Matthieu Palayret – matthieu.palayret@m4x.org

# The MP.jar plugin (Fiji – June 2021)

## Installation steps:

- Installation of Fiji (with Java 8 or more).
- After its installation, update Fiji by following instructions under "Help>Update…". Close Fiji, re-open it, and re-do this step a couple of times.
- In, Fiji, go in "Help>Update…". In the "ImageJ Updater" window that opens, click on "Manage update sites", and select the following boxes: "CALM", "GDSC", "GDSC-SMLM" (but not the "GDSC-SMLM2"!), and "ResultsToExcel". "ImageJ", "Fiji" and "Java-8" boxes should also be selected. Click on "Close", then "Apply changes", and close Fiji at the end of the update.
- Add the plugin "MP.jar" and the library "Flanagan.jar" in the folder "Fiji.app/plugins".
- Then open Fiji. The different plugins are available in "Plugins>MP>…".

## Plugin description:

- **Plugins>MP, « Analyse cell contours only... », MP.Analyse_Contour**

  The *Analyse_Contour* plugin is a shorter version of the *Analyse-Protrusion* plugin: it only analyses the contour of the cells of the movie that it is fed with (but not their protrusions). It takes as input a movie (if none is open when the plugin is started, the user is asked to open one). NB: the plugin can also manage the analysis of a single frame (*i.e.* not a movie).

  An interactive window allows to choose correct parameters to determine the contours of the cells. In blue are plotted detected and selected cell contours; in dark blue, those which are discarded (because of a too be area). Then cell which contours are overlapping in two consecutive frames are linked in trajectories.

  The final movie is saved in "0-Contours.gif" as a GIF movie (only) (with a rate of 3 frames/s.). Final chosen parameters are saved in "Parameters.csv" ; and all detected contours (or cells) are saved in "1-Contour-analysis.csv":
  (1st column) the frame number;
  (2nd-3rd) the (x, y) position of the center of mass of this cell in that frame;
  (4th) the cell number (the same number than in the GIF movie);
  (5th) the area of this cell in that frame;
  (6th) the perimeter of this cell in that frame.

- **Plugins>MP, « Analyse cell contours and protrusions… », MP.Analyse_Protrusions**
  The *Analyse_Protrus*ion plugin, based on the *net.calm.adapt.Adapt* plugin by David Barry, analyses the protrusions of multiple cells in a ".tif" movie. It takes as input a movie (if none is open when the plugin is started, the user is asked to open one). NB: the plugin can also manage the analysis of a single frame (*i.e.* not a movie).

  First, an interactive window allows to choose correct parameters to determine the contours of the cells. In blue are plotted detected and selected cell contours; in dark blue, those which are discarded (because of a too big area). Then cell which contours are overlapping in two consecutive frames are linked in trajectories.

  Then, a second interactive window offers to perform two different actions:

(1) Choose a set of parameters to correctly detect the protrusions of the cells:

(a) the minimal change of curvature in the contour of the cell for a protrusion to be detected (in °);

(b) the maximal surface ratio between the protrusion and the corresponding cell (in % - the protrusion is supposed to be only a small portion of the cell);

(c) a smoothing factor (to take into account only average changes of curvature in the contour of the cell, and not tiny accidental ones);

(d) whether to try to detect the uropod of the cell (corresponding to the protrusion of the cell in the opposite direction from the direction the cell is moving towards, if at least one protrusion is detected).

(2a) Choose some parameters to refine the selection of the cell trajectories:

(a) the minimal length of a trajectory (in s. – if trajectories are shorter, they are discarded);

(b) the dramatic cell area in-/de-crease factor (in % - if a massive increase or decrease is observed in the surface of a cell during its trajectory, it is probably meeting or separating from another cell ; the plugin thus splits the trajectory in two around that massive change).

(2b) Interactively rejecting some cells or trajectories by selecting the correct button ("No rejection" ; "Reject cell in a single frame?"; "Reject a whole cell trajectory?") and directly clicking on the corresponding cell / trajectory on the movie. All initially detected trajectories are plotted, with colours depending on their status:

Dark red: cell rejected in a single frame because of area in-/de-crease;

Dark blue: cell rejected in a single frame;

Dark green: trajectory length shorter than the defined parameter;

Dark purple: whole cell rejected;

[not used except for unknown reason] Dark yellow: cell rejected in a single frame because of 'technical' issue.

NB: No uropod is detected in the 1st frame of a trajectory.

Finally, the final movie is saved in "stack-ini.tif" and "stack-RoiSet.zip" (in order to be easy re-open thanks to the *Open_MP* plugin), and as a GIF movie in "stack.gif" (with a rate of 3 frames/s.).

Final chosen parameters are saved in "Parameters.csv"; a list of the trajectories of all the cells in "0-Trajectories.csv" (with for each cell and each frame, the frame number, the cell number, the status of the trajectory -rejected or not-, and the (x, y) position of the center of mass of this cell in that frame) ; the contours of all protrusions of all cells in "1-Protrusion_contours.csv" ; the trajectory of all detected protrusions (in terms of their centres of mass) in "1-Protrusion_center_of_mass_positions.csv".

Results from additional analyses are saved:

- In "2-Results_per_frame.csv", for each cell and frame (cf. 1st and 3rd columns), it outputs:

  (4th column) the average distance of the protrusions (but not the uropod, if it is detected) to the leading edge of the cell (defined as the opposite of the uropod) and (5th) the distance of the closest protrusion to the uropod (both distances are divided by the perimeter of the cell and given as %);

  (6th) a measure of the circularity of the cell (the ratio of the real area of the cell over the area of a circular cell which would have the same perimeter as the cell);

  (7th) the number of protrusions (but not the uropod, if it is detected) in the cell in the considered frame;

(8th) the average size of the protrusions (but not the uropod, if it is detected) in the cell in the considered frame;
(9th) the area of the uropod of this cell in that frame, if it is detected;
(10th) the area of that cell in that frame.

- In "2-Results.csv", for each cell trajectory (cf. 1st column), it outputs:
(3rd column) the linearity of the trajectory (the trajectory is perfectly linear for a result of 100%);
(4th) the real distance the cell travelled;
(5th) the corresponding average speed of the cell (averaged over the number of frames);
(6th) the global distance the cell travelled (distance between its last and first positions);
(7th) the corresponding global average speed of the cell;
(8th-14th) the average (4th-10th) values of "2-Results_per_frame.csv" averaged over the whole cell trajectory;
(15th) the temporal duration of the trajectory.

- **Plugins>MP, « Analyse 2-colour cell protrusions... », MP.Two_Colour_Analysis**
The *Two_Colour_Analysis* plugin is a version of the *Analyse_Protrusion* plugin adapted for a two-channel movie with two populations of cells to compare: some cells only emit in one channel (they are called the "green-only" or "green" cells) and others emit in both channels (they are called the "red-and-green" or "red" cells).

  Basically, both channels are first analysed independently to detect cells and their contours (in an interactive way). At the same time, the plugin matches cells that emit in both channels, to detect the two populations of cells (the "green-only" and the "red-and-green" cells).

  The cell-trajectory analyses and the protrusion analyses are then performed, as in the *Analyse_Protrusion* plugin, and the outputs are given:
(1) without distinction between the two population of cells;
(2) for the "green-only" cells only (with the "_green.csv" suffix);
and (3) for the "red-and-green" cells only (with the "_red.csv" suffix).

- **Plugins>MP, « Plot 2-colour cell protrusion results... », MP.Open_MP**
This plugin asks for a ".tif" file that has been previously analysed with the *Two_Colour_Analysis* plugin. It opens the ".tif" file and re-draws on it all the detected and analysed contours.

  It does not save any new file. It requires a "stack-RoiSet.zip" to be present in the same folder as the ".tif" file.

- **Plugins>MP, « Combine protrusion results... », MP.Combine_Protrusion_Results**
The *Combine_Protrusion_Results* plugin allows to concatenate results of various *Analyse_Protrusion* and/or *Two_Colour_Analysis* analysis folders. It asks for folders within which are saved either "2-Results.csv", "2-Results_green.csv", and/or "2-Results_red.csv" files. Folders with no such files are ignored.

  For each analysis (or folder), one can decide to combine only the "green" results, or the "red" ones or all of them.

  The plugin outputs **in the first analysed folder** a "2-Combined_Results.csv" file containing the concatenated results.

- **Plugins>MP, « Plot cell trajectories... », MP.Plot_Trajectories**
  This plugin plots the trajectories (of cells or dots within the cells) from pre-processed data. It accepts both
  (1) Imaris-analysed data in a ".xls" format with a sheet called "Position" (with a 1st column called "Position X", a 2nd column "Position Y", a 7th column "Time" and an 8th column "TrackID"),
  and (2) data analysed with one of the MP plugins (*e.g. Get_Trajectories*, or *Analyse_Protrusion*) (".csv" or ".txt" files).

  Several datasets (from potentially both different types of analyses) may be plotted in the same round.

  If a folder (and not a file) is fed in the dialogue box, the plugin will search for a file called "0-Trajectories.csv" in the folder (or it will ignore this folder).

  If no movie is open when the plugin is started, all the trajectories are plotted in a new white window.

  If the original ".tif" file - which was used to detect the detected trajectories which data is fed to the plugin - is open before starting the plugin, the trajectories are plotted over this open movie (the trajectories from the first dataset are coloured blue, then those of the following dataset red, then green, cyan, magenta, orange, and pink).

- **Plugins>MP, « Detect and analyse moving particles in a .tif movie... », MP.Get_Trajectories**
  The *Get_Trajectories* detects and analyses moving particles from a ".tif" movie. The ".tif" movie to analyse needs to be open when the plugin is started. One is then asked to provide a folder for saving the outputs.

  The movie first goes through 3 consecutive filters:
  (1) Fast Fourier-Transform bandpass filter to remove all features < 1 pix and > 10 pix in spatial frequency;
  (2) a temporal walking average filter over 3 frames to smooth the movement of the particles and to avoid intermittent trajectories;
  (3) a contrast adjustment (saturation of 0.35).

  Then the movie is analysed through the *Peakfit* (gdcs_smlm) routine which detects and fit all possible particles. It outputs these first results in "TableFit_PeakFit.txt".

  Results are then interactively filtered (using their width and intensity as parameters), and temporally linked to build trajectories (with the maximal distance between two consecutive fits and the maximal number of frames when the tracked particle is allowed not to be detected as parameters). Filtered results and trajectories are plotted in real time. The chosen parameters are saved in "parameters_AlignTrajectories.txt", the filtered fits in "TableFit_Sorted.txt", and the sorted trajectories in "Table_Trajectories.txt".

  Then, follow a few different analyses :
  (1a) An estimation of the "average trajectory of the cell" (ATC) is obtained by calculating the average displacement of particles which are tracked in consecutive frames. This ATC is saved in "AvgTrajectory.txt" and plotted in black in the "AvgTrajectory.tiff" movie. In

the same movie, each trajectory is also plotted starting from the ATC in the frame it first appears. Each trajectory is colour-coded depending on its relative proximity compared to the ATC (green if following it, red if going in the opposite direction).

(1b) For each trajectory, its angle between its average direction and the average direction of the ATC during the same frames is calculated, and plotted as a radius on a disk (0° = the trajectory is heading in the same direction of the cell; 180° = the trajectory is going in the opposite direction from where the cell is heading). The colour of the radius depends on the local radial density (red for high density of radii within +/- 5°; green for low density). The plot is saved in "AvgAngles.tiff".

(2) For each trajectory, the plugin then calculates
(a) its average speed,
(b) its global speed (distance between last and first position / number of frames),
(c) its directionality (1 if the trajectory is linear; close to 0 if it moves around a fixed position),
(d) its full length (along its path),
(e) its global length (distance between last and first positions),
(f) its total duration ("Time length"),
(g) its average intensity (in arbitrary units),
(h) its average width (standard-deviation of the fitted 2D-Gaussians),
(i) its average area (area of the 2D fitted Gaussians),
(j) the standard deviation of its distance to the "average cell trajectory" (ATC),
and (k) its average angle with the "average cell trajectory" (ATC) (in rad.).
This data is saved in "Track_Analysis.csv".

(3) A pairwise distance analysis: for each couple of trajectories sharing two consecutive frames, the plugin plots their average relative speed against their (initial) distance.
(i) Centripetal attraction should result in negative speeds;
(ii) Centrifugal repulsion should result in positive speeds;
(iii) Randomness should result in speeds close to 0.
The reasoning behind this analysis is to try to detect two populations of particles which would behave differently: at the front of the cell, a flow moving upfront in the direction of the cell; at the rear of the cell, a retroactive flow moving towards the center of the cell. The plot is saved in "Pairwise_Analysis.tiff" and its data in "Pairwise_Analysis.csv".

(4) Finally, the data is analysed through the *Analyse_Trajectories* plugin, in order to perform both an MSD (Mean-Square Displacement) and a JD (Jump-Distance) diffusion analyses. See this plugin for further explanations.
NB: it is then wrongly assumed that 1 frame = 0.1 s and 1 pix = 0.1 µm. The final results from this last step requires to retro-convert the units.

- **Plugins>MP, « Analyse trajectories from an MP-fitted dataset [deprec.]... », MP.Complete_Trajectories**
The *Complete_Trajectories* plugin should not be used anymore. It extract trajectories from a pre-processed result file from the *Get_Trajectories* plugin (a "Table_Trajectories.txt" file), and re-analyse them.

It processes the same analyses (1a), (1b), (2) and (3) (but not (4)) as the *Get_Trajectories* plugin (see the corresponding explanation).

It may be used to re-analyse some fitted data whose analyses were lost or perverted, without redoing the relatively long process of fitting and tracking the particles.

- **Plugins>MP, « Analyse trajectories from an Imaris-fitted dataset... », MP.Analyse_Trajectories**
  The *Analyse_Trajectories* takes a ".xls" Imaris file of trajectories of particles and analyse their speed and behaviours. It performs three analyses of the trajectories:

  (1) It calculates and plots their step distribution histogram and fits it to a log-normal distribution (which is the theoretical curve followed by freely moving particles), which plot is output as "4-StepSizeHistogram.tif" in the folder of the ".xls" input file.

  (2) An MSD (Mean-Square Displacement) analysis to get an estimation of the speed of each trajectory and to measure whether it is freely diffusing or following a directed movement. Only trajectories of length > 2 are considered. A polynomial of degree 2 is fitted to the MSD curve using up to the 15 first distance intervals.

  Different motions theoretically follow three main different MSD curves:
  - Directed movement: MSD = 4D * DeltaT + (V * DeltaT)^2
  - Constrained movement : MSD = 4D * DeltaT^alpha (alpha < 1)
  - Freely diffusive movement: MSD = 4D * DeltaT

  The following outputs are saved in the folder of the ".xls" input file :
  - The MSD plot and the fitted polynomial: "5-RebuildTrajectories_MSD.tif"

  - The histogram distribution of the diffusion coefficient D (in µm^2/s) obtained (by correct fitting - for which the diffusion coefficient is >0 and the R^2 value > 0.6) for all the trajectories: "6-MSD_Diffusion_Histogram.tif" (and its corresponding "-Data.txt" and "-Hist.txt" files).

  - The histogram distribution of the average speed coefficient V (in the case of a directed movement - in µm/s) obtained (in the same conditions) for all the trajectories: "6-MSD_Speed_Histogram.tif" (and its corresponding "-Data.txt" and "-Hist.txt" files).

  (3) A Jump-Distance analysis (equivalent to the MSD analysis, but on the whole population of trajectories, not for each single trajectory) for up to the 11 first step intervals. Three hypotheses are tested by fitting the results with a corresponding theoretical distribution: the trajectories are composed of 1, 2 or 3 populations of diffusing particles. Results are saved in a subfolder "/PopulationSD". Please ask matthieu.palayret@m4x.org if you need more information about these results.

- **Plugins>MP, « Plot trajectories over the initial .tif movie... », MP.Plot_Trajectories**
  See above.

- **Plugins>MP, « Align several cell trajectories from Imaris datasets... », MP.Align_Trajectories**
  The *Align_Trajectories* plugin further analyses outputs from Imaris: it aligns trajectories of various cells on a single *(Ox)* axis, and the behaviour of the movement of their detected particles, depending on whether these are at the front (ahead of the center of mass of the cell, on the *(Ox)* axis) or at the back of the cell.

It thus takes two consecutive inputs: (1) an ".xls" Imaris-derived file with the **cell** tracks, and (2) an ".xls" Imaris-derived file for the **particle** tracks.

It then globally re-align each cell (and their corresponding particle tracks) along a common *(Ox)* axis (defined as the line between the first and last positions of the cell track - as we here suppose the cell trajectory to be roughly linear). The tracks of all the cells (and of their particles - with a twice thinner line) are plotted together in two different plots:

(1) In a first plot (which is not automatically saved), the colour of the trajectories of the particles gets darker with time (to indicate their direction over time).
(2) In the second plot (saved in "Trajectory_plot.tiff"), trajectories are plotted following a colour-coded time-gradient depending on whether they are behind the center of mass of the cell (gradient from yellow to green) or before it (gradient from orange to red).

Finally, trajectories are analysed in a similar way as in the *Get_Trajectories* plugin: for each trajectory of each cell, the plugin calculates and saves :

(1st column) the name of the cell this trajectory belongs to,
(2nd) the name of the trajectory,
(3rd) its average speed,
(4th) its average speed along the *(Ox)* axis,
(5th) its global speed (distance between last and first position / number of frames),
(6th) its directionality (1 if the trajectory is linear; close to 0 if it moves around a fixed position),
(7th) its full length (along its path),
(8th) its global length (distance between last and first positions),
(9th) its total duration ("Time length"),
(10th) whether, when the particle if first detected, it appears before the centre of mass of the cell (on the *(Ox)* axis) ("rear" particle) or after it ("upfront" particle).

This data is saved in "Track_Analysis.csv".

Then, a sum-up of these results is calculated and saved separately for all "rear" particles (saved in "Track_rear_statistics.csv") and for all "upfront" particles (saved in "Track_upfront_statistics.csv"). For each of the 3rd to 9th column (as defined above), both an average value and a standard deviation of all the particles of interest are calculated and saved.

- **Plugins>MP, « Combine Imaris .xls results... », MP.Combine_Excel_Results**
  The *Combine_Excel_Results* plugin allow to concatenate results of various Imaris analysis Excel files.

  The plugin outputs **in the folder of the first analysed file** a "CombinedResults.xls" Excel file containing the concatenated results (sheet by sheet).

  The plugin consistently adds an "Origin" column in all Excel sheet to identify the file name from which the result comes from.
  Moreover, if an Excel sheet called "Track Speed Mean" exists, the plugin adds to it a column "AColumnx60" which multiplies the first column by 60.

- **Plugins>MP, « Separately combine .xls and/or .csv results... », MP.Combine_Results**
  The *Combine_Results* plugin allow to separately concatenate results of
  (1) various Imaris analysis Excel files (using *the Combine_Excel_Results* plugin)

and (2) various results ".csv" files from MP plugins, in a single go.

The plugin can be fed with folders (and not only files). It will then scan the folders for all ".csv" and ".xls" files, and consider them for concatenation.

The plugin outputs **in the folder of the first analysed file (or folder)** a "CombinedResults.xls" Excel file and/or a "CombinedResults.csv" file (as relevant) separately containing the concatenated results.

The plugin consistently adds an "Origin" column in all Excel sheet to identify the file name from which the result comes from.
Moreover, if an Excel sheet called "Track Speed Mean" exists, the plugin adds to it a column "AColumnx60" which multiplies the first column by 60.