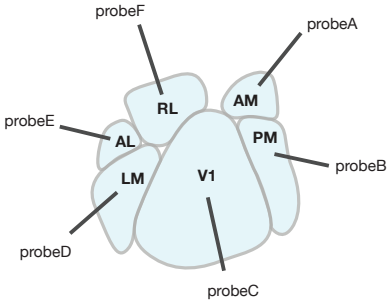
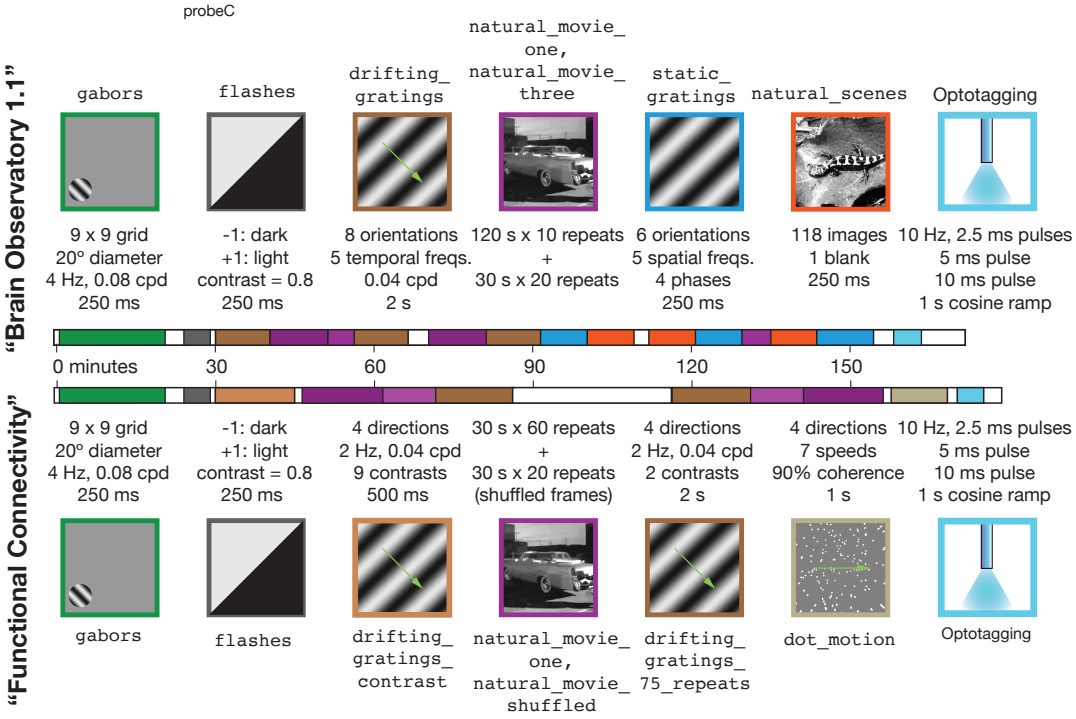
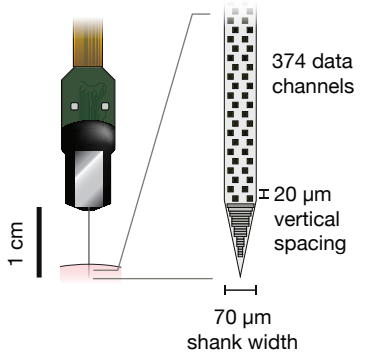
Allen Brain Observatory: Visual Coding Neuropixels Dataset



Metadata	`session.metadata`	`dict`
	`.probes`	`pandas.DataFrame`
	`.channels`	`pandas.DataFrame`
	`.units`	`pandas.DataFrame`
Stimuli	`session.stimulus_presentations`	`pandas.DataFrame`
	`.optogenetic_stimulation_epochs`	`pandas.DataFrame`
Spikes	`session.spike_times[unit_id]`	`numpy.ndarray`
	`.spike_amplitudes[unit_id]`	`numpy.ndarray`
	`.mean_waveforms[unit_id]`	`xarray.DataArray`
LFP	`session.get_lfp(probe_id)`	`xarray.DataArray`
	`.get_current_source_density(probe_id)`	`xarray.DataArray`
Behavior	`session.running_speed`	`pandas.DataFrame`
	`.get_pupil_data()`	`pandas.DataFrame`



Neuropixels probes



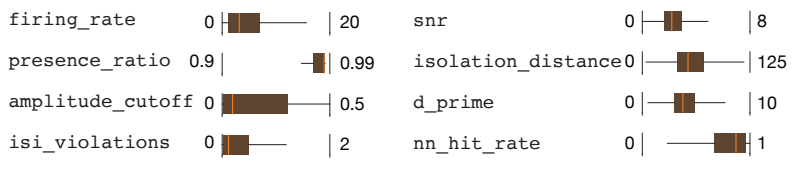
Spike band: ~30 kHz sample rate
500 Hz analog hi-pass
150 Hz digital hi-pass

LFP band: ~2.5 kHz original sample rate
1000 Hz analog lo-pass
625 Hz digital lo-pass
NWB includes every 2nd sample and every 4th channel

Available sessions

	WT	Pvalb	Sst	Vip
Brain Observatory 1.1	16	5	6	5
Functional Connectivity	14	3	6	3

Unit quality metrics



VISUAL CORTEX	primary visual cortex	VISp	3964[1] (8603[2])
	lateromedial area	VISl	2075 (4935)
	rostrolateral area	VISrl	2567 (6013)
	anterolateral area	VISal	3036 (6466)
	posteromedial area	VISpm	1798 (4215)
	anteromedial area	VISam	2959 (6198)
HIPPO-CAMPAL FORMATION	cornu ammonis 1	CA1	5878 (17,104)
	cornu ammonis 3	CA3	815 (3148)
	dentate gyrus	DG	1655 (5832)
	subiculum	SUB	850 (1938)
	prosubiculum	ProS	652 (1522)
THALAMUS	lateral geniculate nuc.	LGd	1306 (2582)
	lateral posterior nuc.	LP	2492 (4849)
MIDBRAIN	anterior pretectal nuc.	APN	1297 (3841)

[1]Total units passing default QC filters
[2]Total units (no QC filters)

Cheat sheet version 1.1, November 2019

AllenSDK Commands

Installation (using conda)

```
$ conda create -n allensdk python=3.7
$ source activate allensdk
$ conda activate allensdk
$ pip install allensdk
```

Where to go for help

Documentation: allensdk.readthedocs.io

Issues: github.com/alleninstitute/allensdk/issues

Forum: community.brain-map.org

Setting up a data cache

In [ ]:
```
from allensdk.brain_observatory.ecephys.ecephys_project_cache import EcephysProjectCache

data_directory = '/path/to/directory' # where the data will be stored
manifest_path = os.path.join(data_directory, 'manifest.json')

cache = EcephysProjectCache.from_warehouse(manifest=manifest_path)
```

Loading data for one session

In [ ]:
```
sessions = cache.get_session_table()

session = cache.get_session_data(sessions.index.values[i],
                                 isi_violations_maximum = np.inf,   # disable default threshold of 0.5
                                 amplitude_cutoff_maximum = np.inf, # disable default threshold of 0.1
                                 presence_ratio_minimum = -np.inf)  # disable default threshold of 0.9
```

Getting stimulus information

In [ ]:
```
session.stimulus_names                    # returns a list of stimulus names
session.get_stimulus_epochs()             # returns a DataFrame of stimulus epochs
session.stimulus_presentations            # returns a DataFrame of stimulus information
session.stimulus_conditions               # returns a DataFrame of unique conditions
session.get_stimulus_table(['flashes']) # returns a DataFrame for one stimulus type
session.optogenetic_stimulation_epochs  # returns a DataFrame of optotagging trial info
```

Aligning spike times to stimuli

In [ ]:
```
df = session.presentationwise_spike_times(
    stimulus_presentation_ids=presentation_ids,
    unit_ids=unit_ids) # returns a DataFrame of spike times aligned to presentation starts

da = session.presentationwise_spike_counts(
    bin_edges=time_bin_edges,
    stimulus_presentation_ids=presentation_ids,
    unit_ids=unit_ids) # returns a DataArray with dimensions of times x presentations x units
```

Accessing information about units across all sessions

In [ ]:
```
units = cache.get_units()
```

In [ ]:
```
analysis_metrics1 = cache.get_unit_analysis_metrics_by_session_type('brain_observatory_1.1')
analysis_metrics2 = cache.get_unit_analysis_metrics_by_session_type('functional_connectivity')

all_metrics = pd.concat([analysis_metrics1, analysis_metrics2], sort=False)
```