Python Documentation

version

May 13, 2020

Contents

Welcome to moseq2-pca's documentation!	1
moseq2-pca	1
moseq2_pca package	1
Subpackages	1
moseq2_pca.helpers package	1
Helpers - Data Module	1
Helpers - Wrapper Module	1
moseq2_pca.pca package	2
PCA - Utilties Module	2
CLI Module	5
cli	5
apply-pca	5
clip-scores	6
compute-changepoints	6
train-pca	7
GUI Module	9
Utilities Module	9
Visualization Module	12
Indices and tables	13
Index	15
Python Module Index	21

Welcome to moseq2-pca's documentation!

moseq2-pca

moseq2_pca package

Subpackages

moseq2_pca.helpers package

Helpers - Data Module

moseq2_pca.helpers.data.get_pca_yaml_data (pca_yaml)
 Reads PCA yaml file and returns metadata

Parameters: pca_yaml (str) (path to pca.yaml)

Returns: use_fft (bool) (indicates whether to use FFT) clean_params (dict) (dict of image filtering

parameters) mask_params (dict) (dict of mask parameters)) missing_data (bool)

(indicates whether to use mask_params)

moseq2_pca.helpers.data.load_pcs_for_cp (pca_file_components, config_data) Load computed Principal Components for Model-free Changepoint Analysis.

Parameters:

• pca_file_components (str) (path to pca h5 file to read PCs)

• config_data (dict) (config parameters)

Returns:

pca_components (str) (path to pca components) changepoint_params (dict) (dict of relevant changepoint parameters) cluster (dask Cluster) (Dask Cluster object.) client (dask Client) (Dask Client Object) missing_data (bool) (Indicates whether to use mask_params) mask_params (dict) (Mask parameters to use when computing CPs)

moseq2_pca.helpers.data.setup_cp_command (input_dir, config_data, output_dir, output_file,
output_directory=None)

Helper function for changepoints_wrapper to perform data-path existence checks.

Parameters:

- input_dir (int) (path to directory containing all h5+yaml files)
- config_data (dict) (dict of relevant PCA parameters (image filtering etc.))
- output_dir (str) (path to directory to store PCA data)
- output_file (str) (pca model filename)
- output_directory (str) (alternative output_dir)

Returns:

config_data (dict) (updated config_data dict with the proper paths) pca_file_components (str) (path to trained pca file) pca_file_scores (str) (path to pca_scores file) h5s (list) (list of relevant pca h5 files) yamls (list) (list of relevant pca metadata yaml files) save_file (str) (path to save changepoints)

Helpers - Wrapper Module

moseq2_pca.helpers.wrappers.apply_pca_wrapper (input_dir, config_data, output_dir,
output_file, output_directory=None, gui=False)
Wrapper function to obtain PCA Scores.

- input_dir (int) (path to directory containing all h5+yaml files)
- config_data (dict) (dict of relevant PCA parameters (image filtering etc.))
- output_dir (str) (path to directory to store PCA data)
- output_file (str) (pca model filename)
- output_directory (str) (alternative output_dir)
- gui (bool) (indicate GUI is running)

Returns: config_data (dict)

Return type: updated config_data variable to write back in GUI API

moseq2_pca.helpers.wrappers.compute_changepoints_wrapper (input_dir, config_data, output_dir, output_file, gui=False, output_directory=None)
Wrapper function to compute model-free (PCA based) Changepoints.

Parameters:

- input_dir (int) (path to directory containing all h5+yaml files)
- config_data (dict) (dict of relevant PCA parameters (image filtering etc.))
- output_dir (str) (path to directory to store PCA data)
- output_file (str) (pca model filename)
- output_directory (str) (alternative output_dir)
- gui (bool) (indicate GUI is running)

Returns: config_data (dict)

Return type: updated config_data variable to write back in GUI API

moseq2_pca.helpers.wrappers.train_pca_wrapper (input_dir, config_data, output_dir,
output_file, output_directory=None, gui=False)
Wrapper function to train PCA.

Parameters:

- input_dir (int) (path to directory containing all h5+yaml files)
- config_data (dict) (dict of relevant PCA parameters (image filtering etc.))
- output_dir (str) (path to directory to store PCA data)
- output_file (str) (pca model filename)
- output directory (str) (alternative output dir)
- gui (bool) (indicate GUI is running)

Returns: config_data (dict)

Return type: updated config_data variable to write back in GUI API

moseq2_pca.pca package

PCA - Utilties Module

moseq2_pca.pca.util.apply_pca_dask (pca_components, h5s, yamls, use_fft, clean_params, save_file, chunk_size, mask_params, missing_data, client, fps=30, gui=False)

"Apply" trained PCA on input frame data to obtain PCA Scores using Distributed Dask cluster.

- pca_components (np.array) (array of computed Principal Components)
- h5s (list) (list of h5 files)
- yamls (list) (list of yaml files)
- use_fft (bool) (indicate whether to use 2D-FFT)
- clean_params (dict) (dictionary containing filtering options)
- save_file (str) (path to pca_scores filename to save)
- chunk_size (int) (size of chunks to process)
- mask_params (dict) (dictionary of masking parameters (if missing data))
- missing_data (bool) (indicates whether to use mask arrays.)
- fps (int) (frames per second)

Returns:

Return type: None

moseq2_pca.pca.util.apply_pca_local (pca_components, h5s, yamls, use_fft, clean_params, save_file, chunk_size, mask_params, missing_data, fps=30)

"Apply" trained PCA on input frame data to obtain PCA Scores using local cluster/platform.

Parameters:

- pca_components (np.array) (array of computed Principal Components)
- h5s (list) (list of h5 files)
- yamls (list) (list of yaml files)
- use_fft (bool) (indicate whether to use 2D-FFT)
- clean_params (dict) (dictionary containing filtering options)
- save_file (str) (path to pca_scores filename to save)
- chunk_size (int) (size of chunks to process)
- mask_params (dict) (dictionary of masking parameters (if missing data))
- missing_data (bool) (indicates whether to use mask arrays.)
- fps (int) (frames per second)

Returns:

Return type: None

moseq2_pca.pca.util.get_changepoints_dask (changepoint_params, pca_components, h5s, yamls, save_file, chunk_size, mask_params, missing_data, client, fps=30, pca_scores=None, progress_bar=False, gui=False)

Computes model-free changepoints using PCs and PC Scores on distributed dask cluster.

- changepoint_params (dict) (dict of changepoint parameters)
- pca_components (np.array) (computed principal components)
- h5s (list) (list of h5 files)
- yamls (list) (list of yaml files)
- save_file (str) (path to save changepoint files)
- chunk_size (int) (size of chunks to process in dask.)
- mask_params (dict) (dict of missing_data mask parameters.)
- missing_data (bool) (indicate whether to use mask_params)
- client (dask Client) (initialized Dask Client object)
- fps (int) (frames per second)
- pca_scores (np.array) (computed principal component scores)
- progress_bar (bool) (display progress bar)
- gui (bool) (indicate GUI use)

Returns:

Return type: None

moseq2_pca.pca.util.mask_data (original_data, mask, new_data)
Create a mask subregion given a boolean mask if missing data flag is used.

Parameters:

- original_data (3d np.ndarray) (input frames)
- mask (3d boolean np.ndarray) (mask array)
- new_data (3d np.ndarray) (frames to use)

Returns: output (3d np.ndarray)

Return type: masked data array

moseq2_pca.pca.util.train_pca_dask (dask_array, clean_params, use_fft, rank, cluster_type, client, workers, cache, mask=None, iters=10, recon_pcs=10, min_height=10, max_height=100)
 Train PCA using dask arrays.

Parameters:

- dask_array (dask array) (chunked frames to train PCA)
- **clean_params (dict)** (dictionary containing filtering parameters)
- use_fft (bool) (indicates whether to use 2d-FFT on images.)
- rank (int) (Matrix rank to use)
- cluster type (str) (indicates which cluster to use.)
- client (Dask.Client) (client object to execute dask operations)
- workers (int) (number of dask workers)
- cache (str) (path to cache directory)
- mask (dask array) (dask array of masked data if missing_data parameter==True)
- iters (int) (number of SVD iterations)
- recon pcs (int) (number of PCs to reconstruct. (if missing data = True))
- min_height (int) (minimum mouse height from floor in (mm))
- max_height (int) (maximum mouse height from floor in (mm))

Returns: output_dict (dict)

Return type: dictionary containing PCA training results.

CLI Module

```
cli
```

```
cli [OPTIONS] COMMAND [ARGS]...
```

apply-pca

```
cli apply-pca [OPTIONS]
```

Options

- -i, --input-dir <input_dir>
 Directory to find h5 files [default: /Users/aymanzeine/Desktop/moseq/moseq2-pca/docs]
- --cluster-type <cluster_type>
 Cluster type [default: local]

Options: local|slurm|nodask

- -o, --output-dir <output_dir>
 Directory to store results [default: /Users/aymanzeine/Desktop/moseq/moseq2-pca/docs/_pca]
- --output-file <output_file>
 Name of h5 file for storing pca results [default: pca_scores]
- --h5-path <h5_path>
 Path to data in h5 files [default: /frames]
- --h5-mask-path <h5_mask_path>
 Path to log-likelihood mask in h5 files [default: /frames_mask]
- --pca-path pca_path>
 Path to pca components [default: /components]
- --pca-file <pca_file>
 Path to PCA results
- --chunk-size <chunk_size>
 Number of frames per chunk [default: 4000]
- --fill-gaps <fill_gaps>
 Fill dropped frames with nans [default: True]
- --fps <fps>
 Fps (only used if no timestamps found) [default: 30]
- --detrend-window <detrend_window>
 Length of detrend window (in seconds, 0 for no detrending) [default: 0]
- --config-file <config_file>
 Path to configuration file
- -d, --dask-cache-path <dask_cache_path>
 Path to spill data to disk for dask local scheduler [default: /Users/aymanzeine/moseq2_pca]
- -q, --queue <queue>
 Cluster queue/partition for submitting jobs [default: debug]
- -n, --nworkers <nworkers>
 Number of workers [default: 10]
- -c, --cores <cores>
 Number of cores per worker [default: 1]
- -p, --processes processes>
 Number of processes to run on each worker [default: 1]
- -m, --memory <memory>

RAM usage per workers [default: 15GB]

-w, --wall-time <wall_time>
Wall time for workers [default: 06:00:00]

--timeout <timeout>

Time to wait for workers to initialize before proceeding (minutes) [default: 5]

clip-scores

Clips PCA scores from the beginning or end

Args:

pca_file (string): Path to PCA scores clip_samples (int): number of samples to clip from beginning or end from_end (bool): if true clip from end rather than beginning

Note that scores are modified in place.

cli clip-scores [OPTIONS] PCA_FILE CLIP_SAMPLES

Options

--from-end

[default: False]

Arguments

PCA_FILE

Required argument

CLIP_SAMPLES

Required argument

compute-changepoints

cli compute-changepoints [OPTIONS]

Options

-i, --input-dir <input_dir>

Directory to find h5 files [default: /Users/aymanzeine/Desktop/moseq/moseq2-pca/docs]

-o, --output-dir <output_dir>

Directory to store results [default: /Users/aymanzeine/Desktop/moseq/moseq2-pca/docs/_pca/]

--output-file <output_file>

Name of h5 file for storing pca results [default: changepoints]

--cluster-type <cluster_type>

Cluster type [default: local]

Options: local|slurm

--pca-file-components <pca_file_components>

Path to PCA components

--pca-file-scores <pca_file_scores>

Path to PCA results

--pca-path <pca_path>

Path to pca components [default: /components]

--neighbors <neighbors>

Neighbors to use for peak identification [default: 1]

--threshold <threshold>

Peak threshold to use for changepoints [default: 0.5]

-k, --klags <klags>

Lag to use for derivative calculation [default: 6]

```
-s, --sigma <sigma>
  Standard deviation of gaussian smoothing filter [default: 3.5]
-d, --dims <dims>
 Number of random projections to use [default: 300]
--fps <fps>
  Fps (only used if no timestamps found) [default: 30]
--h5-path <h5_path>
  Path to data in h5 files [default: /frames]
--h5-mask-path <h5_mask_path>
  Path to log-likelihood mask in h5 files [default: /frames mask]
--chunk-size <chunk_size>
  Number of frames per chunk [default: 4000]
--config-file <config_file>
  Path to configuration file
--dask-cache-path <dask_cache_path>
  Path to spill data to disk for dask local scheduler [default: /Users/aymanzeine/moseq2_pca]
--visualize-results <visualize_results>
  Visualize results [default: True]
-q, --queue <queue>
 Cluster queue/partition for submitting jobs [default: debug]
-n, --nworkers <nworkers>
 Number of workers [default: 10]
-c, --cores <cores>
 Number of cores per worker [default: 1]
-p, --processes cesses>
 Number of processes to run on each worker [default: 1]
-m, --memory <memory>
  RAM usage per workers [default: 15GB]
-w, --wall-time <wall_time>
 Wall time for workers [default: 06:00:00]
--timeout <timeout>
  Time to wait for workers to initialize before proceeding (minutes) [default: 5]
train-pca
cli train-pca [OPTIONS]
                                                 Options
-i, --input-dir <input_dir>
  Directory to find h5 files [default: /Users/aymanzeine/Desktop/moseq/moseq2-pca/docs]
--cluster-type <cluster_type>
  Cluster type [default: local]
         Options: local|slurm
-o, --output-dir <output_dir>
  Directory to store results [default: /Users/aymanzeine/Desktop/moseq/moseq2-pca/docs/_pca]
--gaussfilter-space <gaussfilter_space>
  Spatial filter for data (Gaussian) [default: 1.5, 1]
--gaussfilter-time <gaussfilter_time>
  Temporal filter for data (Gaussian) [default: 0]
```

```
--medfilter-space <medfilter space>
 Median spatial filter [default: 0]
--medfilter-time <medfilter_time>
 Median temporal filter [default: 0]
--missing-data
 Use missing data PCA [default: False]
--missing-data-iters <missing_data_iters>
 Missing data PCA iterations [default: 10]
--mask-threshold <mask_threshold>
 Threshold for mask (missing data only) [default: -16]
--mask-height-threshold <mask_height_threshold>
 Threshold for mask based on floor height [default: 5]
--min-height <min_height>
 Min mouse height from floor (mm) [default: 10]
--max-height <max_height>
 Max mouse height from floor (mm) [default: 100]
--tailfilter-size <tailfilter_size>
 Tail filter size [default: 9, 9]
--tailfilter-shape <tailfilter_shape>
 Tail filter shape [default: ellipse]
--use-fft
 Use 2D fft [default: False]
--recon-pcs <recon_pcs>
 Number of PCs to use for missing data reconstruction [default: 10]
--rank <rank>
 Rank for compressed SVD (generally>>nPCS) [default: 50]
--output-file <output_file>
 Name of h5 file for storing pca results [default: pca]
--chunk-size <chunk_size>
 Number of frames per chunk [default: 4000]
--visualize-results <visualize_results>
 Visualize results [default: True]
--config-file <config_file>
 Path to configuration file
-d, --dask-cache-path <dask_cache_path>
 Path to spill data to disk for dask local scheduler [default: /Users/aymanzeine/moseq2_pca]
--local-processes <local_processes>
 Use processes with local scheduler [default: True]
-q, --queue <queue>
 Cluster queue/partition for submitting jobs [default: debug]
-n, --nworkers <nworkers>
 Number of workers [default: 10]
-c, --cores <cores>
 Number of cores per worker [default: 1]
-p, --processes cesses>
 Number of processes to run on each worker [default: 1]
-m, --memory <memory>
 Total RAM usage per worker [default: 15GB]
-w, --wall-time <wall_time>
```

Wall time for workers [default: 06:00:00]

--timeout <timeout>

Time to wait for workers to initialize before proceeding (minutes) [default: 5]

GUI Module

moseq2_pca.gui.apply_pca_command (input_dir, index_file, config_file, output_dir,
output_file, output_directory=None)

Compute PCA Scores given trained PCA using Jupyter Notebook.

Parameters:

- input_dir (str) (path to directory containing training data)
- index_file (str) (path to index file.)
- config_file (str) (path to config file)
- output_dir (str) (path to output pca directory)
- output_file (str) (name of output pca file.)
- output_directory (str) (alternative output directory path)

Returns: (str)

Return type: success string.

moseq2_pca.gui.compute_changepoints_command (input_dir, config_file, output_dir,
output_file, output_directory=None)

Compute Changepoint distribution using Jupyter Notebook.

Parameters:

- input_dir (str) (path to directory containing training data)
- config_file (str) (path to config file)
- output_dir (str) (path to output pca directory)
- output_file (str) (name of output pca file.)
- output_directory (str) (alternative output directory path)

Returns: (str)

Return type: success string.

moseq2_pca.gui.train_pca_command (input_dir, config_file, output_dir, output_file, output_directory=None)

Train PCA through Jupyter notebook, and updates config file.

Parameters:

- input_dir (str) (path to directory containing training data)
- config_file (str) (path to config file)
- output_dir (str) (path to output pca directory)
- output_file (str) (name of output pca file.)
- output_directory (str) (alternative output directory path)

Returns:

Return type: None

Utilities Module

moseq2_pca.util.clean_frames (frames, medfilter_space=None, gaussfilter_space=None,
medfilter_time=None, gaussfilter_time=None, detrend_time=None, tailfilter=None,
tail_threshold=5)

Filters spatial/temporal noise from frames using Median and Gaussian filters, given kernel sizes for each respective requested filter.

```
Parameters:
```

- frames (3D numpy array) (frames to filter.)
- medfilter_space (list) (median spatial filter kernel.)
- gaussfilter_space (list) (gaussian spatial filter kernel.)
- medfilter_time (list) (median temporal filter.)
- gaussfilter_time (list) (gaussian temporal filter.)
- detrend_time (int) (number of frames to lag for.)
- tailfilter (int) (size of tail-filter kernel.)
- tail_threshold (int) (threshold value to use for tail filtering)

Returns: out (3D numpy array)

Return type: filtered frames.

moseq2_pca.util.command_with_config (config_file_param_name)

moseq2_pca.util.gauss_smooth (signal, win_length=None, sig=1.5, kernel=None) Perform Gaussian Smoothing on a 1D signal.

Parameters:

- signal (1d numpy array) (signal to perform smoothing)
- win_length (int) (window_size for gaussian kernel filter)
- sig (float) (variance of 1d gaussian kernel.)
- kernel (tuple) (kernel size to use for smoothing)

Returns: result (1d numpy array)

Return type: smoothed signal

moseq2_pca.util.gaussian_kernelld (n=None, sig=3)
Get 1D gaussian kernel.

Parameters:

- n (int) (number of points to use.)
- sig (int) (variance of kernel to use.)

Returns: kernel (1d array)
Return type: 1D numpy kernel.

moseq2_pca.util.get_changepoints (scores, k=5, sigma=3, peak_height=0.5, peak_neighbors=1, baseline=True, timestamps=None)

Compute changepoints distribution and CP Curve.

Parameters:

- scores (3D numpy array) (nframes * r * c)
- k (int) (klags Lag to use for derivative calculation.)
- sigma (int) (Standard deviation of gaussian smoothing filter.)
- peak_height (float) (user-defined peak Changepoint length.)
- peak neighbors (int) (number of peaks in the CP curve.)
- baseline (bool) (normalize data.)
- timestamps (array) (loaded timestamps.)

Returns: cps (2D numpy array) (array of values for CP curve) normed_df (1D numpy array) (array of values for bar plot)

moseq2_pca.util.get_metadata_path(h5file)

Return path within h5 file that contains the kinect extraction metadata.

Parameters: h5file (str) (path to h5 file.)

Returns: (str)

Return type: path to acquistion metadata within h5 file.

moseq2_pca.util.get_rps (frames, rps=600, normalize=True)
Get random projections of frames.

Parameters:

- frames (2D or 3D numpy array) (Frames to get dimensions from.)
- rps (int) (Number of random projections.)
- normalize (bool) (indicates whether to normalize frames.)

Returns: rproj (2D or 3D numpy array)

Return type: Computed random projections with same shape as frames

moseq2_pca.util.get_timestamp_path(h5file)

Return path within h5 file that contains the kinect timestamps

Parameters: h5file (str) (path to h5 file.)

Returns: (str)

Return type: path to metadata timestamps within h5 file

moseq2_pca.util.initialize_dask (nworkers=50, processes=1, memory='4GB', cores=1,
wall_time='01:00:00', queue='debug', local_processes=False, cluster_type='local',
scheduler='distributed', timeout=10, cache_path='/Users/aymanzeine/moseq2_pca', **kwargs)
 Initialize dask client, cluster, workers, etc.

Parameters:

- nworkers (int) (number of dask workers to initialize)
- processes (int) (number of processes per worker)
- memory (str) (amount of memory to allocate to dask cluster)
- cores (int) (number of cores to use.)
- wall_time (str) (amount of time to allow program to run)
- queue (str) (logging mode)
- local_processes (bool) (indicate whether the processes are local)
- cluster_type (str) (indicate what cluster to use)
- scheduler (str) (indicate what scheduler to use)
- timeout (int) (number of worker timeouts to allow)
- cache_path (str or Pathlike) (path to store cached data)
- kwargs (extra keyward arguments)

Returns: client (dask Client) (initialized Client) cluster (dask Cluster) (initialized Cluster) workers (dask Workers) (intialized workers or None if cluster_type = 'local') cache (dask Chest) (initialized Chest (cache) object pointing to given cache path)

moseq2_pca.util.insert_nans (timestamps, data, fps=30)
Fills NaN values with 0 in timestamps.

Parameters:

- timestamps (1D array) (timestamp time-strs)
- data (1D array) (timestamp values)
- fps (int) (frames per second)

Returns: filled_data (1D array) (filled missing timestamp values.) data_idx (1D array) (indices of inserted 0s) filled_timestamps (1D array) (filled timestamp-strs)

moseq2_pca.util.read_yaml (yaml_file)

Reads yaml file and returns dictionary representation of file contents.

Parameters: yaml_file (str) (path to yaml file)

Returns: return_dict (dict)

Return type: dict of yaml file contents

```
moseq2_pca.util.recursive_find_h5s
(root_dir='/Users/aymanzeine/Desktop/moseq/moseq2-pca/docs', ext='.h5',
yaml_string='{}.yaml')
   Recursively find h5 files, along with yaml files with the same basename
```

- root_dir (str or os.Pathlike) (path to directory to start recursive search)
- ext (str) (extension to search for, e.g. .h5)

• yaml string (str) (a format to use to name yaml files)

Returns: h5s (list) (list of h5 file paths) dicts (list) (list of dicts containing metadata file contents)

yamls (list) (list of yaml file paths)

moseq2_pca.util.recursively_load_dict_contents_from_group (h5file, path) Reads all contents from h5 and returns them in a nested dict object.

Parameters:

• h5file (str) (path to h5 file)

• path (str) (path to group within h5 file)

Returns: ans (dict)

Return type: dictionary of all h5 group contents

moseq2_pca.util.select_strel (string='e', size=(10, 10))
 Selects Structuring Element Shape

Parameters:

• string (str) (e for Ellipse, r for Rectangle)

• size (tuple) (size of StructuringElement)

Returns: strel (cv2.StructuringElement)

Return type: returned StructuringElement with specified size.

moseq2_pca.util.shutdown_dask (scheduler)

Graceful shutdown dask scheduler. source:

https://github.com/dask/distributed/issues/1703#issuecomment-361291492

Parameters: scheduler (dask Scheduler) (scheduler to shutdown.)

Returns:

Return type: None

Visualization Module

moseq2_pca.viz.changepoint_dist (cps, headless=False)
Creates bar plot describing computed Changepoint Distribution.

Parameters:

• cps (np.ndarray) (changepoints to graph)

• headless (bool) (trim first element in PC list)

Returns: plt (plt.figure) (figure to save/graph) ax (plt.ax) (figure axis variable)

moseq2_pca.viz.display_components (components, cmap='gray', headless=False) Creates grid of computed Principal Components.

Parameters:

- components (2D np.ndarray) (components to graph)
- cmap (str) (color map to use)

• headless (bool) (trim first element in PC list)

Returns: plt (plt.figure) (figure to save/graph) ax (plt.ax) (figure axis variable)

moseq2_pca.viz.scree_plot (explained_variance_ratio, headless=False)
Creates Scree plot describing principal components.

• explained_variance_ratio (1D np.array) (explained variance ratio of each principal

component)

• headless (bool) (trim first element in PC list)

Returns: plt (plt.figure)

Return type: figure to save/graph

Indices and tables

- genindex
- modindex
- search

Index

--gaussfilter-time <gaussfilter_time> cli-train-pca command line option cli-apply-pca command **Symbols** --h5-mask-path <h5_mask_path> line option cli-apply-pca --chunk-size <chunk size> command cli-compute-changepoints line option command line option cli-compute-changepoints --h5-path <h5_path> cli-apply-pca command command line option line option cli-train-pca command cli-compute-changepoints line option command line option command. --input-dir <input_dir> --cluster-type <cluster_type> cli-apply-pca cli-apply-pca command line option line option cli-compute-changepoints cli-compute-changepoints command line option command line option cli-train-pca command cli-train-pca command line option line option cli-apply-pca --config-file <config_file> command --klags <klags> cli-compute-changepoints line option command line option ${\tt cli-compute-changepoints}_{\tt local-processes} \verb| <local_processes > \\$ cli-train-pca command line option command line option cli-train-pca command line option --mask-height-threshold <mask_height_threshold> cli-train-p command --cores <cores> cli-apply-pca command line option line option cli-train-pca --mask-threshold <mask_threshold> cli-compute-changepoints command command line option line option cli-train-pca command cli-train-pca --max-height <max height> line option command line option -dask-cache-path <dask_cache_path> cli-apply-pca command line option cli-train-pca --medfilter-space <medfilter_space> command cli-compute-changepoints line option command line option --medfilter-time <medfilter_time> cli-train-pca cli-train-pca command line option line option --detrend-window <detrend_window> cli-apply-pca --memory <memory> cli-apply-pca command command line option line option cli-compute-changepoints --dims <dims> cli-compute-changepoints command line option command line option cli-apply-pca cli-train-pca command --fill-gaps <fill_gaps> command line line option option --min-height <min_height> cli-train-pca command line cli-apply-pca command --fps <fps> option line option cli-train-pca command --missing-data cli-compute-changepoints line option command line option cli-clip-scores command -missing-data-iters <missing_data_iters> cli-train-pca --from-end command

cli-train-pca

line option

command -neighbors <neighbors>

line option

cli-compute-changepoints command line option

line option

--gaussfilter-space <gaussfilter_space>

nworkers		cli-apply-pca line option	command	threshold	<threshold></threshold>	cli-comput	e-changepoints line option	•
		cli-compute-c		timeout	<timeout></timeout>	cli-apply-pca line option	command	
		cli-train-pca line option	command			cli-compute-c command line		
output-dir	<output_dir></output_dir>	cli-apply-policyline option	ca command	I		cli-train-pca line option	command	
		cli-compute command l	•			cli-train-pca line option	command	
		cli-train-pca	avisual command	ize-results	<visualize_< td=""><td>_results></td><td>cli-compute-ch command line</td><td></td></visualize_<>	_results>	cli-compute-ch command line	
output-file	<pre><output_file< pre=""></output_file<></pre>	line option	า				cli-train-pca line option	comm
		cli-compu command	te-changepoin I line option	wall-time ts	<wall_time></wall_time>	•		
		cli-train-p		nd		cli-comput command	e-changepoints line option	
pca-file	e <pca_file></pca_file>	cli-apply-pc command li				cli-train-pc line option		
-file-components	<pca_file_co< td=""><td>omponents></td><td>cli-compute-c</td><td></td><td></td><td>cli-apply-pca line option</td><td>command</td><td></td></pca_file_co<>	omponents>	cli-compute-c			cli-apply-pca line option	command	
-pca-file-scores	<pca_file_so< td=""><td></td><td>ompute-chang mand line opti</td><td></td><td></td><td>cli-compute-cl command line</td><td>•</td><td></td></pca_file_so<>		ompute-chang mand line opti			cli-compute-cl command line	•	
pca-path		cli-apply-pca line option	command			cli-train-pca line option	command	
		cli-compute-c command line	• .	-d		cli-apply-pca line option	command	
processes	<pre><pre><pre><pre><pre><pre><pre><pre></pre></pre></pre></pre></pre></pre></pre></pre>	cli-apply-pca	command			cli-compute-cl command line		
		cli-compute- command lir	changepoints ne option			cli-train-pca line option	command	
		cli-train-pca line option	command	-i		cli-apply-pca line option	command	
queue <		li-apply-pca ne option	command			cli-compute-cl	• .	
		li-compute-cha ommand line c				cli-train-pca line option	command	
		li-train-pca ne option	command	-k		cli-compute-cl command line		
rank <ra< td=""><td></td><td>cli-train-pca ine option</td><td>command</td><td>-m</td><td></td><td>cli-apply-pca line option</td><td>command</td><td></td></ra<>		cli-train-pca ine option	command	-m		cli-apply-pca line option	command	
recon-po	cs <recon_pc< td=""><td>s> cli-train-p comman option</td><td></td><td></td><td></td><td>cli-compute-cl command line</td><td>•</td><td></td></recon_pc<>	s> cli-train-p comman option				cli-compute-cl command line	•	
sigma <s< td=""><td></td><td>li-compute-cha ommand line o</td><td></td><td></td><td></td><td>cli-train-pca line option</td><td>command</td><td></td></s<>		li-compute-cha ommand line o				cli-train-pca line option	command	
tailfilter-s			•			cli-apply-pca line option	command	
tailfilter-	-size <tailf:< td=""><td>ilter size></td><td>line option</td><td></td><td></td><td>cli-compute-cl command line</td><td>•</td><td></td></tailf:<>	ilter size>	line option			cli-compute-cl command line	•	
		<u></u>	command line option			cli-train-pca line option	command	

-0	cli-apply-pca line option	command	input-dir <input_dir></input_dir>			
	cli-compute-changepoints		memory <memory></memory>			
	command line option		nworkers <nworkers></nworkers>			
	cli-train-pca	command	output-dir <output_dir></output_dir>			
	line option		output-file <output_file></output_file>			
-p	cli-apply-pca line option	command	pca-file <pca_file></pca_file>			
	cli-compute-ch	angepoints	pca-path <pca_path></pca_path>			
	command line	option	processes <pre><pre><pre><pre></pre></pre></pre></pre>			
	cli-train-pca line option	command	queue <queue></queue>			
-q	cli-apply-pca	command	timeout <timeout></timeout>			
-	line option		wall-time <wall_time></wall_time>			
	cli-compute-ch		-C			
	command line	•	-d			
	cli-train-pca line option	command	-i			
-s	cli-compute-ch		-m			
	command line	•	-n			
−M	cli-apply-pca line option	command	-0			
	cli-compute-ch	angepoints	-p			
	command line		-q			
	cli-train-pca	command	-w			
	line option		cli-clip-scores command line optionfrom-end			
A			CLIP_SAMPLES			
apply_pca_command() (in r	nodule moseq2 __	_pca.gui)	PCA_FILE			
apply_pca_dask() (in modul	·		cli-compute-changepoints command line option			
apply_pca_local() (in modul			chunk-size <chunk_size></chunk_size>			
apply_pca_wrapper()	(in	module	cluster-type <cluster_type></cluster_type>			
moseq2_pca.helpers.wrapp	ers)		config-file <config_file></config_file>			
•			cores <cores></cores>			
C			dask-cache-path <dask_cache_path></dask_cache_path>			
changepoint_dist() (in modu	·	•	dims <dims></dims>			
<pre>clean_frames() (in module r cli-apply-pca command lin</pre>	. –.)	fps <fps></fps>			
chunk-size <chunk_si< td=""><td>-</td><td></td><td>h5-mask-path <h5_mask_path></h5_mask_path></td></chunk_si<>	-		h5-mask-path <h5_mask_path></h5_mask_path>			
cluster-type <cluster_< td=""><td></td><td></td><td>h5-path <h5_path></h5_path></td></cluster_<>			h5-path <h5_path></h5_path>			
config-file <config_file></config_file>			input-dir <input_dir></input_dir>			
cores <cores></cores>			klags <klags></klags>			
dask-cache-path <das< td=""><td>sk cache path></td><td>•</td><td>memory <memory></memory></td></das<>	sk cache path>	•	memory <memory></memory>			
detrend-window <detr< td=""><td>•</td><td></td><td>neighbors <neighbors></neighbors></td></detr<>	•		neighbors <neighbors></neighbors>			
fill-gaps <fill_gaps></fill_gaps>			nworkers <nworkers></nworkers>			
fps <fps></fps>			output-dir <output_dir></output_dir>			
h5-mask-path <h5_m< td=""><td>ask nath></td><td></td><td>output-file <output_file></output_file></td></h5_m<>	ask nath>		output-file <output_file></output_file>			
h5-path <h5_path></h5_path>	aon_pani>		pca-file-components <pca_file_components></pca_file_components>			
110-pati1 <110_pati1>						

```
--pca-file-scores <pca_file_scores>
                                                               --processes cesses>
    --pca-path <pca_path>
                                                               --queue <queue>
    --processes cesses>
                                                               --rank <rank>
    --queue <queue>
                                                               --recon-pcs <recon_pcs>
    --sigma <sigma>
                                                               --tailfilter-shape <tailfilter_shape>
    --threshold <threshold>
                                                               --tailfilter-size <tailfilter_size>
    --timeout <timeout>
                                                               --timeout <timeout>
    --visualize-results <visualize_results>
                                                               --use-fft
    --wall-time <wall time>
                                                               --visualize-results <visualize results>
                                                               --wall-time <wall time>
    -C
    -d
                                                               -C
    -i
                                                               -d
    -k
                                                               -i
    -m
                                                               -m
    -n
                                                               -n
    -0
                                                               -0
    -p
                                                               -p
    -q
                                                                -q
    -S
                                                           CLIP SAMPLES
    -W
                                                               cli-clip-scores command line option
cli-train-pca command line option
    --chunk-size <chunk size>
                                                           command_with_config() (in module moseq2_pca.util)
    --cluster-type <cluster type>
                                                           compute changepoints command()
                                                                                                   (in
                                                                                                          module
                                                           moseq2_pca.gui)
    --config-file <config_file>
                                                           compute_changepoints_wrapper()
                                                                                                          module
                                                                                                  (in
    --cores <cores>
                                                           moseq2_pca.helpers.wrappers)
    --dask-cache-path <dask_cache_path>
                                                           D
    --gaussfilter-space <gaussfilter_space>
    --gaussfilter-time <gaussfilter time>
                                                           display_components() (in module moseq2_pca.viz)
    --input-dir <input_dir>
                                                           G
    --local-processes < local processes>
                                                           gauss_smooth() (in module moseq2_pca.util)
    --mask-height-threshold <mask height threshold>
                                                           gaussian_kernel1d() (in module moseq2_pca.util)
    --mask-threshold <mask threshold>
                                                           get_changepoints() (in module moseq2_pca.util)
    --max-height <max_height>
                                                                                                          module
                                                           get_changepoints_dask()
                                                                                              (in
    --medfilter-space <medfilter space>
                                                           moseq2_pca.pca.util)
    --medfilter-time <medfilter time>
                                                           get_metadata_path() (in module moseq2_pca.util)
    --memory <memory>
                                                           get pca yaml data()
                                                                                            (in
                                                                                                          module
    --min-height <min height>
                                                           moseq2_pca.helpers.data)
    --missing-data
                                                           get_rps() (in module moseq2_pca.util)
    --missing-data-iters <missing data iters>
                                                           get_timestamp_path() (in module moseq2_pca.util)
    --nworkers < nworkers>
                                                           Ī
    --output-dir <output_dir>
                                                           initialize_dask() (in module moseq2_pca.util)
    --output-file <output_file>
```

```
insert_nans() (in module moseq2_pca.util)
load pcs for cp()
                             (in
                                             module
moseq2_pca.helpers.data)
M
mask_data() (in module moseq2_pca.pca.util)
moseq2_pca.gui (module)
moseq2_pca.helpers.data (module)
moseq2_pca.helpers.wrappers (module)
moseq2_pca.pca.util (module)
moseq2_pca.util (module)
moseq2_pca.viz (module)
P
PCA FILE
    cli-clip-scores command line option
R
```

read_yaml() (in module moseq2_pca.util) recursive_find_h5s() (in module moseq2_pca.util) recursively_load_dict_contents_from_group() (in module moseq2_pca.util)

scree_plot() (in module moseq2_pca.viz) select_strel() (in module moseq2_pca.util)

setup_cp_command() (in module moseq2_pca.helpers.data)

shutdown_dask() (in module moseq2_pca.util)

T

```
train_pca_command() (in module moseq2_pca.gui)
train_pca_dask() (in module moseq2_pca.pca.util)
train_pca_wrapper() (in module
moseq2_pca.helpers.wrappers)
```

Python Module Index

m

moseq2_pca.gui
moseq2_pca.helpers.data
moseq2_pca.helpers.wrappers
moseq2_pca.pca.util
moseq2_pca.util
moseq2_pca.viz