AutoCell demo

[Installation note: folder ‘AutoCell’ and the 3 subfolders ‘brick’, ‘explor’ and ‘colormaps’ should be added to the Matlab path]

Load the demo data in Matlab:

```
>> load demoAutoCell
>> whos x
  Name       Size                Bytes  Class     Attributes
  x        120x120x280          32256000  double
```

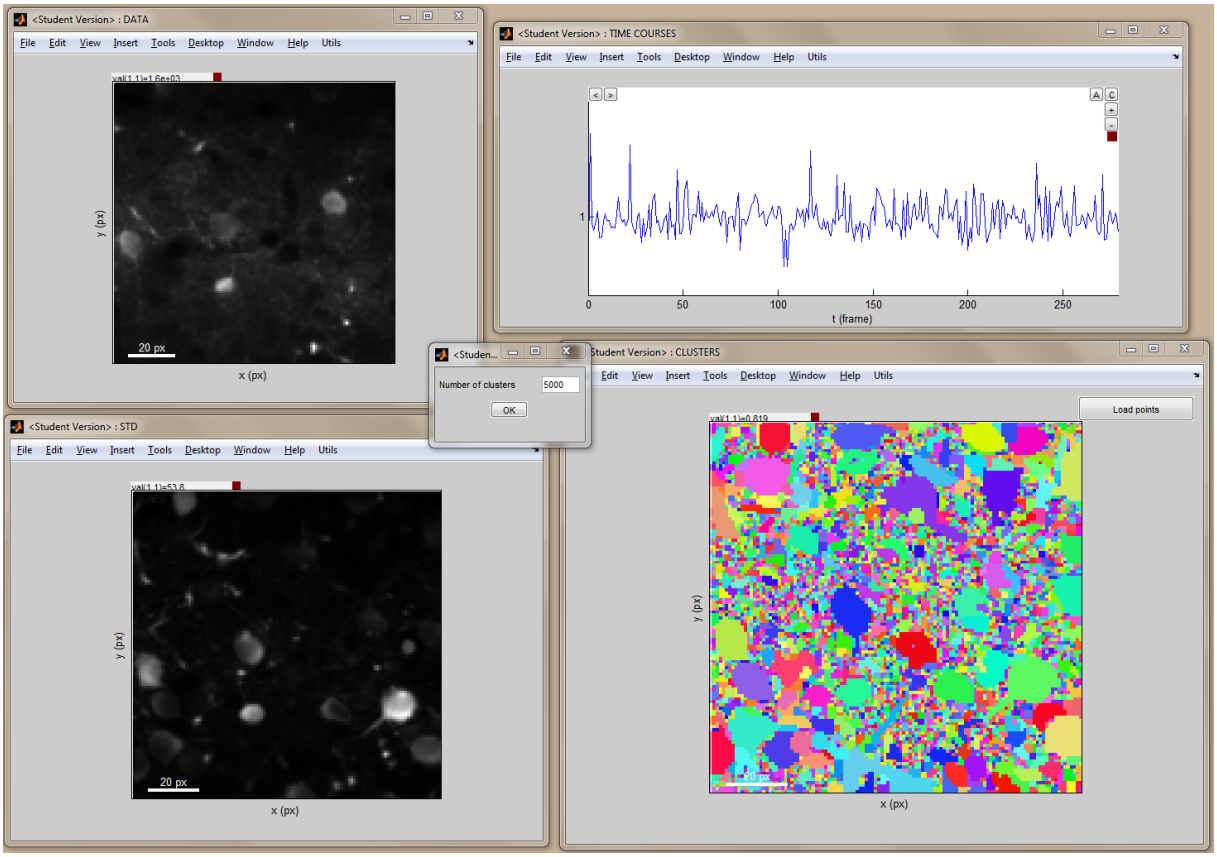
x is a movie made of average frames from the experiment: each of the 280 frames of x is actually the average over one 8s trial: indeed, whereas AutoCell can in principle be applied to any movie, we usually apply it to such movie spanning the full experiment, with some temporal averaging over blocks of 5 to 10s to keep the total number of frames as well as the photonic noise in individual frames small.

In addition here x has been cropped to only a subpart of the imaging field so that this demo data remains small.

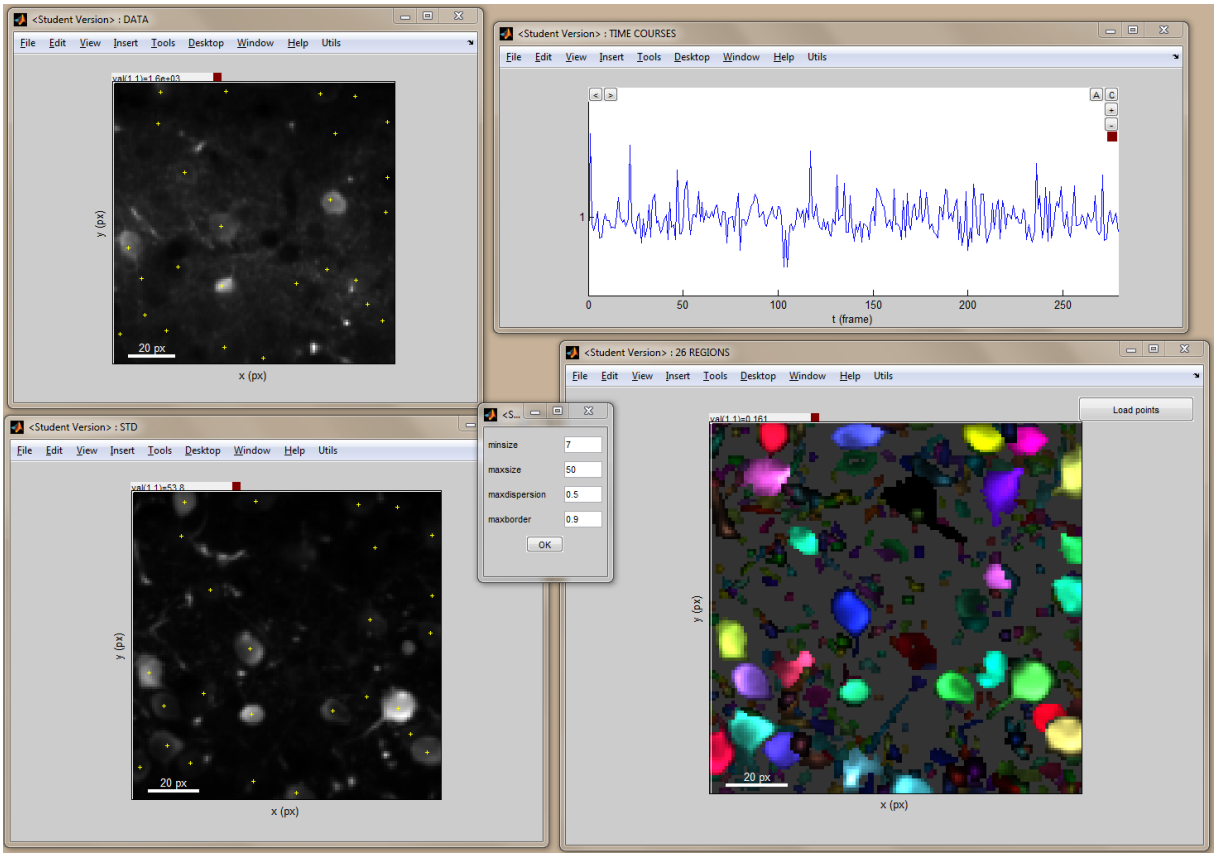
Launch the ROI detection:

```
[ROIprecomp ROIpar A] = AutoCell(x);
```

Several windows appear, the steps of the clustering can be visualized in the ‘CLUSTERS’ window. Then user is prompted to select the number of clusters. Different values can be tried. On this reduced dataset, 5000 is an appropriate value (press OK twice to proceed).



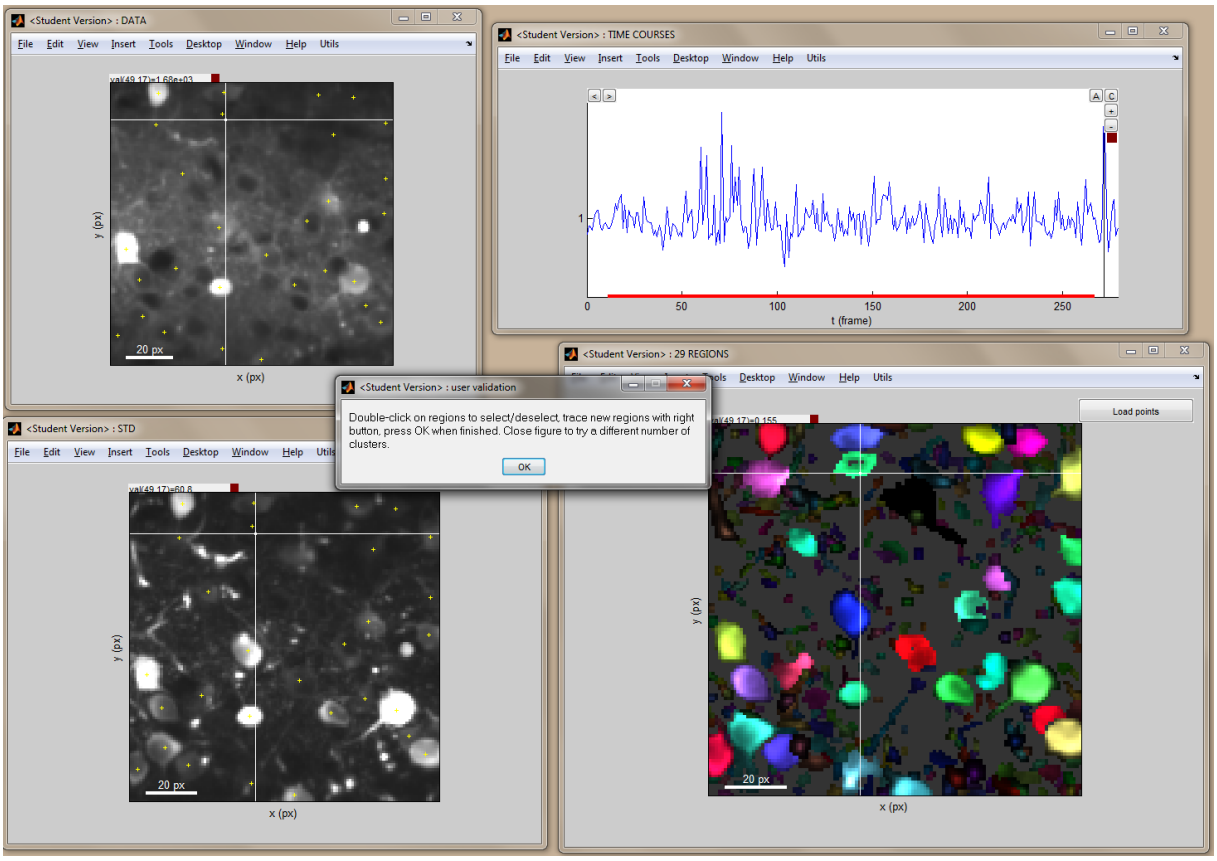
Next, the program compute pixel weights in individual regions and propose a suggested subselection of which regions are likely to be neurons. The parameters for this automatic subselection can be edited (for example here, setting a minimal size of 7 pixels instead of the default 9 results in slightly more neurons to be selected). Press OK again to proceed.



Finally, this subselection can be manually edited by simple clicking the pixels in any of the windows. The graphic tools offer many possibility to navigate through the data in order to visualize cells and their activity.

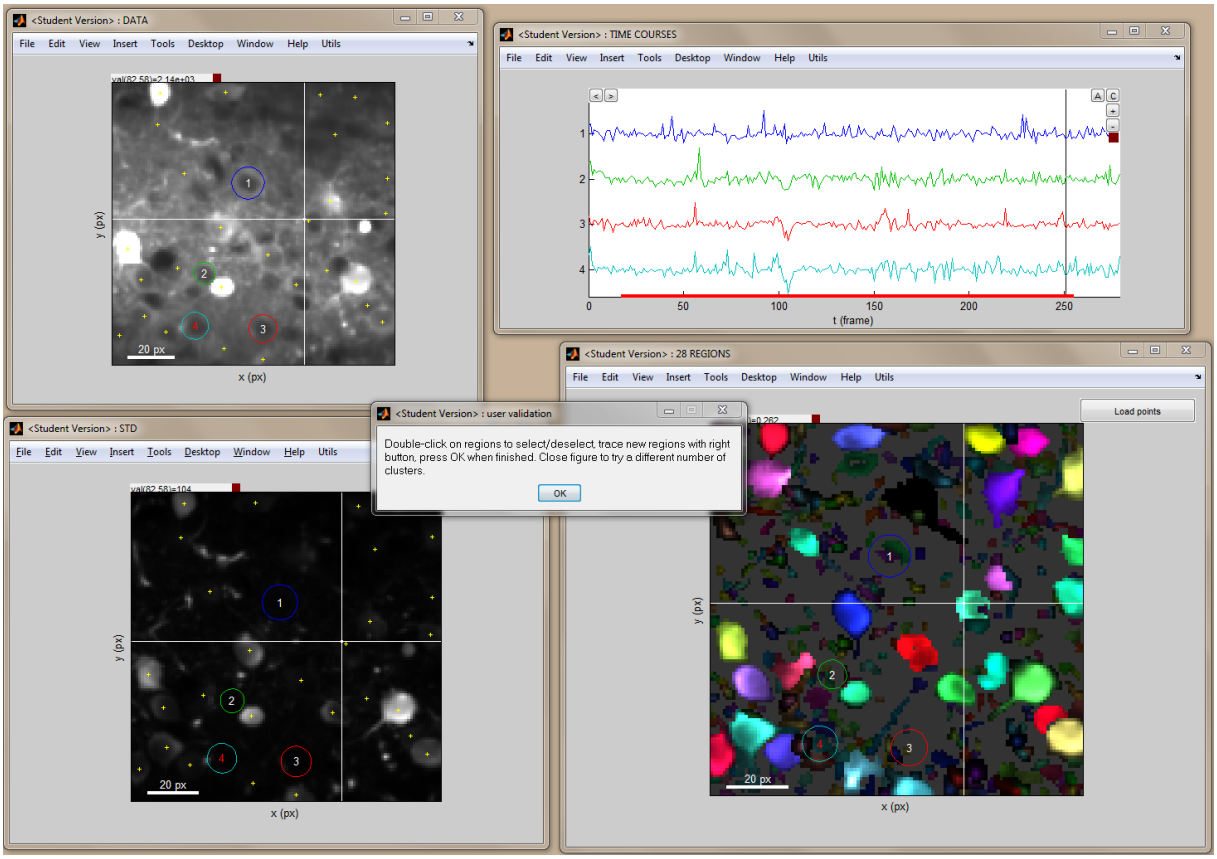
For example here an average (rather than single) frame is shown in ‘DATA’ window, which has been previously defined by a temporal region selection in the ‘TIME COURSES’ window (appears in red in this window; click and hold the mouse right button to define such region).

As another example, the clipping range has been adjusted in ‘DATA’ and ‘STD’ (click and hold the mouse left button on the small red square in the top of each of these displays).



Noticeably, some neurons that can be seen in the ‘DATA’ window were not detected by the algorithm. The reason is that these neurons have no or only very little activity: indeed they do not appear in the ‘STD’ window. The graphic interface allows however to select them by drawing regions (click and hold the right button, in any of the 3 image displays). Note that I usually do not use this feature as I am not interested by the neurons that most probably do not respond to the stimulations.

When finished press OK. If some regions were selected manually, answer YES to the question ‘You selected regions manually. Do you want to keep them as additional neurons?’.

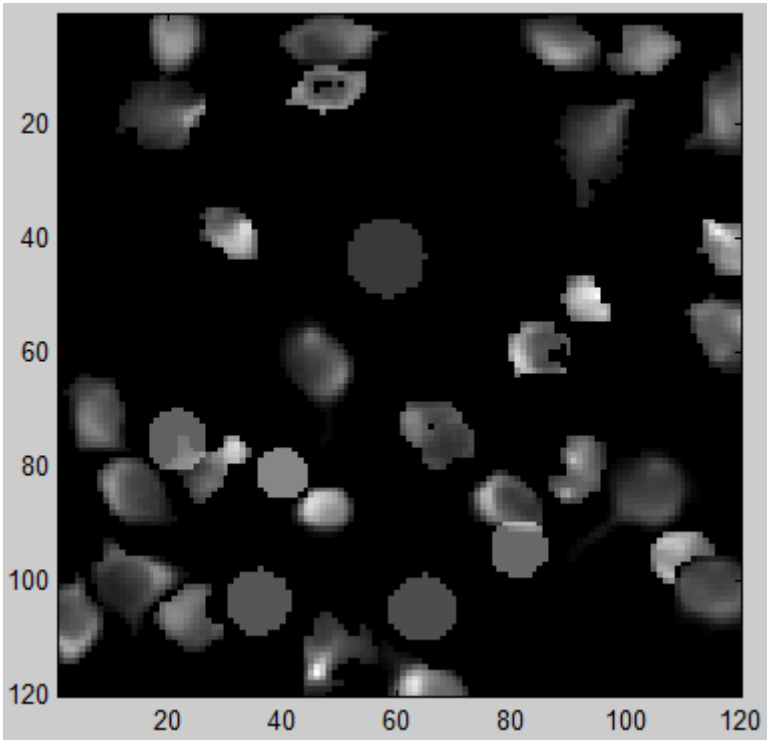


The AutoCell function returned 3 outputs. The ROIprecomp and ROIpar structures gather respectively the clustering information and all the parameters that were possibly edited by the user.

```
>> ROIprecomp

ROIprecomp =

    tree: [2x14400 double]
   cdist: [14400x1 double]

>> ROIpar

ROIpar =

      clustering: [1x1 struct]
       shapepar: [1x1 struct]
       okneuron: [5000x1 logical]
  manualneuropil: []
   manualneurons: {1x6 cell}
```

The output A is a sparse array of size (number of selected regions) x (number of pixels in the image). Each row A(:,i) is the mask of one region, as can be seen in the following display of the sum of all masks. Notice the difference between automatic and manual neurons: the later do not have pixel weights and possibly intersect with other regions.

```
>> whos A
  Name      Size              Bytes  Class     Attributes
  A         35x14400          101752  double    sparse
>> [nx ny nframe] = size(x);
>> figure(1), imagesc(reshape(sum(A,1),[nx ny])'), axis image, colormap gray
```



Now, to extract the signals of individual neurons from experimental movies, a simple multiplication with A is needed!

For example if variable ‘data’ is the data from one trial (here it has 159 frames), the signals from individual neurons are obtained as follows:

```
>> whos data
  Name       Size                Bytes  Class     Attributes
  data     120x120x159          18316800  double
>> signals = A*reshape(data,[nx*ny size(data,3)]);
>> whos signals
  Name        Size            Bytes  Class     Attributes
  signals     35x159           44520  double
```