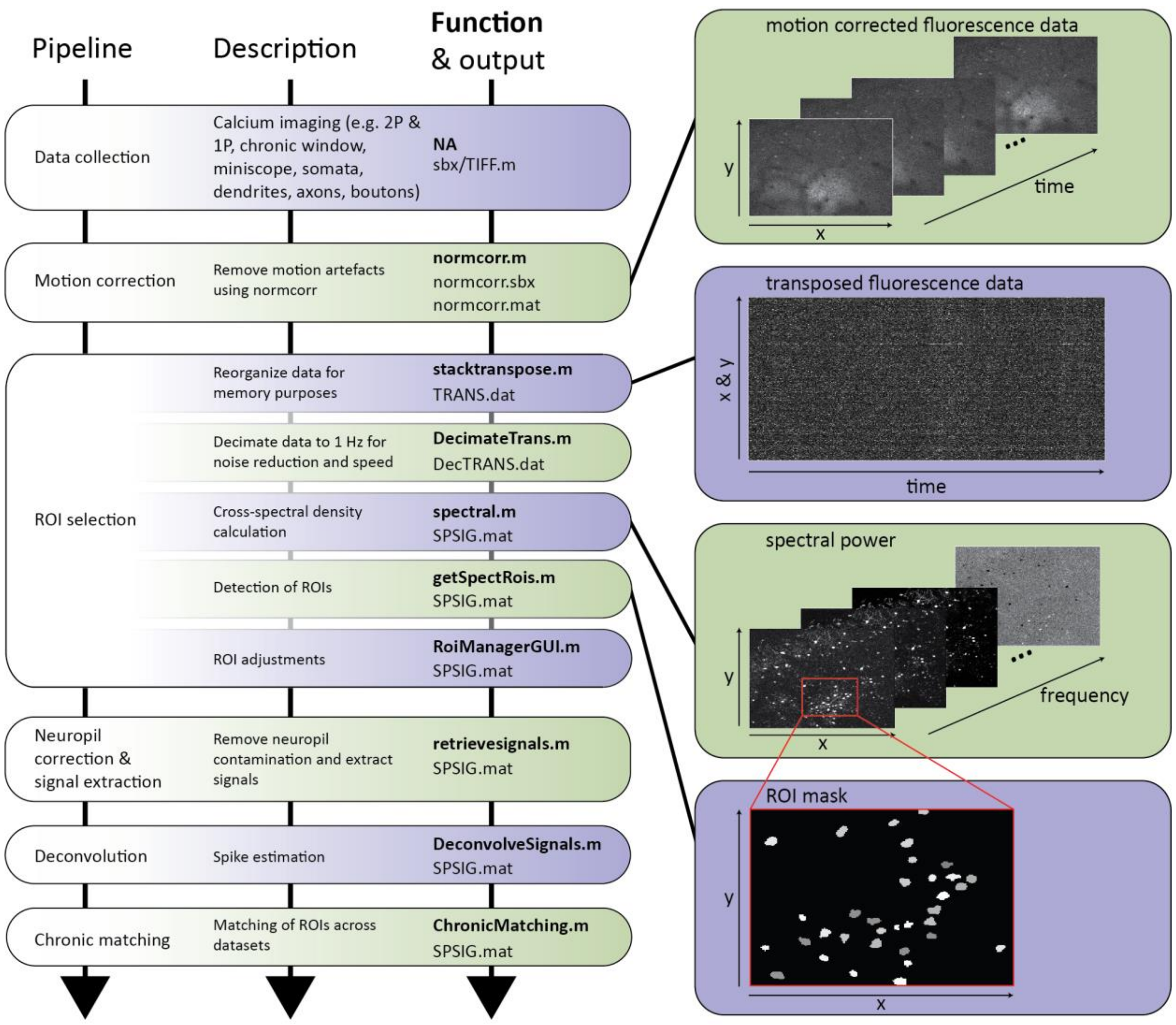
Contents

Introduction 2
Fluorescence data 3
Raw data of non-sbx format 3
Raw sbx data 3
Showsbx.m (to apply motion correction and other things on sbx data) 3
ROI selection 4
Transpose data 4
Decimate data 4
Spectral calculation 4
Representative fluorescence images 4
Get ROIs 5
Edit ROIs 6
Signal extraction 6
Fluorescence signal of ROIs 6
Spike estimation with MLspike 6
Chronic 6
Jumping into chronic without creating ROIs with spectral pipeline. 6
Chronic matching 7
References 8



Introduction

This manual covers practical information on using the Spectral Segmentation toolbox pre-processing pipeline. For more information, see the paper (reference).

There are two easy ways to run the pipeline: Via the **SpectralPipeline** or **AutomatedAnalysis** scripts.

The script **SpectralPipeline**.m in the main folder covers all the main functions of the pre-processing pipeline. All the functions can be called without input, each function will ask for which file to analyse. This script is useful for manually processing the data.

The script **AutomatedAnalysis**.m in the main folder is the script that we use to automatically execute all the pre-processing steps of the analysis for as many .sbx files as you request, until the RoiManagerGUI step, which requires manual input.

The pipeline uses a naming convention which it uses to search for files. If you deviate from this naming convention the toolbox will be harder to use.
Transposed files should end their filename with *_Trans.dat*
Decimated transposed files should end their filename with *_DecTrans.dat*
Spectral/ROI files should end their filename with *_SPSIG.mat*

Fluorescence data

Raw data of non-sbx format

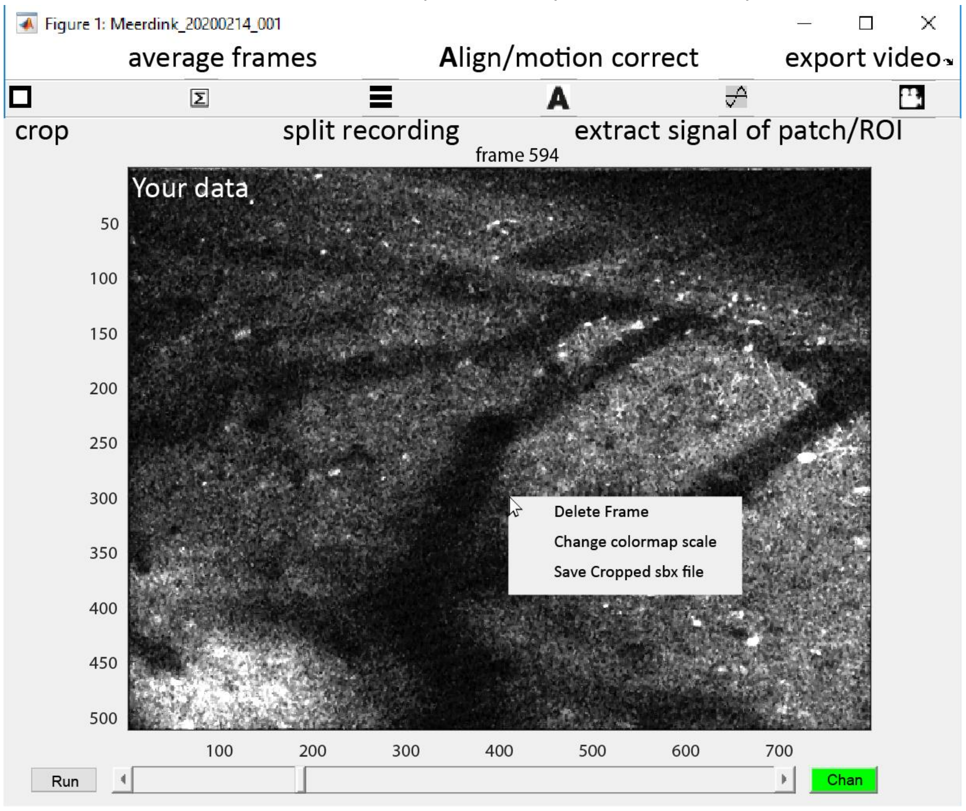
We do not have code to perform motion correction on file formats other than sbx files. But normcorre, which is necessary for the sbx motion correction, can handle a lot of file formats. If your raw data does not consist of .sbx files, we recommend to start the pipeline with .tiff files. Tiff files can be converted to .sbx with the script tif2sbx.m, which is included in toolbox.

Raw sbx data

Raw data needs to be motion corrected. We use the toolbox NoRMCorre (Pnevmatikakis and Giovannucci 2017) for motion correction. However, NoRMCorre cannot recognize sbx files. So we have adapted some of their code so normcorre can be used on sbx files. To be able to apply motion correction on sbx files, download normcorre from github: https://github.com/flatironinstitute/NoRMCorre. Place the normcorre code in your Matlab path **below** SpectralSegmentation folders. The motion correction can be applied by calling the GUI Showsbx.m, or via the AutomatedAnalysis script.

Showsbx.m (to apply motion correction and other things on sbx data)

The Showsbx.m GUI is able to play sbx files and call many functions, including motion correction. Start the GUI by executing *Showsbx* in the Matlab command in window. A prompt will appear to select your .sbx file.
The contrast of the data can be changed by right clicking in the main image (Change colormap scale). The motion correction can be called by pressing the **A** button. If the edges of the recording contain artefacts they can be removed by cropping (square bordered button). When crop is activated, press once in the upper left corner of your data, then press in the lower right corner of your data. Then press enter. A rectangle will appear which denotes what data will be included when aligning your data. This GUI is also a nice first step to check your data for any artefacts.



The Showsbx GUI. Showing a two-photon recording of 1 depth, imaged in a mouse injected with GCaMP6f in V1 layer 2-3.

ROI selection

Transpose data

The transposed data is necessary for the retrieval of fluorescence traces.
The motion corrected sbx data is transposed with StackTranspose.m.
When calling the function without input it will ask which .sbx file to transpose, followed by where to save the output.

Decimate data

To remove noise and decrease the amount of datapoints to deal with we often decimate/subsample the data to ~1Hz. Decimating data takes quite some time, so to save time decimated data is saved into a separate file with DecimateTrans.m. DecimateTrans.m creates a file with the filename of the original recording, with _DecTrans appended.
The decimated data is necessary for the spectral calculation (spectral.m). It is optional for other functions but can be used by many, speeding them up. These functions first search for the decTrans.dat file before the regular transposed file. Functions that can optionally use the decimated trans files are: RoiManagerGUI, getSpectRois, CalcSpatialCorrFun.

Spectral calculation

Run spectral.m to calculate the cross-spectral power. This function creates the SPSIG.mat file.
The spectral calculation requires the decimated transposed data.
The spectral components are stored in the variable *SPic*, the frequencies that correspond to the *SPic* 3rd dimension are stored in *Sax*.

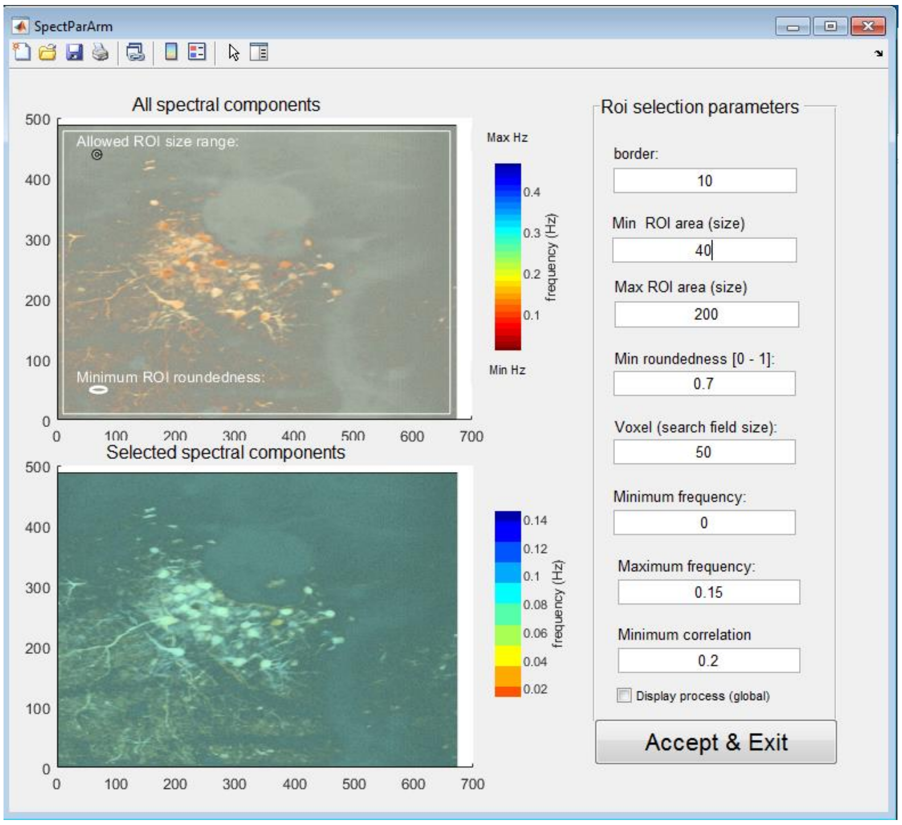
Representative fluorescence images

High quality fluorescence images of your data can be made using the function **BackgroundImgSbx**.m. It uses the regular motion corrected fluorescence data, not the transposed data.
The projections are made by taking short average projections of multiple frames (chunks) to reduce noise, and using these chunks for a maximum and an average projection. So there are two variables that can be changed in this procedure.
- *nframes*: How many frames are averaged together to use as a picture for the average projection: how many frames in each chunk.
- *nchunks*: How many chunks are used for the average projection.
Higher nframes will result in less noisy images, but short peaks in fluorescence get averaged out more. So the max projection end result will look slightly more like an average projection.
Higher nchunks means higher quality, because more data is included, but the procedure takes longer. There is no limit to the number of chunks so you can take overlapping chunks. This means the same frames can be used more than once in the projection. BackgroundImgSbx outputs the average fluorescence image *BImgAverage*, the maximum fluorescence projection is saved in *BImgMax*. Information about how the images were created is stored in the struct *infoImg*, this also contains the percentage of frames used of the recording to create the images in *infoImg.percent*.

Get ROIs

getSpectRois.m is used to automatically create ROIs via the cross-spectral power images and parameters specified by the user. These parameters are stored in the variable *spar* (spectral parameters), which also gets saved to the current folder as spar.mat, so getspectrois can load in the previously used parameters. When calling getSpectRois select the SPSIG file via the pop-up menu, and then set parameters for the ROI selection with the **SpectParArm** UI (see figure below). This UI requires an SPSIG file to show a preview of the data. If you want to set the *spar* variable without loading in an SPSIG file you can use the Spectroiparm.m UI.
In the SpectParArm UI the top image shows all frequency components, with different colors for different frequencies. It also gives feedback on the selected parameters.

Parameters to set:
Border: Defines the minimum distance from the edge of the image for finding ROI centres. The selected border is shown with the white rectangle just inside the top image.
Min & Max ROI area (size): The size of ROIs is calculated in how many pixels of the mask/ image they occupy. Feedback is given by a donut (see top left of top image).
Min roundness: the roundness is calculated as the ratio of the area over the square of the perimeter (length of contour) of the ROI. An estimation of the minimum roundness of ROIs to be found is shown in the lower left corner of the top image.
Minimum & maximum frequency select which frequency components to use for the ROI search. This has a large effect on how many ROIs are going to be found. The lower image shows the spectral components that are selected, so you can estimate whether artefacts or neurons are included.
Minimum correlation: The correlation refers to the covariance of the correlation of the fluorescence signal of the pixel with the highest spectral power value in the ROI with the signal of all other pixels in the ROI. If this value is low, the ROI signal is mainly uncorrelated noise (see paper).



Roi selection parameters

border: 10

Min ROI area (size): 40

Max ROI area (size): 200

Min roundedness [0 - 1]: 0.7

Voxel (search field size): 50

Minimum frequency: 0

Maximum frequency: 0.15

Minimum correlation 0.2

☐ Display process (global)

SpectParArm UI. Select parameters for the ROI selection.

Edit ROIs
ROIs can be edited manually via the RoiManagerGUI. The RoiManagerGUI has a separate manual. See the github page https://github.com/Leveltlab/SpectralSegmentation or the PDF file in the SpectralSegmentation\docs folder.

Signal extraction
Fluorescence signal of ROIs
Retrievesignals.m is the script that retrieves the fluorescence signal from ROIs. It uses the regular transposed data, not the decimated data. For the neuropil correction some variables can be changed. The function outputs the following signals:
sigRaw: raw fluorescence for all the ROIs: the average of all the pixels in each ROI.
sig: Df/f, as calculated with **basecorrect**.m.
sigBack: The Df/f of the neuropil ROIs.
sigCorrected: Df/f with neuropil subtraction before converting to Df/f

Spike estimation with MLspike
DeconvolveSignals uses MLspike (Deneux et al., 2016) for spike estimation of the retrieved fluorescence traces. To be able to use it download the code from brick and spike at https://github.com/MLspike, and add it to the Matlab path. The spike estimates are saved in the same format as the fluorescence signal. The outputted signals are:
Decon: Spiking data of the Df/f from *sig*.
Den: Denoised signal of the Df/f from *sig*.
DeconCorrected: Spiking data of the neuropil corrected Df/f *sigCorrected.*
DenCorrected: Denoised signal of the neuropil corrected Df/f *sigCorrected*.

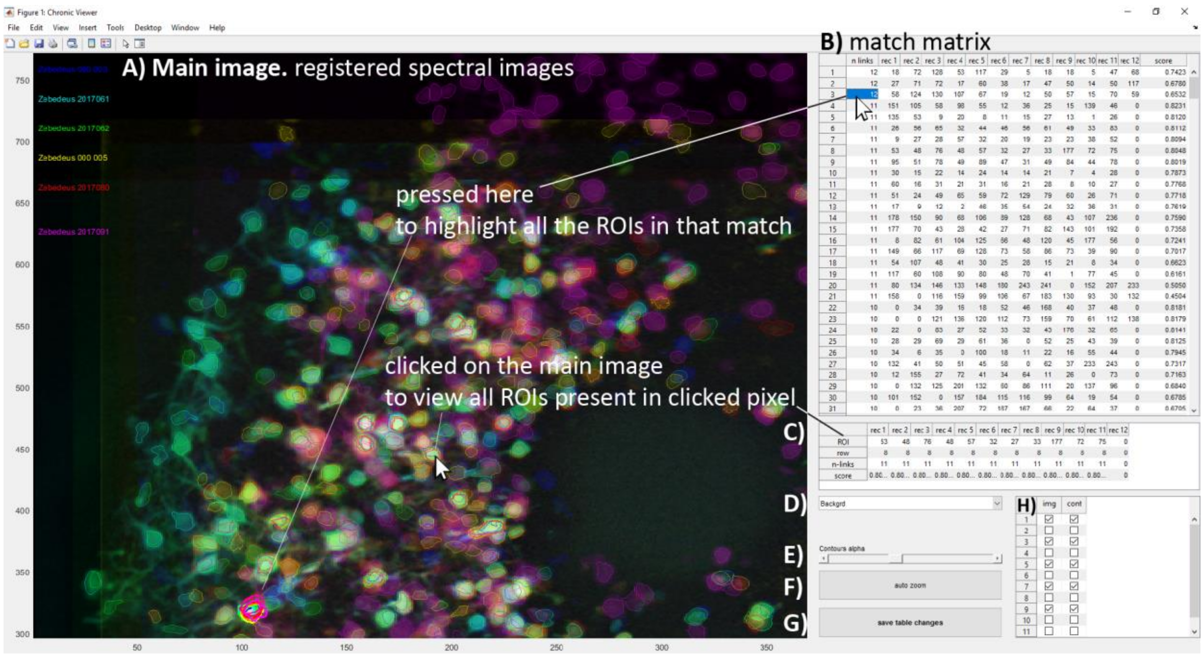
Chronic matching
Jumping into chronic without creating ROIs with spectral pipeline.
The chronic matching provided in the toolbox requires some specific variables that are created in the Spectral processing pipeline. However, you can use ROIs that you have found with other toolboxes. To be able to execute the Chronic matching as provided, you need to have pre-processed the data until the spectral calculation step.
The essential variables for chronic matching are:
- *Mask* (2D double [height x width]): An image that tells where ROIs are. Each pixel corresponds to an ROI by the number it has. 0 means no ROI present. Usually ROIs do not consist of multiple areas; they touch themselves everywhere. This variable is completely essential, and you need to provide this one yourself.
- *PP* (struct): This struct has a lot of info about the ROIs in Mask, like size (PP.A), signal covariance (PP.Rvar), contour coordinates (PP.Con) etc. If you have a mask, the contour and center coordinates of PP can be made via the script **PPfromMask**.m. Then the rest of the ROI properties can be retrieved by **PPModernize**.m. This function requires transposed data.
- *SPic* (3D double [height x width x frequencies]): This variable is used to create the image that is used for registration of recordings. But you could replace it by a maximum fluorescence projection.

Chronic matching

For chronic matching, execute the **ChronicMatching**.m script step by step. When the chronic matching is complete, the last step is to check the matches with the **ChronicViewer** UI (see figure). In this example there are 12 recording sessions, recorded over a period of 105 days in mouse V1 layer 2/3 expressing GCaMP6f. The UI displays the registered spectral images in the main image **A**. The visible neuron population shifted over time, in the early recordings (blue, cyan, green) a lot of neurons are visible on the left side. These were not visible in other recordings. The last recording had the largest differences. **B** contains the match matrix (via a Matlab UI table) which shows the chronically matched ROIs, one per row. The left column says how many ROIs are present in that match. Then the identification number of the matched ROIs are shown for each recording. The final column contains the overlap scores, which is the average of the fraction of ROI pixel overlap between all the different sessions for this match.

Clicking on ROIs or matches in the match matrix will highlight contours of the corresponding ROIs in the main image. The reverse can also be done. Clicking on ROIs in the main image will show information about the ROIs present in the clicked pixel in UI table **C**, with the ROI ID number(s) on the first row, on the second row is their position/row in the match matrix.

Occasionally matches might be incorrect according to the user. The match matrix can be edited by typing ROI ID numbers into the match matrix/ UI table. However, this can be frustrating because there can be a lot of data, every time you edit a cell the table jumps to the top and it can be slow to react if there is as much data as in this example. The advantages over editing the match matrix in the UI instead of in the Matlab variables tab is that the match properties get recalculated. If a ROI is added into a match, that ROI could already be present in another match. The UI will check whether the ROI is present in another match, remove it from that match, and also recalculate that match's properties. Since an edit can cause changes elsewhere in the matrix, it can be hard to undo typos. There is no undo option. If a lot of edit to the matches have to be made it is advised to rerun the chronic matching with a different overlap threshold or retry registration. **D)** Select other views for the main image. **E)** Contour alpha changes the opacity of the ROI contours. **F)** 'auto zoom' toggle enables a zoom to the ROI you clicked on in the match matrix. **G)** Save an edited match table to the Matlab workspace. **H)** Select which recording session image or ROI contours to show in the main image.



	n links	rec 1	rec 2	rec 3	rec 4	rec 5	rec 6	rec 7	rec 8	rec 9	rec 10	rec 11	rec 12	score
1	12	18	72	128	53	117	29	5	18	18	5	47	68	0.7423
2	12	27	71	72	17	60	38	17	47	50	14	50	117	0.6780
3	12	58	124	130	107	67	19	12	50	57	15	70	59	0.6532
4	11	151	105	58	98	55	12	36	25	15	139	46	0	0.8231
5	11	135	53	9	20	8	11	15	27	13	1	26	0	0.8120
6	11	26	56	65	32	44	46	56	61	49	33	83	0	0.8112
7	11	9	27	28	57	32	20	19	23	23	38	52	0	0.8094
8	11	53	48	76	48	57	32	27	33	177	72	75	0	0.8048
9	11	95	51	78	49	89	47	31	49	84	44	78	0	0.8019
10	11	30	15	22	14	24	14	14	21	7	4	28	0	0.7873
11	11	60	16	31	21	31	16	21	28	8	10	27	0	0.7768
12	11	51	24	49	65	59	72	129	79	60	26	71	0	0.7718
13	11	17	9	12	2	46	35	54	24	32	36	31	0	0.7619
14	11	178	150	90	68	106	89	128	68	43	107	236	0	0.7590
15	11	177	70	43	28	42	27	71	82	143	101	192	0	0.7358
16	11	8	82	61	104	125	86	48	120	45	177	56	0	0.7241
17	11	149	66	117	69	128	73	58	86	73	39	90	0	0.7017
18	11	54	107	48	41	30	25	28	15	21	8	34	0	0.6623
19	11	117	60	108	90	80	48	70	41	1	77	45	0	0.6161
20	11	80	134	146	133	148	180	243	241	0	152	207	233	0.5050
21	11	158	0	116	159	99	106	67	183	130	93	30	132	0.4504
22	10	0	34	39	16	18	52	46	168	40	37	48	0	0.8181
23	10	0	0	121	136	120	112	73	159	70	61	112	138	0.8179
24	10	22	0	63	27	52	33	32	43	176	32	65	0	0.8141
25	10	28	29	69	29	61	36	0	52	25	43	39	0	0.8125
26	10	34	6	35	0	100	18	11	22	16	55	44	0	0.7945
27	10	132	41	50	51	45	58	0	62	37	233	243	0	0.7317
28	10	12	155	27	72	41	34	64	11	26	0	73	0	0.7163
29	10	0	132	125	201	132	60	86	111	20	137	96	0	0.6840
30	10	101	152	0	157	184	115	116	99	64	19	54	0	0.6785
31	10	0	23	36	207	72	187	167	66	22	64	37	0	0.6705

	rec 1	rec 2	rec 3	rec 4	rec 5	rec 6	rec 7	rec 8	rec 9	rec 10	rec 11	rec 12
ROI	53	48	76	48	57	32	27	33	177	72	75	0
row	8	8	8	8	8	8	8	8	8	8	8	0
n-links	11	11	11	11	11	11	11	11	11	11	11	0
score	0.80...	0.80...	0.80...	0.80...	0.80...	0.80...	0.80...	0.80...	0.80...	0.80...	0.80...	0

	img	cont
1	☑	☑
2	☐	☐
3	☑	☑
4	☐	☐
5	☑	☑
6	☐	☐
7	☑	☑
8	☐	☐
9	☑	☑
10	☐	☐
11	☐	☐

References

Pnevmatikakis, E. A., & Giovannucci, A. (2017). NoRMCorre: An online algorithm for piecewise rigid motion correction of calcium imaging data. *Journal of neuroscience methods*, 291, 83-94. 10.1016/j.jneumeth.2017.07.031

Deneux, T., Kaszas, A., Szalay, G., Katona, G., Lakner, T., Grinvald, A., Rózsa, B. & Vanzetta, I. (2016) Accurate spike estimation from noisy calcium signals for ultrafast three-dimensional imaging of large neuronal populations in vivo. *Nature Communications* 7, 12190. 10.1038/ncomms12190