

# mbo\_utilities

Cheat Sheet - Python API, CLI & GUI

---

## Python API: Core I/O

`imread(path)`

Lazy-load any supported format

`imwrite(arr, path, ext)`

Stream-write to disk

`get_metadata(path)`

Extract file metadata dict

`get_voxel_size(path)`

Get physical dimensions ( $\mu\text{m}$ )

`get_files(path, **kw)`

Discover files with filtering

## Examples

**# Load data**

```
from mbo_utilities import imread
arr = imread('/path/to/data.tiff')
arr = imread('/path/to/raw/') # ScanImage
```

**# Write data**

```
from mbo_utilities import imwrite
imwrite(arr, 'out.zarr', ext='.zarr')
imwrite(arr, 'out/', planes=[0,1,2])
```

**# Get metadata**

```
from mbo_utilities import get_metadata
meta = get_metadata('/path/to/data')
print(meta['nframes'], meta['shape'])
```

## Utilities & Visualization

`save_mp4(fname, images, **kw)`

Export video from 3D array

`save_png(fname, data)`

Save image via matplotlib

`norm_minmax(images)`

Normalize to 0-1 range

`smooth_data(data, window)`

Temporal smoothing

`subsample_array(arr, factor)`

Downsample array

`files_to_dask(files)`

Build Dask array from files

`expand_paths(paths)`

Expand wildcards/lists

## Examples

**# Video export**

```
from mbo_utilities import save_mp4
save_mp4('movie.mp4', arr[:500],
        framerate=30, temporal_avg=5)
```

**# Dask arrays**

```
from mbo_utilities import files_to_dask
darr = files_to_dask(tiff_files,
                    chunk_t=250)
```

# CLI Commands

|                                      |                        |   |
|--------------------------------------|------------------------|---|
| <code>mbo view [PATH]</code>         | Launch GUI viewer      | <code>--roi 0,1 --widget --metadata</code>              |
| <code>mbo convert IN OUT</code>      | Convert formats        | <code>-e .zarr -p 0,1,2 --fix-phase --register-z</code> |
| <code>mbo info PATH</code>           | Show file metadata     | <code>--metadata</code>                                 |
| <code>mbo scanphase [PATH]</code>    | Analyze scan phase     | <code>-o output/ --format png --show</code>             |
| <code>mbo formats</code>             | List supported formats |   |
| <code>mbo --download-notebook</code> | Get user guide         | <code>[PATH]</code>                                     |

## Quick Examples

```
mbo /path/to/data.tiff          # View TIFF in GUI
mbo convert raw/ out.zarr -e .zarr # Convert to Zarr
mbo convert data.tiff out/ --fix-phase # Fix bidirectional scan
mbo view data/ --roi 0,1        # View specific ROIs
```

# Supported Formats

## Input

|                          |                              |
|--------------------------|------------------------------|
| <code>.tif, .tiff</code> | BigTIFF, OME-TIFF, ScanImage |
| <code>.zarr</code>       | Zarr v3, OME-NGFF            |
| <code>.h5, .hdf5</code>  | HDF5 datasets                |
| <code>.bin</code>        | Suite2p binary + ops.npy     |
| <code>.npy</code>        | NumPy arrays                 |
| <code>.nwb</code>        | Neurodata Without Borders    |
| In-memory                | NumPy/Dask arrays            |

## Output

|                    |                           |
|--------------------|---------------------------|
| <code>.tiff</code> | BigTIFF (streaming write) |
| <code>.zarr</code> | Zarr v3 with OME metadata |
| <code>.h5</code>   | HDF5 with chunking        |
| <code>.bin</code>  | Suite2p binary format     |
| <code>.npy</code>  | NumPy array               |
| <code>.mp4</code>  | Video export              |

## Lazy Array Types (returned by imread)

- `MboRawArray` - Raw ScanImage multi-ROI with phase correction
- `TiffArray` / `MBOTiffArray` - Standard/Dask-backed TIFF
- `TiffVolumeArray` - Directory of planeXX.tiff files
- `Suite2pArray` / `Suite2pVolumeArray` - Suite2p output
- `ZarrArray` - OME-Zarr / Zarr v3 stores
- `H5Array` - HDF5 datasets
- `NumpyArray` - .npy files or in-memory arrays
- `BinArray` - Direct binary file manipulation
- `NWBArray` - Neurodata Without Borders
- `IsoviewArray` - Lightsheet multi-view data

# GUI Features

## Preview & Visualization

- Image Viewer - FastPlotLib 2D/3D rendering with WGPU
- Frame Navigation - Time slider with playback controls
- Z-Plane Slider - Navigate through imaging planes
- Window Functions - mean, max, std, mean-subtracted
- Scan-Phase Correction - Fix bidirectional artifacts
- Contrast Controls - V-Min/V-Max adjustment
- Summary Stats - Per-plane mean, std, SNR tables

## Processing & Export

- Spatial Crop - Select ROI region for processing
- Suite2p Pipeline - Integrated registration & cell detection
- Registration Settings - Rigid/non-rigid, 1P mode options
- Save As Dialog - Export to .tiff/.zarr/.h5/.bin
- Multi-ROI Support - Process ROIs separately or combined
- Suite3D Registration - Axial z-plane alignment
- Phase Correction - Automatic bidirectional scan fix

## ROI Diagnostics (Suite2p Results)

- dF/F Traces - Adjustable baseline (median/percentile)
- Quality Metrics - SNR, skewness, activity histograms
- Filter Sliders - Interactive threshold adjustment
- Auto-save - Syncs iscell.npy to disk on change
- File Watching - Detects external modifications
- Suite2p Sync - Bi-directional with Suite2p GUI
- ROI Statistics - Detailed per-ROI information

## Launching the GUI

# From command line

```
mbo view /path/to/data  
mbo /path/to/data.tiff
```

# From Python

```
from mbo_utilities import run_gui  
run_gui('/path/to/data')  
run_gui() # opens file dialog
```