Deep Brain Motion Correction

%%

Ben Engelhard, Princeton University (2019).

%%

´

Overview: This code runs a motion correction algorithm designed for use on movies generated in deep-brain calcium imaging. A complete description of the algorithm is provided in the paper (Methods section and legend of Extended Data Fig. 3). Here, we provide instructions for using the code.

Notes

1) In the paper we used a version of the code optimized for running on the cluster of the Princeton Neuroscience Institute. Here, we provide a stand-alone version of the code which can be run in any computer that has Matlab. As a consequence, runtime could be slow for large datasets. If you have access to a cluster, please read the section on **optimization** for details on how to optimize the code to your specific cluster.

2) Apart from Matlab, you will need the ImageJ software for generating the patches ROIs (used for motion correction) and the neuronal ROIs (for extracting the traces after motion correction).

3) This code has been tested to work on unix and windows.

4) This code has been tested on datasets where each tiff file was a recording of 50 s of data at 30 Hz, and each frame was 512 by 512 pixels at 16 bits. Other configurations may or may not work.

5) During motion correction, movies will be temporally downsampled by a factor of 2 in order to increase the signal.

Installation
Save all files to your local computer and add the folder to the Matlab path.

Usage
There are two commands that can be used: the first performs the motion correction algorithm:

```
DBMC(input_folder,output_folder,have_red_channel,use_red_channel)
```

The second extracts the neuronal traces after motion correction has been performed and the neuronal ROIs have been saved:

get_neural_traces(output_folder)

Motion correction

The code will motion correct all tiff files in the input directory ('input_folder' argument), and will assume that they belong to a contiguous recording where the order of the name sorting of the tiff files is the same as the temporal order of the movies. The 'output_folder' argument is the directory where all data will be saved. The 'have_red_channel' and 'use_red_channel' arguments are Booleans which denote if the red channel exists, and if it should be used for motion correction (as opposed to the green channel). It is assumed that the red channel is every second frame of every movie.

In order to motion correct, the following steps should be taken:

1- Run the main function: DBMC(input_folder,output_folder,have_red_channel,use_red_channel)

2- After the first phase of motion correction, the file 'template_mov.tif' (and the file 'template_mov_green.tif' if using the red channel) will have been saved in the 'output_folder' directory. Now the user should draw patches and save them using ImageJ ROI manager (this is the only manual step of the motion correction procedure). Each patch should be a contiguous area of the 'template_mov.tif' movie (or the 'template_mov_green.tif' if it exsits). After drawing all patches, they should be saved in a zip file (in the 'output_folder' directory) named 'patches.zip'. Use the following tips for drawing the patches:

   a- Patches are typically 80-160 pixels wide.
   b- Patches typically include 1-5 visible neurons.
   c- Choose neurons that 'drift together' in a patch, and avoid neurons that 'drift apart'. To assess that, you should cycle though the 'template_mov.tif' and look at which neurons are moving together through time, and which are not.
   d- Patches should be based on the first frame of the 'template_mov.tif' file.
   e- Only patches that are drawn will be motion corrected, so be sure each relevant neuron (or process) is inside at least one patch.
   f- Depending on the data, sometimes patches should be larger or smaller. Larger patches have better signal, but may include neurons or processes that do not drift together, whereas smaller patches have the opposite problem. Draw patches of different sizes initially to test which size better works for your data.

3- After drawing the patches and saving them, run the main function again: DBMC(input_folder,output_folder,have_red_channel,use_red_channel)

4- Once the program finishes running, the 'output_folder' should have the motion corrected movies corresponding to each patch. These are saved in tiff files named 'mc_image_stack_full_patch_X.tif' where X corresponds to the patch number. If the movie is large, this might be saved in parts that are smaller than 4 GB, in which case they will be named 'mc_image_stack_full_roi_X_partY.tif' where Y is the file order.

5- In addition to the corrected movies, a folder named 'ds5_files' will be created in the 'output_folder' directory. This folder will contain temporally downsampled versions of the motion corrected files (by a factor of 5) which may be useful for determining neuronal ROIs.

Trace extraction

Before running the trace extraction command, neuronal ROIs should be drawn and saved in the 'output_folder' directory using the ImageJ ROI manager, with the following naming convention: if a single ROI is saved for a given patch, it should be named 'mc_image_stack_full_patch_N_ROI.roi' , where N is the patch number (e.g. mc_image_stack_full_patch_3_ROI.roi). If multiple ROIs are saved for a given patch, they should be saved in a zip file and named 'mc_image_stack_full_patch_N_ROIs.zip' , where N is the patch number (e.g. mc_image_stack_full_patch_3_ROIs.zip).
After saving the neuronal ROIs, run the following command:

```
get_neural_traces(output_folder)
```

Where 'output_folder' is the directory where the motion corrected files and the neuronal ROIs have been saved. The program will save matlab files named 'mc_image_stack_full_patch_X_ROI_Y.mat' where X is the patch number and Y is the name of the ROI file that was saved in ImageJ. Each trace is a cell array where each term is a vector corresponding to the mean pixel values in the neuronal ROI for each frame, in a given motion correction movie file (if the motion corrected files for a given patch have been saved in parts because of size, then each cell array term will correspond to a different part file).

Optimization

The code was originally designed to be parallelized for running in a cluster. Given that different clusters have different commands for running jobs, this release was written to run on any PC, to ensure maximal portability. If you have access to a cluster, it is recommended that each 'parfor' command be switched to a cluster-specific command that launches jobs in the cluster. There are two such commands in the code, one in the DBMC.m file and one in the get_neural_traces.m file. Please contact your system (or cluster) administrator for details on how to change the code.

External credits

The following files were obtained from freely available public repositories:

localnormalize.m: Guanglei Xiong (xgl99@mails.tsinghua.edu.cn) ,
https://www.mathworks.com/matlabcentral/fileexchange/8303-local-normalization
movingmean.m: Glen - https://www.mathworks.com/matlabcentral/fileexchange/41859-moving-average-function
ReadImageJROI.m: Dylan Muir (dylan.muir@unibas.ch) ,
https://github.com/DylanMuir/ReadImageJROI/blob/master/ReadImageJROI.m
saveastiff.m: YoonOh Tak - https://www.mathworks.com/matlabcentral/fileexchange/35684-multipage-tiff-stack

See individual file headers for relevant copyright notices.