session_demonstration_script

November 10, 2021

1 Example code for using session.Session

Note: This notebook covers several relevant methods of the `Session` and `Stim` objects, detailing some of their arguments, as well. For more details, take a look at the docstring associated with a method of interest.

Import notes:

- These packages should be present if installing the conda environment from `osca.yml`.
- `util` is a Github repo of mine, and the correct branch `osca_mult` is automatically installed from `osca.yml`. Errors internal to the codebase involving `util` code and occurring *after* new changes have been pulled from the `OpenScope_CA_Analysis` repo *may* be due to an update of the `osca_mult` branch of `util`. Though I will try to avoid this, consider updating the utility under those circumstances.

1.1 Set paths to main data directory and the mouse dataframe

If you wish to use the same formatting style (and logging format) as I do:

1.2 1. Basics of initializing a Session object

After creating the session, you must run `self.extract_info()`. This wasn't amalgamated into the `__init__` to reduce the amount of information needed to just create a session object.

1.2.1 Loading ROI/running/pupil info

You can load this information when you call `self.extract_info()` or manually later by calling `self.load_roi_info()`, `self.load_run_data()` and `self.load_pup_data()`.

```
Loading stimulus and alignment info…
Loading ROI trace info…
    WARNING: Session 764704289: 3 noisy ROIs (mean below 0 or median above
midrange) are also included in the NaN ROI attributes (but not set to NaN): 244,
298, 305.
Loading running info…
    WARNING: Session 764704289: 211 dropped running frames (~0.1%) (in pre-
processing).
Loading pupil info…
```

```
[5]:      stimulus_type stimulus_template_name  unexpected gabor_frame  \
     0       grayscreen             grayscreen         NaN
     1           gabors                 gabors         0.0           A
     2           gabors                 gabors         0.0           B
     3           gabors                 gabors         0.0           C
     4           gabors                 gabors         0.0           D
     …            …                      …         …         …
     8839       visflow          visflow_right         0.0
     8840       visflow          visflow_right         1.0
     8841       visflow          visflow_right         1.0
     8842       visflow          visflow_right         1.0
     8843    grayscreen             grayscreen         NaN

           gabor_kappa  gabor_mean_orientation  gabor_number  \
     0             NaN                     NaN           NaN
     1            16.0                   135.0          30.0
     2            16.0                   135.0          30.0
     3            16.0                   135.0          30.0
     4            16.0                   135.0          30.0
     …           …                      …           …
     8839          NaN                     NaN           NaN
     8840          NaN                     NaN           NaN
     8841          NaN                     NaN           NaN
     8842          NaN                     NaN           NaN
     8843          NaN                     NaN           NaN

                                           gabor_locations_x  \
     0                                                    []
     1     [-0.998732530996428, -0.7988942745979938, -0.0…
     2     [-0.8273358833992613, -0.32202169430120714, -0…
     3     [-0.1439318404380644, -0.9639223437829889, -0…
     4     [-0.1981534893873622, -0.7603480104179756, -0…
     …                                                  …
     8839                                                 []
     8840                                                 []
     8841                                                 []
     8842                                                 []
     8843                                                 []

                                           gabor_locations_y  \
     0                                                    []
     1     [-0.936204215614872, -0.48115197167416995, -0…
     2     [-0.6021449948480063, -0.6653905125829843, -0…
     3     [-0.725093701321675, -0.021010443830197678, -0…
     4     [-0.23152862741244445, -0.41875478323604776, -…
     …                                                  …
     8839                                                 []
```

```
8840                                                 []
8841                                                 []
8842                                                 []
8843                                                 []

                                            gabor_sizes  …  \
0                                                    []  …
1     [293, 392, 392, 323, 280, 396, 316, 363, 226, …  …
2     [313, 319, 262, 228, 400, 210, 264, 218, 308, …  …
3     [396, 212, 277, 210, 390, 329, 406, 317, 358, …  …
4     [326, 244, 208, 212, 251, 242, 341, 299, 406, …  …
…                                                 …  …
8839                                                 []  …
8840                                                 []  …
8841                                                 []  …
8842                                                 []  …
8843                                                 []  …

     square_proportion_flipped start_frame_stim  stop_frame_stim  \
0                          NaN                0             1800
1                          NaN             1800             1818
2                          NaN             1818             1836
3                          NaN             1836             1854
4                          NaN             1854             1872
…                        …              …              …
8839                      0.00           249960           250020
8840                      0.25           250020           250080
8841                      0.25           250080           250140
8842                      0.25           250140           250200
8843                       NaN           250200           251999

      num_frames_stim  start_frame_twop  stop_frame_twop  num_frames_twop  \
0                1800               143             1046              903
1                  18              1046             1055                9
2                  18              1055             1064                9
3                  18              1064             1073                9
4                  18              1073             1082                9
…               …               …              …              …
8839               60            125552           125582               30
8840               60            125582           125612               30
8841               60            125612           125642               30
8842               60            125642           125672               30
8843             1799            125672           126575              903

      start_time_sec  stop_time_sec  duration_sec
0          14.277090      44.301717     30.024627
1          44.301717      44.602241      0.300524
```

```
2          44.602241     44.902563      0.300322
3          44.902563     45.202768      0.300204
4          45.202768     45.503007      0.300240
…              …             …            …
8839     4183.741890   4184.742721      1.000831
8840     4184.742721   4185.743529      1.000808
8841     4185.743529   4186.744364      1.000835
8842     4186.744364   4187.745223      1.000860
8843     4187.745223   4217.728557     29.983333

[8844 rows x 24 columns]
```

1.2.2 Some information contained in the session object

Note: `Stim` objects (subclasses: `Gabors`, `Visflow`, `Grayscr`) are a separate class from `Session` objects. However, each can by accessed from the other using: - from `Session objects`: `self.stims`, `self.gabors`, `self.visflow` - from `Stim objects`: `self.sess`

```
number of rois: 628
mouse number: 6
mouse ID: 413663
gabor object: Gabors (stimulus from session 764704289)
2p frames per sec: 30.08
stimulus frames per sec: 59.95
```

1.3 2. Identifying segments of interest

From a `Session`'s `Stim`, you can get a list of segments that fit a specific criterion, e.g. **U segments** (unexpected, 3rd segment).

Then, you can access the frame numbers.

Note: Specifying `ch_fl` (check flanks) ensures that only frame numbers whose flanks are within the recording are returned. In other words, any frame number too close to the start of end of the recording (based on `pre`/`post` values), will be dropped.

You can now get the **ROI/running/pupil data** corresponding these reference frames and specified `pre`/`post` periods (in sec).

You can also directly obtain statistics on the data of interest

```
[11]: datatype                roi_traces
      nan_rois_removed               yes
      scaled                          no
      baseline                        no
      integrated                     yes
      smoothing                       no
      fluorescence                   dff
      general ROIs sequences
      stats   None stat_mean    0.026752
```

```
             error_SEM   0.000911
```

Data and statistics are returned in a hierarchical dataframe with **columns** and **indices**.

This has the advantage of allowing metadata to be stored in dummy columns, however extracting data from these dataframes can be tricky, syntaxically.

```
[12]: datatype                    roi_traces
      nan_rois_removed                   yes
      scaled                             yes
      baseline                            no
      integrated                          no
      smoothing                           no
      fluorescence                       dff
      ROIs sequences time_values
      0    0         -1.000000     -0.009556
                     -0.966102     -0.644810
                     -0.932203     -0.214521
                     -0.898305     -0.116127
                     -0.864407     -0.318214
      …                                  …
      643  95         0.864407      0.050568
                      0.898305      0.445153
                      0.932203      0.108850
                      0.966102      0.116475
                      1.000000      0.213779

      [3617280 rows x 1 columns]
```

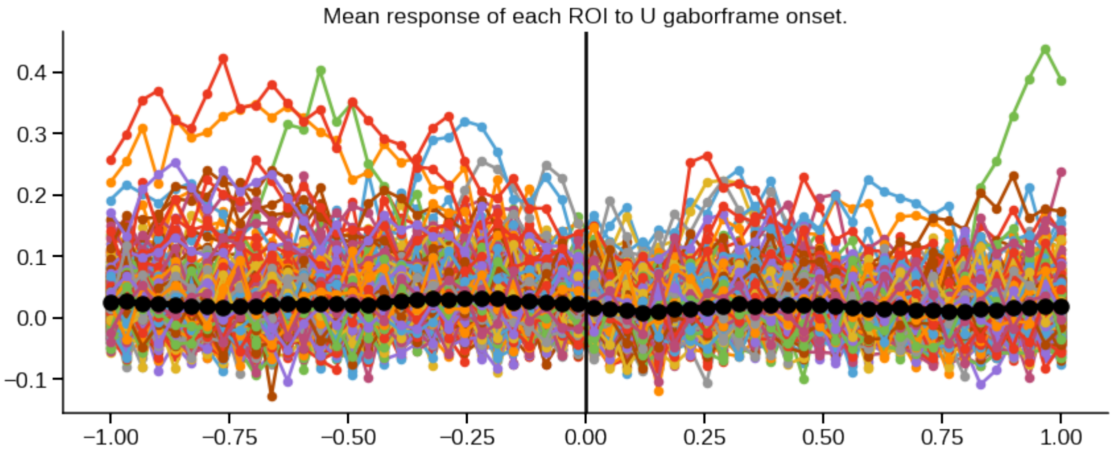
To **extract a numpy array** with the correct dimensions from a hierarchical dataframe, you can use the following utility.

Here, each index level, then column level becomes an axis, **i.e. ROIs x sequences x time_values** (In this case, `squeeze_cols` is set to True to prevent each dummy column from becoming an axis.)

```
ROI data shape: 628 ROIs x 96 sequences x 60 time values
```

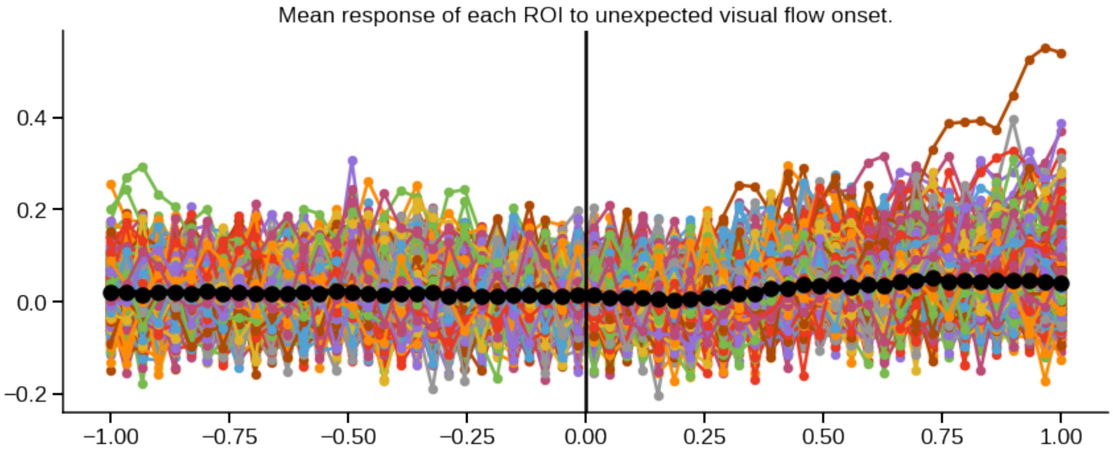
You can also retrieve the time stamps for each frame.

Finally, we can plot each ROIs mean activity across sequences, as well as a mean across ROIs.



1.3.1 The same steps apply for Visflow

```
[16]: <module 'sess_util.sess_data_util' from '../sess_util/sess_data_util.py'>
```



1.4 3. Additional tips on indexing a hierarchical dataframe

```
[18]: scaled                               yes
      baseline                              no
      integrated                            no
      smoothing                             no
      fluorescence                         dff
      ROIs sequences time_values
      0    0         -1.000000    0.045433
                     -0.966102   -0.303140
                     -0.932203   -0.301464
```

```
                -0.898305   -0.219794
                -0.864407    0.065587
…                              …
643  4           0.864407   -0.513739
                 0.898305   -0.143983
                 0.932203    0.000693
                 0.966102    0.140502
                 1.000000    0.283351

[113040 rows x 1 columns]
```

1.4.1 4. Retrieving several Session objects, based on criteria

This function keeps track of which Sessions or Mice must be left out (e.g., due to a problem with the session data or the mouse didn't see the stimulus of interest <- the latter only comes up with pilot data).

You can now retrieve the mouse number, session number and ID that fit specific the criteria,

e.g., **session number 1, 2 or 3**, **production**, **dendritic plane**

```
WARNING: Sorted and unique will be set to False as multiple labels are
requested.
```

```
mouse 6: 764704289 (session 1)
mouse 6: 765193831 (session 2)
mouse 6: 766502238 (session 3)
mouse 8: 777914830 (session 1)
mouse 8: 778864809 (session 2)
mouse 8: 779650018 (session 3)
mouse 9: 826187862 (session 1)
mouse 9: 826773996 (session 2)
mouse 9: 827833392 (session 3)
mouse 10: 826338612 (session 1)
mouse 10: 826819032 (session 2)
mouse 10: 828816509 (session 3)
mouse 11: 823453391 (session 1)
mouse 11: 824434038 (session 2)
mouse 11: 825180479 (session 3)
```

You can now **initialize the Sessions** using this function which does the additional extraction steps automatically.

```
Creating session 764704289…
Loading stimulus and alignment info…
Loading ROI trace info…
    WARNING: Session 764704289: 3 noisy ROIs (mean below 0 or median above
midrange) are also included in the NaN ROI attributes (but not set to NaN): 244,
298, 305.
```

```
Loading running info…
    WARNING: Session 764704289: 211 dropped running frames (~0.1%) (in pre-
processing).
Finished creating session 764704289.

Creating session 765193831…
Loading stimulus and alignment info…
Loading ROI trace info…
    WARNING: Session 765193831: 4 noisy ROIs (mean below 0 or median above
midrange) are also included in the NaN ROI attributes (but not set to NaN): 3,
63, 88, 134.
Loading running info…
    WARNING: Session 765193831: 345 dropped running frames (~0.1%) (in pre-
processing).
Finished creating session 765193831.

Creating session 766502238…
Loading stimulus and alignment info…
Loading ROI trace info…
    WARNING: Session 766502238: 4 noisy ROIs (mean below 0 or median above
midrange) are also included in the NaN ROI attributes (but not set to NaN): 18,
114, 136, 240.
Loading running info…
    WARNING: Session 766502238: 387 dropped running frames (~0.2%) (in pre-
processing).
Finished creating session 766502238.

Creating session 777914830…
Loading stimulus and alignment info…
Loading ROI trace info…
    WARNING: Session 777914830: 1 noisy ROIs (mean below 0 or median above
midrange) are also included in the NaN ROI attributes (but not set to NaN): 45.
Loading running info…
    WARNING: Session 777914830: 381 dropped running frames (~0.2%) (in pre-
processing).
Finished creating session 777914830.

Creating session 778864809…
Loading stimulus and alignment info…
Loading ROI trace info…
Loading running info…
    WARNING: Session 778864809: 630 dropped running frames (~0.3%) (in pre-
processing).
Finished creating session 778864809.

Creating session 758519303…
Loading stimulus and alignment info…
Loading ROI trace info…
Loading running info…
    WARNING: Session 758519303: 175 dropped running frames (~0.1%) (in pre-
processing).
Finished creating session 758519303.
```

Then run through the sessions and do whatever with them.

```
Session ID: 764704289 (mouse 6, session 1)
    visflow: 33 sequences
    gabors: 96 sequences
Session ID: 765193831 (mouse 6, session 2)
    visflow: 34 sequences
    gabors: 98 sequences
Session ID: 766502238 (mouse 6, session 3)
    visflow: 29 sequences
    gabors: 94 sequences
Session ID: 777914830 (mouse 8, session 1)
    visflow: 32 sequences
    gabors: 83 sequences
Session ID: 778864809 (mouse 8, session 2)
    visflow: 29 sequences
    gabors: 88 sequences
Session ID: 758519303 (mouse 1, session 1)
    visflow: 31 sequences
    gabors: 94 sequences
```

1.4.2 5. Retrieving ROI masks from session.

Boolean ROI masks can be obtained for Session.

For **dendritic sessions**, the Session is built to assume that `EXTRACT` (not `allen`) ROI data is to be used. This can be checked by checking `self.dend`. As long as `self.dend` is properly set, the correct masks will be loaded.

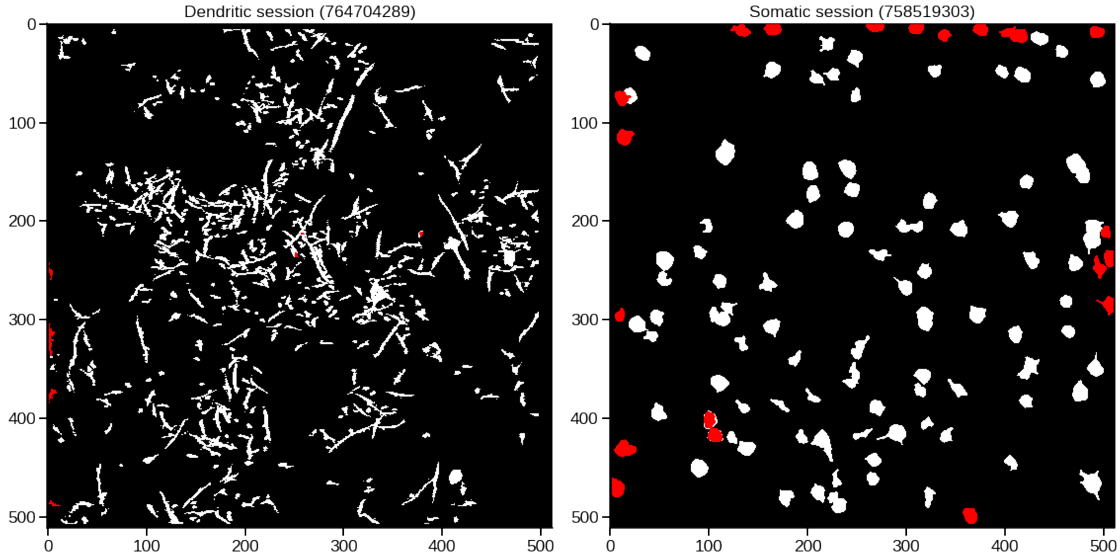
```
Dendritic session, ROI type: extr
Somatic session, ROI type: allen
```

Masks can be loaded as follows, with dimensions: **ROI x height x width**, retrieving only masks for ROIs that are valid for dF/F traces.

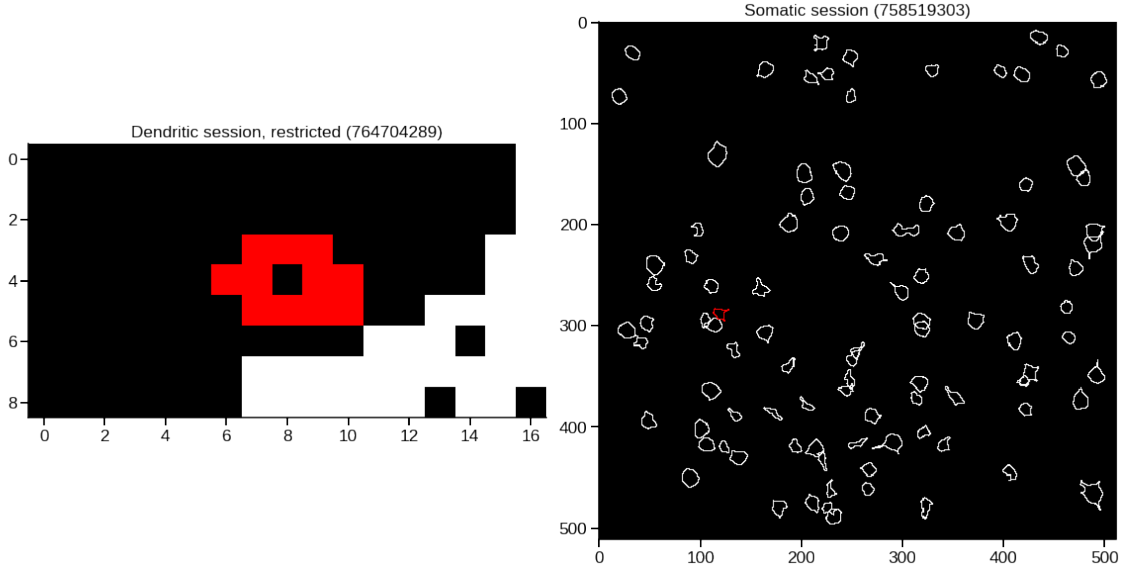
Or if all masks are needed, the attributes can simply be retrieved.

One way to check which ROIs are not valid, is using `self.get_nanrois()`

This is a tool to visualize ROIs, where specific ROIs can be set to red using a `valid_mask`.



This is a tool to visualize ROI contours, optionally localized around an ROI of interest.


Dendritic session, restricted (764704289)
0
2
4
6
8
0
2
4
6
8
10
12
14
16
Somatic session (758519303)
0
100
200
300
400
500
0
100
200
300
400
500

1.4.3 6. Last notes

List the methods/attributes attached to `Session` and `Stim` objects.

```
Session: Session (758519303)
_dend, _extract_sess_attribs, _full_table, _get_roi_facts, _init_directory,
_init_roi_facts_df, _load_stim_df, _load_stim_dict, _load_stims, _nanrois_dff,
_nrois, _nwb, _only_tracked_rois, _roi_masks, _set_dend_type, _set_nanrois,
_set_nanrois_tracked, _set_roi_attributes, _set_tracked_rois, _stim2twopfr,
align_pkl, all_files, any_files, behav_video_h5, check_flanks, convert_frames,
correct_data_h5, data_loaded, date, dend, depth, dir, drop_tol, expdir, expid,
extract_info, gabors, get_active_rois, get_fr_ran, get_frames_timestamps,
get_nanrois, get_nrois, get_plateau_roi_traces, get_pup_data, get_roi_masks,
get_roi_seqs, get_roi_traces, get_run_velocity, get_run_velocity_by_fr,
get_single_roi_trace, get_stim, grayscr, home, line, load_pup_data,
load_roi_info, load_run_data, mouse_df, mouse_dir, mouse_n, mouseid, n_stims,
notes, nwb, only_tracked_rois, pass_fail, plane, procdir, pup_data_h5,
pup_video_h5, roi_extract_json, roi_facts_df, roi_mask_file, roi_masks,
roi_names, roi_objectlist, roi_trace_dff_h5, roi_trace_h5, run_data, runtype,
segid, sess_n, sessid, set_only_tracked_rois, stim2twopfr, stim_df, stim_fps,
stim_pkl, stim_seed, stim_sync_h5, stims, stimtypes, time_sync_h5, tot_stim_fr,
tot_twop_fr, tracked_rois, twop2stimfr, twop_fps, visflow, zstack_h5

Gabors: Session (758519303)
_set_block_params, all_gabfr, all_gabfr_mean_oris, block_params, deg_per_pix,
exp_gabfr, exp_gabfr_mean_oris, exp_max_s, exp_min_s, get_A_frame_1s,
get_A_segs, get_all_unexp_segs, get_all_unexp_stim_fr, get_fr_by_seg,
get_frames_by_criteria, get_n_fr_by_seg, get_pup_diam_data,
get_pup_diam_stats_df, get_roi_data, get_roi_stats_df, get_run, get_run_data,
get_run_stats_df, get_segs_by_criteria, get_segs_by_frame, get_start_unexp_segs,
get_start_unexp_stim_fr_trans, get_stats_df, get_stim_beh_sub_df,
get_stim_df_by_criteria, get_stim_par_by_frame, get_stim_par_by_seg, kappas,
n_patches, n_segs_per_seq, ori_ran, phase, seg_len_s, sess, sf, size_ran,
stim_fps, stimtype, unexp_gabfr, unexp_gabfr_mean_oris, unexp_max_s,
unexp_min_s, win_size

Visflow: Session (758519303)
_set_block_params, block_params, deg_per_pix, exp_max_s, exp_min_s,
get_all_unexp_segs, get_all_unexp_stim_fr, get_dir_segs_exp, get_fr_by_seg,
get_frames_by_criteria, get_n_fr_by_seg, get_pup_diam_data,
get_pup_diam_stats_df, get_roi_data, get_roi_stats_df, get_run, get_run_data,
get_run_stats_df, get_segs_by_criteria, get_segs_by_frame, get_start_unexp_segs,
get_start_unexp_stim_fr_trans, get_stats_df, get_stim_beh_sub_df,
get_stim_df_by_criteria, main_flow_direcs, n_squares, prop_flipped, seg_len_s,
sess, speed, square_sizes, stim_fps, stimtype, unexp_max_s, unexp_min_s,
win_size

Grayscr: Session (758519303)
get_all_fr, get_start_fr, get_stop_fr, sess
```

List Stim object attribute values.

```
Gabors: Gabors (stimulus from session 758519303)
{'all_gabfr': ['A', 'B', 'C', 'D', 'U'],
 'all_gabfr_mean_oris': [0.0, 45.0, 90.0, 135.0, 180.0, 225.0],
 'block_params':    start_seg  stop_seg  num_segs  start_time_sec  stop_time_sec
duration_sec  \
0          1      6801      6800        43.540279    2085.237221   2041.696942
```

```
   start_frame_stim  stop_frame_stim  num_frames_stim  start_frame_twop  \
0              1800           124200           122400              1045   

   stop_frame_twop  num_frames_twop  gabor_kappa  
0            62456            61411           16  ,
 'deg_per_pix': 0.06251912565744862,
 'exp_gabfr': ['A', 'B', 'C', 'D'],
 'exp_gabfr_mean_oris': [0.0, 45.0, 90.0, 135.0],
 'exp_max_s': 90,
 'exp_min_s': 30,
 'kappas': [16],
 'n_patches': 30,
 'n_segs_per_seq': 5,
 'ori_ran': [0, 360],
 'phase': 0.25,
 'seg_len_s': 0.3,
 'sess': Session (758519303),
 'sf': 0.04,
 'size_ran': [159.9510532951381, 319.9021065902762],
 'stim_fps': 59.95049782968851,
 'stimtype': 'gabors',
 'unexp_gabfr': ['U'],
 'unexp_gabfr_mean_oris': [90.0, 135.0, 180.0, 225.0],
 'unexp_max_s': 6,
 'unexp_min_s': 3,
 'win_size': [1920, 1200]}

Visflow: Visflow (stimulus from session 758519303)
{'block_params':    start_seg  stop_seg  num_segs  start_time_sec  stop_time_sec
duration_sec  \
0       6802      7822      1020     2115.262241    3136.110155   1020.847915   
1       7823      8843      1020     3166.135568    4186.983399   1020.847832   

   start_frame_stim  stop_frame_stim  num_frames_stim  start_frame_twop  \
0            126000           187200            61200             63359   
1            189000           250200            61200             94967   

   stop_frame_twop  num_frames_twop main_flow_direction  square_size  \
0            94064            30705        right (temp)          128   
1           125671            30704        left (nasal)          128   

   square_number  
0            105  
1            105  ,
 'deg_per_pix': 0.06251912565744862,
 'exp_max_s': 90,
 'exp_min_s': 30,
 'main_flow_direcs': ['left (nasal)', 'right (temp)'],
 'n_squares': [105],
 'prop_flipped': 0.25,
 'seg_len_s': 1,
 'sess': Session (758519303),
 'speed': 799.7552664756905,
 'square_sizes': [128],
 'stim_fps': 59.95049782968851,
 'stimtype': 'visflow',
 'unexp_max_s': 4,
 'unexp_min_s': 2,
 'win_size': [1920, 1200]}

Grayscr: Grayscr (session 758519303)
{'sess': Session (758519303)}
```