.obs.sources.n.cb.fits \leadsto interpolated background in regularized component of sources
 <fieldname>.s{w|N}.WWW.obs.fits \leadsto original observed image
 <fieldname>.s{w|N}.sources.ell.WWW.obs.fits \leadsto half-maximum ellipses overlaid on observed image
 <fieldname>.s{w|N}.sources.ell.WWW.obs.src.fits \leadsto half-maximum ellipses on component of sources
 <fieldname>.s{w|N}.sources.ell.WWW.obs.src.n.fits \leadsto half-maximum ellipses on regularized sources
 <fieldname>.s{w|N}.sources.ell.WWW.obs.fil.fits \leadsto half-maximum ellipses on component of filaments
 <fieldname>.s{w|N}.sources.ell.WWW.obs.fil+ske.global.fits \leadsto half-max ellipses on global skeletons
 <fieldname>.s{w|N}.sources.foo.WWW.obs.fits \leadsto footprint ellipses overlaid on observed image
 <fieldname>.s{w|N}.sources.foo.WWW.obs.src.fits \leadsto footprint ellipses on component of sources
 <fieldname>.s{w|N}.sources.foo.WWW.obs.src.n.fits \leadsto footprint ellipses on regularized sources
 <fieldname>.s{w|N}.sources.foo.WWW.obs.fil.fits \leadsto footprint ellipses on component of filaments
 <fieldname>.s{w|N}.sources.foo.all.fits \leadsto footprint ellipses accumulated over all wavebands

<fieldname>.s{w|N}.WWW.det.foots.fits ~> initial footprints of detected sources
<fieldname>.s{w|N}.WWW.obs.foots.fits ~> final footprints of measured sources

7.5.5 Directory for derived skeletons of filaments

The directory runs/results/+skeletons contains the following files:

<fieldname>.f{w|N}.det.fil.ske.KKKKas.fits ~> smoothed accumulated skeletons on scales around KKKK
<fieldname>.f{w|N}.det.fil.ske.KKKKas.sGG.fits ~> skeletons with significance GG obtained on scale KKKK
<fieldname>.f{w|N}.det.fil.ske.all.fits ~> skeletons accumulated over scales of interest (technical)
<fieldname>.f{w|N}.det.fil.ske.global.m.fits ~> intermediate image for creation of global skeletons
<fieldname>.f{w|N}.det.fil.ske.global.n.fits ~> scale-independent global skeletons (normalized to 1)
<fieldname>.f{w|N}.det.fil.ske.global.fits ~> final (segmented) scale-independent global skeletons

7.5.6 Directory for skeletons, normals, and footprints

The directory runs/results/+filaments/+images/WWW contains the following files:

<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~angles.fits ~> position angles at each skeleton point
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~curvsmo.fits ~> smoothed curvatures at skeleton points
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~curvmea.fits ~> mean curvatures of skeletons
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~curvtot.fits ~> total curvatures of skeletons
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~foots.fits ~> two-sided footprints of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~footsa.fits ~> one-sided footprints of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~footsb.fits ~> one-sided footprints of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~lindens.fits ~> linear densities along crests of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~normals.fits ~> normals to skeletons at each point
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~widths.fits ~> average widths of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~coo.hires.reg ~> high-res skeletons as a ds9 region
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~angles.fits ~> position angles at global skeleton points
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~curvsmo.fits ~> smoothed curvatures at global skeleton points
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~curvmea.fits ~> mean curvatures of global skeletons
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~curvtot.fits ~> total curvatures of global skeletons
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~foots.fits ~> two-sided footprints of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~footsa.fits ~> one-sided footprints of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~footsb.fits ~> one-sided footprints of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~lindens.fits ~> linear densities along crests of filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~normals.fits ~> normals to global skeletons at each point
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~widths.fits ~> average widths of filaments

7.5.7 Directory for catalogs and profiles of filaments

The directory runs/results/+filaments/+profiles/WWW contains the following files:

<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~coo.hires.cat ~> high-res coordinates of skeleton points
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~curva.cat ~> average curvatures of skeletons
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~lindens.cat ~> average linear densities and masses
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~lindens.fi.NNNN.cat ~> linear densities along fil NNNN
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~profiles.fi.NNNN.mea.cat ~> mean profiles for NNNN
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~profiles.fi.NNNN.med.cat ~> median profiles for NNNN
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~profiles.fi.NNNN.sel.cat ~> selected profiles for NNNN
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~selwidth.cat ~> filament widths along selected normals
<fieldname>.f{w|N}.WWW.obs.fil.ske.KKKKas.sGG~widths.cat ~> widths averaged along filaments
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~coo.hires.cat ~> high-res coordinates of global skeleton points
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~curva.cat ~> average curvatures of global skeletons
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~lindens.cat ~> average linear densities and masses
<fieldname>.f{w|N}.WWW.obs.fil.ske.global~lindens.fi.NNNN.cat ~> linear densities along filament NNNN

<fieldname>.f{w|N}.WWW.obs.fil.ske.global~profiles.fi.NNNN.mea.cat ~> mean profiles for filament NNNN
 <fieldname>.f{w|N}.WWW.obs.fil.ske.global~profiles.fi.NNNN.med.cat ~> median profiles for filament NNNN
 <fieldname>.f{w|N}.WWW.obs.fil.ske.global~profiles.fi.NNNN.sel.cat ~> selected profiles for fil NNNN
 <fieldname>.f{w|N}.WWW.obs.fil.ske.global~selwidth.cat ~> widths of filaments along selected normals
 <fieldname>.f{w|N}.WWW.obs.fil.ske.global~widths.cat ~> widths averaged along filaments

7.5.8 Directory for visualization of extracted filaments

The directory `runs/results/+filaments/+visuals` contains the following files:

<fieldname>.f{w|N}.WWW.obs.ske.KKKKas.sGG.fits ~> skeletons obtained at scale KKKK with significance GG
 <fieldname>.f{w|N}.WWW.obs.ske.KKKKas.sGG.thick.fits ~> skeletons (thick) obtained at scale KKKK with GG
 <fieldname>.f{w|N}.WWW.det.filaments+ske.KKKKas.sGG.fits ~> skeletons at KKKK with GG on filaments
 <fieldname>.f{w|N}.WWW.obs.filaments+ske.KKKKas.sGG.fits ~> skeletons at KKKK with GG on filaments
 <fieldname>.f{w|N}.WWW.det.filaments+ske.global.fits ~> global skeletons on detection image of filaments
 <fieldname>.f{w|N}.WWW.obs.filaments+ske.global.fits ~> global skeletons on measurement image of filaments
 <fieldname>.f{w|N}.WWW.obs.filaments.fits ~> measurement image of filaments to overlay global skeletons
 <fieldname>.f{w|N}.WWW.obs.filaments.reg ~> global skeletons in a *ds9* region file to be overlaid

8 User agreement

In what follows, *this software* refers to the scripts *getsf*, *comfun*, *compile*, *convolve*, *hires*, *installg*, *iospeed*, *pre-pobs*, *resample*, and the associated utilities *cleanbg*, *ellipse*, *expanda*, *extractx*, *fitconv*, *finalcat*, *fitfluxes*, *fmeasure*, *imgstat*, *makecube*, *modfits*, *operate*, *readhead*, *sfinder*, *smeasure*, *splitcube*, and the library *tools*; and *the author* refers to Alexander Men’shchikov of the Département d’Astrophysique, IRFU, CEA Saclay, France.

It is assumed that anyone using this software (*the user*) has read, understood, and agreed to the following conditions of use:

1. This software is freely available for download and use. A user is free to share this software with co-workers and students, but when it is requested by a colleague not working directly with the user, the latter is asked to redirect that person to the *getsf* web site: <http://irfu.cea.fr/Pisp/alexander.menshchikov/> or to the author: alexander.menshchikov@cea.fr
2. This software shall be used exclusively for research, education, non-profit, and non-military purposes. Specific written permission from the author must be obtained before any commercial use of this software may be undertaken.
3. The banners of this software, these conditions of use, and information about the author shall remain with this software and any descendent developed from and still based substantially upon this software.
4. The name of the institution with which the author is affiliated shall not be used to publicize any data and/or results generated by this software. All findings and their interpretation are the opinions of the user and do not necessarily reflect those of the author nor the institutions with which the author is affiliated.
5. The author makes no representations about the suitability of this software for any purpose. Subject to the above conditions, this software is provided *as is* without any expressed or implied warranty.

9 Citations

The algorithms used in this software are described in: Men’shchikov (2021b, 2017, 2016, 2013); Palmeirim et al. (2013); Men’shchikov et al. (2012, 2010). Detailed benchmarking of *getsf* and comparisons with its predecessor *getsources* (Men’shchikov et al. 2012) is published by Men’shchikov (2021a). Any publication reporting results obtained using this software or any of its derivatives should include a citation of Men’shchikov (2021b) or a phrase like this: “Use of *getsf*, developed by A. Men’shchikov at the DAp, IRFU, CEA Saclay, France, is hereby acknowledged” (if only individual scripts or utilities were used, then *getsf* may be substituted with their names).

References

Abrahamoff, M. D., Magalhães, P. J., & Ram, S. J. 2004, *Biophotonics International*, 11, 36

- Bertin, E., Mellier, Y., Radovich, M., et al. 2002, in *Astronomical Society of the Pacific Conference Series*, Vol. 281, *Astronomical Data Analysis Software and Systems XI*, ed. D. A. Bohlender, D. Durand, & T. H. Handley, 228
- Men'shchikov, A. 2013, *A&A*, 560, A63
- Men'shchikov, A. 2016, *A&A*, 593, A71
- Men'shchikov, A. 2017, *A&A*, 607, A64
- Men'shchikov, A. 2021a, *A&A*, 654, A78
- Men'shchikov, A. 2021b, *A&A*, 649, A89
- Men'shchikov, A., André, P., Didelon, P., et al. 2010, *A&A*, 518, L103+
- Men'shchikov, A., André, P., Didelon, P., et al. 2012, *A&A*, 542, A81
- Mink, D. J. 2002, in *Astronomical Society of the Pacific Conference Series*, Vol. 281, *Astronomical Data Analysis Software and Systems XI*, ed. D. A. Bohlender, D. Durand, & T. H. Handley, 169–+
- Palmeirim, P., André, P., Kirk, J., et al. 2013, *A&A*, 550, A38
- Pence, W. 1999, in *Astronomical Society of the Pacific Conference Series*, Vol. 172, *Astronomical Data Analysis Software and Systems VIII*, ed. D. M. Mehringer, R. L. Plante, & D. A. Roberts, 487–+