# From Migrating Cell to Shape Space

If you have never used Matlab before: you should read this section.

You should have a folder full of DV files.

All folders for the program.

Before you start: adding the files to the file path.

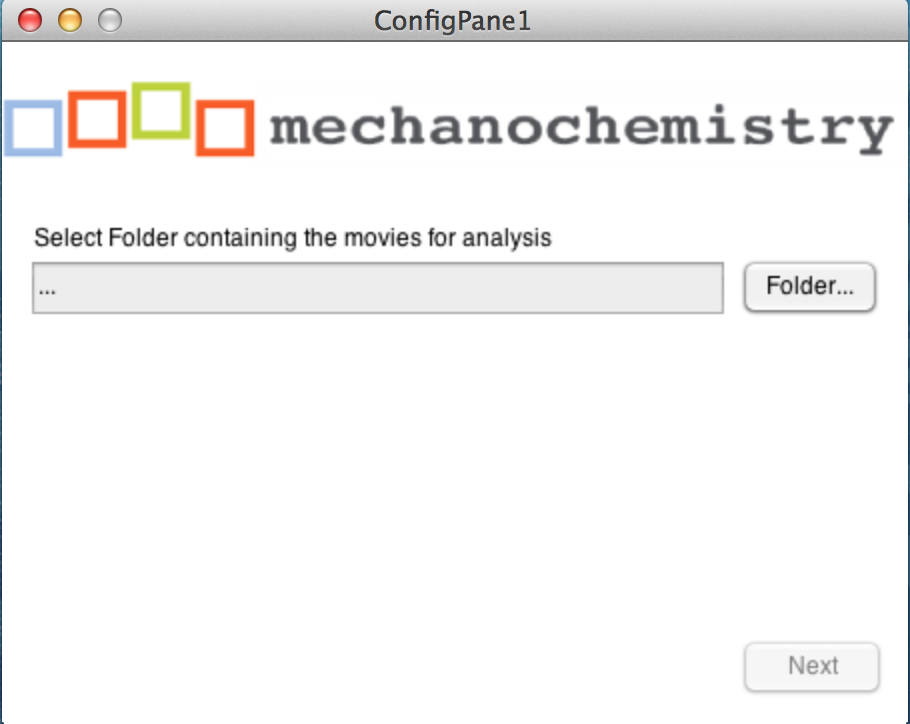
Console.   
How to run a program.

1 Cell Segmentation

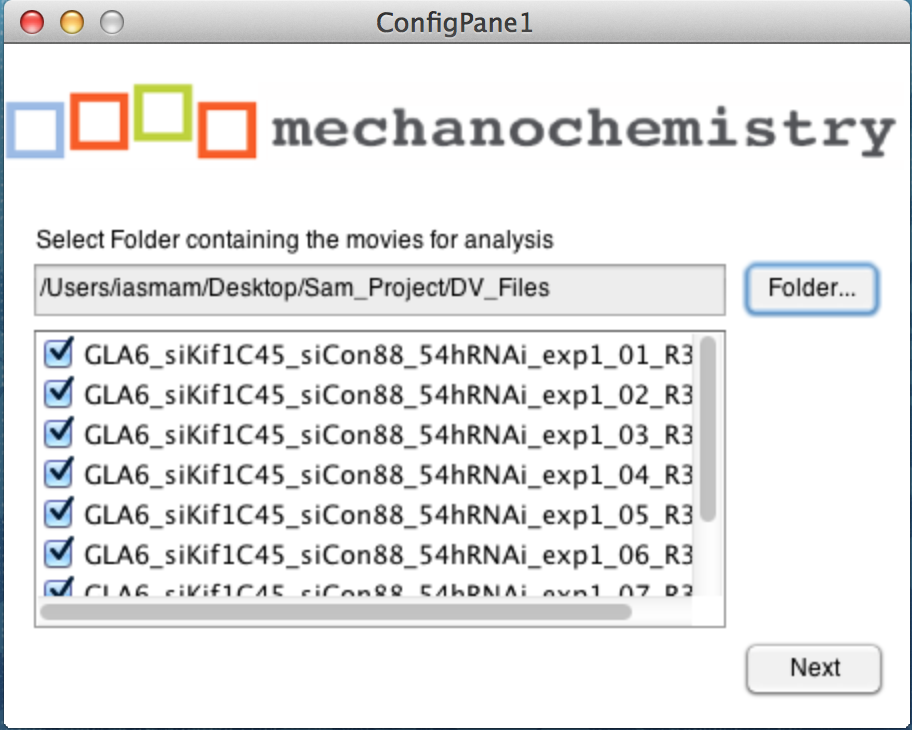
To run cell segmentation, you need to have a folder where your “.dv” files are stored, and a second folder, where you will save all the results of this analysis. In this tutorial, this folder will be called “ the Analysis Folder.”

In this tutorial, the folder is called “Demo\_Analysis”. You can call this folder the way you like, to avoid issues, we recommend to not having white space in the folder name.

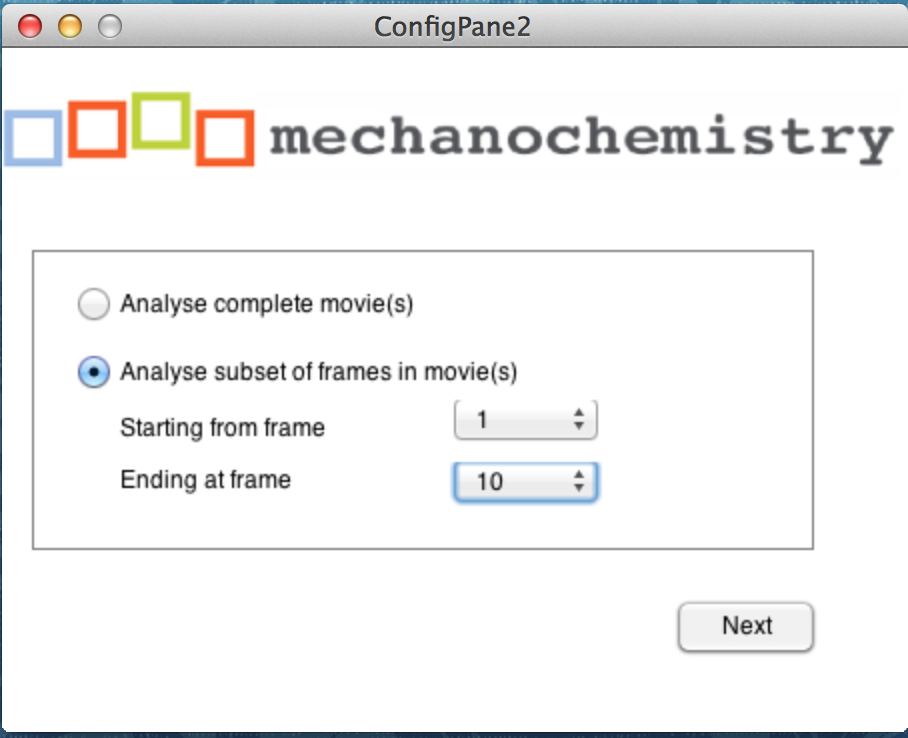
To run cell segmentation, you go to the folder “1-ImageSegmentationFull”, select the file “Run\_CellSegmentation.m” open it, and click on run: you should see the following interface:



There you should select the folder containing your .”dv” files. After selecting the movies, the interface changes to this:



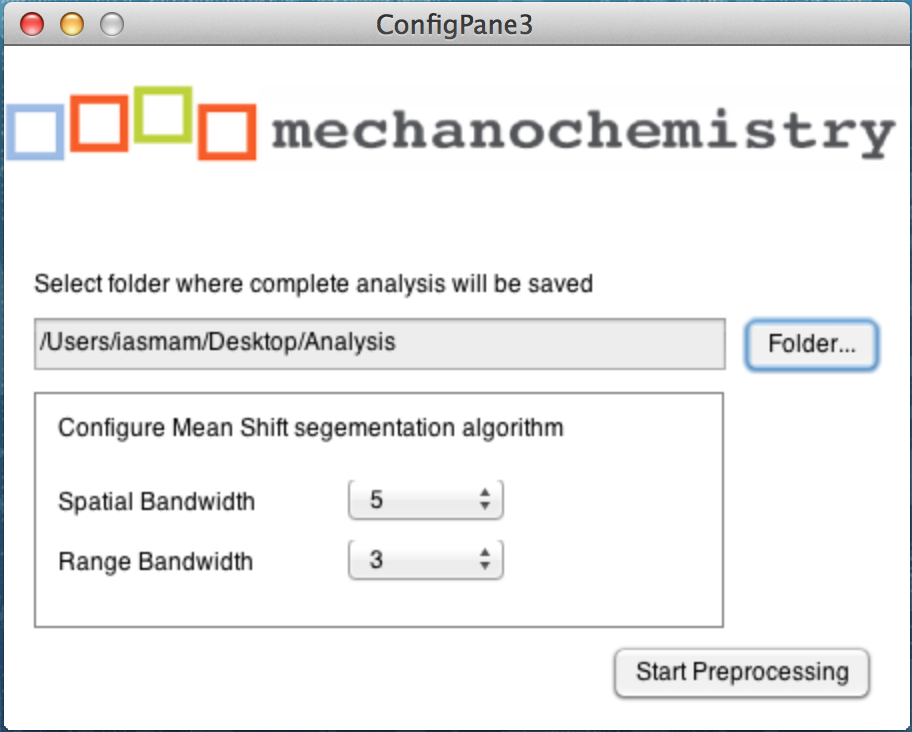
The interface allows you to select the movies you would like to analyze. Please note, that at least one movie files need to be selected to continue. Once you are happy with your selection, you click on the “Next” button, and should see the following interface.



Note, the loading of the interface is slow, so please be patient. It is slow, because it calculates the number of frames of your shortest movie from the selection.

In this new frame, you can select to analyze the complete movie sets from beginning to end. However, sometimes, it might be an advantage to only select a subset, because after a certain amount of frames, the microscope went out of focus, for example. Then you go for the second choice “Analyse subset of frames in movie(s)” and there, you can then define the subset of frames.

Once you are happy with the frame selection of your movie, you click the “Next” button and should see the following interface:



Here you need to select the Analysis Folder, where all the data from this tutorial should be saved. Unless you know what you do, you are ready to start the cell segmentation by clicking on the button “Start Segmentation.”

The cell segmentation is performed using an implementation of the Mean Shift Algorithm, using the EDISON wrapper (**E**dge **D**etection and **I**mage **S**egmentati**ON**). Relevant reference is:   
D. Comanicu, P. Meer: "Mean shift: A robust approach toward feature space analysis." IEEE Trans. Pattern Anal. Machine Intell., 24, 603-619, May 2002.

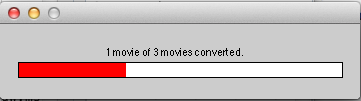
Anyhow, if the segmentation does not work for your, you can play around with the parameter. How these parameter matter can be understood by reading the reference, or Wikipedia: <http://en.wikipedia.org/wiki/Mean_shift>. Overall, we choose good default values, which should work for most cases.

However, the interface permits you to play with the spatial and the range component of the algorithm.

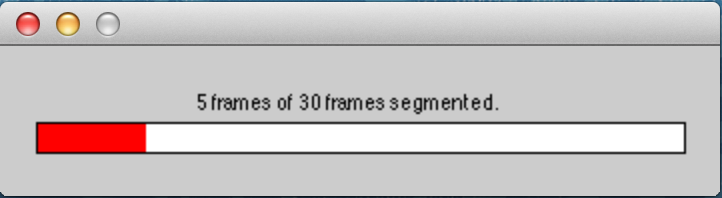
Generally speaking, the range bandwidth influences how much noise the algorithm picks up. If there are many little dots that are not cells detected, than you should increase this parameter.

The second parameter, the spatial bandwidth, influences on how large the radius of a detected shape is, where the selected three pixel (?) is referred to as smallest possible radius. Depending on your imaging conditions and average size of your cell, you can increase your parameter, if your cells are much larger than the example cells in this demo.

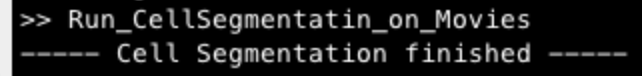
Once you segmentation starts, two progress bar appears. Note this is a slow process and may take considerable time.

First the program needs to convert the movies, and you should see: 

This is followed by segmenting each frame individually.



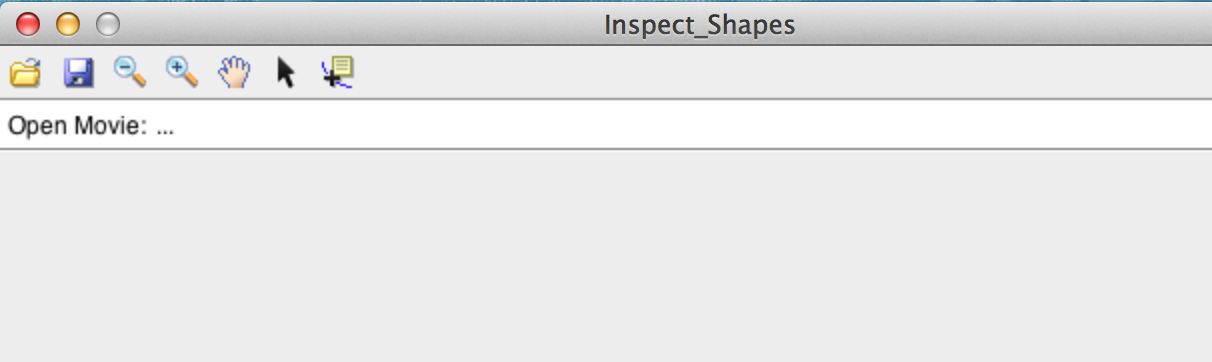
Once the program finishes, you should see the following message on the console:



When you open the Analysis folder, then the following should have happened. For selected file from your movie folder, there should be a new File, that is called “ImageStackXXX.mat”. The first movie in your selection will be saved as “ImageStack001.mat” and so on. For each of this files, a second file with the segmentation information is creates, so for the first movie, this file will be called “ImageStack001CurveData.mat” and so on.

Manual correction of Cell Segmentation

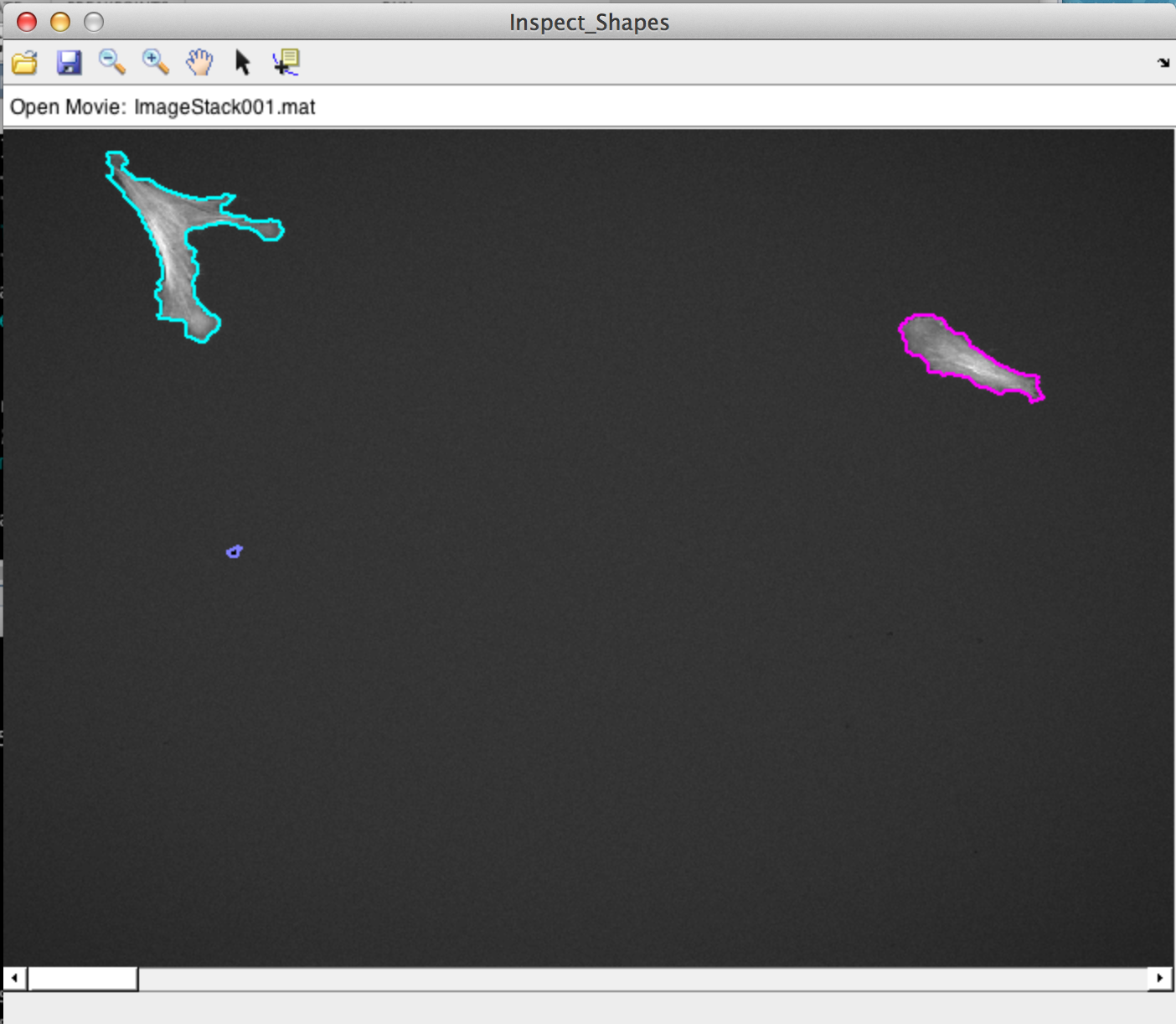
Once the cell segmentation has been finished, you might like to inspect the output of the cell segmentation program. In order to do so, open in the folder “1-ImageSegmentationFull” the program “Inspect\_Shapes” and click “Run” you should see the following graphical user interface:



Each movie you have selected for cell anotation the previous stack, the program has created a file of the following format: “ImageStack001.mat” for the first stack, ImageStack002.mat for the second stack in your analysis folder.

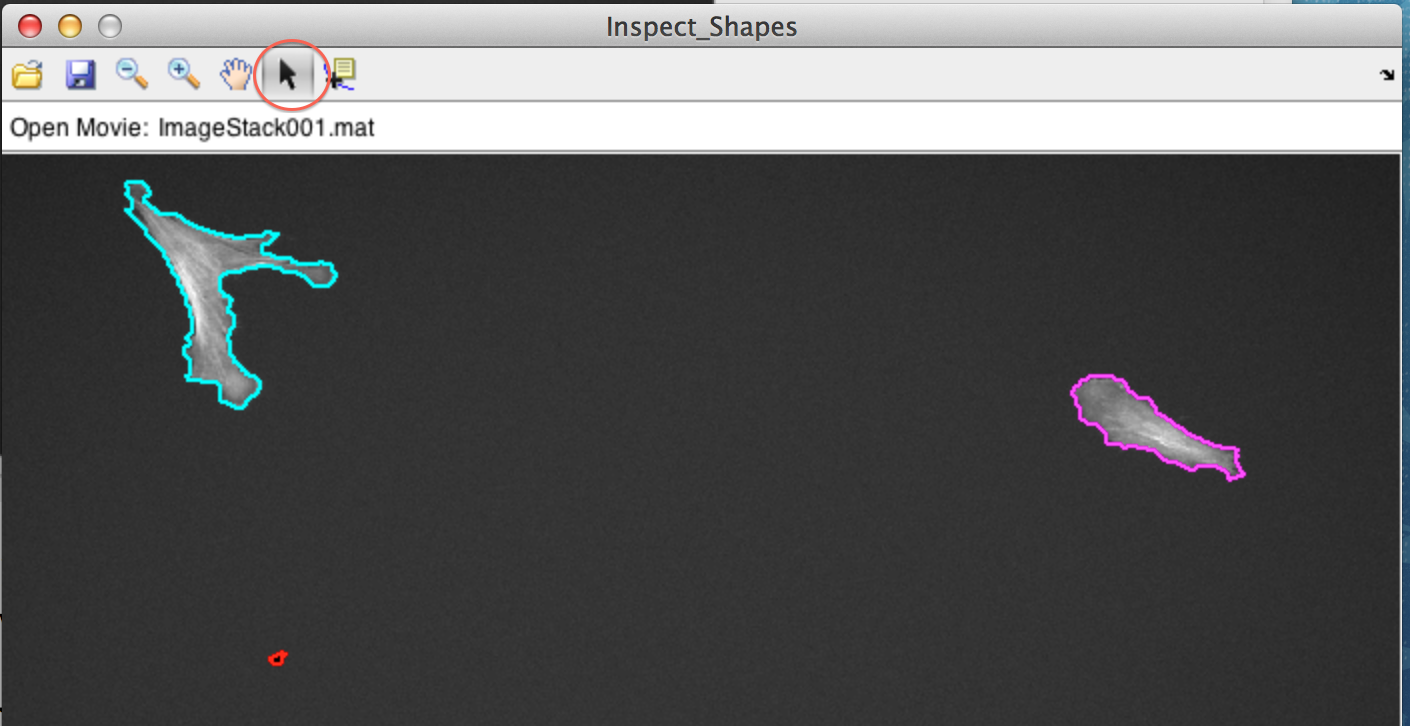
To manually correct, you have to open each of this stacks individually, by clicking on the “Folder” icon in the program, and select for example “ImageStack001.mat”.

After selecting a file, you should see something like the following in the graphical user interface:



Each detected cell is colored differently, and you can scroll the movie up and down to inspect what the program did not select properly.

For example, here we would like to remove the small blob in purple, since it is not properly corrected. To do so, we need to click on the arrow button in the tool bar and then click on the small cell.



Any cell colored in red, will be excluded. In order to remember this new setting, we need to save the modification by clicking on the save icon in the tool bar. If we made an arrow, all we need to do is to click again on a red coloured cell, and then it turns back to normal.

Since removing an error from a movie is a lot of work, we can speed this up by clicking on the cell and pressing the CTR button. Then, this cell will be removed from the current frame up till the end of the movie.

The next step involves opening a guid

In the folder 1-ImageSementationFull there is a file called “CSG\_v2.m”

Open in Matlab this file and start it. You should see:



Click on “New Experiment”

And enter the path, where you have been saving things, in all three boxes.



If you are a programmer, the 3 paths would make sense, or if you want to do some advanced output. Otherwise, if you have just been following this tutorial, you have to put 3 times the same path in.

If you were successful, you should see a dialog window, that should look like this: Important: it should have a number!



# Now to open a stack. You have to enter the experiment number and the number of the stack (see below)

# Macintosh HD:Users:iasmam:Desktop:Screen Shot 2015-02-07 at 14.42.13.png

# If everthing was working out. You should see your stack, with the outline of the cells.



This interface permits you to manually correct things. Say I would like to remove the small detected cell here, I would Shft+Click on it, and then save the data. Saving the data, means that I generate 3 new files for the stack.   
CellArray001.mat

CellFrameData001.mat

Handledata\_exp1.Stack.mat



# Options in the GUI

* You can delete cell shapes (also froma specific point)
* You can merge cells (even with merge point)
* There are instruction on the movie.

You can toggle off the short lived ones, and you can toggle of the small ones. Be careful as the two annul each other. Which is pretty annoying.

You can only load one stack at the time.  
You save them in your folder.

# Once all your files are done

Say, you have corrected all files, then you are ready for the next step: in Folder 1-ImageSegmantation Full, there is the file MakeBigStructs.m

Open it in Matlab and click on the “Run” Button.



There should be an explorer that opens, and you should enter as the directory, the experminet directory you have been using.



After selecting the folder, the program should create 2 new files in the folder, namely Bigcellarrayandindex.mat and BigCellDataStruct.mat (see below).



“2-ShapeManifoldEmbedding”

open the script “RunShapeManifoldEmbedding.m” in Matlab. Click on the “Run” Button in Matlab, and the script should ask you for experiment folder (as before). Select the folder and when the script complets correctly, you should see the following dialog box.



Also, if the program run correctly, you should have the file “CellShapeData.mat” in your experiment folder.

Now you are ready to move to the steps in Folder

# 3 Extended Affinity Propagation

This step will cluster your sample according to the similarity of cell shapes. All files to complete this steps, will be in the folder   
“3-Extended\_Affinity\_Propagation”

The first preparatory stop consists in running the Affinity propagation. To do so, open the file “Run\_Affinity\_Propagation.m” in Matlab and click “Run”. The program should open a File Chooser for your Experiment Folder, the folder where you have saved the files of the previous steps. After selecting the folder, the program should start automatically. Progress of the program can be seen on the console. When the program finishes, you should see the following progress bars.



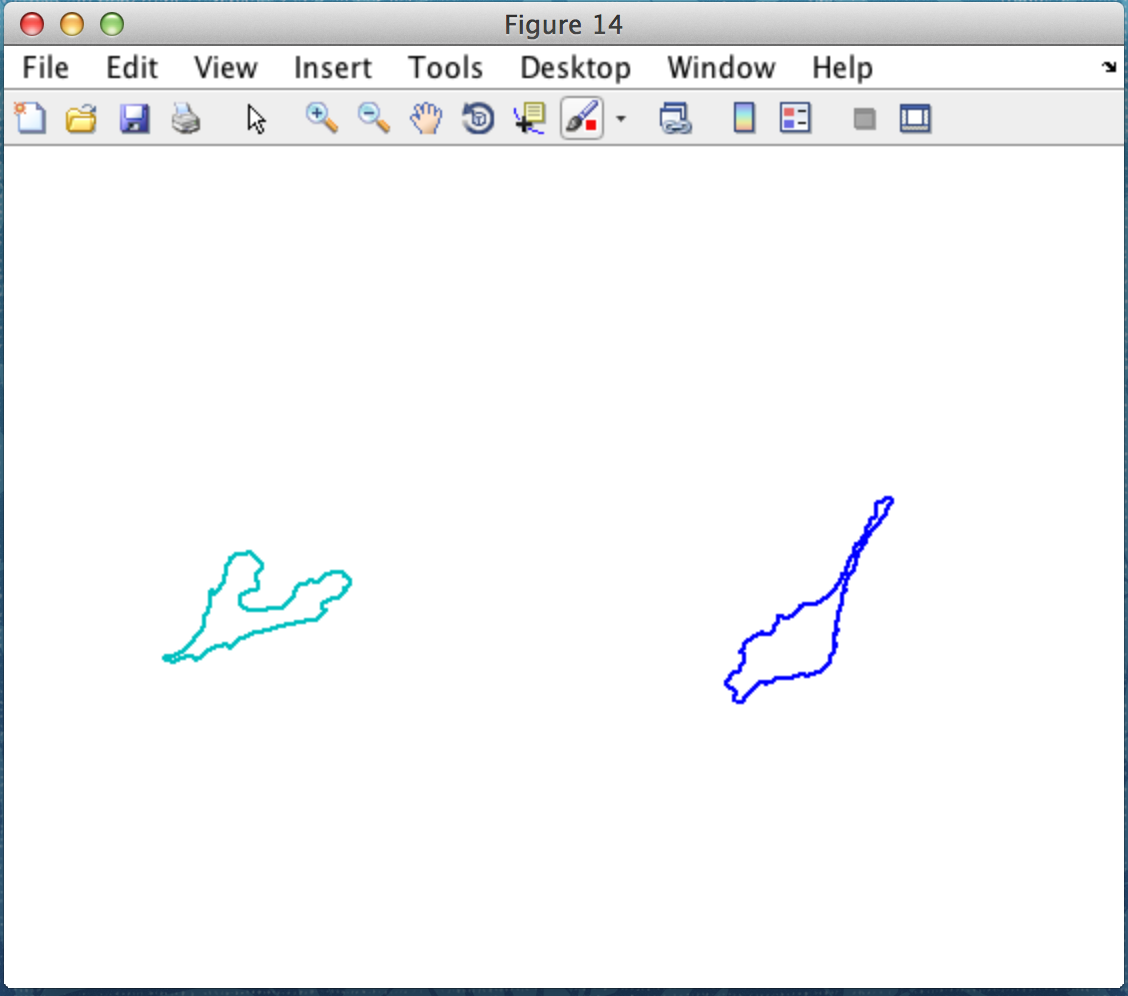
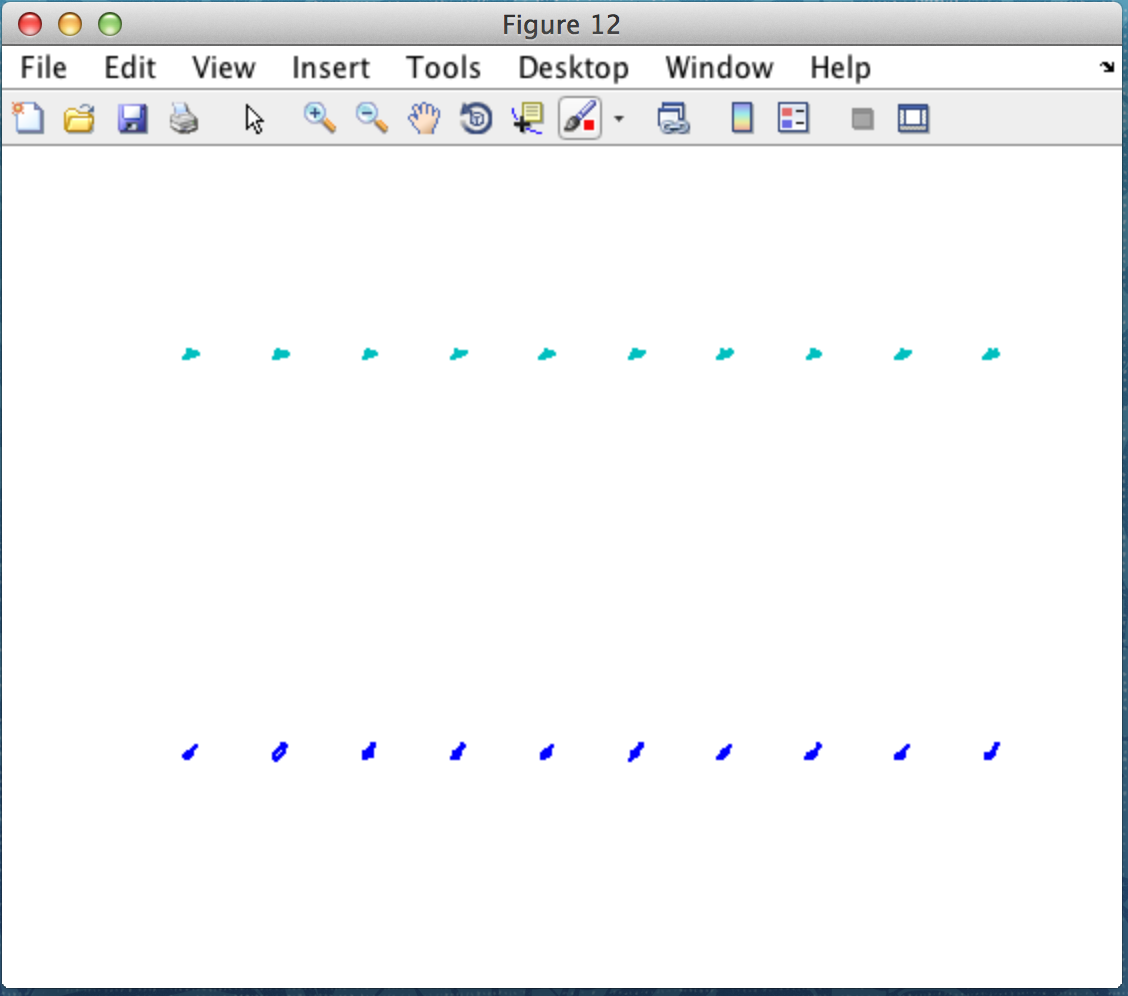
Once the program finishes successful, there should be nine new files in the folder, which should have the following names:



Inspecting the classification of shapes

Now you are ready to inspect the classification of shapes.

Open “Inspect\_Shapes.m” and click on Run. The program should open a Folder Chooser, where you can select the Experiment Folder. After selecting the file, the program should run straight away, and produces the following two files.

The figure on the left shows a representative sample for each cluster, while the figure on the right shows ten representatives for each clusters. The program saves automatically the figures under “Experiment Folder/Figures”

With the following file names:

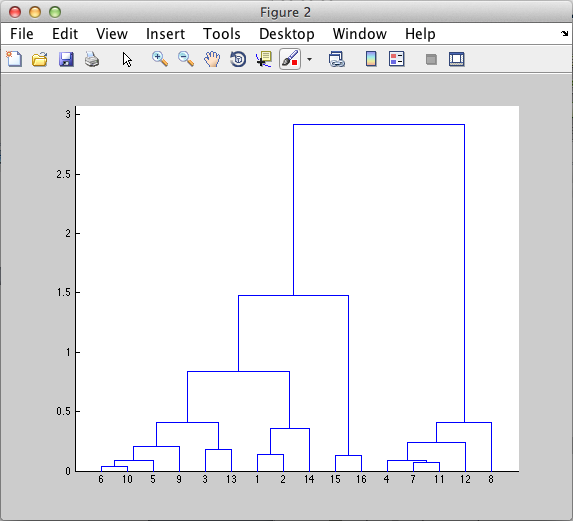
* 3\_AllShapes\_Example\_foreach\_Cluster.fig
* 3\_AllShapes\_Example\_foreach\_Cluster.fig

Dendogram of Shapes

Sometimes, the previous step might produce too many shape classes. You can force the classification of shapes to use less clusters. To make an informed choice, you should to have a look at the dendogram of shapes first.

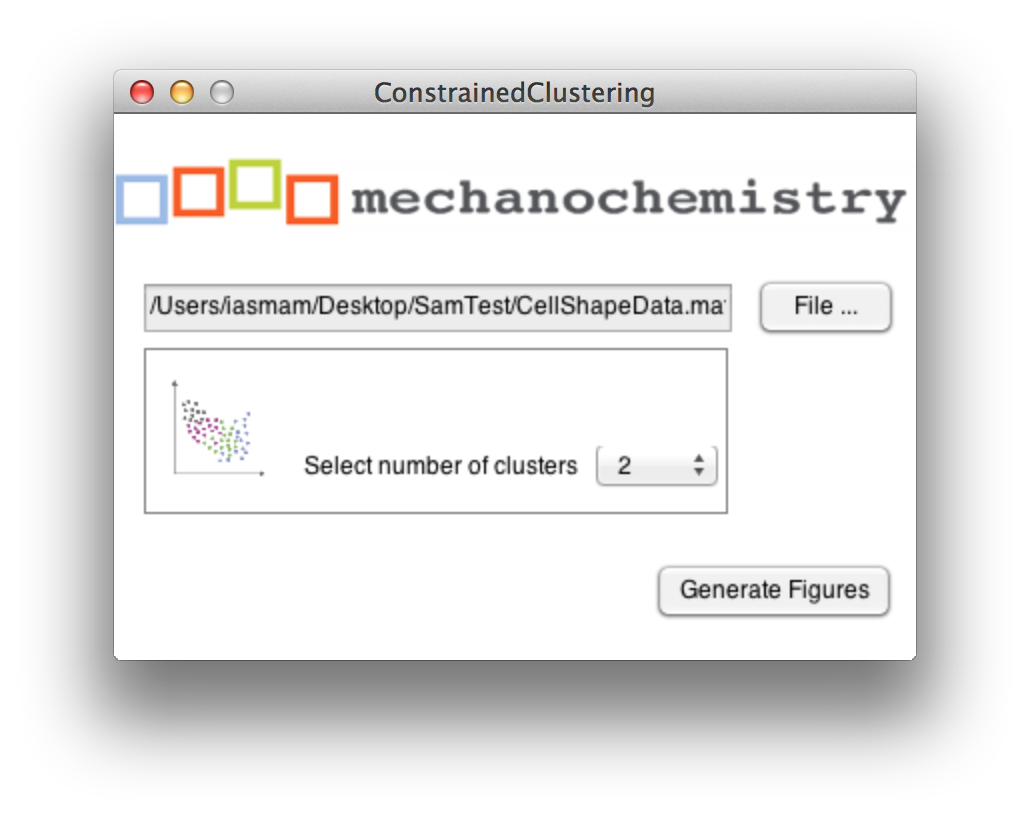
To generate the dendogram of shapes, open “Run\_Dendogram\_of\_Shapes( )” and click on “Run”.

The program should open a File Chooser and you should to select the Experiment folder. Upon successful completion, the program produces a dendogram, which is automatically saved as  
“Experiment Folder/Figures/ 3\_Dendrogram\_of\_Shapes.fig



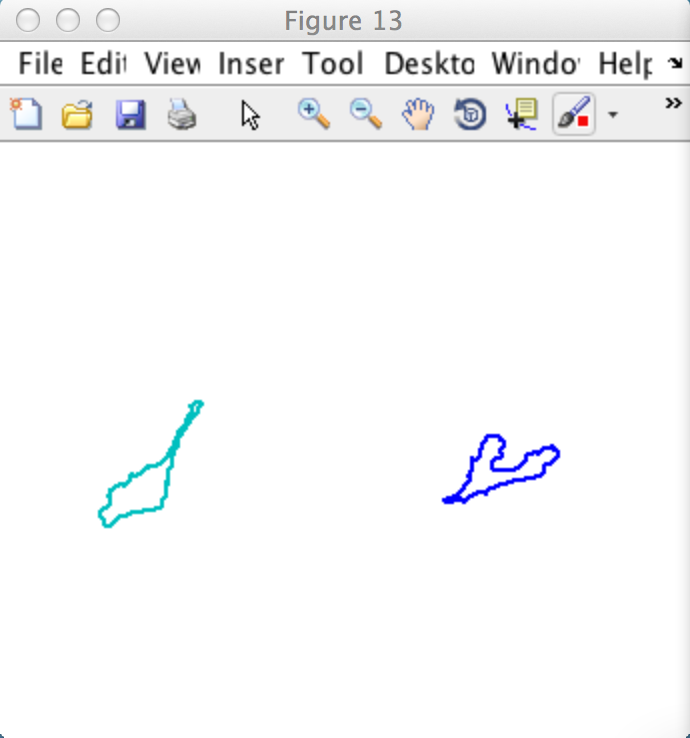
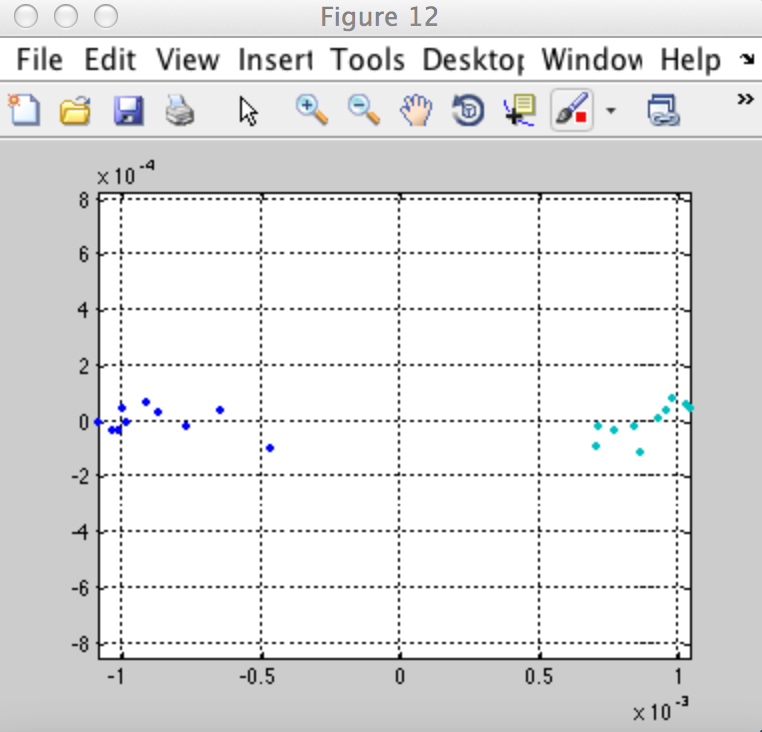
After running the dendogram, you should determine a number that is more appropriate for clustering your shapes. This number is important for the next step.

Then you are ready to run the next program: Open “Run\_constrained\_Clustering()” and click “Run” you should see the following graphical interface:



You need to click on the “File” button and select the file “CellShapeData.mat” in your experiment folder. Then you can select the number of classes you would like to limit you shape clustering. This number must be smaller than the maximum number of clusters you have in the dendogram. Ideallly your choice should be informed by the dendogram you generated before.

After selecting an appropriate number and clicking on “Generate Figure” you should get the following two figure:



The figure on the left shows the coloured clusters in shape space. The figure on the right shows a typical representative for this cluster. These two figures are automatically saved in the Experiment Folder/Figure/ with the following in the following files:

* 3\_Avg\_Shape\_for\_Clusters.fig
* 3\_Coloured\_Shape\_in\_ShapeSpace.fig

# 4 Shape Slicer

To run the shape slicer, you need to go to folder “4-Shape\_Averaging”, open the file “Run\_SpaceSlicer”, and click on the “Run” button in Matlab. You should then see the following graphical interface. You click on the button “File” to select the Matlab file “CellShapeData.mat” from your experiment Folder. Then you click on the “Generate Figure” to run the program.

The graphical interface allows you to configure the slices on the x and y axis, and whether you would like to force the two aces to be on the same scale.

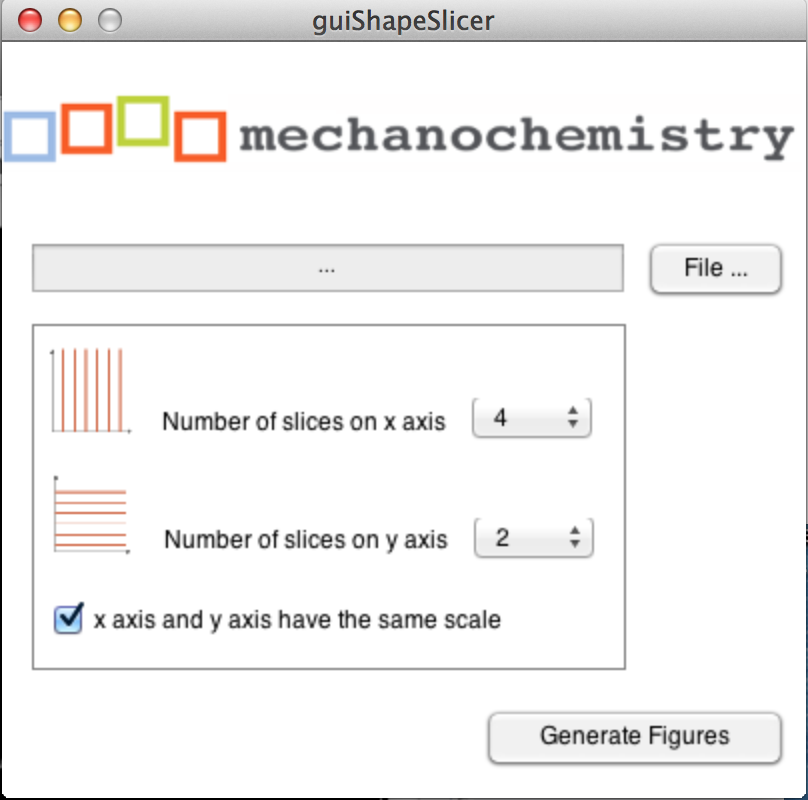


Figure Shape Slicer GUI: Select CellShapeData.mat and configure the slices for the x and y axis.

When the program runs successfully, you should receive four different figures as output. One of those figures should look somewhat similar to the figure below. If you click on the figure under File 🡪 Save As… you can save the figure in vector format and modify with any graphic program.

Note: The program automatically saves in your “Experiment Folder🡪 Figures” all the figures that are generated. However, they are overwritten each time you generate new figures. All four figures begin with a name “4\_ShapeSlicer\_\*\*\*” so that you know they are from your fourth step in the process.



Figure Sliced Shape Space: blue figures indicate average space in slices along the x-axis, green figures indicate average shape in slices along the y-axis.

However, the program also produces the single parts of the programs individually, e.g. the average figures along the axis as well as only the content, permitting maximum flexibility in graph creation.



Figure : Partial output: left average figure along the x-axis, middle average figure along the y-axis, and shape in the sliced grid.

# 5 Out of sample extension

First step: create a folder called OOSE in the Experiment Folder.

You also need to have done all the steps from 1 to 4, and have an experiment folder with all necessary files.

* Run\_CellSEgmentation (as described in Section) but save all the results in the OOSE folder.
* Correct Data manually, using “Inspect Data” as described before.
* Run “Make\_Big\_Structure in the OOSE folder.

These are the preparatory steps before you are ready to start.

LP\_OoSE\_train(CellShapeData, '/Users/iasmam/Desktop/SamTest/OOSse' )

CellShape Data comes from CellShapeData.mat from the big training set (not from the small set!)

After this, we have generated in OOse, the file: LP\_trained.mat



The structure insiside LP has a Nx5 matrix. N is the number of frames we analised.

Next step, we run LP\_OoSE\_run;

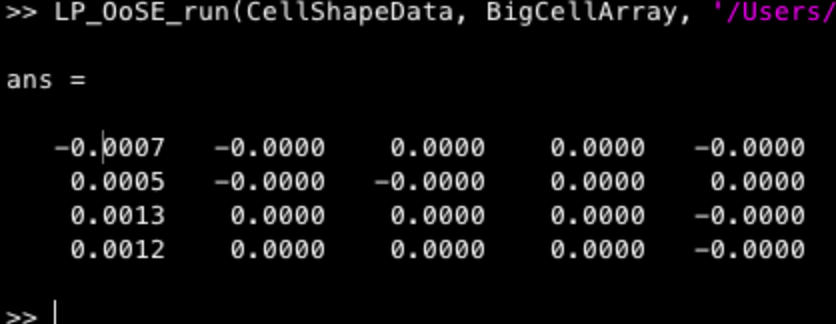
* Where CellShapeData comes from the big set
* BigCellArray comes from the new possible small set.

LP\_OoSE\_run(CellShapeData, BigCellArray, '/Users/iasmam/Desktop/SamTest/OOSse/LP\_trained.mat', '/Users/iasmam/Desktop/SamTest/OOSse/' )

Your console look like this.

And you should have this new files in your folder

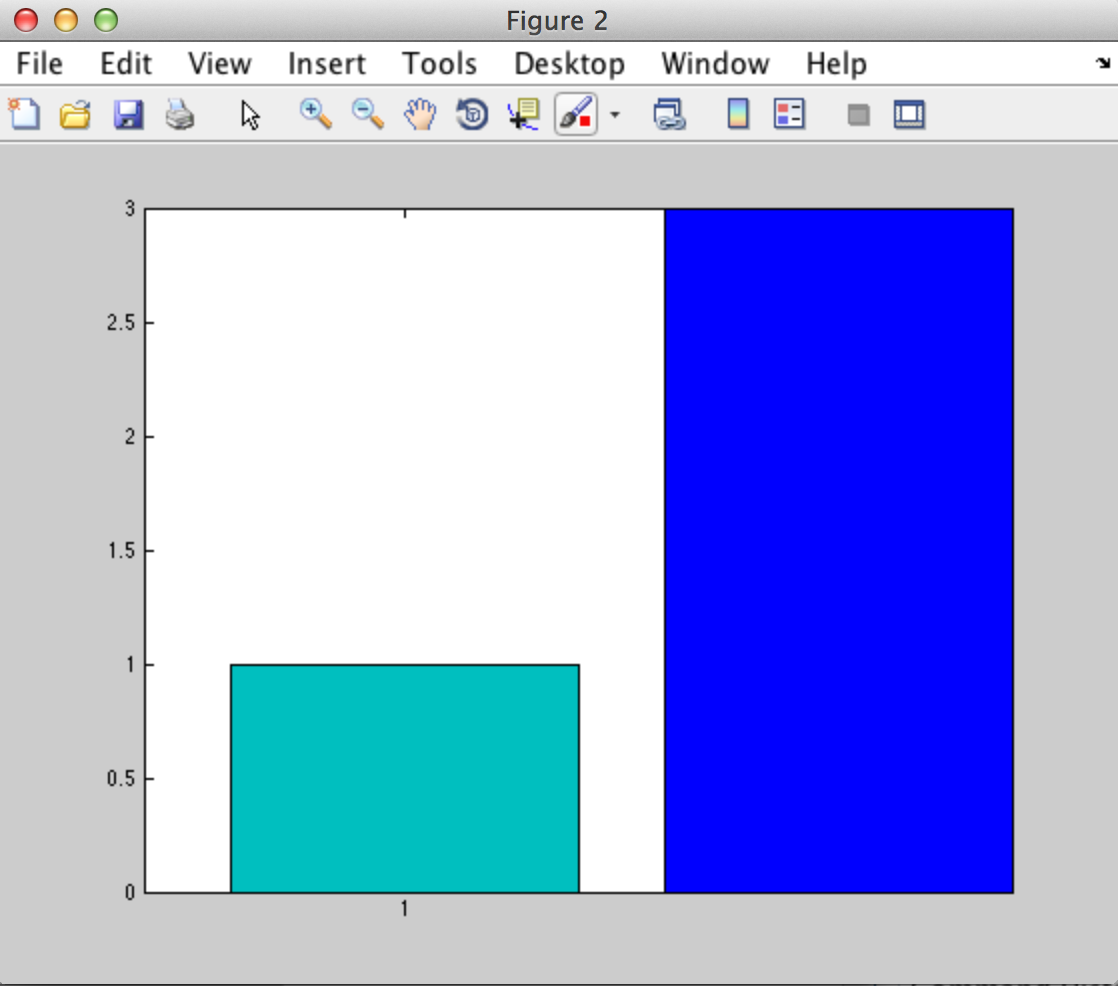
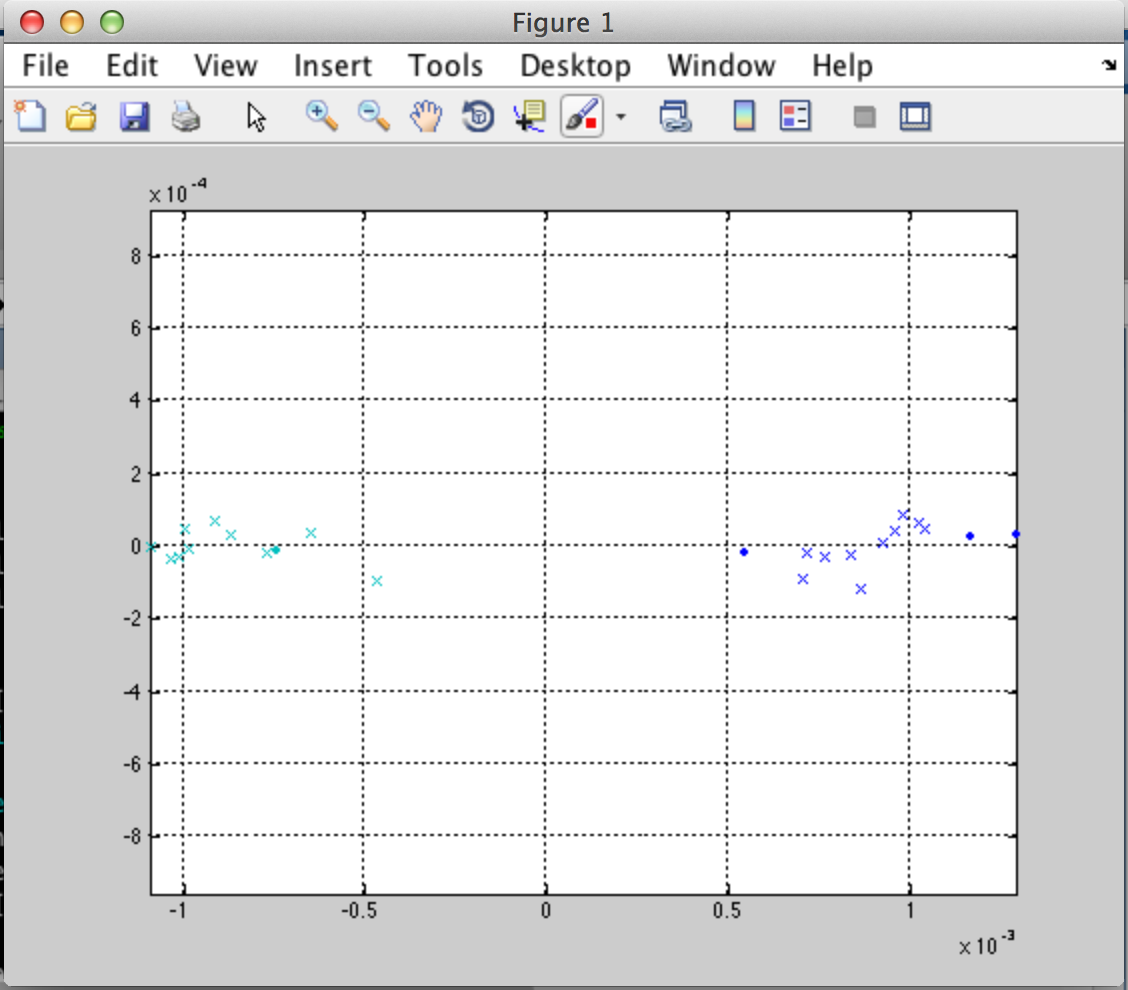
* Dist\_mat.mat
* OoSE\_embedding.mat



Now you are ready to the 3rst step:

To get it to run, enter the following:

OoSE\_bargraphs(2, CellShapeData, '/Users/iasmam/Desktop/SamTest/', '/Users/iasmam/Desktop/SamTest/OOSse')



Now, we see with the crosses the two clusters of shape we have had from the big set. However, the two dots (two green and two blue) are the points that came from the out of sample addition. 3 of the new ones are blues, while one is a greenish one.

# 6 Out of sample extension

Run first MorpProps as for example with

Morph\_Props(BigCellArray, '/Users/iasmam/Desktop/SamTest')

BigCellArray comes from opening “Bigcellarrayandindex.mat” from the big training sample.

This process can be close. After running this process, we have a new file in the folder.

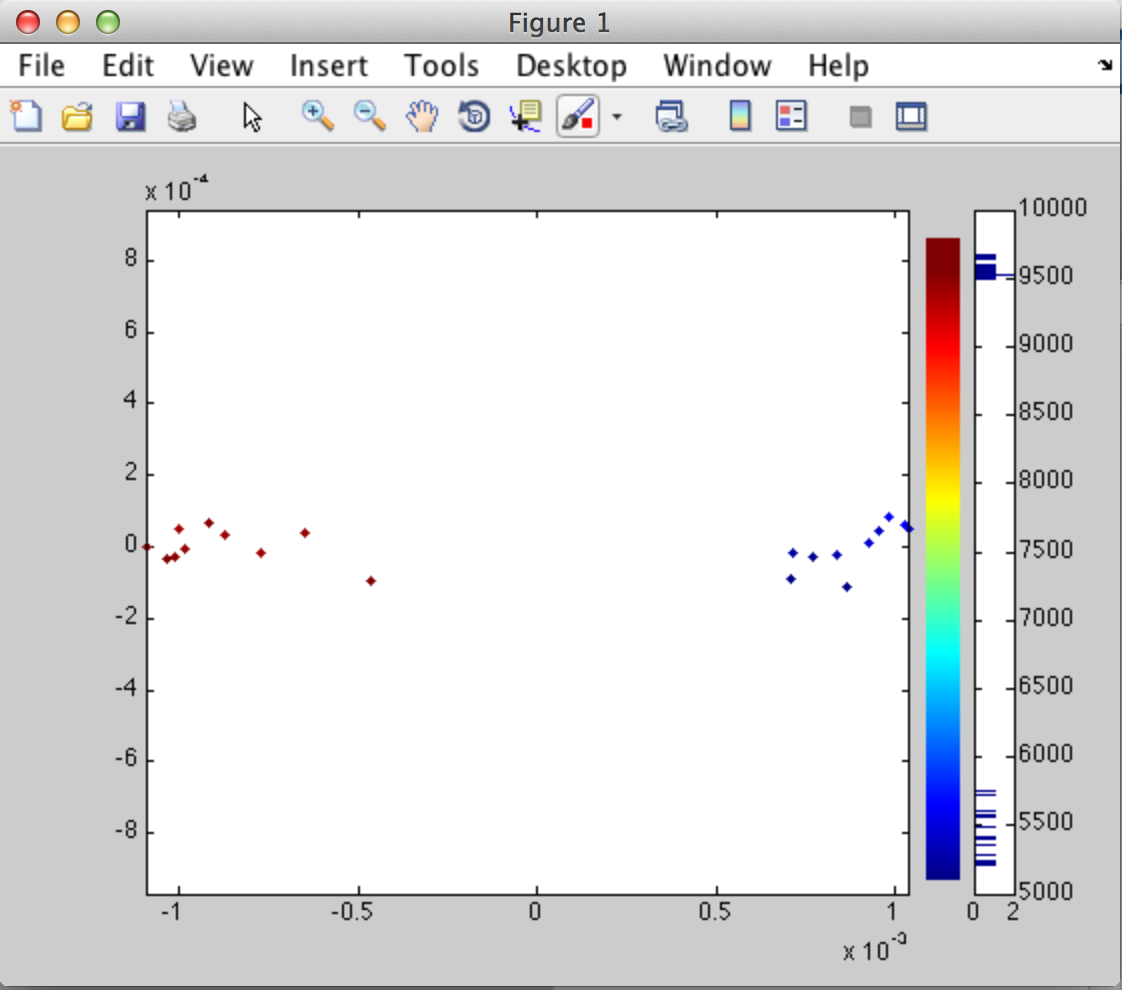


Now we are ready to produce a graph with the following command

Prop\_display(CellShapeData, morphframe, 'Area')

Where

* CellShapeData comes from the big set
* Morphframe from previous operation
* ‘Area’ is one of the possibilities we can select and get



# Step 7 Shape Space Dynamic

To start it, you type for your example this:

ShapeSpaceDynamics(cell\_indices, CellShapeData, '/Users/iasmam/Desktop/SamTest')

* cell\_indices is from the big sample from the file “Bigcellarrayandindex.mat”
* CellShapeData comes from the “CellShapeData.” Also from the big data set.

After this operation we have a new file, DynamicData.mat



in the structure we get we have the following fields:

* Track (is track per cell)
* Speed (is the speed of the cell)
* Average speed is the average speed per tack
* Angles is the angle per cell
* Av\_displacement\_direction : average displacement direction the cell does in its track.

If we have N cells segmented, we should have N structures in this array.

So the first plot we can generate is with this line

Local\_direction\_rose\_plots(DynamicData)

We can influence in how many slices we want to cut the data. “Nbins=[10 6];” in the source code is the important line.

We get than this graph: This shows the local directions in the grids.

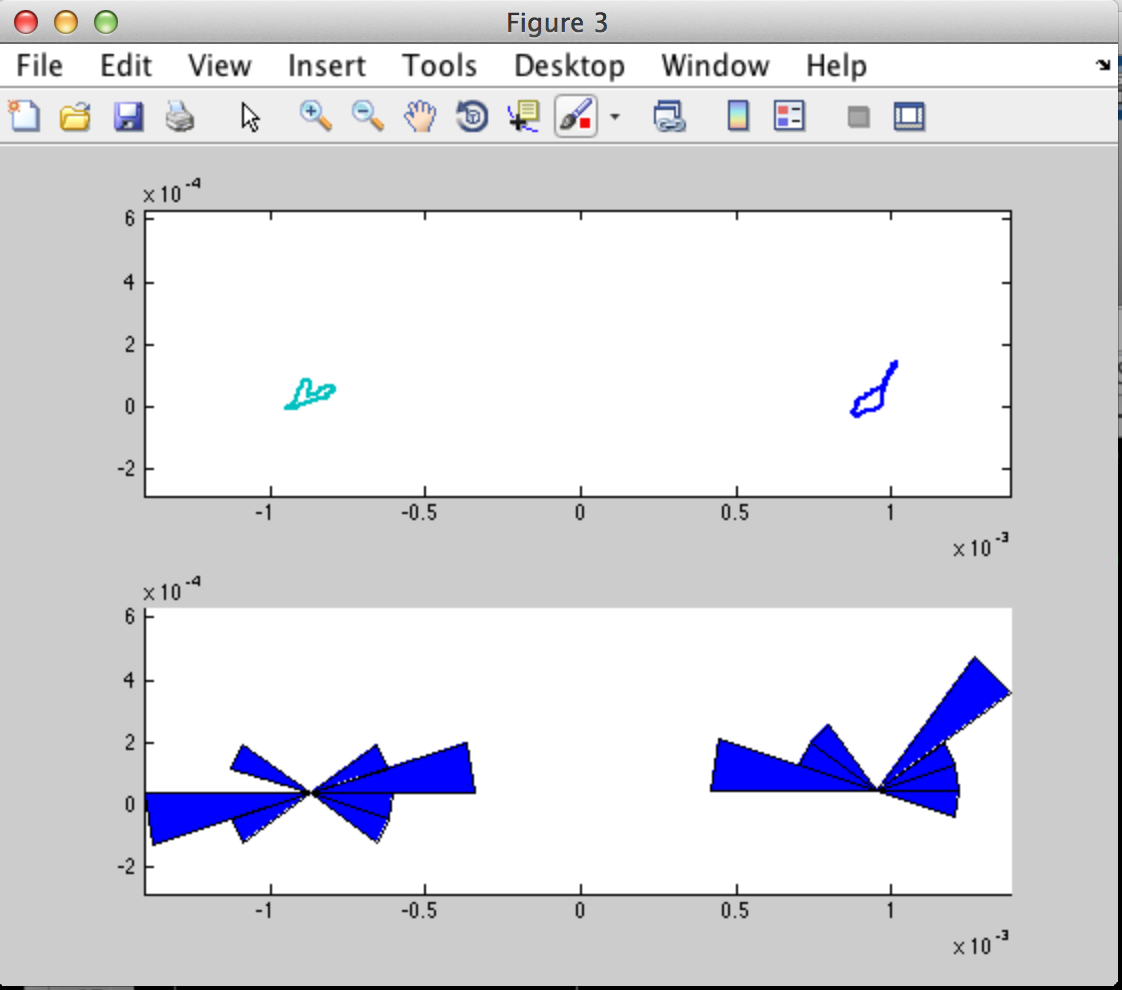


The other figure we can create is by typing this:

Exemplar\_direction\_rose\_plots(DynamicData, CellShapeData, 2, '/Users/iasmam/Desktop/SamTest')

* Dynamic Data comes from Folder 7 first step.
* CellShapeData (big sample)
* 2 is the number of cluster, we have determined this in step 3, by looking at the dendogram.
* Foldername, to get the AffinityDatasample files, in detail, they are
  + APclusterOutput.mat
  + Wish\_list.mat
  + Linkagemat.mat

The graph we get is the on in the bottom.



Things to explain at the beginning.

* The experiment folder.
* Run a program in matlab
* Console in matlab.
* What to do if program does not finish as expected.
  + Do you use a matlab version that we have tested (see list)
  + Do you have followed the step as before and used the guidelines in where to place your files
  + Ask a technical person in your team, ideally, someone who is a matlab programmer
  + Ask the authors.
  + How to save figures in matlab in any format you like.
  + File Chooser