Working with the JeAn - Jet Analyzer package (v.3.1.3)

Alexey G. Stupishin (agstup@yandex.ru)
Tatyana I. Kaltman
Sergey A. Anfinogentov

2020-2023, St.Petersburg, Russia

Getting Started

Requirements:

- IDL (project developed and tested under IDL version 8.2 (Windows) and version 8.5.1 (Linux))
- Set of SolarSoft packages (SSW IDL), including *SSW/gen*, *SSW/sdo*, *SSW/vobs* (*SSW/vobs/gen*, *SSW/vobs/ontology*)
- JeAn – Jet Analyzer package on GitHub – main package
- AS-IDL-Library package on GitHub – additional utilities used by main package

Be sure that all packages are available in IDL Paths.

IMPORTANT! IDL should be started by the command *sswidl.bat* from the root of the SSW package.

Call and Parameters

Main entry point is *pipeline_aia.pro* function. There are 2 required parameters:

a. *config_file*: path to the configuration file with input parameters in json format (see Configuration file section). Example *config_sample.json* can be found in the root package directory.
b. *work_dir*: directory where the results of the work will be saved. Result subdirectory structure and stored files explained in Directories and Files Created section.

Optional parameters can be useful for some specific cases:

a. *no_visual* (integer): flag to ignore creating joint images of intensity and running difference and the creation of video files (default: not set).
b. *no_cand* (integer): flag to ignore searching jet-like candidates (default: not set). Note: use both keys */no_visual, /no_cand* to download fits only.
c. *no_details* (integer): flag to ignore creating individual details movies (default: not set). Note: this flag have meaning only if jet-like candidates were required (key */no_cand* is not set).
d. *no_vis_hmi* (integer): flag to ignore creating HMI images, if requested (like *no_visual* key) (default: not set).
e. *presets_file* (string): path to presets file (see this document, Algorithm and Presets File).
f. *fps* (integer): frames per second for movies (default = 5).
g. *ref_images* (integer): flag to save images for full disk for the beginning, middle and end of event.

Algorithm for jet-like event searching is described in Algorithm and Presets File section and in Appendix A.

Configuration File

Configuration file fields:

- "TIME_START": beginning time of the analyzed sequence,
- "TIME_STOP": end time,
- "X_CENTER": the position of the center of the cutout area in longitude (arcsec),
- "Y_CENTER": the position of the center of the cutout area in latitude (arcsec),
- "WIDTH_ARC": the longitude size of the cutout area (arcsec),
- "HEIGHT_ARC": the latitude size of the cutout area (arcsec),
- "WAVES": An array of SDO/AIA wavelengths (Å).
- "MAG": An array of HMI magnetogram cadences (45, 720) (s).

See also *config_sample.json* example:

```
{
	"TIME_START":"2016-04-28 09:35:00",
	"TIME_STOP":"2016-04-28 10:25:00",
	"X_CENTER":463.0,
	"Y_CENTER":21.0,
	"WIDTH_ARC":300,
	"HEIGHT_ARC":300,
	"WAVES":[171, 304],
	"MAG":[45, 720]
}
```

Fields "WAVES" and "MAG" can be omitted or be an empty arrays ([]). In such case corresponding

Additional configuration file fields:

- "WIDTH_PIX": the longitude size of the cutout area (pixels),
- "HEIGHT_PIX": the latitude size of the cutout area (pixels).

These fields are kept for backward compatibility with previous versions of JeAn.
Note, that if both WIDTH_PIX and WIDTH_ARC (or HEIGHT_PIX and HEIGHT_ARC) are defined, value of `*_ARC` will be used (and `*_PIX` will be ignored).

Algorithm and Presets File

Algorithm for jet-like events searching is based on the analysis of the difference between time-neighbor images (running difference).

As the first step AIA images are downloaded from the server according the values from configuration file.

Then algorithm performs intensity alignment of the images intensity to provide homogeneous time-series of the images. Then it selects pixels on a running difference that significantly differ from the average level, and forms clusters of such pixels (according to the principle of spatio-temporal proximity and mathematical morphology). Further, some metrics are calculated for each cluster, and clusters are considered as jets (more precisely, jet-like), if they are

- sufficiently large (according to various estimates of the number of pixels);
- sufficiently long in time;
- sufficiently elongated (according to different estimates of length and width);
- sufficiently fast (according to various estimates of the speed of movement);
- sufficiently intense (according to additional estimates of the intensity distribution).

(see Appendix A.4). Cluster that passed the filter enumerated and called ‘event details’.

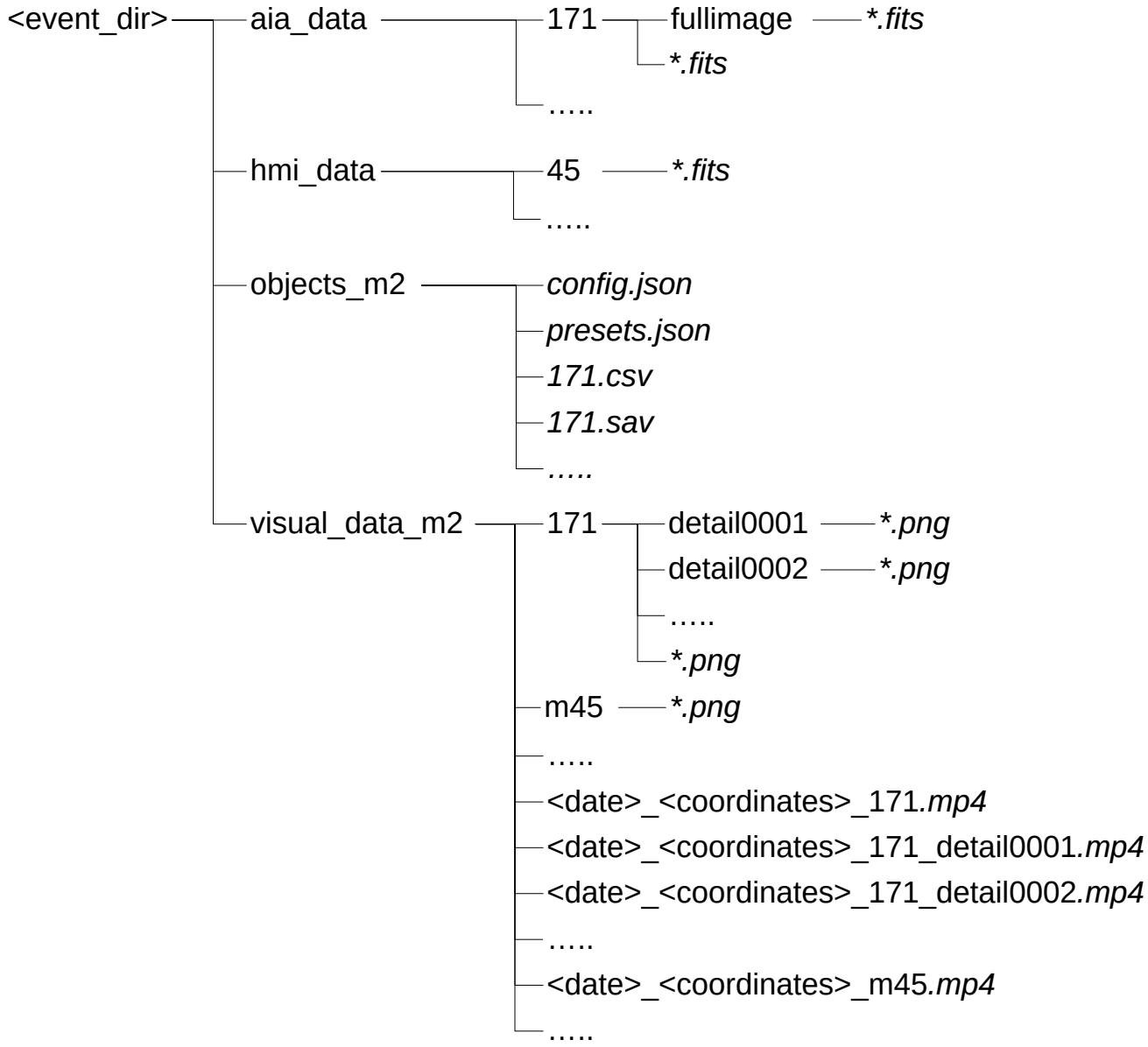
Parameters for such filtering listed in the presets file, sample (*presets_default.json*) can be found in the root of package. Default parameters are set to work with SDO/AIA 171 Å channel. See details in Appendix A.

More detailed explanations[1] can be found in [1].

Directories and Files Created

[1] But note that in this paper previous version of algorithm implementation was described. Actual implementation keeps the main idea, but default values of some parameters was tuned for better detail selection, and clusters are defined not frame-by-frame, but in 3d (coordinates-time) domain.

After execution, the event directory (named by the beginning and end times and the position and size of the area) created in *work_dir*, and the structure of subdirectories:



- *aia_data*: for each requested wavelength, a subdirectory with a sequence of cutout fits (in addition *fullimage* subdirectory can be created, see key */ref_images* description in comments to the function)
- *hmi_data*: for each requested HMI cadence, a subdirectory with a sequence of cutout fits
- *objects_m2*:
  - copies of configuration and presets files
  - for each wavelength, a csv-file *<wavelength>.csv* with general information about each found event detail (the list of parameters can be found in Appendix A.3)
  - for each wavelength, a sav-file *<wavelength>.sav* with detailed information about each found event detail (for further display and detailed analysis)
- *visual_data_m2*:
  - for each requested wavelength, a subdirectory (names by wavelength) with a sequence of generated images for each fits. The pictures are arranged as follows:
    Left pane: Intensity, observation time shown above;
    Right pane: Running difference, the found details are outlined in red, number of details with cardinality (in parentheses) shown on Fig.1.

In addition, if there are found event details, in the same subdirectory subdirectories are created, named by the number of the found detail.

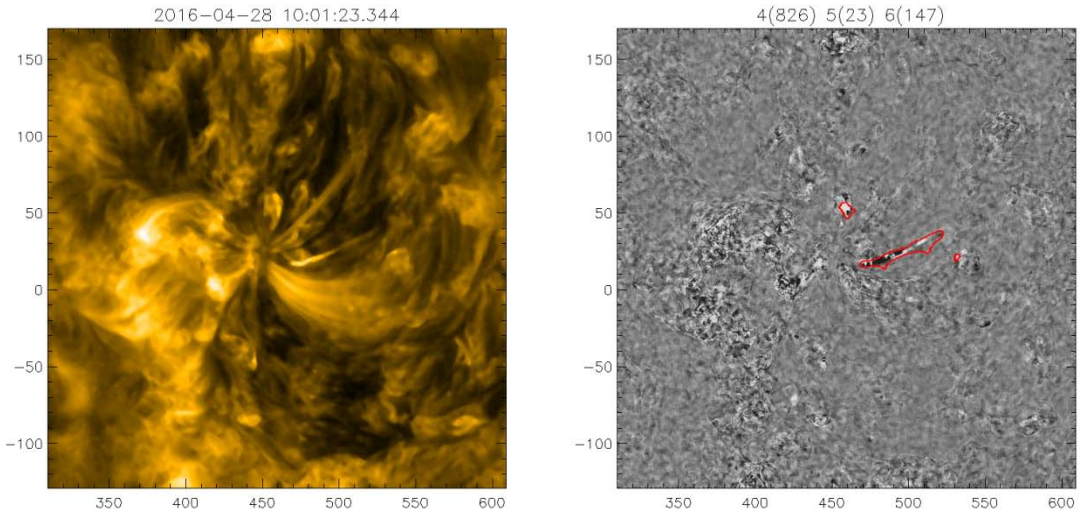


Fig.1. Example of AIA image with running difference and marked jet-like events.

- The most interesting thing: for each wavelength, a video is created with the name *<date>_<coordinates>_<wavelength>.mp4*. In addition, for each detail, a video file is created with the name *<date>_<coordinates>_<wavelength>_detail<number>.mp4*.
- In the same way, for each requested HMI cadence subdirectory with the with a sequence of generated images (see Fig.2) for each fits created, and corresponding video file is created with name *<date>_<coordinates>_m<cadence>.mp4*.

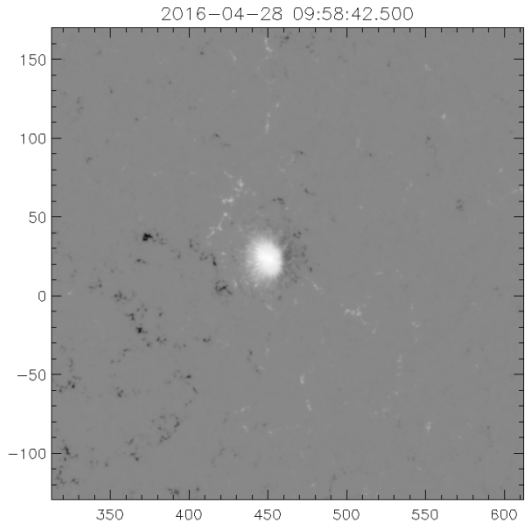


Fig.2. Example of HMI image.

Appendix A

There are 3 subsets of the parameters:

- intensity alignment parameters
- forming clusters (mathematical morphology) parameters
- filtering parameters.

Default values of parameters mentioned below (and marked by *green*) can be found in *presets_default.json*.

A.1. Intensity Alignment

1. Each intensity image in sequence preprocessed with median filter with width = `presets.aia_median` and median value of each image stored (`meds[*]`).
2. Median value of all image medians are calculated (`mmeds`).
3. Limits of accessible median values deviation are defined: `dmax = 1d + presets.median_lim, dmin = 1d/dmax`
4. For i^{th}-image, if `meds[i]/mmeds` is in the appropriate range `[dmin, dmax]`, intensity is normalized to `mmeds` value (`intensity[i] *= mmeds/meds[i]`).
5. Those images which median is out of mention range, considered as "corrupted", and replaced to the image interpolated between two neighbor "non-corrupted" ones.
6. Running differences (RDs) of aligned images are calculated.

A.2. Create Clusters

1. Each `abs(RD)` passed though median filter (with width = `presets.std_median`), and pixels that exceed median level more than `presets.mask_threshold` times marked for the following clusterisation using methods of Mathematical Morphology (MM)[2].
2. Clusterisation performs in two steps:
    a. Removing small stand-alone details by MM opening operation (consequent erosion then dilation MM-operations). Structuring element for this operation is an ellipse in 3D-domain (coordinates – time). Half-width in coordinates dimensions is set by `presets.min_size`, in time dimension by `presets.min_size_t`.
    b. Filling large gaps by MM closing operation (consequent dilation then erosion MM-operations). Structuring element for this operation is an ellipse in 3D-domain (coordinates – time). Half-width in coordinates dimensions is set by `presets.fill_size`, in time dimension by `presets.fill_size_t`.
3. Each set of neighbouring pixels marked as cluster.

A.3. Calculate Cluster Metrics

Before metrics calculation, preliminary principal component analysis (PCA) [3] is performed for each frame ($PCA_f[*]$). Also whole cluster considered in space domain only (i.e. considering all pixels of the cluster in x-y coordinate space ignoring time dimension) (PCA_w).

The metrics are calculated for each cluster (together with columns specification in csv-file *<wavelength>.csv*) are listed in the following table:

[2] Description of the Mathematical Morphology (MM) can be found in a lot of resources (see Wiki for the simple view), more formal explanation can be found in [2].

[3] Corresponding mathematics can be found in a lot of resources, (see Wiki for the simple view), more formal explanation can be found in, e.g., [3].

Metric	Col. #	Col. Title	Description
	1	T start	Start time (YYYY-MM-DD hh:mm:ss.sss)
	2	T max	Time of the frame with maximum cardinality (YYYY-MM-DD hh:mm:ss.sss)
	3	T end	Start time (YYYY-MM-DD hh:mm:ss.sss)
	4	#	Detail number
	5	Duration	Duration (hours-minutes-seconds)
length			number of frames from start to end
total_card	6	Total. card.	total cardinality (number of the pixels in the cluster)
max_card	7	Max. card.	maximal cardinality (number of pixels in the frame with maximal number of pixels);
total_asp	8	Jet asp. ratio	aspect ratio (AsR) of major semi-axis to minor semi-axis of PCA_w
total_wasp	9	LtoW asp. ratio	*total_lng / av_width*
max_asp	10	Max. asp. ratio	maximum AsR by frames (PCA_f)
max_basp			maximum ratio of bounding rectangle size (oriented along semi-axes) by frames (PCA_f)
max_wasp	11	Max. LtoW asp. ratio	likewise *total_wasp*, but maximum by frames (PCA_f)
total_speed	12	Speed est.	of mass center (MC) from first to last frame (km/s)
max_speed	13	Max speed est.	moving of MC between neighbor frames, maximum by frames (km/s)
av_speed	14	Av speed est.	average moving of MC between neighbor frames (km/s)
med_speed	15	Med speed est.	median moving of MC between neighbor frames (km/s)
from_start_speed	16	Base speed est.	moving of MC between first and current frames, maximum by frames (km/s)
total_lng	17	Total length	maximum distance between pixels along major axe of PCA_w (pixels)
av_width	18	Av. width	average pixel distant from the major axis of PCA_w (pixels)
quartiles	19-21	25%, 50%, 75%	quartiles (25%, 50%, 75%) of pixel distribution by intensity of RD (after median filtering)
	22-25	X from, X to, Y from, Y to	coordinates of bounding rectangle, arcsec

A.4. Cluster Filtering

1. The following criteria should be fulfilled to consider cluster as jet-like candidate:
    a. *total_card* >= presets.*min_area**presets.*area_duration*
    b. *max_card* >= presets.*min_max_card*
    c. *total_card / length* (sec) >= presets.*min_av_card*
    d. *length* >= presets.*min_duration*
    e. *total_asp* >= presets.*min_aspect_3d*
    f. *total_wasp* >= presets.*min_waspect_3d*
    g. *max_asp* >= presets.*min_aspect*
    h. *max_speed* >= presets.*min_max_speed*
    i. *av_speed* >= presets.*min_av_speed*
    j. *quartiles*[**0**] >= presets.*min_q25*
    k. *quartiles*[**2**] >= presets.*min_q75*

References

[1] Stupishin, A., Anfinogentov, S., Kaltman, T. Diagnostics of Parameters of Hot Jets in the Solar Corona in Time Series of Images**.** 2021**,** Geomagnetism and Aeronomy, **61**, Issue 8, p.1108. DOI, ADS

[2] Serra, Jean Paul. Image Analysis and Mathematical Morphology: Theoretical advances. GB, Academic Press, 1982. ISBN: 0-8194-0845-X.

[3] Jolliffe, I.T. Principal Component Analysis. Springer New York, 2002, ISBN: 978-0-387-22440-4, pp., 150-166, DOI