



Manual



## Contents

<b>1. Introduction</b>	<b>4</b>
<b>2. Installation and dependencies</b>	<b>4</b>
2.1. Structure	4
2.2. TFMLAB installation	4
<b>3. Running TFMLAB step by step</b>	<b>6</b>
3.1. Input file selection	7
3.2. Image filtering and enhancement	9
3.2.1. Common parameters and tools	9
3.2.2. Beads	10
3.2.3. Fibers	10
3.2.4. Cell	10
3.2.5. Cell segmentation	10
3.3. Pre-shift correction	11
3.4. Check shift correction	12
3.5. Displacement measurement	12
3.6. Visualization file generation	14
3.7. Force calculation	14
<b>4. Other dependencies</b>	<b>17</b>
<b>5. References</b>	<b>20</b>

# 1. Introduction

This is the manual for the 4D-Traction Force Microscopy toolbox TFMLAB. A detailed description of the installation and all the important processing parameters is provided.

If you use this toolbox, please cite references [1]–[3].

For questions, please feel free to contact [jorge.barrasafano@kuleuven.be](mailto:jorge.barrasafano@kuleuven.be).

## 2. Installation and dependencies

### 2.1. Structure

TFMLAB is available at the Gitlab repository: [https://gitlab.kuleuven.be/MATrix/Jorge/tfmlab\\_public](https://gitlab.kuleuven.be/MATrix/Jorge/tfmlab_public)

To use TFMLAB, several software are required, three of which are optional:

- MATLAB: with the Signal Processing Toolbox and the Image Processing Toolbox. Version 2019a or above. Licensed software.
- DIPImage toolbox: freely available at: <http://www.diplib.org/download>
- Paraview (optional): only required to visualize 3D renders of the results. Freely available at: <https://www.paraview.org/download/>
- Abaqus (optional): only required for force calculation. Licensed software.
- FreeFEM (optional): only required for force calculation (and in absence of Abaqus). Freely available at: <https://github.com/FreeFem/FreeFem-sources/releases>. We recommend to use version v4.7-1.

The rest of the necessary software is distributed with TFMLAB (see section 4).

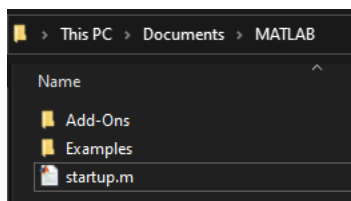
This toolbox is only compatible with Windows systems.

### 2.2. TFMLAB installation

There are a couple of simple steps required to get TFMLAB up and running. Trust us... go through them and you will soon be able to use the software!

- Download TFMLAB.
- Download and install DIPImage (see section 2.1).

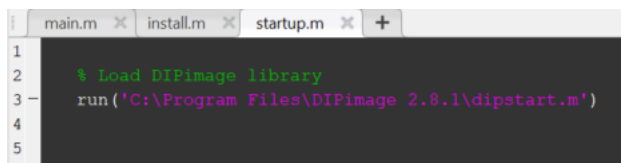
Once installed, DIPImage should be always be added to the MATLAB path before running TFMLAB. The easiest way to do this is to create or modify a file called startup.m in the default working directory of MATLAB (normally in \Documents).



In this file, write the following line (please note that you might have to adapt this line according to your installation path and the version of DIPImage that you use):

```
run('C:\Program Files\DIPimage 2.8.1\dipstart.m');
```

and save it. This will run 'dipstart.m' automatically at the beginning of every session and you will not have to worry about DIPImage anymore.

A screenshot of a MATLAB script editor window. The window has three tabs: 'main.m', 'install.m', and 'startup.m'. The 'startup.m' tab is active. The script content is as follows:

```
1  
2 % Load DIPimage library  
3 run('C:\Program Files\DIPimage 2.8.1\dipstart.m')  
4  
5
```

The rest of the external functions and toolboxes are included in the TFMLAB package. If you plan to calculate tractions, you will need to do two things:

- Install the freely available MinGW-w64 C/C++ compiler in MATLAB. On the MATLAB tab HOME, click on Add-Ons→Get Add-Ons, search for 'MinGW' and install it.
- Run the script install\_tfmlab.m. This command compiles the toolbox SPQR from SuiteSparse. This might take some minutes and some weird text in the command line, but eventually you will get the message 'TFMLAB is ready to use'.

You are now ready to run TFMLAB!

### 3. Running TFMLAB step by step

We developed TFMLAB in the MATLAB framework to provide robust and user-friendly processing of the microscopy data. The software has one main function `main.m`, which calls other routines at every step of computational TFM. However, the user only needs to interact with TFMLAB's GUIs.

Sample data is provided with TFMLAB. A .lif file can be found in `/test_data/PaperData.lif`. This file contains two image series (image stacks), one corresponding to the Stressed state (mechanically active cells, stored in series 1) and one corresponding to the Relaxed state (cells after relaxation with Cytochalasin D, stored in series 2). Each series contains 4 channels: cell channel corresponding to LifeAct-GFP2 signal (channel 1), bead channel corresponding to the fluorescence of 0.2  $\mu\text{m}$  red fluorescent carboxylated polystyrene microspheres (channel 2), transmission light channel (channel 3, not used for TFM), and a nuclei channel corresponding to Hoechst signal (channel 4, it can be included in the workflow but it does not play a role in the calculations).

Briefly, TFMLAB has the following steps: (1) microscope image reading, (2) image processing, (3) displacement measurement and (4) force calculation. Open the script `main.m` in MATLAB and press Run. In this section, you will find a description of all the buttons and parameters present in TFMLAB's GUIs.

### 3.1. Input file selection

The screenshot shows the 'TFMLAB: Input data' window. It contains several input fields and buttons, each with a green circular callout number:

- 1**: 'Select input file...' button
- 2**: 'Select output folder...' button
- 3**: 'Experiment name:' text input field
- 4**: 'Z plane range to be kept (separated by commas):' text input field with value '1,1000'
- 5**: 'Time point range to be kept (separated by commas):' text input field with value '1,1000'
- 6**: 'Series number for the 'stressed' state:' text input field with value '1'
- 7**: 'Series number for the 'relaxed' state:' text input field with value '1'
- 8**: 'Channel 0', 'Channel 1', and 'Channel 2' headers in a table
- 9**: '+' button next to the table
- 10**: 'Skip this step >>' button
- 11**: 'Start' button

Additional text in the window includes the TFMLab logo, copyright information for KU Leuven (Belgium) and Universidad de Sevilla (Spain), and a note about channel names: 'Define the channel names (leave empty if you don't want to use a channel). Accepted names: cell, beads, fibers or nuclei:'.

	Channel 0	Channel 1	Channel 2
+	cell	beads	fibers

- 1) Browse through your folders to select a raw microscopy file. Please, select the option 'All files (\*.\*)' when browsing. Select your microscopy file. Most of the microscopy formats are accepted (TFMLAB uses the Bio-formats<sup>1</sup> library to read data).
- 2) Browse through your folders to select an output folder. A new folder will be created in the selected directory where all the results will be stored.
- 3) Defines a name for this experiment. It will be name of the output results folder.
- 4) Defines the Z plane range that you want to use from the images. If the second number is higher than the number of Z planes, the entire stack is used.

<sup>1</sup> <https://docs.openmicroscopy.org/bio-formats/6.1.0/index.html>

- 5) Defines a range of time points that you want to use. If the second number is higher than the number of time points, the entire time lapse is used.
- 6) Give the series identifier (i.e., as numbered by ImageJ) corresponding to the stressed state in your microscopy file. This series can have multiple time points. You can also include the relaxed state as the last time point in this series (see 7)).
- 7) Give the series identifier (i.e., as numbered by ImageJ) corresponding to the relaxed state in your microscopy file. If your relaxed state was already included in the stressed series and you do not have a separate series for the relaxed state, you can set this field to 0. This series can have multiple time points but only the last one will be considered as purely relaxed and will be the reference for displacement calculation.
- 8) Define what modality is present in each channel. The software only accepts cell (e.g. imaging the cells after transduction with LifeAct), beads (e.g. imaging the fluorescence of the beads) or fibers (e.g. imaging collagen by means of second harmonic generation). Additionally, it also accepts a nuclei channel but it will not play a role in the calculations. If the microscopy file has modalities that are not used in TFMLAB (e.g. bright field) in a specific channel, leave that channel empty. You can add more channels to the table if needed (see 9)).
- 9) Add more channels to the table described in 8).
- 10) If you already have input data that has been converted to tiff files and arranged following TFMLAB's output folder structure, you can skip this step and select the A\_rawImagesTiff folder.
- 11) Start extracting the data from the microscopy file and go to the next step.

After pressing 'Start', the output folder will be created and the images from the different channels will be stored in a systematic fashion.

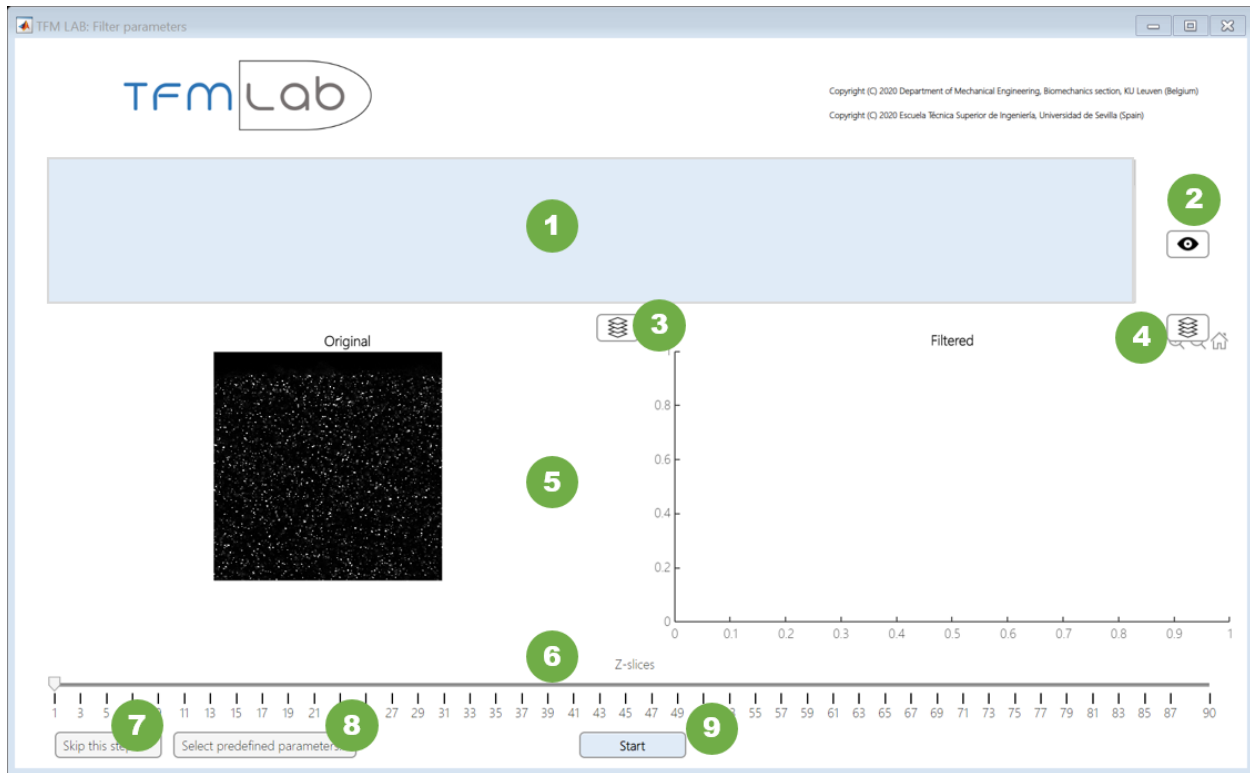
In your output folder, a folder called 'A\_rawImagesTiff\_' followed by the x, y, z resolution (voxel size) in microns will be created. Inside, a .mat file called input\_param.mat containing all parameters chosen in the GUI will be present for reproducibility. Furthermore, there will be a folder for each channel (called 'cell', 'fibers'...). Inside these channel folders, there will be one .tif image per time point. The last or relaxed time point is always named as tp\_R.tif.



## 3.2. Image filtering and enhancement

There will be as many tabs as provided channels (excluding nuclei). Here we explain the parameters for each channel.

### 3.2.1. Common parameters and tools



- 1) Channel specific filtering parameters.
- 2) See the result of filtering with current parameters.
- 3) Generate a maximum intensity projection of the original image.
- 4) Generate a maximum intensity projection of the filtered image.
- 5) Plot area for the original image (left) and the filtered image (right). The first time point of the time series is plotted.
- 6) Z-slice navigator. You can scroll through the z slices of your image. It affects to both the original image and the filtered image. Mouse scroll has the same effect.
- 7) If you already have filtered images from a previous run, you can skip this step and select the B\_filteredImages folder.
- 8) If you want to use the same filtering parameters as in a previous run, you can select the filter\_param.mat which is stored in the B\_filteredImages folder.
- 9) Start filtering with current parameters.

### 3.2.2. Beads

- 1) Sigma for the first Gaussian filter of the DoG filter. (It must be lower than var2).
- 2) Sigma for the second Gaussian filter of the DoG filter. (It must be higher than var1).
- 3) Amplitude for the second filter of the DoG filter (controls the amount of filtering).
- 4) Sigma for further low pass filtering (only applied if >0).
- 5) Intensity value adjustment. Maps the introduced values to [0, 255].

### 3.2.3. Fibers

- 1) Recommended when the fiber structures are dim. Performs a logarithmic or exponential operation.
- 2) Choose between logarithmic or root transformation.
- 3) Exponential value for the root transformation.
- 4) Sigma for further low pass filtering (only applied if >0).
- 5) Intensity value adjustment. Maps the introduced values to [0, 255].

### 3.2.4. Cell

The parameters are the same as in 3.2.3.

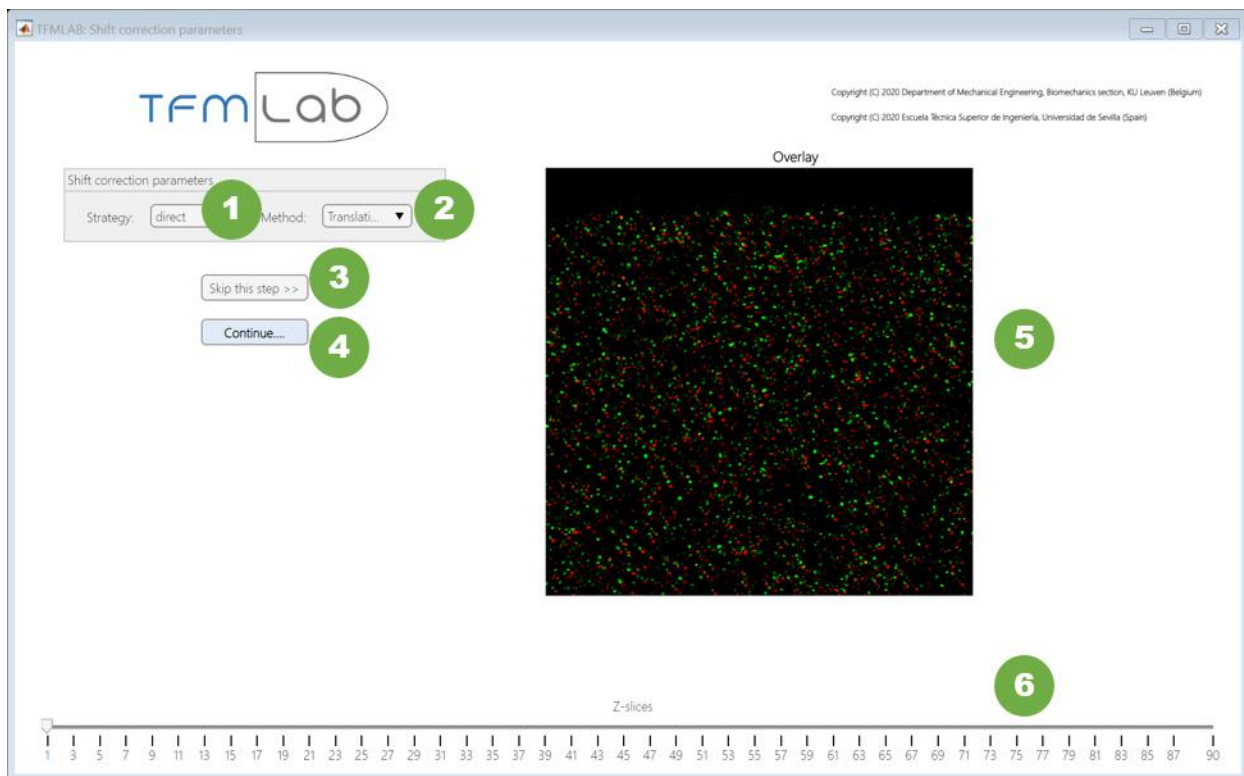
### 3.2.5. Cell segmentation

- 1) Default, the program chooses a threshold based on Otsu's algorithm. With this parameter, you can readjust Otsu's threshold by a certain percentage.
- 2) Choose minimum amount of voxels required for a structure to be considered a cell (removes small noisy regions).

- 3) Generate a 3D render of the cell segmentation
- 4) Generate a maximum intensity projection of the filtered cell image with an overlay of the segmentation.

In your output folder, a folder called 'B\_filteredImages' will be created. Inside, a .mat file called filt\_param.mat containing all parameters chosen in the GUI will be present for reproducibility. Furthermore, there will be a folder for each channel (called 'cell', 'fibers'...). Inside these channel folders, there will be one .mat file per time point. The last or relaxed time point is always named as tp\_R.tif.

### 3.3. Pre-shift correction



- 1) Strategy choice. 'direct': all the time points are registered with respect to the middle one (recommended when there isn't a large shift between the first time point and the relaxed one). 'seq': all the time points are registered with respect to the subsequent one (recommended when there is a large shift between the first time point and the relaxed one).
- 2) Method choice: type of rigid registration. 'Translation phaseCorr (fast)' uses a phase correlation based image registration (recommended). 'Translation Elastix (medium)' uses the Elastix translational rigid registration. 'Euler Elastix (slow)' uses the Elastix Euler registration (not recommended, unstable).
- 3) If you already have shift corrected images from a previous run, you can skip this step and select the C\_shiftCorrectedImages folder.

- 4) Run rigid registration with current parameters.
- 5) Plot area. An overlay of the stressed image in green (time point 1) and the relaxed image in red is shown. An image with no shift should have a majority of yellow (perfect overlap between beads/fibers) except for the areas where cells are interacting with the matrix.
- 6) Z-slice navigator. You can scroll through the z slices of your image. Mouse scroll has the same effect.

In your output folder, a folder called 'C\_shiftCorrectedImages' will be created. Inside, a .mat file called shift\_param.mat containing all parameters chosen in the GUI will be present for reproducibility. Furthermore, there will be a folder for each channel (called 'cell', 'fibers'...). Inside these channel folders, there will be one .mat file per time point. The last or relaxed time point is always named as tp\_R.mat.

### 3.4. Check shift correction

The program includes this GUI to verify that the rigid shifts were correctly removed. Here, the user should see a majority of yellow in the image overlay (perfect overlap between beads/fibers) except for the areas where cells are interacting with the matrix. It should be relatively easy to know where the cells are interacting with the matrix just by looking at the image overlay. Moreover, a quantitative similarity metric between the images is also provided. Normally, this number should be above 60%.

As a rule of thumb, if the pre-shift correction step (using the 'direct' strategy and the 'phaseCorr' method) did not perform well (showing little overlay/yellow and low similarity metric), it is not recommended to continue. Typically, this is due to strange aberrations in the hydrogel (non rigid deformations that are not caused by the cells and that heavily modify the hydrogel structure).

Based on this information the user can decide whether to move on to the next step, to cancel or to go back to the pre-shift correction GUI to change a parameter and repeat the process.

### 3.5. Displacement measurement

TFMLAB uses the FFD algorithm for displacement calculation. FFD uses the software Elastix for efficient non-rigid image registration. The user is kindly referred to our article on FFD [4] and Elastix's excellent documentation<sup>2</sup> to get a better insight on the parameters used. TFMLAB allows for tuning just a basic selection of registration parameters from the large list that Elastix provides. Throughout the years, we have optimized the rest of the hidden parameters for 3D TFM applications.

---

<sup>2</sup> <https://elastix.lumc.nl/download/elastix-5.0.0-manual.pdf>

Displacement measurement parameters:

Grid spacing XY (um):  1 Number of iterations:  2

Grid spacing Z (um):  3

☒ Use cell mask 4 ☐ Get Jacobian matrix 5

Strategy:  Time step (only for sequential):

☐ Artifact correction

Metric:  6  9

Optimizer:  7  10

Marker type:  8 11

- 1) XY spacing between grid nodes.
- 2) Number of iterations per scale for the optimizer.
- 3) Z spacing between grid nodes.
- 4) Use the cell mask to ignore the beads that fall inside the cell segmentation. Normally, this is due to engulfment by the cell and ignoring these beads is desirable, as their movements have no mechanical meaning.
- 5) Generate a Jacobian matrix in the results. It can be useful if the user wants to compute strain field from the results. Use only if really needed as it increases the computational time and it has higher RAM demands.
- 6) Registration metric. Some typical metrics from Elastix are included as options. We recommend 'normXcorr' (Normalised Correlation Coefficient in Elastix documentation) for TFM applications.
- 7) Registration optimizer. Some typical optimizers from Elastix are included as options. We recommend 'adapStochGradDesc' (AdaptiveStochasticGradientDescent in Elastix documentation) for TFM applications.
- 8) Define which type of marker you want to use for the registration (beads or fibers). If your images contain both channels, you can choose 'both' and two registrations will take place, first with beads and then with fibers.
- 9) If you want to use the same parameters as in a previous run, you can select the file disp\_param.mat, which is stored in the D\_2\_displacements folder.
- 10) If you already have measured displacements in a previous run, you can skip this step and select the D\_2\_displacements folder.
- 11) Start measuring displacements with current parameters.

In your output folder, three new folders called will be created.

'D\_1\_calcElastix' contains one folder per channel used. Inside, the .bat files used to run Elastix are stored for reproducibility. Moreover, the parameters and the output strings generated by Elastix are also stored.

In 'D\_2\_displacements' there is a .mat file called disp\_param.mat containing all parameters chosen in the GUI for reproducibility. Furthermore, there will be a folder for each channel (called 'cell', 'fibers'...). Inside these channel folders, there will be one .mat file per time point. The last or relaxed time point is not included as it has been the reference image for the registration. Zero displacements at this time point can be assumed.

After displacement measurement, a post-shift correction is automatically done. Typically, FFD can capture displacements more accurately than a basic phase correlation operation (see 3.3). If there were any spurious rigid shifts not corrected by the rigid registration, we can remove them with this operation by subtracting the mean from each displacement component (X, Y and Z).

'D\_3\_postShiftCorrectedImages', there is a .mat file that contains the detected shifts. Furthermore, there will be a folder for each marker used for image registration. Inside, there is a folder per channel (called 'cell', 'fibers'...). Inside these channel folders, there will be one .mat file per time point with the corrected image.

### 3.6. Visualization file generation

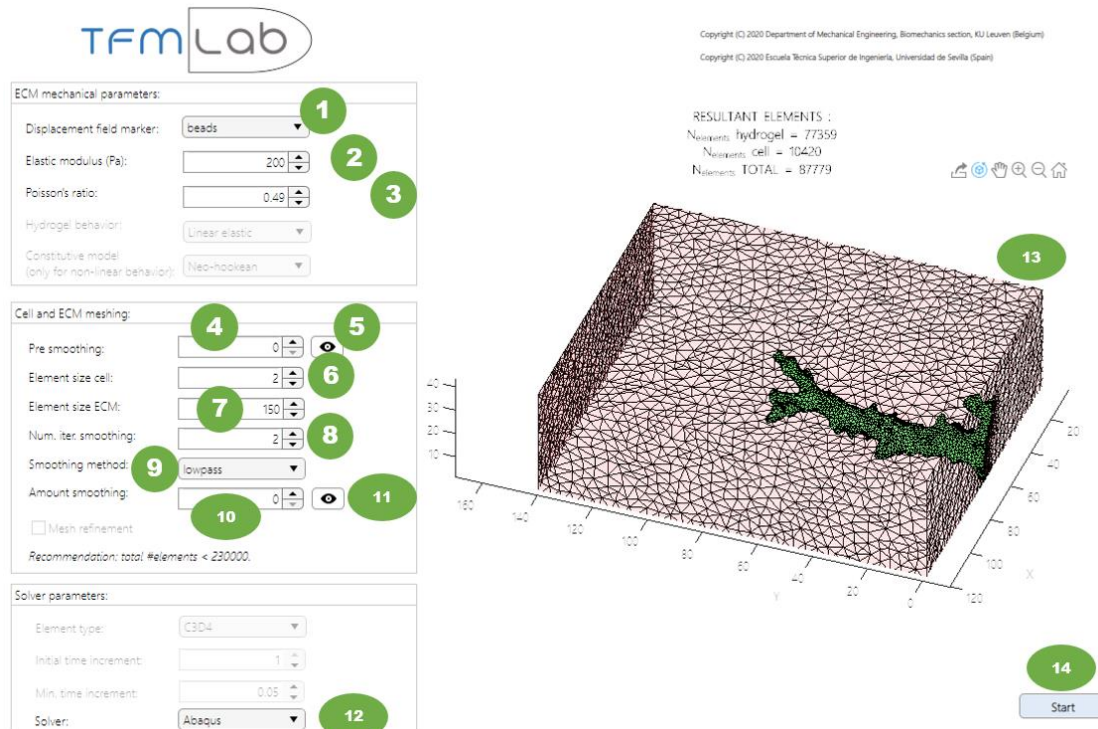
The software will then ask you to crop a region of interest by manually drawing a rectangle on a 2D projection of the displacement magnitude (and if provided, the cell segmentation). The rectangle will define the crop in X and Y while will preserve all the Z slices. The data contained in this crop will be exported into Paraview files. In your output folder, a folder called 'E\_resultsParaview' will be created. Inside, a template.pvsm that can be used to visualize the results in Paraview is provided. Furthermore, there will be a folder for each marker used for image registration. Inside, there will be a .mat file called paraview\_param.mat where the coordinates of the region of interest are stored and the .vtk files for cell and displacements for every time point. Please check Paraview's manual<sup>3</sup> to learn how to open and visualize these files.

### 3.7. Force calculation

This step can only be performed if Abaqus/FreeFEM is installed in the system and if a cell channel was provided.

---

<sup>3</sup> <https://www.paraview.org/paraview-guide/>



- 1) Choose the marker that was used to measure the displacements from which the program will compute forces.
- 2) Define the elastic modulus of the hydrogel in Pa.
- 3) Define the Poisson's ratio of the hydrogel.
- 4) Parameter to smooth the cell segmentation before creating the FE mesh. Set to 0 if no smoothing is required. This can be helpful to remove spikes in the cell surface that could hamper the meshing. Normally, a relatively smooth cell surface is desired.
- 5) Generates a figure to visualize the cell segmentation after smoothing in 3D.
- 6) Maximum radius of the Delaunay sphere (element size) that is allowed for the cell surface meshing.
- 7) Maximum tetrahedra element volume that is allowed for the ECM meshing.

The next three parameters regulate the smoothing after meshing. They are based on function 'smoothsurf' from the iso2mesh toolbox<sup>4</sup>. Use them to remove spikes in the surface if any.

- 8) Smoothing iteration number (corresponds to parameter 'iter' from the function 'smoothsurf').
- 9) Smoothing method (corresponds to parameter 'usermethod' from the function 'smoothsurf').
- 10) Amount of smoothing after meshing. Set to 0 if no smoothing is required (corresponds to 'useralpha' from the function 'smoothsurf').
- 11) Visualize the FE mesh.
- 12) Select the FE solver. Options: Abaqus or FreeFem.

<sup>4</sup> Check iso2mesh function list for more details: <http://iso2mesh.sourceforge.net/cgi-bin/index.cgi?Doc/FunctionList>

- 13) FE mesh plotting area. Use the MATLAB tools to rotate, and zoom into the mesh. Initially, the cells will be encapsulated in the ECM, by rotating and zooming, the user can access the cell surface. The resultant number of elements is also shown. We recommend to avoid running problems with total number of elements >230,000 to avoid RAM problems (based on tests on computers with 16MB of RAM). However, the user must decide based on their system's memory specifications. A possible way to reduce dimensionality is to make the region of interest smaller in step 3.6.
- 14) Start force calculation with current parameters.

In your output folder, a folder called 'F\_tractions' will be created. Inside, there will be a folder with the name of the marker chosen (beads/fibers). Inside this folder, a .mat file called trac\_param.mat containing all parameters chosen in the GUI will be present for reproducibility. A .txt file with the parameters and the resultant number of elements is also given to ease rapid verifications. The intermediate and final result files provided by Abaqus/FreeFEM (which are used by TFMLAB in the background while computing forces) are stored in 'results\_intermediate' and 'results\_abaqus'/'results\_freefem', respectively. This also allows for reproducibility and to check for possible intermediate problems. A folder called 'results\_paraview' will be generated containing a .pvsm template and .vtk files for cell and the two main mechanical variables (displacements and tractions). Finally, the folder 'results\_raw' contains text files with all the FE coordinates and all the mechanical variables (displacements, strains, stress and tractions) for every time point. 'sr' stands for strain, 'ss' stand for stress, 't' stands for tractions and 'u' stands for displacements. Letters 'f' and 'i' at the end of the name of these files indicate if the variable has been computed forwardly (directly from the measured displacements) or using the PBNIM algorithm, respectively. We recommend using the PBNIM (files ending in 'i') as it proved to be more accurate in previous studies [2], [3].



## 4. Other dependencies

Other external functions and toolboxes **are provided with TFMLAB** with their corresponding license file:

- bfmtlab: <https://www.openmicroscopy.org/bio-formats/downloads/>
- getLargestCC: Ran Shadmi (2020). Get Largest Connected Components (<https://www.mathworks.com/matlabcentral/fileexchange/39094-get-largest-connected-components>), MATLAB Central File Exchange. Retrieved May 18, 2020.
- iso2mesh: <http://iso2mesh.sourceforge.net/cgi-bin/index.cgi>
- NifTI\_tools: Jimmy Shen (2020). Tools for NifTI and ANALYZE image (<https://www.mathworks.com/matlabcentral/fileexchange/8797-tools-for-nifti-and-analyze-image>), MATLAB Central File Exchange. Retrieved May 18, 2020.
- PATCH\_3Darray: Adam A (2020). Plot a 3D array using patch (<https://www.mathworks.com/matlabcentral/fileexchange/28497-plot-a-3d-array-using-patch>), MATLAB Central File Exchange. Retrieved May 18, 2020.
- polygon2voxel\_version1j: Dirk-Jan Kroon (2020). Polygon2Voxel (<https://www.mathworks.com/matlabcentral/fileexchange/24086-polygon2voxel>), MATLAB Central File Exchange. Retrieved May 18, 2020.
- smoothn: Damien Garcia (2020). Smoothn (<https://www.mathworks.com/matlabcentral/fileexchange/25634-smoothn>), MATLAB Central File Exchange. Retrieved May 18, 2020.
- smoothpatch\_version1b: Dirk-Jan Kroon (2020). Smooth Triangulated Mesh (<https://www.mathworks.com/matlabcentral/fileexchange/26710-smooth-triangulated-mesh>), MATLAB Central File Exchange. Retrieved May 18, 2020.
- Elastix: freely available at: <https://elastix.lumc.nl/download.php>
- SuiteSparse: <http://faculty.cse.tamu.edu/davis/suitesparse.html>

## 5. How to include your own displacement measurement modules

While we recommend to use the proposed workflow implemented in TFMLAB, one can modify the code to include new displacement field measurement functions.

The script *main.m* is divided in sections (sections are parts of the code with a title preceded by '%'). Find section '% Displacement calculation' (it should be around line 215). You will see that the first function that is called is *set\_dispCalc*. This is the GUI where the displacement measurement parameters for the FFD are selected. If you want to include a new displacement measurement module, I propose that you do the following:

1. First, you could modify the GUI *set\_dispCalc.m* and include the necessary parameter buttons for your specific displacement field measurement function. I would propose that you add a new tab in the GUI (similar to how the GUI for image processing is arranged – with different tabs for beads, fiber and cell processing) to keep the functionalities of FFD within TFMLAB. I recommend to include a dropdown menu so that the user can select between the available algorithms. Within the code of the GUI, you should store all your parameters in the struct variable *dispParam*, since this is the one that is passed into the base workspace once the user presses the button Start.
2. Once you do that, lower in the *main.m* you will see that it calls to the function *timeSeriesDispCalc*. This function performs the displacement calculation by means of the FFD algorithm. This could be a good place to include your function. Here you could check the value of the dropdown menu to handle which algorithm to use (maybe encapsulating the options with a switch-case statement).
3. You might have to modify your function so that it reads the input images from the output folder structure of TFMLAB. Basically, your function should read the content of the folder *C\_shiftCorrectedImages*. Inside this folder there will be a folder for each channel (called 'cell', 'fibers'...). Inside these channel folders, there will be one .mat file per time point. The last or relaxed time point is always named as *tp\_R.mat*. Typically, your function should take *tp\_R.mat* of the beads/fibers folder as a reference to measure displacements in the rest of the time points. Finally, your function should store the displacement field results in a struct variable called *dispField* with 3 different fields (X, Y, Z), each of them of the size of the images (**Important note:** for some displacement field measurement algorithms like those based on particle image velocimetry, the output displacement field is typically of a smaller size than the original image because one obtains one value per block. We recommend that your function interpolates the calculated values to the rest of the voxels in the image to avoid further manipulations of the code). This variable should be stored in .mat files for each time point in folder *D\_2\_displacements* (*D\_1\_calcElastix* is used to store performance metrics from Elastix as a control for the FFD algorithm; if it is interesting that your function also stores similar information, you could create another *D\_1* folder with a different suffix and store this data).

You can take inspiration on how to loop through files and store them from the code of the function *timeSeriesDispCalc*.

4. If these previous steps are done correctly, the *main* function should be able to proceed with the next steps of the workflow.

Please do not hesitate in contacting us if you need help with this task: [jorge.barrasafano@kuleuven.be](mailto:jorge.barrasafano@kuleuven.be)

## 6. References

- [1] J. Barrasa-Fano, A. Shapeti, A. Jorge-Penas, M. Barzegari, J. A. Sanz-Herrera, and H. Van Oosterwyck, "TFMLAB: a MATLAB toolbox for 4D traction force microscopy," *bioRxiv*, p. 2020.12.18.423056, Dec. 2020.
- [2] J. Barrasa-Fano, A. Shapeti, J. De Jong, A. Ranga, J. A. Sanz-Herrera, and H. Van Oosterwyck, "Advanced in silico validation framework for three-dimensional Traction Force Microscopy and application to an in vitro model of sprouting angiogenesis," *bioRxiv*, p. 2020.12.08.411603, Dec. 2020.
- [3] J. A. Sanz-Herrera, J. Barrasa Fano, M. C ndor, and H. Van Oosterwyck, "Inverse method based on 3D nonlinear physically constrained minimisation in the framework of traction force microscopy.," *Soft Matter*, 2020.
- [4] A. Jorge-Pe nas *et al.*, "Free form deformation-based image registration improves accuracy of traction force microscopy," *PLoS One*, vol. 10, no. 12, pp. 1–22, 2015.