This document aims to describe the Matlab code MainGeneProfile.m and its subfunctions and guide the reader through the user interface. The original codes were developed for the analysis (Thouvenin et al., eLife, 2020) and the user interface was developed for the associated bioprotocol(Thouvenin et al., Bioprotocol, 2020). The major objective of this code is to compute a flow profile from timelapse 2D bead measurements, and we implemented a feature to fit the flow profile to the bidirectional cerebrospinal fluid (CSF) flow model we have developed in (Thouvenin et al., eLife, 2020).

A first description of the analysis workflow is provided. Then, a walkthrough of the user interface is provided as well as a detailed description of the functions called by the interface.

*Specifically for this protocol, we developed a user-interface platform to allow users to generate CSF flow profiles as easily as possible. Here, we present the analysis workflow (Figure 1) and how to generate a first CSF flow profile from the fluorescent beads measurements. More subtle fine tuning of parameters is available within the user interface to adapt to variable imaging conditions, which is fully described here. As input, the analysis takes 2D time lapses of beads flowing in the central canal (Figure 1A1). For a given dorsoventral position, we swap the axes so that the X axis corresponds to the rostrocaudal position and the Y axis to time (kymographs) and then, the beads trajectories appear as lines whose slopes reflects the direction and speed of the particules along the rostrocaudal axis. (Figure 1B1). In order to build the flow profile, the program filters each kymograph and performs automatic segmentation of all lines in each kymograph (Figure 1B2). It then extracts the slope of each line, and converts it into the particle velocity, in order to build a histogram of velocities for each dorso-ventral position (Figure 1B3). By calculating the average velocity at each position, we generate the CSF flow profile (Figure 1C1).*

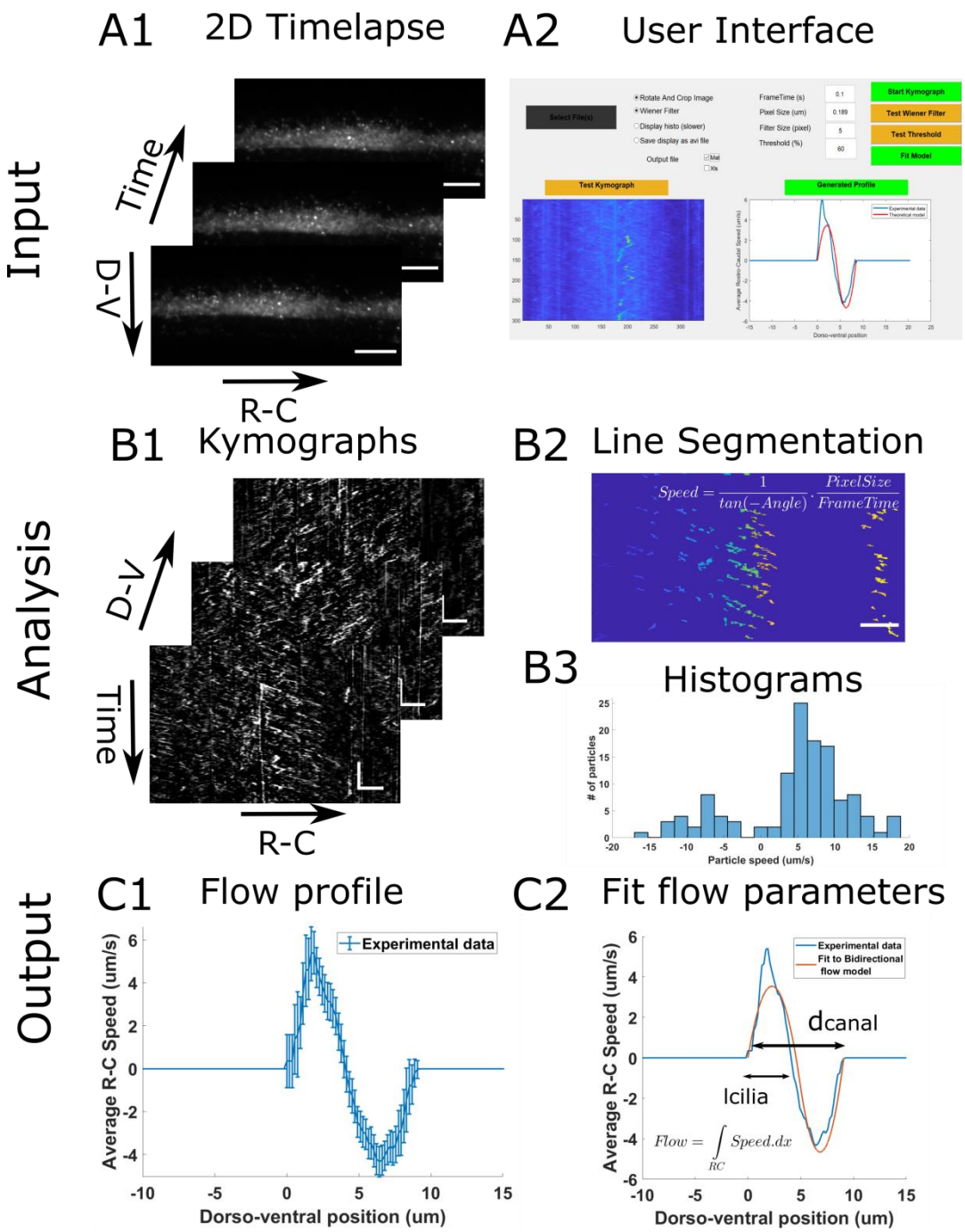



**Figure 1. Principle of the CSF flow profile analysis workflow.** The CSF flow profiles are calculated using 2D time lapses of fluorescent beads flowing in the central canal (A1), and our custom user interface software (A2). The Start Kymograph button starts the analysis by first calculating kymographs from 2D time lapses by swapping dimensions (B1). Each kymograph is filtered, and all lines, corresponding to one particle trajectory, are segmented (B2). The angle of each line is then transformed to a particle velocity value, and aggregated into a speed histogram for each D-V position (B3). The flow profile is then calculated by extracting the mean and standard error of each histogram (C1). The flow profile can finally be fitted to our bidirectional flow model, to extract quantitative flow parameters (C2). Horizontal scale bars are 15 µm and vertical scale bar is 5 s.

**User interface walktrough:**

1. Launch program
   a. Either run Main_GeneProfile.m (requires MATLAB 2018b or later).
   b. Alternatively, download and install the standalone application. Once it is installed, go to the command window and navigate to the installed folder. Run: application\GeneProfile.
2. The user interface window in Figure 2 opens. The user interface is controlled by the function GUI.m, and can be modified either directly in the code (for experts) or by typing 'guide' in the Matlab command window (still not so much for beginners).

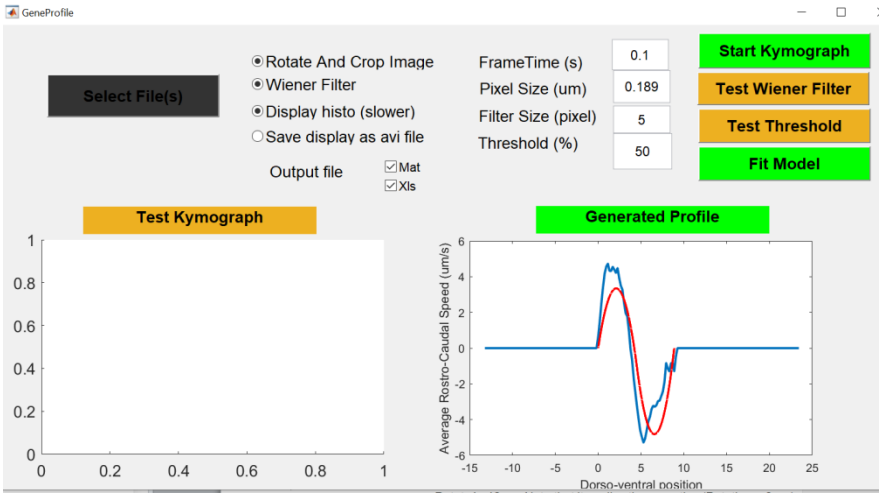



**Figure 2. Gene Profile User Interface.** This interface allows selecting the files and diverse options, tuning parameters, and visualizing filtered images and calculated profiles.

3. ***Select File(s)*** button: When clicking this button, a new window opens (Figure 3) and asks to select the .tif file(s) corresponding to the timelapse 2D beads acquisition. At this stage, Matlab only registers the path to the files and did not load the data in the workspace yet. Multiple selection is authorized, in which case all files will be

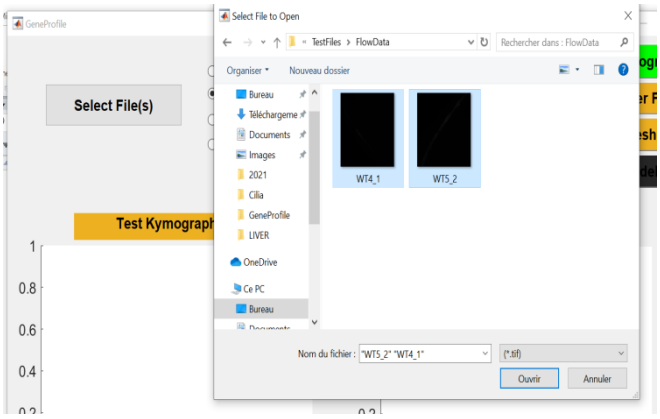



**Figure 3. Select File button description.** A new window pops up and files can be manually selected.

analyzed sequentially for the rest of the analysis. After the files are selected, the button turns to dark grey (see Figure 2), but this doesn't prevent to click again on the button to choose new files (in this case however, the first selection will be erased. When clicking on the button, it calls the following Matlab function:

***[Files,Path]=uigetfile('.tif','MultiSelect','on');***

4. Choose the different options by clicking the associated radio buttons. Here again, the program does not compute anything, but simply saves binary variables (0 if the option is not selected, 1 otherwise). The detailed description of all options will be done later in this manual.

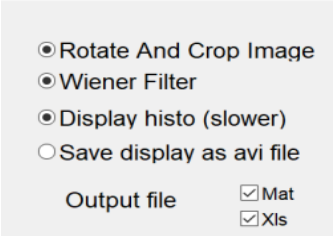



**Figure 4. Select options.** By a simple click on the buttons

   a. RotateAndCrop (Variable: ***Rotate***). If selected, each analysis will start by calling the function RotateAndCrop.m (in the *subfunctions* folder), which performs semi-automatic rotation and crop of the .tif files. For the analysis to work, it is essential that the central canal is as horizontal and as flat as possible. This rotation and crop can be done within the user interface (by clicking the Rotate And Crop Image button) or manually directly in the .tif files (in which case, we advise not to click the button). Because the rotation and crop steps are asked everytime an analysis is performed in the user interface, it can become quite time consuming. The RotateAndCrop.m function can be used independently of the user interface to modify the .tif files in a first step (in which case, don't forget to save the new modified .tif files).

   b. WienerFilter (Variable: ***Wiener***). If selected, each analysis will start by applying a Wiener filter (of size *FilterSize* controllable in the main interface (Figure 2)) to the raw data. The wiener filter enables to increase the beads signal-to-noise ratio (SNR). It applies the following code :

   c. Display Histo (Variable: ***Display***): If selected, an additional visualization window (See later -Figure N) will be displayed when the kymograph is generated (See later). This feature is mostly there to illustrate the different steps of the algorithm, and to verify that the computation works well. Because it displays an image for

each dorso-ventral position, it slows down the flow profile calculation, and should better be off when many profiles are processed simultaneously.

d. SaveAsAvi (Variable: ***Avi***): If selected, and (only) if Display Histo is selected as well, it saves the visualization window as an .avi file (so that it can be controlled afterwards).

e. Output file (.mat /.xls – Variable: Output= 1-.mat/2-.xls/3 -both). If selected, the data corresponding to the flow profile plot are saved, either in .mat, or in .xlsx. The structure of the saved data will be explained in more details later.

5. Define the different parameters by changing the value in the white boxes. The frame time and pixel size depends on the imaging parameters that can vary from one microscope to the other. The wiener and threshold parameters are imaging processing parameters that may be tuned (see later in the manual) to optimize particle detection and will depend on the image SNR which can also vary from one group to the other.

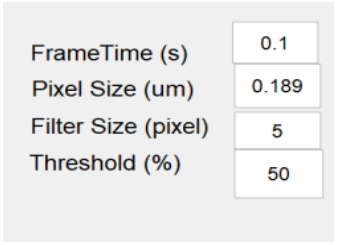



**Figure 5.** Definition of the experimental and imaging processing parameters.

   a. FrameTime (in seconds): defined by the experimental parameters. Inverse of the acquisition frequency.
   b. Pixel size (in µm): defined by the experimental parameters
   c. Filter Size (in pixels): Defines the size of the Wiener filter
   d. Threshold (in %) : Defines the intensity threshold for the detection of the lines (that correspond to the particle trajectories) in the kymographs.

6. **Rotation and Crop of the images.** Before the kymographs are generated, it is important to choose a flat portion of central canal, because all particle trajectories in one plane are averaged and should correspond to the same dorso-ventral position. Using the user interface, a semi-automatic rotation can be performed with manual fine adjustment. Then, a crop interface allows a manual selection of a straight portion of central canal. The procedure rotation and crop is repeated twice to perform one rough alignement, and a second finer one. The rotation interface (Figure 6-Left) proposes a first guess to align the central canal to the horizontal axis. It is then possible to accept this guess, turn the central canal upside down (180° rotation – the

ventral side of central canal is more straight than the dorsal side), or manually add another angle. In this case, another rotation window opens with the modified angle, which can be fine tuned by successive trials. The rotation interface can be closed by clicking the Error button, but it will issue an error in the Matlab code. Once the angle was set (by clicking the Correct button), the crop interface opens, and a manual rectangle can be drawn to define the optimal region of interest. Make sure that the selection has a few pixels outside the central canal as well (Figure 6).

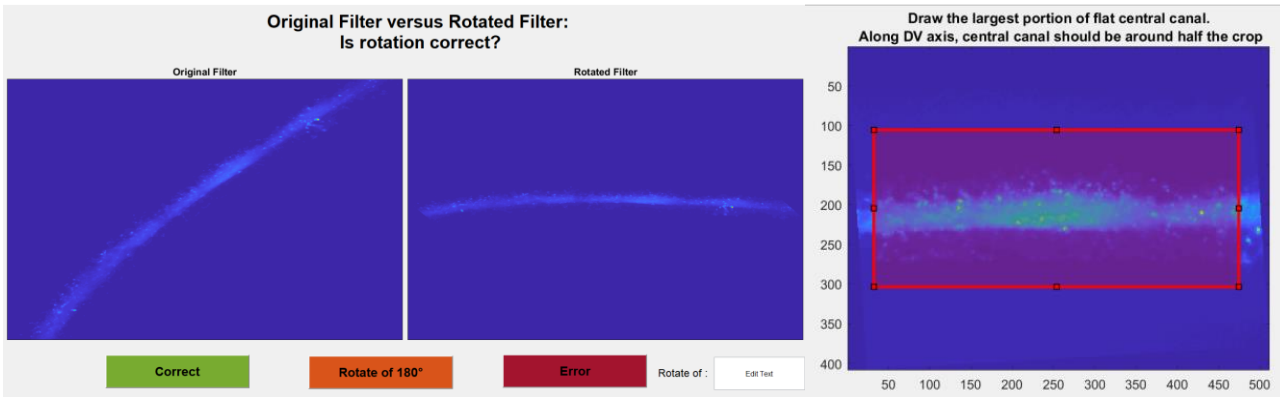
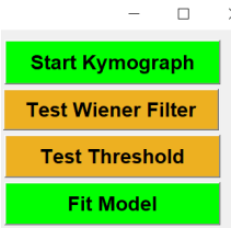



**Figure 6. Rotation and crop interfaces.** The rotation interface (left panel) aims to automatically rotate the image to have the central canal along the horizontal axis and with the ventral side at the bottom side. The angle can be manually tuned to correct for errors. The crop interface (right panel) aims to choose the largest portion of central canal (final region of interest-red box) where the ventral side is as flat as possible.

7. Test the FilterSize and Threshold parameters. If the principle of the kymograph analysis is not clear yet, we advise to go the next section before reading this one. The following tests are not necessary, especially if the same protocol is used for all acquisitions. Nonetheless, the Wiener filter and mainly the threshold value may have to be tuned if new configurations (new beads, different SNR, etc.) are tried. Note that the provided test files have too high SNR to demonstrate the real interest of the Wiener filter, and we had to add random noise to generate figures 7 and 8. Yet, because the additional noise is random, the noise statistics is spatially homogenous, so that it is hard to demonstrate the risks of choosing a too large *FilterSize* value. Nonetheless, in experimental conditions with low SNR (low bead concentration, low laser power, higher framerate, etc..), the Wiener filter can greatly improve the analysis. The wiener filter relies on the estimation of noise surrounding the particles

in a region of size *FilterSizexFilterSize*. The size of the Wiener filter is then important to 1) have enough pixels to correctly estimate the statistics of the noise 2) stay in a region of homogenous noise. In the case of the central canal, it is likely that the noise statistics is different in the central canal than outside. In order to improve the Wiener filtering, a rectangular region of interest (instead of a square one) could be calculated. The threshold parameter corresponds to the intensity threshold to create the binary image that is used to segment the particle trajectories. Briefly, we automatically find the center of the central canal, and calculate the pixel value histogram of the kymograph at this position. We then calculate the intensity value corresponding the *Threshold* percentile (e.g. median for *Threshold* value of 50), and apply a binary threshold (0 for all pixel values below this value- 1 otherwise) to all kymographs. Because the calculated value rely on the plane of maximal intensity, and because several other geometrical filters are also applied, we advise to choose a not too stringent value of *Threshold* (e.g. the default value is 50%). Examples of inadequate *Threshold* values are provided in Figure 9.




a. Test Wiener Filter. By clicking the button '*Test Wiener Filter'*, the program first loads the chosen dataset in the Matlab workspace. If selected, the (rotation + crop) processing is performed twice, and the Wiener Filter of size *FilterSizexFilterSize* is sequentially applied to all images.

```
if Wiener
    for ii=1: NImages
        NewIma(:,:,ii)=wiener2(NewIma(:,:,ii),[FilterSize FilterSize]);
    end
end
```

Then, the center of central canal (from the maximum of fluorescence intensity) is estimated, and the kymograph at this region is displayed in the bottom left axes (see figure 2). Correct value is found when the lines in the kymographs (corresponding to particle movement) show enough contrast. Examples of Wiener filter with respective size of 0, 5, 10, 50 pixels are shown in figure 7 on direct images, and in figure 8 on kymographs. Note that the Wiener Filter option has to be selected; otherwise, only the raw kymograph (without filter) is displayed.

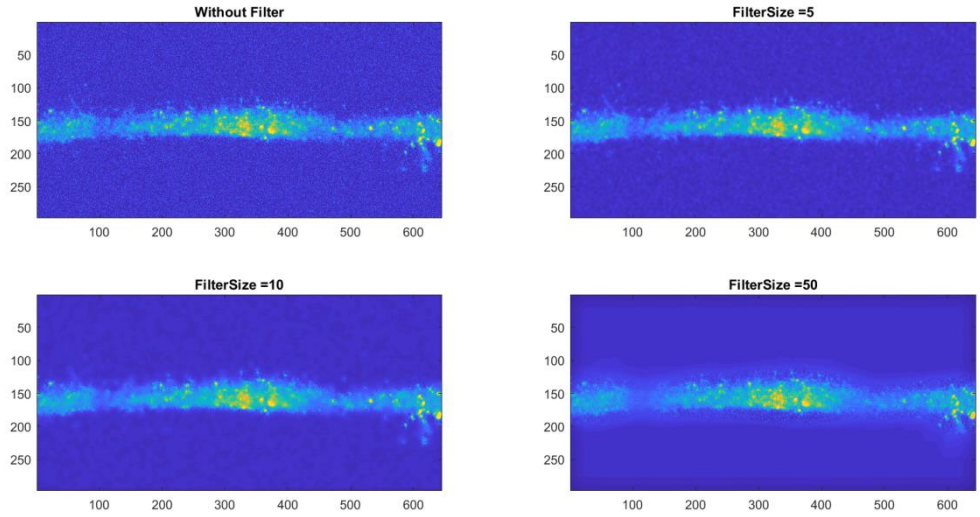



**Figure 7. Wiener filter of different *FilterSize* values on direct images.** Without filter (top left panel), the image has important random noise, which limits the image SNR. With the wiener filter of increasing size, the noise is progressively filtered out, while the particles show the same signal. If the FilterSize value is too high (bottom right panel), regions of the central canal with lower SNR can be mistaken for noise and averaged out, preventing a successful estimation of trajectories in these regions.

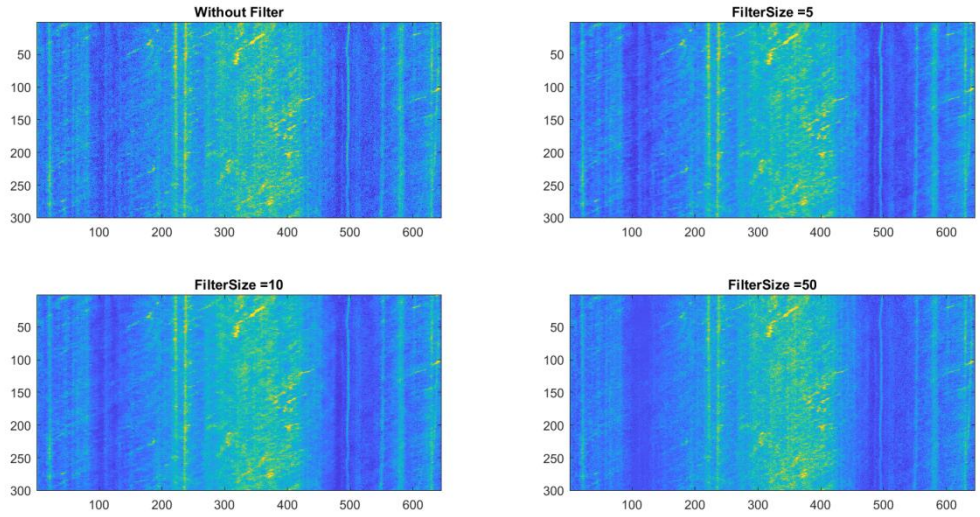



**Figure 8. Wiener filter of different *FilterSize* values on kymographs.** Without filter (top left panel), the image has important random noise, which limits the image SNR, and prevent the correct detection of lines (particle trajectories). With the wiener filter of increasing size, the SNR of the lines progressively increases, but, if the *FilterSize* value is too high (bottom right panel), the signal from some lines is decreased and some lines become more difficult to segment.

b. Test Threshold. By clicking the button '*Test Threshold*', the program first loads the chosen dataset in the Matlab workspace. If selected, the (rotation + crop) processing is performed twice, and the Wiener Filter of size *FilterSizexFilterSize* is sequentially applied to all images. Then, the kymograph is calculated by permuting the dimensions of the dataset (Time becomes the vertical axis, rostrocaudal direction stays horizontal, and dorsoventral direction becomes the

third dimension). The center of central canal (from the maximum of fluorescence intensity) is calculated, and the kymograph at this region is displayed in the bottom left axes (see figure 2). The histogram of the pixel values of this kymograph is calculated and the *Threshold* percentile value is calculated. All values below this threshold are assigned the threshold value, and all values are rescaled between 1 and 2. Then, each column in the kymograph is normalized by its avergaed value (to correct for temporal illumination heterogeneities), and 3 succesive kymographs are averaged. A binary image is then created by keeping only pixels which value is above 1.01 (at least 1% above mean value of each column). The resulting binary image at the center of central canal is plotted in theuser interface bottom right axes (see figure 2). The associated portion of the code is shown below, as well as figure 9 that shows examples of adequate and inadequate values of the *Threshold* parameter. If the value is too low, too many pixels are selected, so that the particle trajectoires cannot be segmented from the surrounding noise. If the Threhsold value is too high, only the brightest particles can be followed and only a few events are detected.

```
Kymo=permute(NewIma,[3 2 1]);
[~,PosMax]=max(mean(mean(Kymo)));

imagesc(ListHandles(23),Kymo(:,:,PosMax))
Kymo=rescale(Kymo,0,1);
Min=quantile(Kymo(:,:,PosMax),Threshold,'all');
Kymo(Kymo<Min)=Min;
Kymo=rescale(Kymo,1,2);
Kymo=Kymo./repmat(mean(Kymo,1),NImages,1);

Kymo_Avg=mean(Kymo(:,:,PosMax-1:PosMax+1),3);

imagesc(ListHandles(24),Kymo_Avg>1.01)
set(ListHandles(15),'BackgroundColor',[0.93,0.69,0.13]);
```

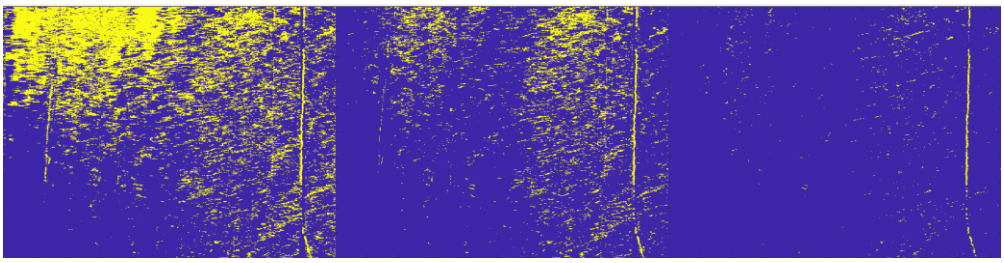
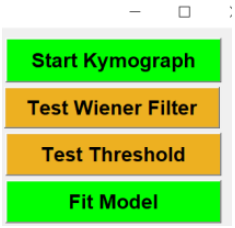


Figure 5. Test Threshold value. All panels show the kymograph at the dorso-ventral position where the signal is maximum. With optimal filter value, the threshold value is tuned from low to high value (left to right), resulting in lines being either not separable (and counted as only one gigantic region of interest) or missed. Keep in mind that this is the image with maximum intensity, so that it is better to have a few errors in this plane and not to miss too many traces in other planes.

c. If multiple files were selected, the test Wiener Filter and Test Threshold will only apply on the first .tif file, but the kymograph analysis will apply to all files.




8. **Run the main analysis to generate the flow profile.** By clicking the *Start Kymograph* button. If several .tif files were selected, all files will be analyzed sequentially. First, the program loads the chosen dataset in the Matlab workspace. If selected, the (rotation + crop) processing is performed twice. If the Wiener filter option is selected, a Wiener Filter of size *FilterSizexFilterSize* is sequentially apply to all images in the dataset. Then, the kymograph is calculated by permuting the dimensions of the dataset (Time becomes the vertical axis, rostrocaudal direction stays horizontal, and dorsoventral direction becomes the third dimension).

```
Kymo=permute(Ima,[3 2 1]);
```

The center of central canal (from the maximum of fluorescence intensity) is calculated, and the kymograph at this region is displayed in the bottom left axes (see figure 2). The histogram of the pixel values of this kymograph is calculated and the *Threshold* percentile value is calculated. All values below this threshold are assigned the threshold value, and all values are rescaled between 1 and 2.

```
Min=quantile(Kymo(:,:,CanalCenter),Threshold,'all');
Kymo(Kymo<Min)=Min;
Kymo=rescale(Kymo,1,2);
```

Then, each column in the kymograph is normalized by its avergaed value (to correct for temporal illumination heterogeneities).

```
Kymo=Kymo./repmat(mean(Kymo,1),NImages,1);
```

For the rest of the analysis, we will scan successive kymograph at each dorso-ventral position, aiming to segment as many straight lines (particle trajectories) as possible. If the *Display* option is selected, at each position, a figure similar to figure 10 is displayed, and if *Save display* option is selected, the movie corresponding to the display of each figure for all dorso-ventral positions is saved. First, we averaged three successive kymographs (with a moving average) to account for the fact that, due to Brownian motion, particles can slightly drift along the dorsoventral axis.

```
Kymo_Avg=mean(Kymo(:,:,ii:ii+NAvg),3);
```

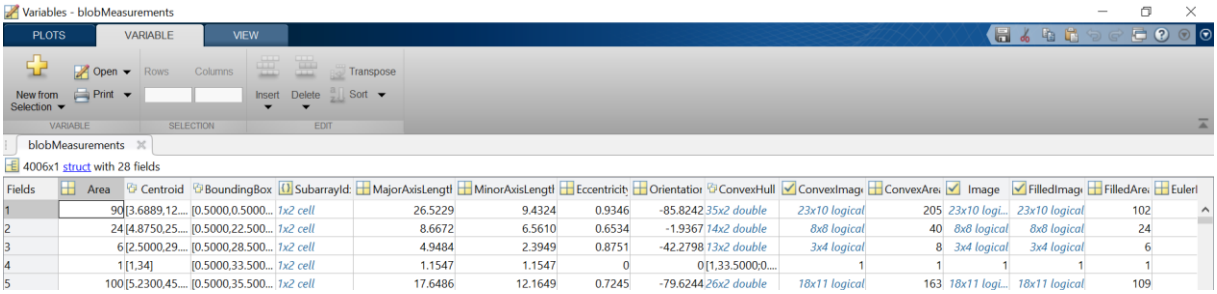
If the Display option is selected, the obtained image is displayed in the top left panel (figure 10). A binary image is then created by keeping only pixels which value is above 1.01 (at least 1% above mean value of each column).

```
Binary=Kymo_Avg>1.01; % Not very stringent filter
```

All groups of connected pixels are labeled with a different identification number (and color in the display), which corresponds to the top right panel in Figure 10. Each group of pixels is then segmented and its properties (Size, Position, Mean Intensity, etc..) are measured.

```
Binary= bwlabel(Binary, 8);
blobMeasurements = regionprops(Binary, Kymo_Avg, 'all');
```

At this point, a structure should be obtained with many properties measured for each group of pixels. There is little chance that all groups of pixels represent particle trajectories, hence we will filter a subpart of these groups that are likely to correspond to straight lines and particle trajectories.




| Fields | Area | Centroid | BoundingBox | SubarrayIdx | MajorAxisLength | MinorAxisLength | Eccentricity | Orientation | ConvexHull | ConvexImage | ConvexArea | Image | FilledImage | FilledArea | EulerN |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 90 | [3.6889,12.... | [0.5000,0.5000... | 1x2 cell | 26.5229 | 9.4324 | 0.9346 | -85.8242 | 35x2 double | 23x10 logical | 205 | 23x10 logi... | 23x10 logical | 102 | |
| 2 | 24 | [4.8750,25.... | [0.5000,22.500... | 1x2 cell | 8.6672 | 6.5610 | 0.6534 | -1.9367 | 14x2 double | 8x8 logical | 40 | 8x8 logical | 8x8 logical | 24 | |
| 3 | 6 | [2.5000,29.... | [0.5000,28.500... | 1x2 cell | 4.9484 | 2.3949 | 0.8751 | -42.2798 | 13x2 double | 3x4 logical | 8 | 3x4 logical | 3x4 logical | 6 | |
| 4 | 1 | [1,34] | [0.5000,33.500... | 1x2 cell | 1.1547 | 1.1547 | 0 | 0 | [1,33.5000;0.... | 1 | 1 | 1 | 1 | 1 | |
| 5 | 100 | [5.2300,45.... | [0.5000,35.500... | 1x2 cell | 17.6486 | 12.1649 | 0.7245 | -79.6244 | 26x2 double | 18x11 logical | 163 | 18x11 logi... | 18x11 logical | 109 | |

First, we want to detect lines, so that the eccentricity (Each group of pixels is fitted by an ellipse with a given eccentricity and angle. An eccentricity close to 1 means a group of pixels with one axis much longer than the other-hence likely corresponding to a line). Second, we want that the lines correpond to particle trajectories. If small particles are detected, we expect that the spatial dimensions of the particle correspond to the impulse response (PSF) of the microscope, which, if the sampling is correct, should correspond to 2 or 3 pixels. Additionally, we assume that the particle should be followed for at least 5 images so that we can consider its trajectory. In total, we expect that the group of pixels corresponding to particle trajectories have a significantly large area of at least 15 pixels (3x5 pixels). Finally, we want to consider only moving particles (hence remove trapped particles that will be detected on all the frames, corresponding to vertical lines in the

kymographs), and avoid line CMOS camera noise (corresponding to horizontal lines in the kymographs).

In total, we filter the group of pixels with 4 conditions: The eccentricity should be over 0.9, the size above 15, and the lines should not be vertical and horizontal (cosinus and sinus of the angle should be above 0.1 in absolute value).

```
EffectiveLines=blobMeasurements([blobMeasurements.Eccentricity]>0.9&[blobMeasuremen
ts.Area]>15&abs(sin(pi/180*[blobMeasurements.Orientation]))>0.1&abs(cos(pi/180*[blo
bMeasurements.Orientation]))>0.1);

    %Create an image with all regions of interest kept (bottom left)

    keeperIndexes =
find([blobMeasurements.Eccentricity]>0.9&[blobMeasurements.Area]>15&abs(sin(pi/180*
[blobMeasurements.Orientation]))>0.1&abs(cos(pi/180*[blobMeasurements.Orientation])
)>0.1);
    keeperBlobsImage = ismember(Binary, keeperIndexes);
    keeperBlobsImage= bwlabel(keeperBlobsImage, 8);
```

The filtered binary kymograph with the remaining lines is displayed in the bottom left panel (Figure 10). For all the segmented lines, the angle is then extracted and transformed into a particle velocity, using the FrameTime and PixelSize values.

```
Orient_Kymo=[EffectiveLines.Orientation];

    %Calculate the speed from the angle

    Speed=1./tan(-Orient_Kymo*pi/180)*(PixelSize/FrameTime);
```

Finally, if more than 5 lines are detected, the average and standard error speed value are calculated, and the speed histogram displayed if the corresponding option is selected.

```
if isempty(Speed)||length(Speed)<5
        LocalMean(ii)=0;
        LocalSE(ii)=0;

    else
        LocalMean(ii)=mean(Speed);
        LocalSE(ii)=std(Speed,0)/sqrt(length(Speed));
        SpeedAll(ii,1:length(Speed))=Speed;
    end
```

This gives the average particle velocity for a given dorsoventral position. It thus remains to perform such segmentation for all kymographs (all dorso-ventral positions) to obtain the flow profile. The calculated flow profile is finally displayed in the bottom right panel of the user interface (Figure 2). If several .tif files are selected, each profile is displayed

sequentially, and the flow profile data saved into a structure called CSFProfile in the Matlab workspace. Additionally, if some ouputs are selected, the flow profile data are saved as .mat, .xlsx, or both. For the .xlsx output, excel should be downloaded on the computer. **The .mat generates a structure that contains, for each file, the dorso-ventral position, the flow at each position, the standard error, as well as the speed histograms at the extrema of the flow profile. In the .xlsx output, the data from each file is saved in a different sheet, and the dorso-ventral position, the flow at each position, and the standard error are saved.**

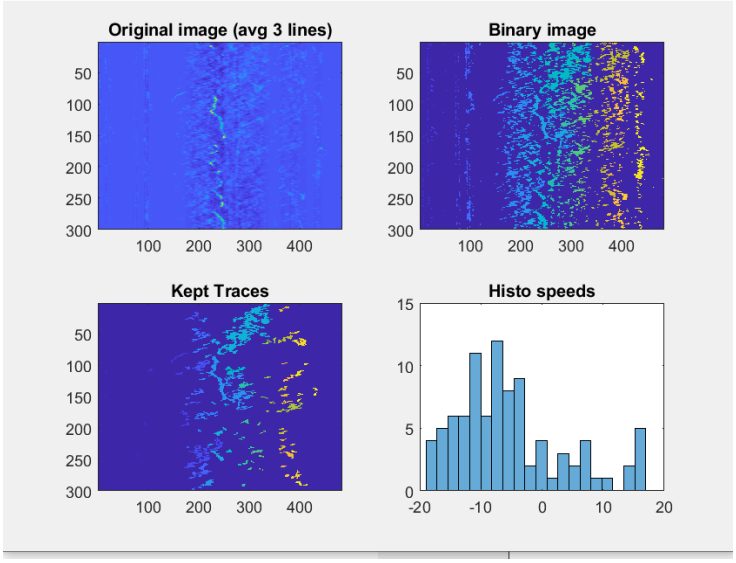



**Figure 10. Display window showing kymograph analysis.** Filtered kymograph at one dorso ventral position (top left corner), and the binary image after intensity thresholding (top right) are shown. The traces selected for analysis (bottom left) are filtered to be long and straight enough. The angle of all traces selected for analysis is converted into corresponding bead velocity and aggregated into a histogram (bottom right).

## Fit the flow profile with the bidirectional flow model

*Depending whether the velocity profile is bidirectional or not, it could be relevant to adjust it with a model providing information on the force generated by the cilia. In (Thouvenin et al., 2020), we developed a simple model accounting for both the force locally generated by the cilia and the constraint of «no positive flux» imposed by the closed geometry of the central canal. We showed that the averaged velocity profile can be fairly described by a piecewise second-order polynomial, defined as:*

$$v_{dorsal}(y) = a_1 y^2 + a_2 y + a_3 \quad (1)$$

$$v_{ventral}(y) = b_1 y^2 + b_2 y \quad (2)$$

*where $a_1$, $a_2$, $a_3$, $b_1$ and $b_2$ are constant expressed as a function of the parameters of the problem:*

$$a_1 = \frac{dP/dx}{2\mu},$$
$$a_2 = -\frac{dP/dx}{\mu} \cdot \frac{3d}{4},$$
$$a_3 = \frac{dP/dx}{\mu} \cdot \frac{d^2}{4}$$
$$b_1 = \frac{dP/dx - f_v}{2\mu}$$
$$b_2 = -\frac{dP/dx - f_v}{\mu} \cdot \frac{d}{4}$$

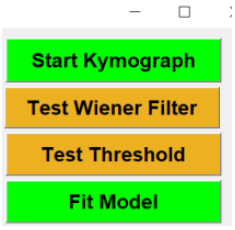
*With $d$ is the diameter of the channel and $\mu$ the viscosity of the CSF, $\frac{dP}{dx}$ is the pressure gradient and $f_v$ is the average force per unit volume generated by a cilium. The latter two parameters can be adjusted experimentally from the CSF velocity profiles processed with the GeneProfile interface.*

*We propose for the user an automated velocity profile fitting tool. Before launching the fitting tool, we provide the user an estimate of the bidirectionality of the flow, called $\beta$ defined as:*

$$\beta = \left(1 - \frac{\left|\int_0^d v(y)dy\right|}{\int_0^d |v(y)|dy}\right) \times 100$$

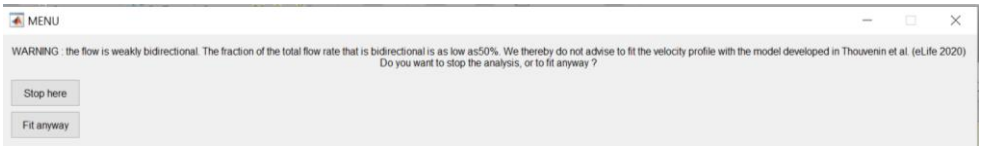
*$\beta$ varies between 0% for a purely monodirectional flow and 100% for a purely bidirectional flow (the flow rate advected towards the tail equals the flow rate advected towards the brain).*

*We advise the users not to perform the fitting of their velocity profiles for values of $\beta$ < 70%, a threshold empirically defined.*

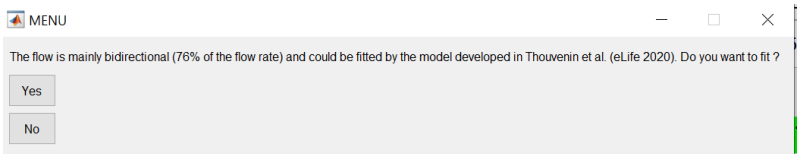



1. Once the flow profile is calculated, “Fit Model” button on the right turns green. The fit can be performed by clicking on this button.
2. Two possibilities can arise:
   a. if $\beta$ < 70%, we estimate that the flow is not bidirectional enough to fit the velocity with our model, and display the warning message “We advise the user not to go

further". Click on "Stop here".




b. if $\beta$ > 70%, the flow can be reasonably fitted with the simple model, and a verification message is sent to the user. Click on "Yes".




3. If the fit is selected, a spline interpolation (function defined piecewise by polynomials) is performed, in order to obtain one polynomial function in the dorsal region (equation 1) and a second polynomial in the ventral region (equation 2), and obtain parameters $a_{1\text{-}3}$ and $b_{1,2}$. From these parameters, the volumetric force, the central canal diameter, and the pressure gradient can be recovered. Additionally, we extract the fitted curve, and the size of the ventral region (where the polynomial function changes in the spline interpolation).
4. The following structure called *FitProfile* is thus obtained in the Matlab workspace :

```
FitProfile(nn).Profile=PP;
FitProfile(nn).Flow=round((1-Ratio_Mean)*100);
FitProfile(nn).Diameter=Pos_dim(end)-Pos_dim(1);
FitProfile(nn).l_cilia = AA.breaks(2);
mu_CSF=10^-3; %Pa.s
FitProfile(nn).f_v=abs(AA.coefs(1,1)-AA.coefs(2,1))*2*mu_CSF*10^6;%unit: Pa/m
FitProfile(nn).GradP=abs(AA.coefs(1,1))*2*mu_CSF*10^6;
```

The Profile field gives the fit values, the flow gives the parameter β, the diameter gives the fitted central canal diameter, l_cilia gives the fitted height of the cilia region (ventral region), and f_v and Grad P respectively give the volumetric force $f_v$ generated by the cilia and the pressure gradient dp/dx opposing this force.

5. The fitted velocity profile (Profile field) displayed in red on the same plot as the experimental profile (in blue) on the bottom right axes of the user interface (Figure 2 –Figure 11). The plot is also saved in .fig and .png format in the same folder as the experimental .tif files. The .fig file allows the user to modify the plot, and do aesthetic changes, as well as to save in vector formats (.eps, .pdf, .svg).

6. Several information can be extracted from the fit, which are detailed below. The data are stored in either an .xls file, and/or a .mat file, called “TheoreticalProfiles”, where:
    a. The velocity is stored in a 10000x1 vector called “Profile” (in µm/s) and the rostro-caudal position in a 10000x1 vector called “X” (in µm).
    b. The volumic force $f_v$ generated by the cilia is stored in the variable called “f_v” (in $N.m^{-3}$).
    c. The common pressure gradient $dP/dx$, opposing the cilia beating due to the closed geometry of the central canal, is stored in the variable “GradP” (in $N.m^{-3}$).
    d. The width of the ciliary region, where the volumic force $f_v$ is generated in the model, is stored in the variable “l_cilia” (in µm).
    e. The measured diameter of the central canal is stored in the variable “Diameter” (in µm). Note that, if required, we advise to use a direct measurement with TexasRed to measure the central canal diameter, since deducing the diameter from the kymograph may be imprecise.

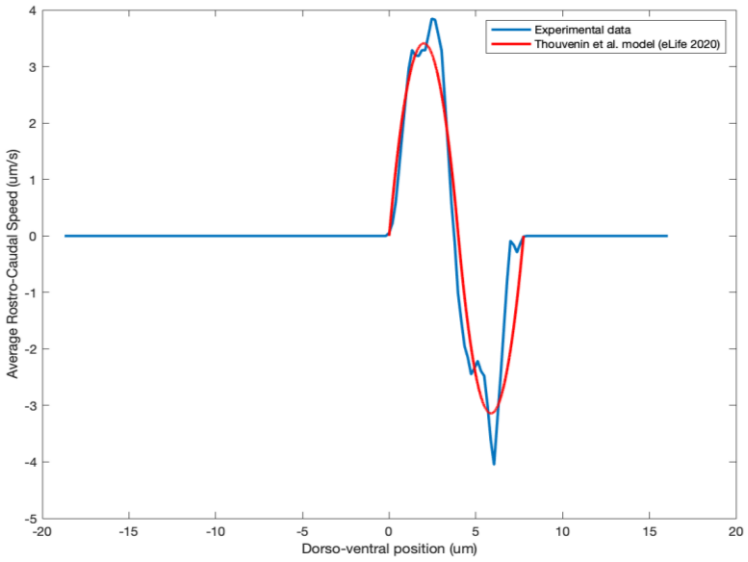



**Figure 11. Example of experimental and theoretical CSF flow profile.** Using the user interface, and the test data set, the experimental CSF flow profile is plotted in blue, with its associated model fit in red.