HELIOS

# Installation

Clone the following GitHub repository

<https://github.com/Nemezijus/HELIOS/tree/HUB-precursor>

using either gitbash or GitHub Desktop.

Add the cloned repository to your MatLab path. Done!

# User Guide

## Creating hdf5 file.

1. Using HUB root files (hrf) from HUB toolbox. HUB root files contain all the information about data file location and necessary parameters of the given experiment. To create the h5 file you need to specify three variables:
2. hrfloc – HUB root file location (a full path specified as a string);
3. hdf5loc – the path to the directory where the H5 file has to be save (a string);
4. pars – a parameter struct with information needed to perform data processing. The fields of the pars struct are: stimtype, dffmethod, tostitch, bgmethod, bgcorrmethod.
   1. stimtype – a string specifying the visual stimulus protocol. Available options are: ‘gray60Hz’, ‘8s\_gray60Hz’, ’13.5s\_gray60Hz’, ‘14s\_gray60Hz’, ‘spat\_truncated\_duration\_15s’.
   2. dffmethod – a string specifying the df/f calculation method. Available options are ‘median’, ‘mode’, ‘mean’, ‘percentile’, ‘gaussian’.
   3. tostitch – a logical 1 or 0, indicating whether you want to perform analysis on stitched traces. Stitching waveforms allows a better estimate of slow waveform changes and in particular helps estimating more accurate df/f.
   4. bgmethod – a string specifying what background estimation method to use. Available options are: ‘skip’, ‘staticpixels’, ‘dynamicpixels’.
   5. Bgcorrmethod – a string specifying what background subtraction (from raw data) method to use. Available options are: ‘customao’, ‘linear’ and ‘contaminationr’.

To create h5 file using this method, specify these variables and run

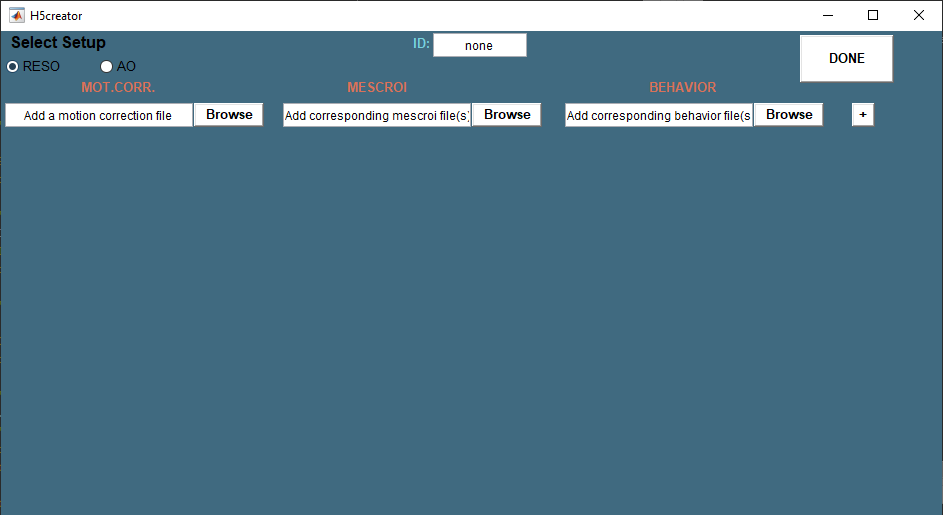
**createhdf5(hrfloc, hdf5loc, pars)**

for example:

hrfloc = ‘C:\data\’10A\_1.mat’;  
hdfloc = ‘C:\data\h5file’;  
pars.stimytpe = ‘8s\_gray60Hz’;  
pars.dffmethod = ‘median’;  
pars.tostitch = 1;  
pars.bgmethod = ‘dynamicpixels’;  
pars.bgcorrmethod = ‘linear’;  
createhdf5(hrfloc, hdf5loc, pars);

H5 creation in this case can take several hours, depending on the size of the data. Using this method it is not possible to embed behavioral information in the H5 file.

1. Using a GUI. To launch the GUI run **h5creator** in MatLab.



Select setup type – AO or RESO.

Specify the ID of the animal.

Add the locations of the data from the first day – Motion corrected data file; corresponding mescroi file(s) and corresponding behavior (.csv) file(s) – using ‘Browse’ buttons.

Using + you can add data from another day for the same animal.

Motion corrected files typically contain several recording units. When specifying mescroi files you can select multiple of them, one for each unit or one mescroi file for all units. If the number of mescroi files is less than the units in motion corrected file, the last mescroi file will be used on all the remaining unmatched units. There is no option to specify mescroi files to specific units.

The number of behavior files MUST match the number of units in the motion corrected file.

When all days have been specified for this experiment, click on ‘DONE’ and H5 creation will begin. The h5 file will be stored in the parent directory of the motion corrected file location.

This H5 creation method so far does not support multiple stimuli and repetitions being present.

Eventually method 1 will be completely replaced by method 2 and method 2 will be the only way to create h5 files.

## Reading hdf5 file and data.

To collect data from the H5 file you need first to create an experiment object:

**ex = experiment(h5filelocation);**

where h5filelocation is a string specifying the full path to the H5 file.

If the H5 file contains behavior data, you can extract it into matlab struct with the following command:

**B = behavior(ex);**

To collect the time series data from the H5 file use the following function:

**W = traces(ex, idxs, type);**

idxs is a cell of four elements: {ROIID, STAGEID, STIMID, REPID}  
ROIID – the numerical indices of ROIs  
STAGEID – the numerical index of stage/day  
STIMID – the numerical indices of stimuli  
REPID – the numerical indices of repetitions.

You can specify more than one index for ROIID, STIMID and REPID using vector notation []. However, it is generally recommended to use combinations of STIMID and REPID, leaving ROIID a single value. For STIMID and REPID you can also specify 0 – then all possible cases will be collected.

Type is a string identifier specifying which data to collect. Available options are: ‘dff’, ‘raw’ and ‘bg’.

W is a waveform object.

You can visualize the waveform object using plot method.

**W.plot;**

Additionally, you can specify the color, linestyle etc. options like in regular matlab plot function.

If W is a ‘raw’ data waveform you can perform df/f of your specified type by using **W.dff(mode)**, where ‘mode’ is a string specifying the df/f method to use (see above). Dff method works for multiple-trace waveform objects for ‘mode’ and ‘median’ cases, but not the others.

You can stack the two aforementioned methods together, for example: **W.dff(‘median’).plot(‘r-‘)**;

For H5 files where stimulus was varied, you can visualize the so called ‘profile’ of a ROI by calling

**profile(ex, ROIID)**;

where ROIID is the ROI numerical identifier. The figure shows the mean df/f traces for every stimulus for all days as well as significant peak counts, polar plot, OSI values and more.

Most of the data in profile is calculated with a method ‘response’. You can also call it manually:

**R = response(ex, ROIID, istage);**

Where ROIID is the roi numerical identifier, and istage – the stage/day numerical identifier. R is a struct which contains various analysis results (see below).

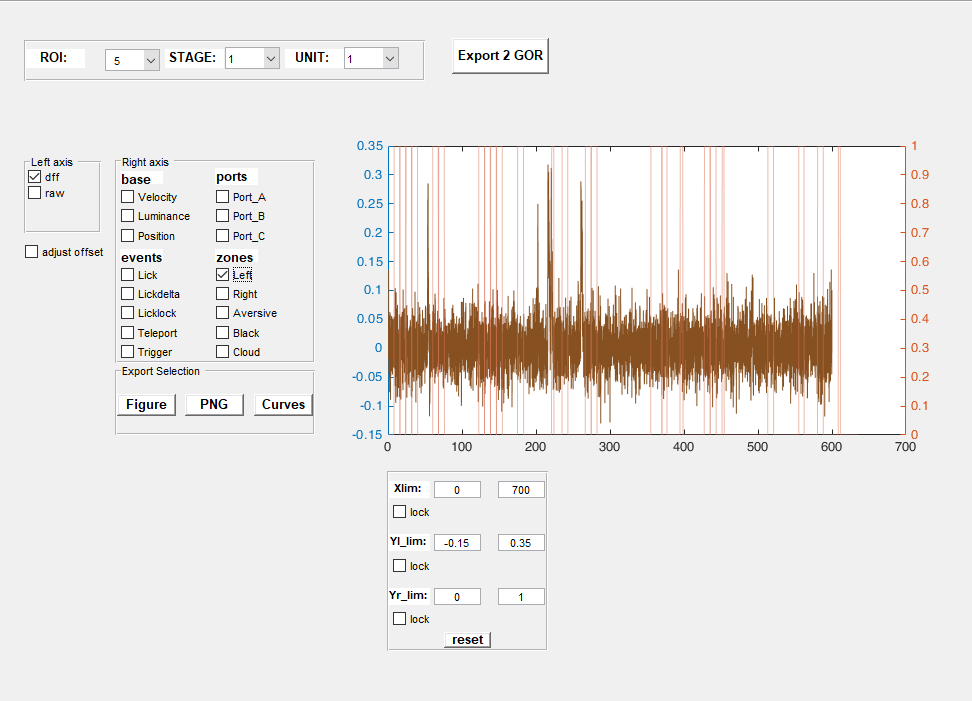
If H5 file contains varied stimuli data, you can visualize the ROI encapsulated are through a movie. To do so, run:

**[IMmovie,frames] = playroi(ex, ROIID, istage, Npix)**

ROIID is the ROI numerical index, istage – stage/day numerical index, Npix – the amount of pixels (border) around the ROI. If there is no output required then the movie will be played instantly, otherwise movie will be stored in the IMmovie variable.

If H5 file contains behavior data, you can directly visualize both imaging and behavior data through an interactive GUI. For that call:

**show (ex);**

****

Here you can select ROI, STAGE and UNIT of interest and what kind of imaging and behavior information to plot. In the panel below you can specify x and y axes (left and right) ranges and lock them so that they won’t change when selecting different parameters. Adjust offset option shifts imaging curves to match the imaging beginning with behavioral data. The data/images can also be exported.

Export 2 GOR - exports all data into gor object and saves it in the same location (but own directory) where H5 file is.  
Figure – creates a separate figure with the same data as is visualized in the main field.  
PNG – the main field image is saved as png and stored in the same directory where H5 file is.  
Curves – the selection of what to visualize is exported to gor object and it is saved in the same directory where H5 file is.

# Classes

Two main classes – **waveform** and **experiment** – form the core of the HELIOS toolbox.

# Waveform

Objects of this class store the waveforms with their time samples and additional meta information.

To create a waveform object:

<**output**> **= waveform (data, time, data\_type, time\_units, data\_units, tag);**

data – the waveform vector or matrix;

time – the corresponding time sampling vector (optional);

data\_type – a string specifying the type of data (e.g. ‘raw’,’dff’) (optional);

time\_units – a string specifying units of time (‘us’, ‘ms’, ‘s’) (optional);

data\_units – a string specifying the units of data (e.g. ‘a.u.’) (optional);

W = waveform;

%fields

**data:** [1x100 double]

**data\_type:** ‘random’

**data\_units:** ‘a.u’

**time:** [1:100 double]

**time\_units:** ‘samples’

**Fs:** NaN

**tag**: {}

tag – a string with custom information (optional) or h5 internal paths used to collect the data;

By default, only data input is required. If data is a matrix the time vector will represent all the rows in the data matrix. If even data is not provided the constructor will build a data of 100 random values.

Defaults:

Time – numbers of samples in the first data row;

Data\_type – NaN

Time\_units – NaN

Data\_units – NaN

Fs – NaN

Tag – {} empty cell

Fs is the sampling frequency and in case the time units are specified as ‘us’ (microseconds), ‘ms’ or ‘s’ it will be calculated automatically.

There are several methods currently available to use on waveform objects.

## Waveform methods

dff, plot

## dff

**[w, dff] = dff(w, method, ex);**

Estimates the df/f on the data inside the waveform object w. If w is already a ‘dff’ type, then no df/f calculation will happen. Some of the methods don’t work on multiple-trace waveforms. However, the most common ones – ‘median’ and ‘mode’ do.

W – waveform object;  
method – a string specifying which df/f method to use;  
ex – experiment object.

## Plot

**[pl,ax] = plot(obj,ax,varargin);**

Plots the data in the waveform object vs own time axis. If ax (the axes handle) is specified, the plot will happen on that axes. Varargin can be specified in the same fashion as for plot function in MatLab in short notation, for example ‘b-.’, ‘k—’ and so on. The propertyname-propertyvalue pairs will not work here.

# Experiment

Experiment class is a result of scanning an experiment HDF5 (H5) file. It collects the information about number of recordings, internal hdf5 file path structure as well as metadata (ID, setup etc.). Only those H5 files, which have been created with HELIOS, can be used to define an experiment object due to their structure. Most of the methods and operations in HELIOS rely on experiment object.

To create an experiment object:

**ex = experiment(<h5 file path>);**

ex = experiment('D:\DATA \m45B\_1.h5');

%fields

**file\_loc:** 'D:\DATA \m45B\_1.h5**'**

**id:** '45B\_1'

**setup:** 'ao'

**stim\_type:** '13.5s\_gray60Hz'

**dff\_type:** 'percentile'

**bg\_corrected:** 1

**N\_roi:** 105

**N\_stages:** 6

**N\_stim:** [9 9 9 9 9 9]

**N\_reps:** [12 12 12 12 12 12]

**paths:** {2×4 cell}

**restun:** {1×6 cell}

The fields of experiment:

file\_loc – H5 file path;

id – the experiment identifier;

setup – experimental setup;

stim\_type – stimulus protocol;

dff\_type – df/f method used;

bg\_corrected – is background correction done;

N\_roi – number of ROIs;

N\_stages – number of recording days;

N\_stim – number of stimuli presented each day;

N\_reps – number of repetitions for each stimulus presented each day;

paths – cell of strings with unique paths in the H5 file;

restun – matrix of numbers showing how recording number (**un**it) relates to **st**imulus number and stimulus **re**petition (one cell for one day).

H5 files currently are created for two types of experiments, with and without visual stimulation. Based on that the H5 file structure is different for them, resulting in different information collected for the experiment object.

# Experiment methods

Behavior, bestcorr, dff, dffparams, exportprofiles, extractbg, playroi, profile, response, show, stitch, storeonaciddff, subtractbg, traces

If **ex** is experiment object:

## Behavior

**B = behavior(ex);**

If experiment file contains behavior data, it is collected into the struct B. The layout of the struct is a the same as in H5 file.

## Bestcorr

**C = bestcorr(ex, iroi);**

Here iroi – a double, indicating ROI number (eg. 1, 2, 20 etc.);

If the H5 file has mean images stored, bestcorr will calculate the correlation between the very first acquired image and all subsequent ones (across all days). The higher the value of C vector the more correlated are the ROI areas.

## dff

**ex = dff(ex, method, tostitch, aoexception)**

performs df/f calculation and stores it in the h5 file! Do not run use this method unless you know what are you doing. This method takes quite some time and overwrites df/f values in the h5 file. To check df/f performance on the traces, use dff method of the waveform class.

method - a string indicating which df/f calculation method to use:

‘median’ – based on method used in Allen institute; uses long timescale median filter for baseline subtraction

‘mode’ – same as ‘median’ but based on mode filter

‘mean’ – F0 is estimated using sliding moving average window, picking the lowest value

‘percentile’ – uses (8th by default) percentile to estimate baseline

‘gauss’ – Gauss smoothened histogram of events is used to estimate F0

The recommended and overall best methods are ‘median’ and ‘percentile’.

Tostitch – logical 1 or 0, indicating whether the df/f should be calculated on stitched traces as opposed to individual cuts based on repetitions. In general stitching improves df/f quality, as detrending is captured more accurately.

Aoexception – logical 1 or 0, indicating whether the df/f should be stored in a different path in the h5 file. Typically, this is used in AO experiment files where background information is not available. The default path for dff in H5 file is DFF but it is calculated on background-corrected signal. An additional path in h5 called DFFBASE is used to store df/f for only raw data with no background-correction. For AO this is the only case (since background is not available). If aoexception is 1, the df/f will be stored in DFFBASE.

## Dffparams

**E = dffparams(ex);**

Collects default parameters of df/f methods into one struct E. Normally not meant for user call, dffparams is used internally by dff method.

## Profile

**F = profile(ex, iroi);**

Creates a figure with representative mean df/f curves, peak counts and polar plots for a given ROI. Applicable only for those experiments which have varied stimuli and repetitions. Creation of such figure can take up to 2 minutes, depending on data readout speed.

F is a figure handle, if the profile creation was successful.

iroi – the numerical index of the ROI.

## Exportprofiles

**exportprofiles(ex, saveloc);**

creates profiles (see above) for ALL ROIs and stores them in the specified location saveloc. If no location is specified, the user will be prompted to choose one.

Works only for those experiments which have varied stimuli presented.

## Extractbg

**ex = extractbg(ex, method);**

calculates the background information from the data, stores it in the h5 file. Takes couple of hours to run. Usually not used as a standalone method, but rather when h5 file is being created.

Method - a string specifying the background calculation method:

‘staticpixels’ – the slowest method. Finds the pixel ids with the lowest signal around the roi, but within the bounding box. Their value mean is the bg signal;

‘dynamicpixels’ – the faster method. For every time frame uses the mean of the 10 lowest valued pixels. This creates dynamic background.

## Playroi

**[IMmovie,frames] = playroi(ex, iroi, istage, Npix)**

Creates a movie of the mean frames for a given ROI and day (stage). If no outputs are requested then the movie is instantly played, otherwise the movie is collected in a IMmovie variable.

iroi – the ROI numerical index

istage – the stage/day numerical index

Npix – the amount of pixels to be used to frame the ROI contour. Default is 5 pixels.

## Response

**R = response(ex, iroi, istage)**

Estimates how strong is the response during the stimulus window for a given ROI and day/stage. Additionally computes the OSI and counts the significant peaks in the stimuli windows.

Iroi – the ROI number

Istage – the stage/day number

R fields:

Stimulus – list of stimuli used;

Strength – the response strength for each stimulus;

Stimwinsd – standard deviation of signal within stimulus window for each stimulus;

Stage – stage identifier;

Osi – orientation selectivity index;

Dominantstimulus – stimulus with the largest response;

Peaksinstimwin – number of significant peaks within stimulus window in each repetition for each stimulus;

Peakmask – logical mask that picked out the peaks for peaksinstimwin.

## Show

**show(ex)**

An interactive data browser and exporter. Usable only for those experiments which contain behavior data.

## Stitch

**S = stitch(ex, iroi, istage, type)**

Stitches the waveforms in their sequential acquisition time for a given ROI and day/stage.

Iroi – the ROI number

Istage – the stage/day number

Type – string identifier telling which data to stitch. Choose from ‘raw’, ‘bg’ and ‘dff’.

S is a waveform object.

## Storeonaciddff

**storeonaciddff(ex, dff, stimlist, stimsequence)**

stores the df/f estimated by OnAcid algorithm in the h5 file. Usually is not called by the user and is run internally when creating h5 file.

Dff – the df/f matrix produced by OnAcid

Stimlist - a sequence of stimuli identifiers till the pattern repeats

Stimsequence - a full list of stimuli identifiers from the first recording till the last. [Optional]

## Subtractbg

**ex = subtractbg(ex, info)**

Reads out the background information from the h5 file and subtracts it from the far data. After the subtraction it calculates the df/f and stores it. This method should not be run standalone by a user; it rather is used by h5 creation functions.

Info – the information struct about the necessary parameters for these operations.

Bgcorrmethod – a string specifying which method of background correction to use – ‘linear’, ‘customao’ or 'contaminationr';

Dffmethod – a string specifying which df/f method to use (see dff method);

Tostitch – logical 1 or 0 specifying whether to stitch the waveforms before df/f calculation.

## Traces

**W = traces(ex,idxs,type)**

Collects the waveforms from H5 file for the specified parameters. The output W is a waveform object.

Idxs – a cell array of structure {ROIID, STAGEID, STIMID, REPID}, where each entry is either a single number or a vector identifying:

ROIs – ROIID

Stages/days – STAGEID

Stimuli – STIMID

Repetitions – REPID

Type – ‘raw’, ‘bg’ or ‘dff’

For example w = traces(ex, {2, 1, [1,2,3,5], 1},’dff’) – collects df/f curves for ROI 2 from day/stage 1 for stimuli 1,2,3,5, repetition 1.