
NWB Format Specification

Release 2.7.0

unknown

Feb 07, 2024

TABLE OF CONTENTS

I	Introduction	1
1	Overview	3
1.1	neurodata_type : Assigning types to specifications	4
1.2	NWBContainer, NWBData, NWBDataInterface: Base neurodata_types for containers and datasets	4
1.3	Time Series : A base neurodata_type for storing time series data	4
1.4	Data Processing Modules: Organizing processed data	5
1.5	Extending the format	5
1.5.1	Extending TimeSeries and NWBContainer	5
1.6	Common attributes	6
2	Comments and Definitions	7
2.1	Notation	7
2.2	Storing Time Values	8
2.3	Links and data references	8
2.4	Design notes	8
2.5	Tables and ragged arrays	9
2.6	Doubly ragged arrays	10
2.7	References to rows of a table	10
II	Format Specification	11
3	Format Overview	15
3.1	Namespace – NWB core	15
4	Type Specifications	17
4.1	Base data types	17
4.1.1	NWBData	17
4.1.2	TimeSeriesReferenceVectorData	18
4.1.3	Image	19
4.1.4	ImageReferences	20
4.1.5	NWBContainer	21
4.1.6	NWBDataInterface	22
4.1.7	TimeSeries	23
4.1.7.1	Groups: sync	27
4.1.8	ProcessingModule	28
4.1.8.1	Groups: <NWBDataInterface>	29
4.1.8.2	Groups: <DynamicTable>	29
4.1.9	Images	30
4.2	Devices	31

4.2.1	Device	31
4.3	Epochs	32
4.3.1	TimeIntervals	32
4.4	Image data	35
4.4.1	GrayscaleImage	35
4.4.2	RGBImage	36
4.4.3	RGBAImage	37
4.4.4	ImageSeries	38
4.4.4.1	Groups: sync	44
4.4.5	ImageMaskSeries	45
4.4.5.1	Groups: sync	50
4.4.6	OpticalSeries	52
4.4.6.1	Groups: sync	57
4.4.7	IndexSeries	59
4.4.7.1	Groups: sync	63
4.5	NWB file	64
4.5.1	ScratchData	64
4.5.2	NWBFile	65
4.5.2.1	Groups: /acquisition	69
4.5.2.2	Groups: /acquisition/<NWBDataInterface>	69
4.5.2.3	Groups: /acquisition/<DynamicTable>	69
4.5.2.4	Groups: /analysis	69
4.5.2.5	Groups: /analysis/<NWBContainer>	70
4.5.2.6	Groups: /analysis/<DynamicTable>	70
4.5.2.7	Groups: /scratch	70
4.5.2.8	Groups: /scratch/<NWBContainer>	71
4.5.2.9	Groups: /scratch/<DynamicTable>	71
4.5.2.10	Groups: /processing	71
4.5.2.11	Groups: /processing/<ProcessingModule>	72
4.5.2.12	Groups: /stimulus	72
4.5.2.13	Groups: /stimulus/presentation	73
4.5.2.14	Groups: /stimulus/presentation/<TimeSeries>	73
4.5.2.15	Groups: /stimulus/templates	73
4.5.2.16	Groups: /stimulus/templates/<TimeSeries>	74
4.5.2.17	Groups: /stimulus/templates/<Images>	74
4.5.2.18	Groups: /general	74
4.5.2.19	Groups: /general/<LabMetaData>	77
4.5.2.20	Groups: /general/devices	77
4.5.2.21	Groups: /general/devices/<Device>	78
4.5.2.22	Groups: /general/subject	78
4.5.2.23	Groups: /general/extracellular_ephys	78
4.5.2.24	Groups: /general/extracellular_ephys/<ElectrodeGroup>	79
4.5.2.25	Groups: /general/extracellular_ephys/electrodes	79
4.5.2.26	Groups: /general/intracellular_ephys	81
4.5.2.27	Groups: /general/intracellular_ephys/<IntracellularElectrode>	83
4.5.2.28	Groups: /general/intracellular_ephys/sweep_table	83
4.5.2.29	Groups: /general/intracellular_ephys/intracellular_recordings	83
4.5.2.30	Groups: /general/intracellular_ephys/simultaneous_recordings	84
4.5.2.31	Groups: /general/intracellular_ephys/sequential_recordings	84
4.5.2.32	Groups: /general/intracellular_ephys/repetitions	84
4.5.2.33	Groups: /general/intracellular_ephys/experimental_conditions	84
4.5.2.34	Groups: /general/optogenetics	84
4.5.2.35	Groups: /general/optogenetics/<OptogeneticStimulusSite>	85
4.5.2.36	Groups: /general/optophysiology	85

	4.5.2.37	Groups: /general/optophysiology/<ImagingPlane>	85
	4.5.2.38	Groups: /intervals	86
	4.5.2.39	Groups: /intervals/epochs	86
	4.5.2.40	Groups: /intervals/trials	87
	4.5.2.41	Groups: /intervals/invalid_times	87
	4.5.2.42	Groups: /intervals/<TimeIntervals>	87
	4.5.2.43	Groups: /units	87
	4.5.3	LabMetaData	88
	4.5.4	Subject	89
4.6		Miscellaneous neurodata_types.	92
	4.6.1	AbstractFeatureSeries	92
	4.6.1.1	Groups: sync	96
	4.6.2	AnnotationSeries	98
	4.6.2.1	Groups: sync	102
	4.6.3	IntervalSeries	103
	4.6.3.1	Groups: sync	107
	4.6.4	DecompositionSeries	108
	4.6.4.1	Groups: bands	112
	4.6.4.2	Groups: sync	113
	4.6.5	Units	114
4.7		Behavior	120
	4.7.1	SpatialSeries	120
	4.7.1.1	Groups: sync	125
	4.7.2	BehavioralEpochs	126
	4.7.2.1	Groups: <IntervalSeries>	126
	4.7.3	BehavioralEvents	127
	4.7.3.1	Groups: <TimeSeries>	127
	4.7.4	BehavioralTimeSeries	128
	4.7.4.1	Groups: <TimeSeries>	128
	4.7.5	PupilTracking	129
	4.7.5.1	Groups: <TimeSeries>	129
	4.7.6	EyeTracking	130
	4.7.6.1	Groups: <SpatialSeries>	130
	4.7.7	CompassDirection	131
	4.7.7.1	Groups: <SpatialSeries>	131
	4.7.8	Position	132
	4.7.8.1	Groups: <SpatialSeries>	132
4.8		Extracellular electrophysiology	133
	4.8.1	ElectricalSeries	133
	4.8.1.1	Groups: sync	137
	4.8.2	SpikeEventSeries	139
	4.8.2.1	Groups: sync	143
	4.8.3	FeatureExtraction	145
	4.8.4	EventDetection	147
	4.8.5	EventWaveform	149
	4.8.5.1	Groups: <SpikeEventSeries>	149
	4.8.6	FilteredEphys	150
	4.8.6.1	Groups: <ElectricalSeries>	150
	4.8.7	LFP	151
	4.8.7.1	Groups: <ElectricalSeries>	151
	4.8.8	ElectrodeGroup	152
	4.8.9	ClusterWaveforms	154
	4.8.10	Clustering	156
4.9		Intracellular electrophysiology	158

4.9.1	PatchClampSeries	158
4.9.1.1	Groups: sync	163
4.9.2	CurrentClampSeries	164
4.9.2.1	Groups: sync	169
4.9.3	IZeroClampSeries	170
4.9.3.1	Groups: sync	175
4.9.4	CurrentClampStimulusSeries	176
4.9.4.1	Groups: sync	180
4.9.5	VoltageClampSeries	181
4.9.5.1	Groups: sync	187
4.9.6	VoltageClampStimulusSeries	188
4.9.6.1	Groups: sync	192
4.9.7	IntracellularElectrode	193
4.9.8	SweepTable	196
4.9.9	IntracellularElectrodesTable	198
4.9.10	IntracellularStimuliTable	200
4.9.11	IntracellularResponsesTable	202
4.9.12	IntracellularRecordingsTable	204
4.9.12.1	Groups: electrodes	207
4.9.12.2	Groups: stimuli	207
4.9.12.3	Groups: responses	207
4.9.12.4	Groups: <DynamicTable>	207
4.9.13	SimultaneousRecordingsTable	208
4.9.14	SequentialRecordingsTable	210
4.9.15	RepetitionsTable	212
4.9.16	ExperimentalConditionsTable	214
4.10	Optogenetics	216
4.10.1	OptogeneticSeries	216
4.10.1.1	Groups: sync	220
4.10.2	OptogeneticStimulusSite	221
4.11	Optical physiology	223
4.11.1	OnePhotonSeries	223
4.11.1.1	Groups: sync	228
4.11.2	TwoPhotonSeries	230
4.11.2.1	Groups: sync	236
4.11.3	RoiResponseSeries	237
4.11.3.1	Groups: sync	241
4.11.4	DfOverF	242
4.11.4.1	Groups: <RoiResponseSeries>	242
4.11.5	Fluorescence	243
4.11.5.1	Groups: <RoiResponseSeries>	243
4.11.6	ImageSegmentation	244
4.11.6.1	Groups: <PlaneSegmentation>	244
4.11.7	PlaneSegmentation	245
4.11.7.1	Groups: reference_images	248
4.11.7.2	Groups: reference_images/<ImageSeries>	248
4.11.8	ImagingPlane	249
4.11.8.1	Groups: <OpticalChannel>	253
4.11.9	OpticalChannel	254
4.11.10	MotionCorrection	255
4.11.10.1	Groups: <CorrectedImageStack>	255
4.11.11	CorrectedImageStack	256
4.11.11.1	Groups: corrected	257
4.11.11.2	Groups: xy_translation	257

4.12	Retinotopy	258
4.12.1	ImagingRetinotopy	258
5	Schema Sources	265
5.1	Namespace – NWB core	265
5.2	Base data types	266
5.2.1	NWBData	266
5.2.2	TimeSeriesReferenceVectorData	267
5.2.3	Image	268
5.2.4	ImageReferences	269
5.2.5	NWBContainer	270
5.2.6	NWBDataInterface	271
5.2.7	TimeSeries	272
5.2.8	ProcessingModule	276
5.2.9	Images	277
5.3	Devices	278
5.3.1	Device	278
5.4	Epochs	279
5.4.1	TimeIntervals	279
5.5	Image data	281
5.5.1	GrayscaleImage	281
5.5.2	RGBImage	282
5.5.3	RGBAImage	283
5.5.4	ImageSeries	284
5.5.5	ImageMaskSeries	289
5.5.6	OpticalSeries	294
5.5.7	IndexSeries	299
5.6	NWB file	302
5.6.1	ScratchData	302
5.6.2	NWBFile	303
5.6.3	LabMetaData	312
5.6.4	Subject	313
5.7	Miscellaneous neurodata_types.	314
5.7.1	AbstractFeatureSeries	314
5.7.2	AnnotationSeries	318
5.7.3	IntervalSeries	321
5.7.4	DecompositionSeries	324
5.7.5	Units	328
5.8	Behavior	332
5.8.1	SpatialSeries	332
5.8.2	BehavioralEpochs	336
5.8.3	BehavioralEvents	337
5.8.4	BehavioralTimeSeries	338
5.8.5	PupilTracking	339
5.8.6	EyeTracking	340
5.8.7	CompassDirection	341
5.8.8	Position	342
5.9	Extracellular electrophysiology	343
5.9.1	ElectricalSeries	343
5.9.2	SpikeEventSeries	347
5.9.3	FeatureExtraction	351
5.9.4	EventDetection	352
5.9.5	EventWaveform	353
5.9.6	FilteredEphys	354

5.9.7	LFP	355
5.9.8	ElectrodeGroup	356
5.9.9	ClusterWaveforms	357
5.9.10	Clustering	358
5.10	Intracellular electrophysiology	359
5.10.1	PatchClampSeries	359
5.10.2	CurrentClampSeries	363
5.10.3	IZeroClampSeries	367
5.10.4	CurrentClampStimulusSeries	371
5.10.5	VoltageClampSeries	374
5.10.6	VoltageClampStimulusSeries	379
5.10.7	IntracellularElectrode	382
5.10.8	SweepTable	383
5.10.9	IntracellularElectrodesTable	384
5.10.10	IntracellularStimuliTable	385
5.10.11	IntracellularResponsesTable	386
5.10.12	IntracellularRecordingsTable	387
5.10.13	SimultaneousRecordingsTable	389
5.10.14	SequentialRecordingsTable	390
5.10.15	RepetitionsTable	392
5.10.16	ExperimentalConditionsTable	393
5.11	Optogenetics	394
5.11.1	OptogeneticSeries	394
5.11.2	OptogeneticStimulusSite	397
5.12	Optical physiology	398
5.12.1	OnePhotonSeries	398
5.12.2	TwoPhotonSeries	403
5.12.3	RoiResponseSeries	408
5.12.4	DfOverF	411
5.12.5	Fluorescence	412
5.12.6	ImageSegmentation	413
5.12.7	PlaneSegmentation	414
5.12.8	ImagingPlane	417
5.12.9	OpticalChannel	420
5.12.10	MotionCorrection	421
5.12.11	CorrectedImageStack	422
5.13	Retinotopy	423
5.13.1	ImagingRetinotopy	423
III	History and Legal	429
6	Making a Pull Request	431
7	Merging PRs and Making Releases	433
8	Making a Release Checklist	435
9	The “dev” Branch and Releases	437
10	Release Notes	439
10.1	2.7.0 (February 7, 2024)	439
10.1.1	Minor changes	439
10.2	2.6.0 (January 17, 2023)	439
10.2.1	Minor changes	439

10.3	2.5.0 (June 14, 2022)	439
10.3.1	Major changes	439
10.3.2	Minor changes	440
10.4	2.4.0 (Aug. 11, 2021)	440
10.4.1	Major changes	440
10.4.2	Deprecations	441
10.4.3	Bug Fixes	441
10.5	2.3.0 (May 12, 2021)	441
10.6	2.2.5 (May 29, 2020)	442
10.7	2.2.4 (April 14, 2020)	442
10.8	2.2.3 (April 13, 2020)	442
10.9	2.2.2 (March 2, 2020)	442
10.10	2.2.1 (January 14, 2020)	442
10.11	2.2.0 (January 6, 2020)	443
10.12	2.1.0 (September 2019)	444
10.13	2.0.2 (June 2019)	446
10.14	2.0.1 (March 2019)	446
10.15	2.0.0 (January 2019)	446
10.15.1	Added new base data types: <code>NWBContainer</code> , <code>NWBData</code> , <code>NWBDataInterface</code>	446
10.15.2	Support general data structures for data tables and vector data	447
10.15.2.1	Support row-based and column-based tables	447
10.15.2.2	Enable efficient storage of large numbers of vector data elements	447
10.15.3	Use new table and vector data structures to improve data organization	448
10.15.3.1	Improved organization of electrode metadata in <code>/general/extracellular_ephys</code>	448
10.15.3.2	Improved storage of lab-specific meta-data	449
10.15.3.3	Improved storage of Spectral Analyses (Signal Decomposition)	449
10.15.3.4	Improved storage of Images	450
10.15.3.5	Improved storage of ROIs	450
10.15.3.6	Improved storage of unit-based data	451
10.15.3.7	Improved support for sweep-based information	452
10.15.4	Improved specification of reference time stamp(s)	453
10.15.4.1	Improve standardization of reference time specification using ISO8061	453
10.15.4.2	Improved specification of reference time	453
10.15.5	Improved storage of time intervals	453
10.15.5.1	Improved storage of epoch data	453
10.15.5.2	Improved support for trial-based data	454
10.15.5.3	Generalized storage of time interval	454
10.15.6	Replaced Implicit Links/Data-Structures with Explicit Links	454
10.15.7	Improved consistency, identifiably, and readability	455
10.15.7.1	Improved identifiably of objects	455
10.15.7.2	Simplified extension of subject metadata	455
10.15.7.3	Reduce requirement for potentially empty groups	456
10.15.7.4	Added missing metadata	456
10.15.7.5	Improved Consistency	456
10.15.7.6	Added <code>keywords</code> field	457
10.15.7.7	Removed <code>source</code> field	457
10.15.7.8	Removed <code>ancestry</code> field	457
10.15.8	Improved organization of processed and acquisition data	457
10.15.8.1	Improved organization of processed data	457
10.15.8.2	Simplified organization of acquisition data	458
10.15.9	Other changes:	458
10.15.10	Improved governance and accessibility	458
10.15.11	Specification language changes	458
10.15.11.1	Specification of dataset dimensions	458

10.15.11.2	Added Link type	459
10.15.11.3	Removed datasets defined via autogen	459
10.15.11.4	Removed 'neurodata_type=Custom'	460
10.16	1.0.x (09/2015 - 04/2017)	460
10.16.1	1.0.6, April 8, 2017	460
10.16.2	1.0.5i_beta, Dec 6, 2016	460
10.16.3	1.0.5h_beta, Nov 30, 2016	460
10.16.4	1.0.5g_beta, Oct 7, 2016	460
10.16.5	1.0.5f_beta, Oct 3, 2016	461
10.16.6	1.0.5e_beta, Sept 22, 2016	461
10.16.7	1.0.5d_beta, Sept 6, 2016	461
10.16.8	1.0.5c_beta, Aug 17, 2016	461
10.16.9	1.0.5b_beta, Aug 16, 2016	461
10.16.10	1.0.5a_beta, Aug 10, 2016	461
10.16.11	1.0.5_beta Aug 2016	462
10.16.12	1.0.4_beta June 2016	462
10.16.13	1.0.3 April, 2016	462
10.16.14	1.0.2 February, 2016	462
10.16.15	1.0.1 October 7th, 2015	462
10.16.16	1.0.0 September 28 th , 2015	463
11	Credits	465
11.1	Acknowledgments	465
11.2	Authors	465
11.2.1	NWB: Version 2.0.0 and later	465
11.2.2	NWB: Version 1.0.x and earlier	465
12	Legal	467
12.1	Copyright	467
12.2	Licence	467

Part I

Introduction

OVERVIEW

The **NWB Format** is a core component of the **Neurodata Without Borders (NWB)** project. The NWB format is designed to store general optical and electrical physiology data in a way that is both understandable to humans as well as accessible to programmatic interpretation. The format is designed to be friendly to and usable by software tools and analysis scripts, and to impose few a priori assumptions about data representation and analysis.

The NWB format uses the following main primitives to hierarchically organize neuroscience data :

- A *Group* is similar to a folder and may contain an arbitrary number of other groups and datasets,
- A *Dataset* describes an n-dimensional array and provides the primary means for storing data,
- An *Attribute** is a small dataset that is attached to a specific group or dataset and is typically used to store metadata specific to the object they are associated with, and
- A *Link* is a reference to another group or dataset.

The NWB format is formally described via formal specification documents using the **NWB specification language** . HDF5 currently serves as the main format for storing data in the NWB format (see <http://nwb-storage.readthedocs.io/en/latest/> for details). The **PyNWB** API is available to enable users to efficiently interact with NWB format files.

The NWB format uses a modular design in which all main semantic components of the format have a unique *neurodata_type* (similar to a Class in object-oriented design)(Section 1.1). This allows for reuse and extension of types through inclusion and inheritance. All datasets and groups in the format can be uniquely identified by either their name and/or *neurodata_type*.

Two important base types in the NWB format are *NWBContainer* and *TimeSeries*. *NWBContainer* defines a generic container for storing collection of data and is used to define common features and functionality across data containers (see Section 1.2). *TimeSeries* is a central component in the NWB format for storing complex temporal series (see Section 1.3). In the format, these types are then extended to define more specialized types. To organize and define collections of processed data from common data processing steps, the NWB format then defines the concept of *ProcessingModule* where each processing step is represented by a corresponding *NWBDataInterface* (an extension of *NWBContainer*) (see Section 1.4 for details).

At a high level, data is organized into the following main groups:

- *acquisition/* : data streams recorded from the system, including ephys, ophys, tracking, etc.,
- *intervals/* : experimental intervals,
- *stimulus/* : stimulus data,
- *general/* : experimental metadata, including protocol, notes and description of hardware device(s).
- *processing/* : standardized processing modules, often as part of intermediate analysis of data that is necessary to perform before scientific analysis,
- *analysis/* : lab-specific and custom scientific analysis of data.

The high-level data organization within NWB files is described in detail in [Section 4.5.2](#). The top-level datasets and attributes are described in [Table 4.22](#) and the top-level organization of data into groups is described in [Table 4.23](#).

1.1 neurodata_type : Assigning types to specifications

The concept of a *neurodata_type* is similar to the concept of a Class in object-oriented programming. In the NWB format, groups or datasets may be given a unique *neurodata_type*. The *neurodata_type* allows the unique identification of the type of objects in the format and also enable the reuse of types through the concept of inheritance. A group or dataset may, hence, define a new *neurodata_type* while extending an existing type. E.g., *AbstractFeatureSeries* defines a new type that inherits from *TimeSeries*.

1.2 NWBContainer, NWBData, NWBDataInterface: Base neurodata_types for containers and datasets

NWBContainer is a specification of a group that defines a generic container for storing collections of data. *NWBContainer* serves as the base type for all main data containers (including *TimeSeries*) of the core NWB data format and allows us to define and integrate new common functionality in a central place and via common mechanisms (see [Section 4.1.5](#)).

NWBDataInterface extends *NWBContainer* and serves as base type for primary data (e.g., experimental or analysis data) and is used to distinguish in the schema between non-metadata data containers and metadata containers (see [Section 4.1.6](#)).

NWBData is a specification of a Dataset that functions as a common base *neurodata_type* for datasets with an assigned *neurodata_type* (see [Section 4.1.1](#)).

Note: The concept of *NWBContainer* and *NWBData* have been introduced in NWB 2. *NWBDataInterface* (also introduced in NWB 2) replaces *Interface* from NWB 1.x. *Interface* was renamed to *NWBDataInterface* to ease intuition and the concept was generalized via *NWBContainer* to provide a common base for data containers (rather than being specific to *ProcessingModules* as in NWB 1.x).

1.3 Time Series : A base neurodata_type for storing time series data

The file format is designed around a data structure called a *TimeSeries* which stores time-varying data. A *TimeSeries* is a superset of several *neurodata_types*, including signal events, image stacks and experimental events. To account for different storage requirements and different modalities, a *TimeSeries* is defined in a minimal form and it can be extended (i.e., subclassed) to account for different modalities and data storage requirements (see [Section 1.5](#)).

Each *TimeSeries* has its own HDF5 group, and all datasets belonging to a *TimeSeries* are in that group. In particular, a *TimeSeries* defines components to store *data* and *time*.

The *data* element in the *TimeSeries* will typically be an array of any valid HDF5 data type (e.g., a multi-dimensional floating point array). The data stored can be in any unit. The attributes of the data field must indicate the SI unit that the data relates to (or appropriate counterpart, such as color-space) and the multiplier necessary to convert stored values to the specified SI unit.

TimeSeries support provides two time objects representations. The first, *timestamps*, stores time information that is corrected to the experiment's time base (i.e., aligned to a master clock, with time-zero aligned to the starting time of the experiment). This field is used for data processing and subsequent scientific analysis. The second, *sync*, is an

optional group that can be used to store the sample times as reported by the acquisition/stimulus hardware, before samples are converted to a common time-base and corrected relative to the master clock. This approach allows the NWB format to support streaming of data directly from hardware sources.

In addition to data and time, the *TimeSeries* group can be used to store additional information beyond what is required by the specification. I.e., an end user is free to add additional key/value pairs as necessary for their needs via the concept of extensions. It should be noted that such lab-specific extensions may not be recognized by analysis tools/scripts existing outside the lab. Extensions are described in section (see [Section 1.5](#)).

1.4 Data Processing Modules: Organizing processed data

NWB uses *ProcessingModule* to store data for—and represent the results of—common data processing steps, such as spike sorting and image segmentation, that occur before scientific analysis of the data. Processing modules store the data used by software tools to calculate these intermediate results. All processing modules are stored directly in the group */processing*. The name of each module is chosen by the data provider (i.e. processing modules have a “variable” name). The particular data within each processing module is specified by one or more *NWBDataInterface*, which are groups residing directly within a processing module. Each *NWBDataInterface* has a unique *neurodata_type* (e.g., *ImageSegmentation*) that describes and defines the data contained in the *NWBDataInterface*. For *NWBDataInterfaces* designed for use with processing modules, a default name (usually the same as the *neurodata_type*) is commonly specified to further ease identification of the data in a file. However, to support storage of multiple instances of the same subtype in the same processing module, NWB allows users to optionally define custom names as well.

1.5 Extending the format

The data organization presented in this document constitutes the *core* NWB format. Extensibility is handled via the concept of extensions, allowing users to extend (i.e., add to) existing and create new *neurodata_types* definitions for storing custom data. To avoid collisions between extensions, extensions are defined as part of custom namespaces (which typically import the core NWB namespace). Extensions to the format are written using the *Specification Language*. To ease development of extensions, the *PyNWB* (and *HDMF* used by *PyNWB*) API provides dedicated data structures that support programmatic creation and use of extensions. An example for extending NWB using *PyNWB* is available at <https://pynwb.readthedocs.io/en/stable/tutorials/general/extensions.html> and additional details are also available as part of the *PyNWB* tutorials <https://pynwb.readthedocs.io/en/stable/tutorials/index.html>.

Creating extensions allows adding and documenting new data to NWB, interaction with custom data via the API, validation of custom data contents, sharing and collaboration of extensions and data. Popular extensions may be proposed and added to the official format specification after community discussion and review. To propose a new extensions for the NWB core format you may file an issue at <https://github.com/NeurodataWithoutBorders/nwb-schema/issues>.

1.5.1 Extending TimeSeries and NWBContainer

Like any other *neurodata_type*, *TimeSeries* can be extended via extensions by defining corresponding derived *neurodata_types*. This is typically done to represent more narrowly focused modalities (e.g., electrical versus optical physiology) as well as new modalities (e.g., video tracking of whisker positions). When a *neurodata_type* inherits from *TimeSeries*, new data objects (i.e., datasets, attributes, groups, and links) can be added while all objects of the parent *TimeSeries* type are inherited and, hence, part of the new *neurodata_type*. Section [Section 4.1.7](#) includes a list of all *TimeSeries* types.

Extending *NWBContainer* works in the same way, e.g., to create more specific types for data processing.

1.6 Common attributes

All NWB Groups and Datasets with an assigned `neurodata_type` have three required attributes: *neurodata_type*, *namespace*, and *object_id*.

- `neurodata_type` (variable-length string) is the name of the NWB primitive that this group or dataset maps onto
- `namespace` (variable-length string) is the namespace where `neurodata_type` is defined, e.g. “core” or the namespace of an extension
- `object_id` (variable-length string) is a universally unique identifier for this object within its hierarchy. It should be set to the string representation of a random UUID version 4 value (see [RFC 4122](#)) upon first creation. It is **not** a hash of the data. Files that contain the exact same data but were generated in different instances will have different `object_id` values. Currently, modification of an object does not require its `object_id` to be changed.

COMMENTS AND DEFINITIONS

2.1 Notation

The description of the format is divided into subsection based on *neurodata_type*. Each *neurodata_type* section includes:

- A basic description of the type
- An optional figure describing the organization of data within the type
- A set of tables describing the datasets, attributes and groups contained in the type.
- An optional set of further subsections describing the content of subgroups contained in the given *neurodata_type*.

In the tables we use the following notation in the **Id** column to uniquely identify datasets, groups, attributes:

- ``name`` describes the unique name of an object
- ``<neurodata_type>`` describes the ``neurodata_type`` of the object in case that the object does not have a unique name
- ``-`` prefixes are used to indicate the depth of the object in the hierarchy to allow identification of the parent of the object. E.g., an object with a ``--`` prefix will belong to the previous object with a ``-`` prefix.

Here a quick example:

Table 2.1: Example illustrating the description of the contents of ``neurodata_types``.

Id	Type	Description
<code><MyTimeSeries></code>	group	Top level group for the <i>neurodata_type</i> . The group the <i>neurodata_type</i> <i>MyTimeSeries</i> but no fixed name
<code>—myattr</code>	at-tribute	Attribute with the fixed name <i>myattr</i> defined on <code><MyTimeSeries></code>
<code>—mydata</code>	dataset	Required dataset with a unique name contained in <code><MyTimeSeries></code>
<code>——unit</code>	at-tribute	Attribute <i>unit</i> defined on the dataset <i>.mydata</i>
<code>—myotherdata</code>	dataset	Optional dataset with a unique name contained in <code><MyTimeSeries></code>
<code>—<ElectricalSeries></code>	group	Optional set of groups with the <i>neurodata_type</i> <i>ElectricalSeries</i> that are contained in <code><MyTimeSeries></code>

2.2 Storing Time Values

All times are stored in seconds using double precision (64-bit) floating point values. A smaller floating point value, e.g., 32-bit, is **not** permitted for storing times. This is because significant errors for time can result from using smaller data sizes. Throughout this document, sizes (number of bits) are specified for many datatypes (e.g., float32). For fields with a specified size, larger sizes can be used, so long as the selected size encompasses the full range of data, and for floats, without loss of significant precision.

2.3 Links and data references

Soft Links: In some instances, the specification refers to links. When links are made within a file, NWB uses soft-links and not hard-links. This is because soft-links distinguish between the link and the target of the link, whereas hard-links cause multiple names (paths) to be created for the target, and there is no way to determine which of these names are preferable in a given situation. If the target of a soft link is removed (or moved to another location), then the soft link will “dangle,” that is point to a target that no longer exists. For this reason, moving or removing targets of soft links should be avoided unless the links are updated to point to the new location.

Object References: Object references are similar to SoftLinks but instead of being stored as elements in the data hierarchy (similar to a Group or Dataset) the object reference defines a data type on a Dataset, i.e., the object references are stored as elements in a Dataset.

Region References: Region references are similar to object references but instead of pointing to another Group/Dataset as a whole a region reference defines an additional selection and as such describe a reference to a subset of a dataset.

2.4 Design notes

Data type sizes

The listed size of integers and floating point values is the minimum size. The size of the data type should be large enough to store the required data, and preferably not larger. 64-bit floating point (double) is required for timestamps, while 32-bit floating point is largely sufficient for other uses.

Extra fields

All parts of an NWB file should be governed by either the core schema or defined in a neurodata extension (NDX). *Extra fields* are any datasets, attributes, groups, links etc. that are included in a file but which are not described by the NWB schema or a neurodata extension (NDX). Extra fields are not considered part of the NWB file and as such, any NWB API may ignore extra fields. For API's this specifically means:

- an NWB file that includes extra fields should be readable by the API as long as the file is otherwise valid,
- an API is permitted to ignore extra fields on read,
- an API is permitted to ignore (including remove) extra fields on write.

In practice, the use of extra fields is highly discouraged and instead neurodata extensions (NDX) should be used to extend NWB to include additional fields if necessary.

Why do timestamps_link and data_link record linking between datasets, but links between epochs and timeseries are not recorded?

Epochs have a link to entire timeseries (i.e., the HDF5 group). If 100 epochs link to a time series, there is only one time series. The data and timestamps within it are not shared anywhere (at least from the epoch linking). An epoch is an entity that is put in for convenience and annotation so there isn't necessarily an important association between what epochs link to what time series (all epochs could link to all time series).

The `timestamps_link` and `data_link` fields refer to links made between time series, such as if timeseries A and timeseries B, each having different data (or time) share time (or data). This is much more important information as it shows structural associations in the data.

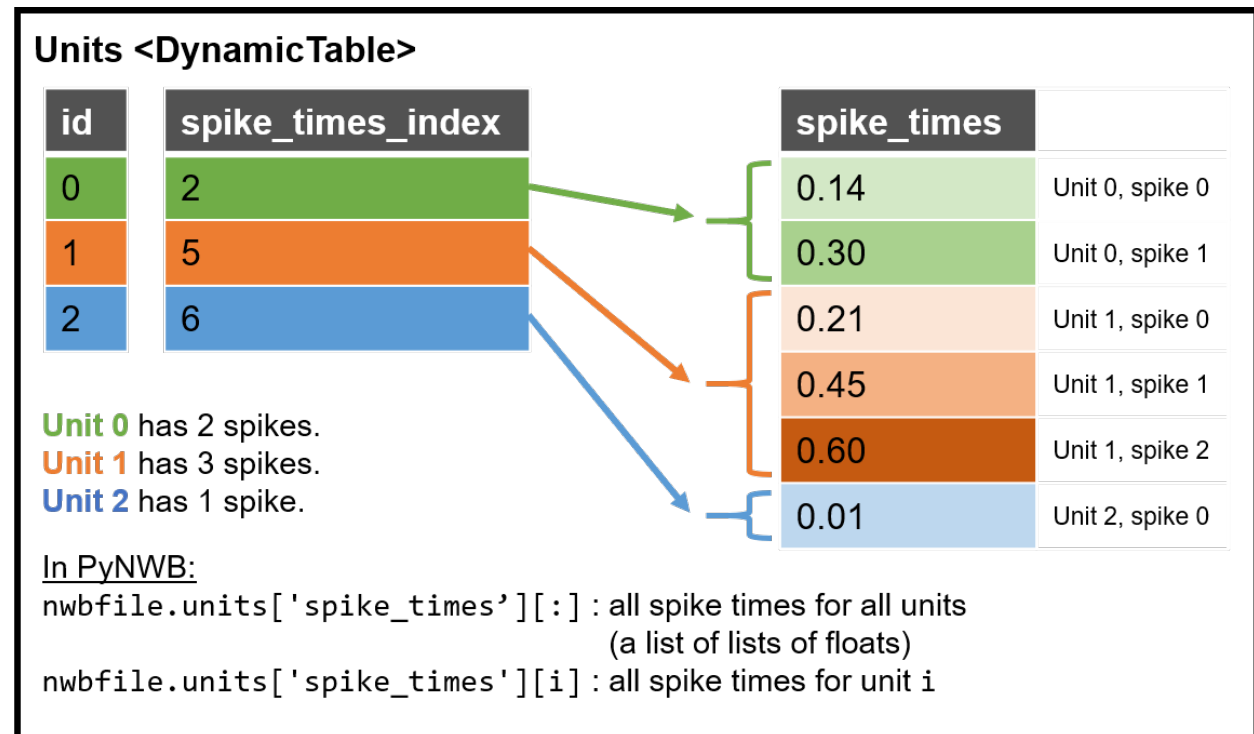
2.5 Tables and ragged arrays

The NWB schema includes several tables, such as for storing data/metadata about trials, epochs, single units and multi-units, electrodes, and ROIs. All of the tables in NWB derive from the base data type, `DynamicTable`. `DynamicTable` is a column-based representation of a table that allows users to add custom columns (of type `VectorData`) that are not pre-defined in the specification. This is useful for handling types of data where every experiment or lab may want to store information unique to that experiment or lab, e.g., metadata related to the trials in a session or spike sorting metrics.

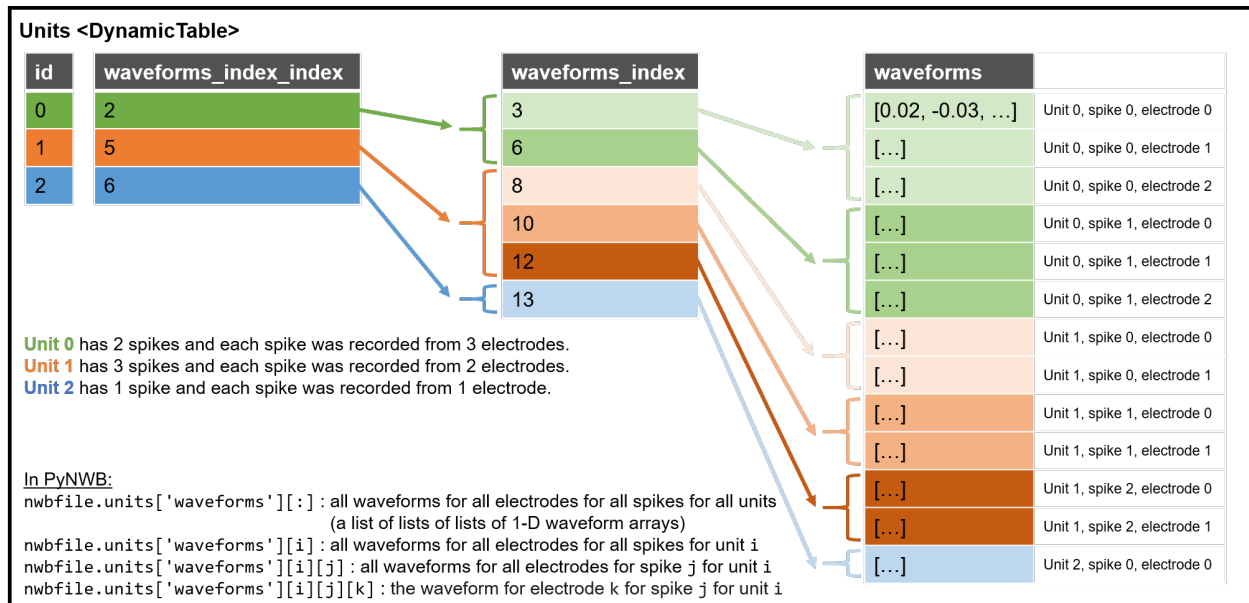
`DynamicTable` objects typically contain columns that are of equal length, where the *i*-th element of a column corresponds to the *i*-th element of all of the other columns. In other words, each row has a single item in each column. However, in some situations, users may wish to store and associate multiple items in a single column for each row. For example, in the Units table, each row represents a single sorted unit and each unit has multiple spike times associated with it, where the number of spike times differs between units (rows). This is sometimes called a ragged array or jagged array.

Ragged array columns can be created by creating a primary `VectorData` column that contains all of the data values (e.g., spike times) and creating a secondary `VectorIndex` column that contains a mapping from rows to elements of its target `VectorData` column. The `VectorIndex` column has the same number of elements (rows) as the rest of the table.

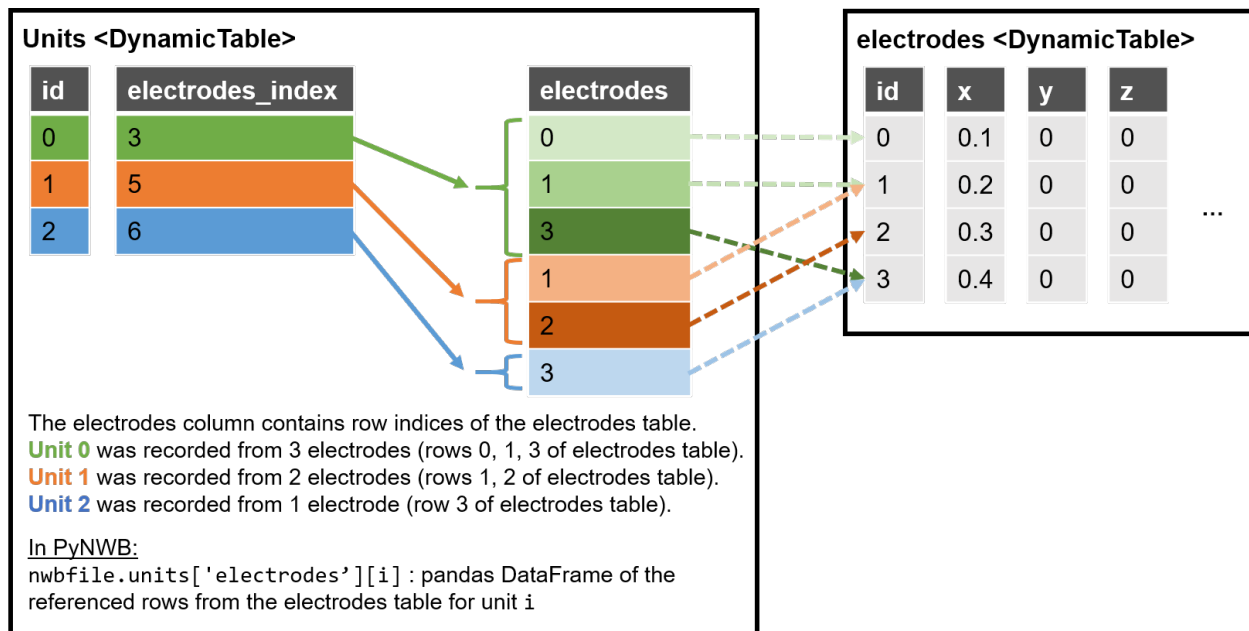
The values of the `VectorIndex` column follow the mapping such that the data associated with the first row is at `VectorData[0:VectorIndex[0]]`, and the data associated with the second row is at `VectorData[VectorIndex[0]:VectorIndex[1]]`, and so on.



2.6 Doubly ragged arrays



2.7 References to rows of a table



Part II

Format Specification

Version 2.7.0 Feb 07, 2024

FORMAT OVERVIEW

3.1 Namespace – NWB core

- **Description:** NWB namespace
- **Name:** core
- **Full Name:** NWB core
- **Version:** 2.6.0-alpha
- **Authors:**
 - Andrew Tritt
 - Oliver Ruebel
 - Ryan Ly
 - Ben Dichter
 - Keith Godfrey
 - Jeff Teeters
- **Contacts:**
 - ajtritt@lbl.gov
 - oruebel@lbl.gov
 - rly@lbl.gov
 - bdichter@lbl.gov
 - keithg@alleninstitute.org
 - jteeters@berkeley.edu
- **Schema:**
 - **namespace:** hdmf-common
 - **doc:** This source module contains base data types used throughout the NWB data format.
 - **source:** nwb.base.yaml
 - **title:** Base data types
 - **doc:** This source module contains neurodata_types for device data.
 - **source:** nwb.device.yaml
 - **title:** Devices

- **doc:** This source module contains neurodata_types for epoch data.
- **source:** nwb.epoch.yaml
- **title:** Epochs
- **doc:** This source module contains neurodata_types for image data.
- **source:** nwb.image.yaml
- **title:** Image data
- **doc:** Main NWB file specification.
- **source:** nwb.file.yaml
- **title:** NWB file
- **doc:** Miscellaneous types.
- **source:** nwb.misc.yaml
- **title:** Miscellaneous neurodata_types.
- **doc:** This source module contains neurodata_types for behavior data.
- **source:** nwb.behavior.yaml
- **title:** Behavior
- **doc:** This source module contains neurodata_types for extracellular electrophysiology data.
- **source:** nwb.ecephys.yaml
- **title:** Extracellular electrophysiology
- **doc:** This source module contains neurodata_types for intracellular electrophysiology data.
- **source:** nwb.icephys.yaml
- **title:** Intracellular electrophysiology
- **doc:** This source module contains neurodata_types for opto-genetics data.
- **source:** nwb.ogen.yaml
- **title:** Optogenetics
- **doc:** This source module contains neurodata_types for optical physiology data.
- **source:** nwb.ophys.yaml
- **title:** Optical physiology
- **doc:** This source module contains neurodata_type for retinotopy data.
- **source:** nwb.retinotopy.yaml
- **title:** Retinotopy

TYPE SPECIFICATIONS

4.1 Base data types

This source module contains base data types used throughout the NWB data format.

4.1.1 NWBData

Overview: An abstract data type for a dataset.

NWBData extends Data and includes all elements of Data with the following additions or changes.

- **Extends:** Data
- **Primitive Type:** Dataset
- **Inherits from:** Data
- **Subtypes:** *GrayscaleImage*, *ImageReferences*, *RGBAImage*, *RGBImage*, *Image*, *ScratchData*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.1

4.1.2 TimeSeriesReferenceVectorData

Overview: Column storing references to a TimeSeries (rows). For each TimeSeries this VectorData column stores the start_index and count to indicate the range in time to be selected as well as an object reference to the TimeSeries.

TimeSeriesReferenceVectorData extends VectorData and includes all elements of VectorData with the following additions or changes.

- **Extends:** VectorData
- **Primitive Type:** Dataset
- **Data Type:** Compound data type with the following elements:
 - **idx_start:** Start index into the TimeSeries ‘data’ and ‘timestamp’ datasets of the referenced TimeSeries. The first dimension of those arrays is always time. (*dtype*= int32)
 - **count:** Number of data samples available in this time series, during this epoch (*dtype*= int32)
 - **timeseries:** The TimeSeries that this index applies to (*dtype*= object reference to *TimeSeries*)
- **Default Name:** timeseries
- **Inherits from:** VectorData, Data
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.2

Table 4.1: Datasets, Links, and Attributes contained in <TimeSeriesReferenceVectorData>

Id	Type	Description
<TimeSeriesReferenceVectorData>	Dataset	Top level Dataset for <TimeSeriesReferenceVectorData> • Neurodata Type: TimeSeriesReferenceVectorData • Extends: VectorData • Data Type: Compound data type with the following elements: – idx_start: Start index into the TimeSeries ‘data’ and ‘timestamp’ datasets of the referenced TimeSeries. The first dimension of those arrays is always time. (<i>dtype</i>= int32) – count: Number of data samples available in this time series, during this epoch (<i>dtype</i>= int32) – timeseries: The TimeSeries that this index applies to (<i>dtype</i>= object reference to <i>TimeSeries</i>) • Default Name: timeseries
—description	Attribute	Description of what these vectors represent. • Data Type: text • Name: description

4.1.3 Image

Overview: An abstract data type for an image. Shape can be 2-D (x, y), or 3-D where the third dimension can have three or four elements, e.g. (x, y, (r, g, b)) or (x, y, (r, g, b, a)).

Image extends NWBData and includes all elements of *NWBData* with the following additions or changes.

- **Extends:** *NWBData*
- **Primitive Type:** Dataset
- **Data Type:** numeric
- **Dimensions:** [['x', 'y'], ['x', 'y', 'r, g, b'], ['x', 'y', 'r, g, b, a']]
- **Shape:** [[None, None], [None, None, 3], [None, None, 4]]
- **Inherits from:** *NWBData*, *Data*
- **Subtypes:** *GrayscaleImage*, *RGBAImage*, *RGBImage*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.3

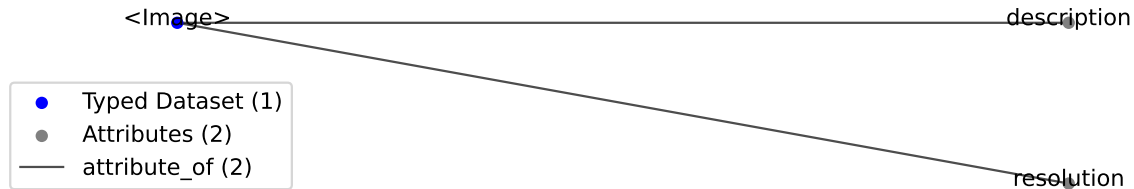


Table 4.2: Datasets, Links, and Attributes contained in <Image>

Id	Type	Description
<Image>	Dataset	Top level Dataset for <Image> • Neurodata Type: Image • Extends: <i>NWBData</i> • Data Type: numeric • Dimensions: [['x', 'y'], ['x', 'y', 'r, g, b'], ['x', 'y', 'r, g, b, a']] • Shape: [[None, None], [None, None, 3], [None, None, 4]]
—resolution	At-tribute	Pixel resolution of the image, in pixels per centimeter. • Data Type: float32 • Required: False • Name: resolution
—description	At-tribute	Description of the image. • Data Type: text • Required: False • Name: description

4.1.4 ImageReferences

Overview: Ordered dataset of references to Image objects.

ImageReferences extends NWBData and includes all elements of *NWBData* with the following additions or changes.

- **Extends:** *NWBData*
- **Primitive Type:** Dataset
- **Data Type:** object reference to *Image*
- **Dimensions:** ['num_images']
- **Shape:** [None]
- **Inherits from:** *NWBData*, *Data*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see [Section 5.2.4](#)

4.1.5 NWBContainer

Overview: An abstract data type for a generic container storing collections of data and metadata. Base type for all data and metadata containers.

NWBContainer extends Container and includes all elements of Container with the following additions or changes.

- **Extends:** Container
- **Primitive Type:** Group
- **Inherits from:** Container
- **Subtypes:** *SpatialSeries, LabMetaData, VoltageClampStimulusSeries, EventDetection, NWBFile, Device, NWBDataInterface, DecompositionSeries, Position, TimeSeries, FeatureExtraction, CorrectedImageStack, OptogeneticStimulusSite, OpticalChannel, Fluorescence, ClusterWaveforms, ImagingRetinotopy, PatchClampSeries, SpikeEventSeries, CurrentClampSeries, OptogeneticSeries, ProcessingModule, CompassDirection, EyeTracking, BehavioralEvents, EventWaveform, Subject, IndexSeries, AnnotationSeries, ElectricalSeries, BehavioralTimeSeries, ElectrodeGroup, IntracellularElectrode, BehavioralEpochs, IntervalSeries, RoiResponseSeries, FilteredEphys, AbstractFeatureSeries, Images, OnePhotonSeries, VoltageClampSeries, Clustering, CurrentClampStimulusSeries, ImageSegmentation, MotionCorrection, OpticalSeries, TwoPhotonSeries, PupilTracking, ImagingPlane, ImageMaskSeries, DfOverF, IZeroClampSeries, ImageSeries, LFP*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.5

4.1.6 NWBDataInterface

Overview: An abstract data type for a generic container storing collections of data, as opposed to metadata.

NWBDataInterface extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, Container
- **Subtypes:** *SpatialSeries*, *VoltageClampStimulusSeries*, *EventDetection*, *DecompositionSeries*, *Position*, *TimeSeries*, *FeatureExtraction*, *CorrectedImageStack*, *Fluorescence*, *ClusterWaveforms*, *ImagingRetinotopy*, *PatchClampSeries*, *SpikeEventSeries*, *CurrentClampSeries*, *OptogeneticSeries*, *CompassDirection*, *EyeTracking*, *BehavioralEvents*, *EventWaveform*, *IndexSeries*, *AnnotationSeries*, *ElectricalSeries*, *BehavioralTimeSeries*, *BehavioralEpochs*, *IntervalSeries*, *RoiResponseSeries*, *FilteredEphys*, *AbstractFeatureSeries*, *Images*, *OnePhotonSeries*, *VoltageClampSeries*, *Clustering*, *CurrentClampStimulusSeries*, *ImageSegmentation*, *MotionCorrection*, *OpticalSeries*, *TwoPhotonSeries*, *PupilTracking*, *ImageMaskSeries*, *DfOverF*, *IZeroClampSeries*, *ImageSeries*, *LFP*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.6

4.1.7 TimeSeries

Overview: General purpose time series.

TimeSeries extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Subtypes:** *SpatialSeries*, *VoltageClampStimulusSeries*, *DecompositionSeries*, *PatchClampSeries*, *SpikeEventSeries*, *CurrentClampSeries*, *OptogeneticSeries*, *IndexSeries*, *AnnotationSeries*, *ElectricalSeries*, *IntervalSeries*, *RoiResponseSeries*, *AbstractFeatureSeries*, *OnePhotonSeries*, *VoltageClampSeries*, *CurrentClampStimulusSeries*, *OpticalSeries*, *TwoPhotonSeries*, *ImageMaskSeries*, *IZeroClampSeries*, *ImageSeries*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.7

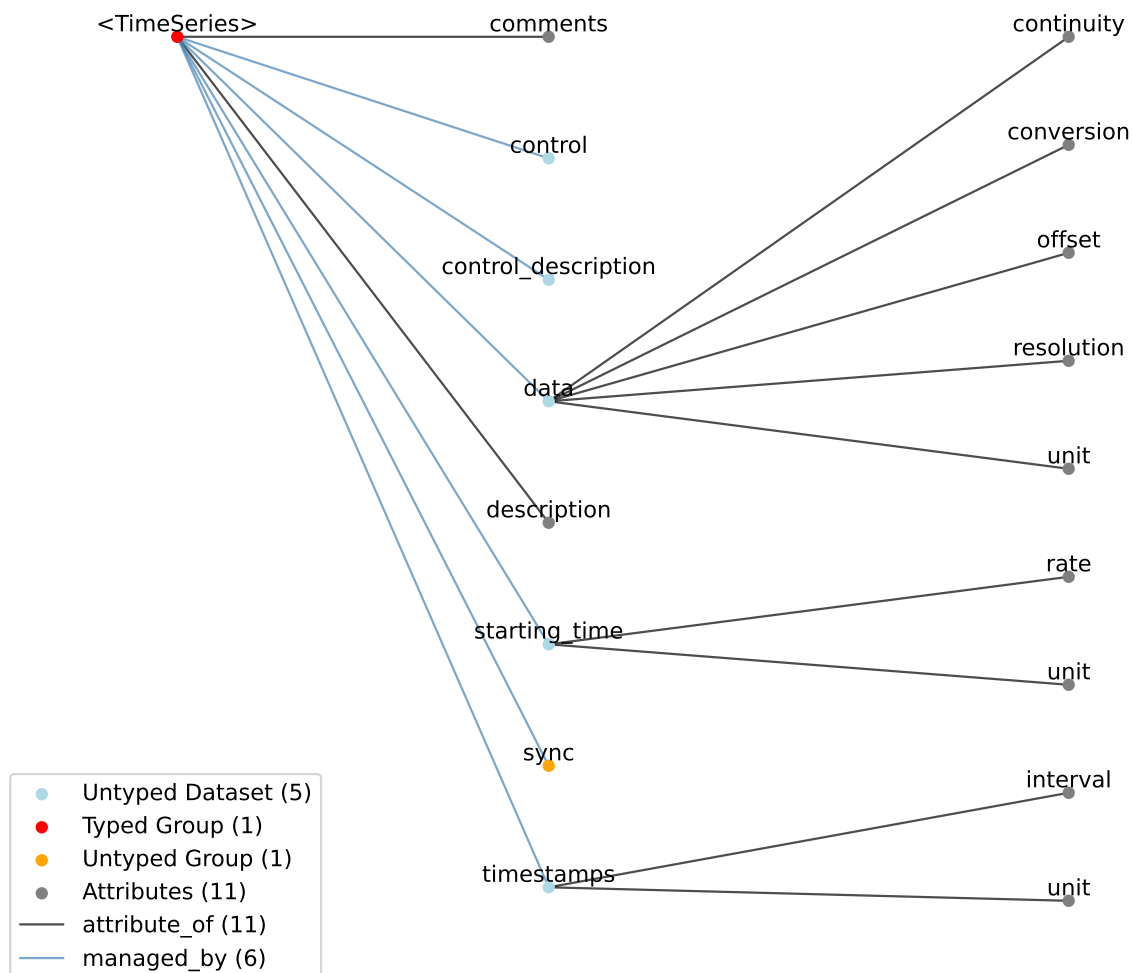


Table 4.3: Datasets, Links, and Attributes contained in <TimeSeries>

Id	Type	Description
<TimeSeries>	Group	Top level Group for <TimeSeries> • Neurodata Type: <i>TimeSeries</i> • Extends: <i>NWBDataInterface</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Data values. Data can be in 1-D, 2-D, 3-D, or 4-D. The first dimension should always represent time. This can also be used to store binary data (e.g., image frames). This can also be a link to data stored in an external file. • Dimensions: [['num_times'], ['num_times', 'num_DIM2'], ['num_times', 'num_DIM2', 'num_DIM3'], ['num_times', 'num_DIM2', 'num_DIM3', 'num_DIM4']] • Shape: [[None], [None, None], [None, None, None], [None, None, None, None]] • Name: data
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion

continues on next page

Table 4.3 – continued from previous page

Id	Type	Description
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Name: unit
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate

continues on next page

Table 4.3 – continued from previous page

Id	Type	Description
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description

Table 4.4: Groups contained in <TimeSeries>

Id	Type	Description
<TimeSeries>	Group	Top level Group for <TimeSeries> • Neurodata Type: <i>TimeSeries</i> • Extends: <i>NWBDataInterface</i>
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.1.7.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.1.8 ProcessingModule

Overview: A collection of processed data.

ProcessingModule extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.8

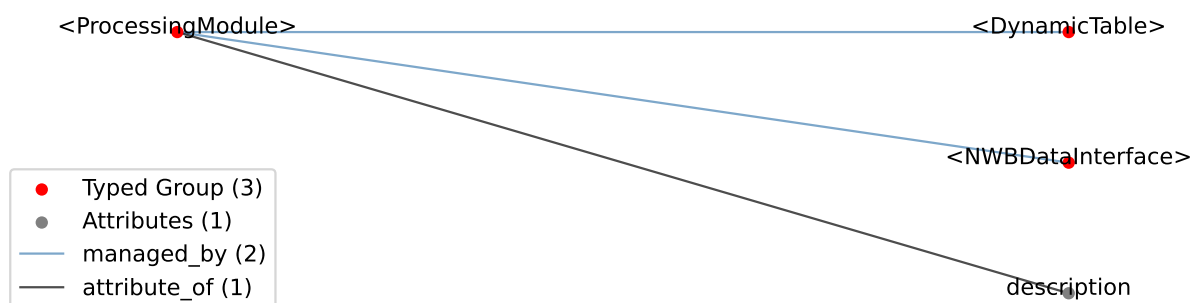


Table 4.5: Datasets, Links, and Attributes contained in <ProcessingModule>

Id	Type	Description
<ProcessingModule>	Group	Top level Group for <ProcessingModule> • Neurodata Type: <i>ProcessingModule</i> • Extends: <i>NWBContainer</i>
—description	At-tribute	Description of this collection of processed data. • Data Type: text • Name: description

Table 4.6: Groups contained in <ProcessingModule>

Id	Type	Description
<ProcessingModule>	Group	Top level Group for <ProcessingModule> • Neurodata Type: <i>ProcessingModule</i> • Extends: <i>NWBContainer</i>
—<NWBDataInterface>	Group	Data objects stored in this collection. • Extends: <i>NWBDataInterface</i> • Quantity: 0 or more

continues on next page

Table 4.6 – continued from previous page

Id	Type	Description
—<DynamicTable>	Group	Tables stored in this collection. • Extends: DynamicTable • Quantity: 0 or more

4.1.8.1 Groups: <NWBDatInterface>

Data objects stored in this collection.

- **Extends:** *NWBDatInterface*
- **Quantity:** 0 or more

4.1.8.2 Groups: <DynamicTable>

Tables stored in this collection.

- **Extends:** [DynamicTable](#)
- **Quantity:** 0 or more

4.1.9 Images

Overview: A collection of images with an optional way to specify the order of the images using the “order_of_images” dataset. An order must be specified if the images are referenced by index, e.g., from an IndexSeries.

Images extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** Images
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.base.yaml
- **Source Specification:** see Section 5.2.9

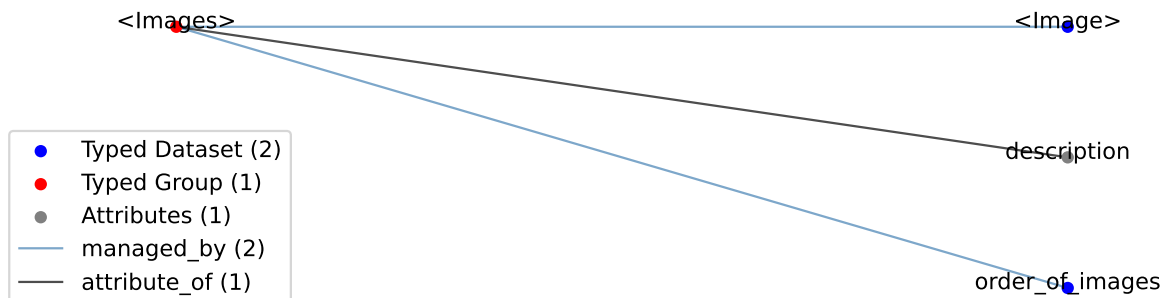


Table 4.7: Datasets, Links, and Attributes contained in <Images>

Id	Type	Description
<Images>	Group	Top level Group for <Images> • Neurodata Type: <i>Images</i> • Extends: <i>NWBDataInterface</i> • Default Name: Images
—description	At-tribute	Description of this collection of images. • Data Type: text • Name: description
—<Image>	Dataset	Images stored in this collection. • Extends: <i>Image</i> • Quantity: 1 or more
—order_of_images	Dataset	Ordered dataset of references to Image objects stored in the parent group. Each Image object in the Images group should be stored once and only once, so the dataset should have the same length as the number of images. • Extends: <i>ImageReferences</i> • Quantity: 0 or 1 • Name: order_of_images

4.2 Devices

This source module contains `neurodata_types` for device data.

4.2.1 Device

Overview: Metadata about a data acquisition device, e.g., recording system, electrode, microscope.

Device extends `NWBContainer` and includes all elements of `NWBContainer` with the following additions or changes.

- **Extends:** `NWBContainer`
- **Primitive Type:** Group
- **Inherits from:** `NWBContainer`, `Container`
- **Source filename:** `nwb.device.yaml`
- **Source Specification:** see [Section 5.3.1](#)

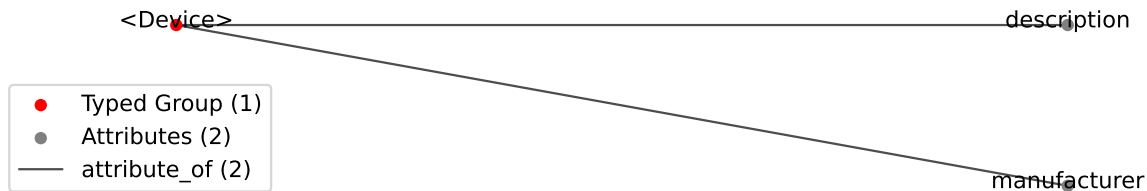


Table 4.8: Datasets, Links, and Attributes contained in `<Device>`

Id	Type	Description
<code><Device></code>	Group	Top level Group for <code><Device></code> • Neurodata Type: <code>Device</code> • Extends: <code>NWBContainer</code>
<code>—description</code>	At-tribute	Description of the device (e.g., model, firmware version, processing software version, etc.) as free-form text. • Data Type: text • Required: False • Name: description
<code>—manufacturer</code>	At-tribute	The name of the manufacturer of the device. • Data Type: text • Required: False • Name: manufacturer

4.3 Epochs

This source module contains `neurodata_types` for epoch data.

4.3.1 TimeIntervals

Overview: A container for aggregating epoch data and the `TimeSeries` that each epoch applies to.

`TimeIntervals` extends `DynamicTable` and includes all elements of `DynamicTable` with the following additions or changes.

- **Extends:** `DynamicTable`
- **Primitive Type:** Group
- **Inherits from:** `DynamicTable`, `Container`
- **Source filename:** `nwb.epoch.yaml`
- **Source Specification:** see [Section 5.4.1](#)

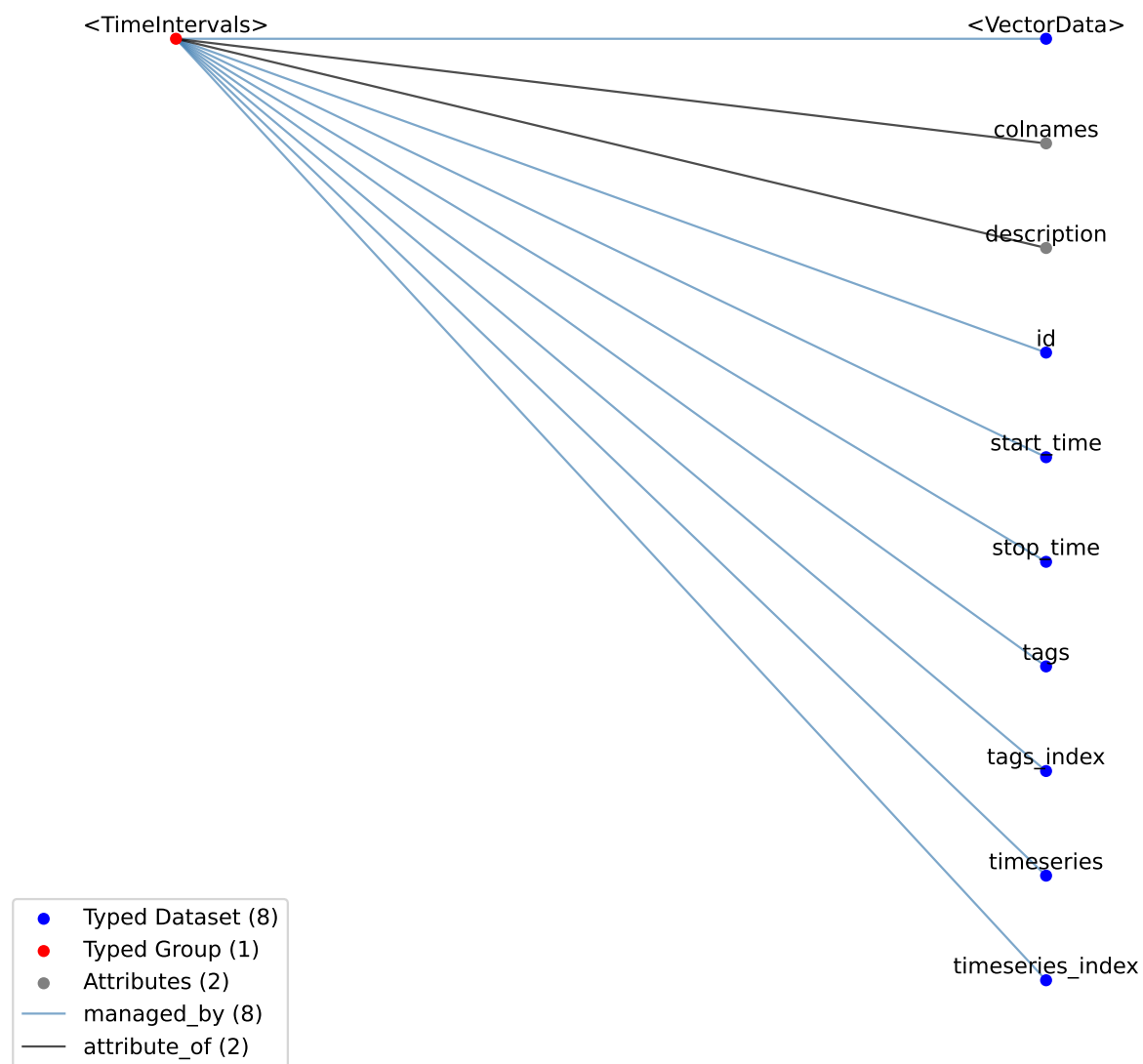
Table 4.9: Datasets, Links, and Attributes contained in `<TimeIntervals>`

Id	Type	Description
<code><TimeIntervals></code>	Group	Top level Group for <code><TimeIntervals></code> • Neurodata Type: <code>TimeIntervals</code> • Extends: <code>DynamicTable</code>
<code>—colnames</code>	Attribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: [<code>num_columns</code>] • Shape: [None] • Name: <code>colnames</code>
<code>—description</code>	Attribute	Description of what is in this dynamic table. • Data Type: text • Name: <code>description</code>
<code>—start_time</code>	Dataset	Start time of epoch, in seconds. • Extends: <code>VectorData</code> • Data Type: float32 • Name: <code>start_time</code>
<code>—stop_time</code>	Dataset	Stop time of epoch, in seconds. • Extends: <code>VectorData</code> • Data Type: float32 • Name: <code>stop_time</code>

continues on next page

Table 4.9 – continued from previous page

Id	Type	Description
—tags	Dataset	User-defined tags that identify or categorize events. • Extends: VectorData • Quantity: 0 or 1 • Data Type: text • Name: tags
—tags_index	Dataset	Index for tags. • Extends: VectorIndex • Quantity: 0 or 1 • Name: tags_index
—timeseries	Dataset	An index into a TimeSeries object. • Extends: TimeSeriesReferenceVectorData • Quantity: 0 or 1 • Name: timeseries
—timeseries_index	Dataset	Index for timeseries. • Extends: VectorIndex • Quantity: 0 or 1 • Name: timeseries_index
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more



4.4 Image data

This source module contains neurodata_types for image data.

4.4.1 GrayscaleImage

Overview: A grayscale image.

GrayscaleImage extends Image and includes all elements of *Image* with the following additions or changes.

- **Extends:** *Image*
- **Primitive Type:** Dataset
- **Data Type:** numeric
- **Dimensions:** ['x', 'y']
- **Shape:** [None, None]
- **Inherits from:** *Image*, *NWBData*, *Data*
- **Source filename:** nwb.image.yaml
- **Source Specification:** see [Section 5.5.1](#)

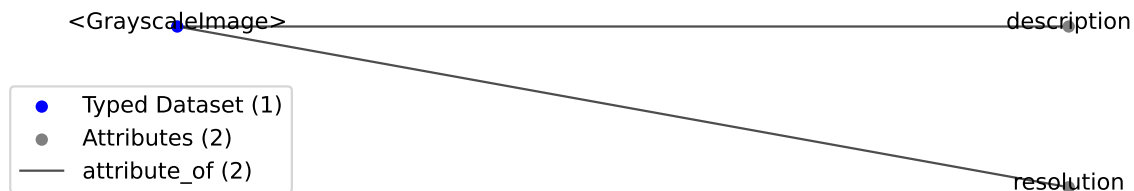


Table 4.10: Datasets, Links, and Attributes contained in <GrayscaleImage>

Id	Type	Description
<GrayscaleImage>	Dataset	Top level Dataset for <GrayscaleImage> • Neurodata Type: GrayscaleImage • Extends: <i>Image</i> • Data Type: numeric • Dimensions: ['x', 'y'] • Shape: [None, None]
—resolution	At-tribute	Pixel resolution of the image, in pixels per centimeter. • Data Type: float32 • Required: False • Name: resolution
—description	At-tribute	Description of the image. • Data Type: text • Required: False • Name: description

4.4.2 RGBImage

Overview: A color image.

RGBImage extends Image and includes all elements of Image with the following additions or changes.

- **Extends:** Image
- **Primitive Type:** Dataset
- **Data Type:** numeric
- **Dimensions:** ['x', 'y', 'r, g, b']
- **Shape:** [None, None, 3]
- **Inherits from:** Image, NWBData, Data
- **Source filename:** nwb.image.yaml
- **Source Specification:** see Section 5.5.2

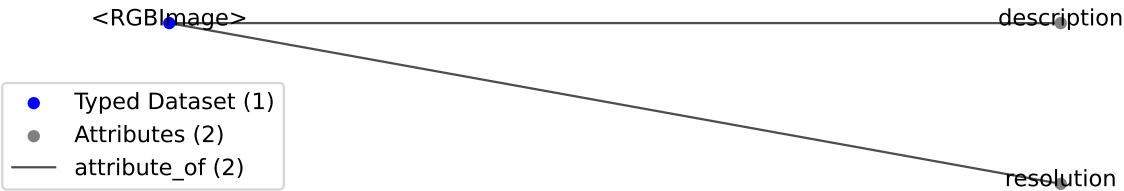


Table 4.11: Datasets, Links, and Attributes contained in <RGBImage>

Id	Type	Description
<RGBImage>	Dataset	Top level Dataset for <RGBImage> • Neurodata Type: RGBImage • Extends: Image • Data Type: numeric • Dimensions: ['x', 'y', 'r, g, b'] • Shape: [None, None, 3]
—resolution	At-tribute	Pixel resolution of the image, in pixels per centimeter. • Data Type: float32 • Required: False • Name: resolution
—description	At-tribute	Description of the image. • Data Type: text • Required: False • Name: description

4.4.3 RGBImage

Overview: A color image with transparency.

RGBImage extends Image and includes all elements of *Image* with the following additions or changes.

- **Extends:** *Image*
- **Primitive Type:** Dataset
- **Data Type:** numeric
- **Dimensions:** ['x', 'y', 'r, g, b, a']
- **Shape:** [None, None, 4]
- **Inherits from:** *Image*, *NWBData*, *Data*
- **Source filename:** nwb.image.yaml
- **Source Specification:** see [Section 5.5.3](#)

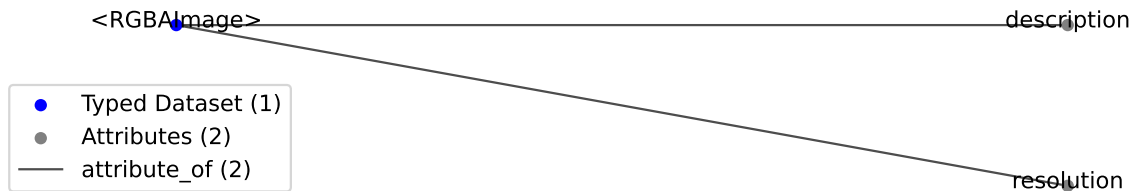


Table 4.12: Datasets, Links, and Attributes contained in <RGBImage>

Id	Type	Description
<RGBImage>	Dataset	Top level Dataset for <RGBImage> • Neurodata Type: RGBImage • Extends: <i>Image</i> • Data Type: numeric • Dimensions: ['x', 'y', 'r, g, b, a'] • Shape: [None, None, 4]
—resolution	At-tribute	Pixel resolution of the image, in pixels per centimeter. • Data Type: float32 • Required: False • Name: resolution
—description	At-tribute	Description of the image. • Data Type: text • Required: False • Name: description

4.4.4 ImageSeries

Overview: General image data that is common between acquisition and stimulus time series. Sometimes the image data is stored in the file in a raw format while other times it will be stored as a series of external image files in the host file system. The data field will either be binary data, if the data is stored in the NWB file, or empty, if the data is stored in an external image stack. [frame][x][y] or [frame][x][y][z].

ImageSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Subtypes:** *ImageMaskSeries*, *OpticalSeries*, *TwoPhotonSeries*, *OnePhotonSeries*
- **Source filename:** nwb.image.yaml
- **Source Specification:** see [Section 5.5.4](#)

Table 4.13: Datasets, Links, and Attributes contained in <ImageSeries>

Id	Type	Description
<ImageSeries>	Group	Top level Group for <ImageSeries> • Neurodata Type: <i>ImageSeries</i> • Extends: <i>TimeSeries</i>
—description	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	Attribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Binary data representing images across frames. If data are stored in an external file, this should be an empty 3D array. • Data Type: numeric • Dimensions: [['frame', 'x', 'y'], ['frame', 'x', 'y', 'z']] • Shape: [[None, None, None], [None, None, None, None]] • Name: data

continues on next page

Table 4.13 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Name: unit

continues on next page

Table 4.13 – continued from previous page

Id	Type	Description
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—dimension	Dataset	Number of pixels on x, y, (and z) axes. • Quantity: 0 or 1 • Data Type: int32 • Dimensions: ['rank'] • Shape: [None] • Name: dimension
—external_file	Dataset	Paths to one or more external file(s). The field is only present if format='external'. This is only relevant if the image series is stored in the file system as one or more image file(s). This field should NOT be used if the image is stored in another NWB file and that file is linked to this file. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_files'] • Shape: [None] • Name: external_file
—starting_frame	At-tribute	Each external image may contain one or more consecutive frames of the full ImageSeries. This attribute serves as an index to indicate which frames each file contains, to facilitate random access. The 'starting_frame' attribute, hence, contains a list of frame numbers within the full ImageSeries of the first frame of each file listed in the parent 'external_file' dataset. Zero-based indexing is used (hence, the first element will always be zero). For example, if the 'external_file' dataset has three paths to files and the first file has 5 frames, the second file has 10 frames, and the third file has 20 frames, then this attribute will have values [0, 5, 15]. If there is a single external file that holds all of the frames of the ImageSeries (and so there is a single element in the 'external_file' dataset), then this attribute should have value [0]. • Data Type: int32 • Dimensions: ['num_files'] • Shape: [None] • Name: starting_frame

continues on next page

Table 4.13 – continued from previous page

Id	Type	Description
—format	Dataset	Format of image. If this is ‘external’, then the attribute ‘external_file’ contains the path information to the image files. If this is ‘raw’, then the raw (single-channel) binary data is stored in the ‘data’ dataset. If this attribute is not present, then the default format=‘raw’ case is assumed. • Quantity: 0 or 1 • Data Type: text • Name: format
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

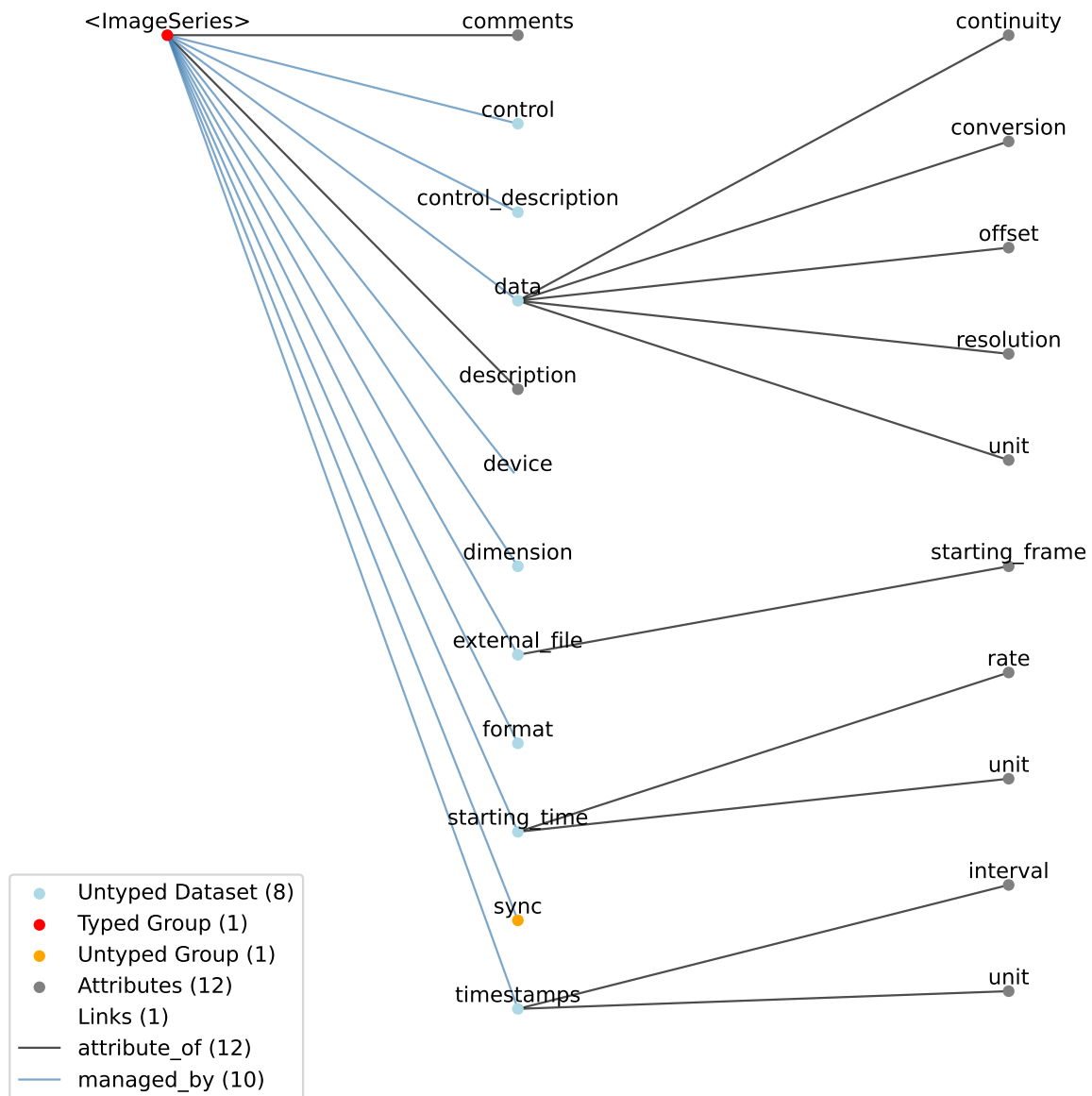
continues on next page

Table 4.13 – continued from previous page

Id	Type	Description
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

Table 4.14: Groups contained in <ImageSeries>

Id	Type	Description
<ImageSeries>	Group	Top level Group for <ImageSeries> • Neurodata Type: <i>ImageSeries</i> • Extends: <i>TimeSeries</i>
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync



4.4.4.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.4.5 ImageMaskSeries

Overview: An alpha mask that is applied to a presented visual stimulus. The ‘data’ array contains an array of mask values that are applied to the displayed image. Mask values are stored as RGBA. Mask can vary with time. The timestamps array indicates the starting time of a mask, and that mask pattern continues until it’s explicitly changed.

ImageMaskSeries extends ImageSeries and includes all elements of *ImageSeries* with the following additions or changes.

- **Extends:** *ImageSeries*
- **Primitive Type:** Group
- **Inherits from:** *ImageSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.image.yaml
- **Source Specification:** see [Section 5.5.5](#)

Table 4.15: Datasets, Links, and Attributes contained in <ImageMaskSeries>

Id	Type	Description
<ImageMaskSeries>	Group	Top level Group for <ImageMaskSeries> • Neurodata Type: <i>ImageMaskSeries</i> • Extends: <i>ImageSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Binary data representing images across frames. If data are stored in an external file, this should be an empty 3D array. • Data Type: numeric • Dimensions: [['frame', 'x', 'y'], ['frame', 'x', 'y', 'z']] • Shape: [[None, None, None], [None, None, None, None]] • Name: data

continues on next page

Table 4.15 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Name: unit

continues on next page

Table 4.15 – continued from previous page

Id	Type	Description
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—dimension	Dataset	Number of pixels on x, y, (and z) axes. • Quantity: 0 or 1 • Data Type: int32 • Dimensions: ['rank'] • Shape: [None] • Name: dimension
—external_file	Dataset	Paths to one or more external file(s). The field is only present if format='external'. This is only relevant if the image series is stored in the file system as one or more image file(s). This field should NOT be used if the image is stored in another NWB file and that file is linked to this file. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_files'] • Shape: [None] • Name: external_file
—starting_frame	At-tribute	Each external image may contain one or more consecutive frames of the full ImageSeries. This attribute serves as an index to indicate which frames each file contains, to facilitate random access. The 'starting_frame' attribute, hence, contains a list of frame numbers within the full ImageSeries of the first frame of each file listed in the parent 'external_file' dataset. Zero-based indexing is used (hence, the first element will always be zero). For example, if the 'external_file' dataset has three paths to files and the first file has 5 frames, the second file has 10 frames, and the third file has 20 frames, then this attribute will have values [0, 5, 15]. If there is a single external file that holds all of the frames of the ImageSeries (and so there is a single element in the 'external_file' dataset), then this attribute should have value [0]. • Data Type: int32 • Dimensions: ['num_files'] • Shape: [None] • Name: starting_frame

continues on next page

Table 4.15 – continued from previous page

Id	Type	Description
—format	Dataset	Format of image. If this is ‘external’, then the attribute ‘external_file’ contains the path information to the image files. If this is ‘raw’, then the raw (single-channel) binary data is stored in the ‘data’ dataset. If this attribute is not present, then the default format=‘raw’ case is assumed. • Quantity: 0 or 1 • Data Type: text • Name: format
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.15 – continued from previous page

Id	Type	Description
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—masked_imageseries	Link	Link to ImageSeries object that this image mask is applied to. • Target Type <i>ImageSeries</i> • Name: masked_imageseries
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

Table 4.16: Groups contained in <ImageMaskSeries>

Id	Type	Description
<ImageMaskSeries>	Group	Top level Group for <ImageMaskSeries> • Neurodata Type: <i>ImageMaskSeries</i> • Extends: <i>ImageSeries</i>
—masked_imageseries	Link	Link to ImageSeries object that this image mask is applied to. • Target Type <i>ImageSeries</i> • Name: masked_imageseries
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

continues on next page

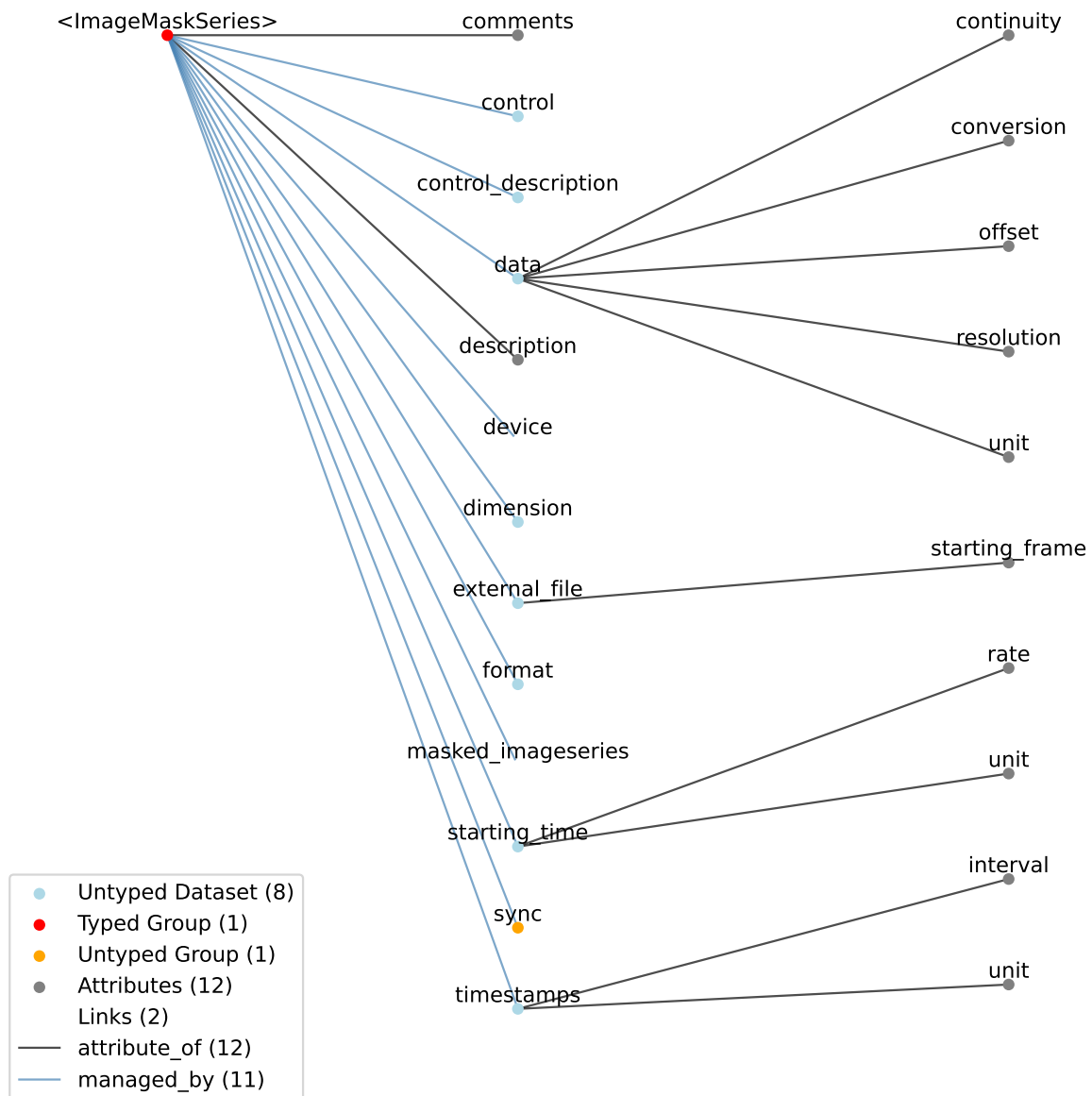
Table 4.16 – continued from previous page

Id	Type	Description
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.4.5.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync



4.4.6 OpticalSeries

Overview: Image data that is presented or recorded. A stimulus template movie will be stored only as an image. When the image is presented as stimulus, additional data is required, such as field of view (e.g., how much of the visual field the image covers, or how what is the area of the target being imaged). If the OpticalSeries represents acquired imaging data, orientation is also important.

OpticalSeries extends ImageSeries and includes all elements of *ImageSeries* with the following additions or changes.

- **Extends:** *ImageSeries*
- **Primitive Type:** Group
- **Inherits from:** *ImageSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.image.yaml
- **Source Specification:** see Section 5.5.6

Table 4.17: Datasets, Links, and Attributes contained in <OpticalSeries>

Id	Type	Description
<OpticalSeries>	Group	Top level Group for <OpticalSeries> • Neurodata Type: <i>OpticalSeries</i> • Extends: <i>ImageSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—distance	Dataset	Distance from camera/monitor to target/eye. • Quantity: 0 or 1 • Data Type: float32 • Name: distance
—field_of_view	Dataset	Width, height and depth of image, or imaged area, in meters. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['width, height'], ['width, height, depth']] • Shape: [[2], [3]] • Name: field_of_view

continues on next page

Table 4.17 – continued from previous page

Id	Type	Description
—data	Dataset	Images presented to subject, either grayscale or RGB • Data Type: numeric • Dimensions: [['frame', 'x', 'y'], ['frame', 'x', 'y', 'r, g, b']] • Shape: [[None, None, None], [None, None, None, 3]] • Name: data
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
—offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
—resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
—unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply 'data' by 'conversion' and add 'offset'. • Data Type: text • Name: unit

continues on next page

Table 4.17 – continued from previous page

Id	Type	Description
—continuity	Attribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—orientation	Dataset	Description of image relative to some reference frame (e.g., which way is up). Must also specify frame of reference. • Quantity: 0 or 1 • Data Type: text • Name: orientation
—dimension	Dataset	Number of pixels on x, y, (and z) axes. • Quantity: 0 or 1 • Data Type: int32 • Dimensions: ['rank'] • Shape: [None] • Name: dimension
—external_file	Dataset	Paths to one or more external file(s). The field is only present if format='external'. This is only relevant if the image series is stored in the file system as one or more image file(s). This field should NOT be used if the image is stored in another NWB file and that file is linked to this file. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_files'] • Shape: [None] • Name: external_file

continues on next page

Table 4.17 – continued from previous page

Id	Type	Description
—starting_frame	At-tribute	Each external image may contain one or more consecutive frames of the full ImageSeries. This attribute serves as an index to indicate which frames each file contains, to facilitate random access. The ‘starting_frame’ attribute, hence, contains a list of frame numbers within the full ImageSeries of the first frame of each file listed in the parent ‘external_file’ dataset. Zero-based indexing is used (hence, the first element will always be zero). For example, if the ‘external_file’ dataset has three paths to files and the first file has 5 frames, the second file has 10 frames, and the third file has 20 frames, then this attribute will have values [0, 5, 15]. If there is a single external file that holds all of the frames of the ImageSeries (and so there is a single element in the ‘external_file’ dataset), then this attribute should have value [0]. • Data Type: int32 • Dimensions: [‘num_files’] • Shape: [None] • Name: starting_frame
—format	Dataset	Format of image. If this is ‘external’, then the attribute ‘external_file’ contains the path information to the image files. If this is ‘raw’, then the raw (single-channel) binary data is stored in the ‘data’ dataset. If this attribute is not present, then the default format=‘raw’ case is assumed. • Quantity: 0 or 1 • Data Type: text • Name: format
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps

continues on next page

Table 4.17 – continued from previous page

Id	Type	Description
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
——control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
——control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description
——device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

Table 4.18: Groups contained in <OpticalSeries>

Id	Type	Description
<OpticalSeries>	Group	Top level Group for <OpticalSeries> • Neurodata Type: <i>OpticalSeries</i> • Extends: <i>ImageSeries</i>
——device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

continues on next page

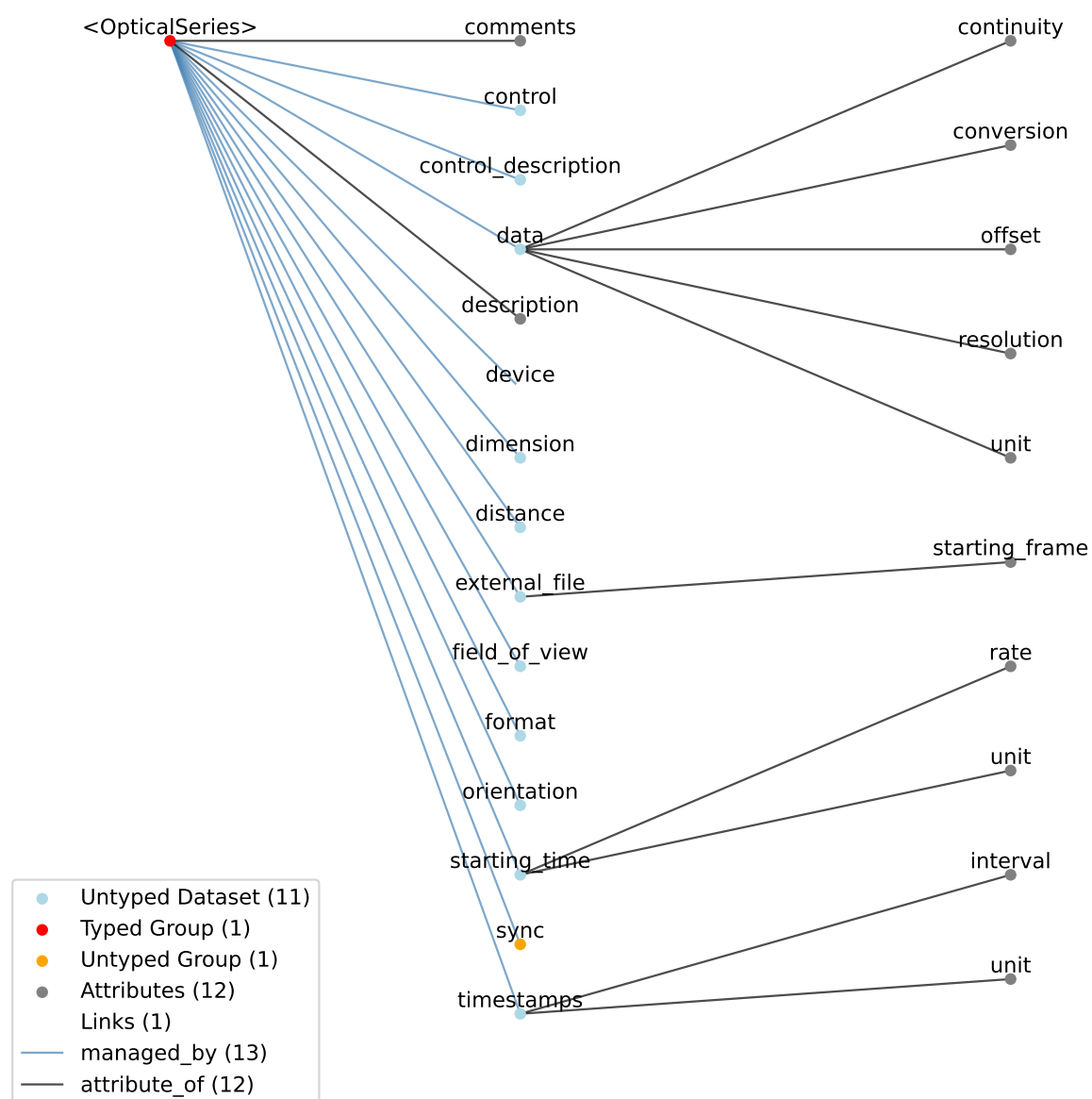
Table 4.18 – continued from previous page

Id	Type	Description
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.4.6.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync



4.4.7 IndexSeries

Overview: Stores indices to image frames stored in an ImageSeries. The purpose of the IndexSeries is to allow a static image stack to be stored in an Images object, and the images in the stack to be referenced out-of-order. This can be for the display of individual images, or of movie segments (as a movie is simply a series of images). The data field stores the index of the frame in the referenced Images object, and the timestamps array indicates when that image was displayed.

IndexSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.image.yaml
- **Source Specification:** see [Section 5.5.7](#)

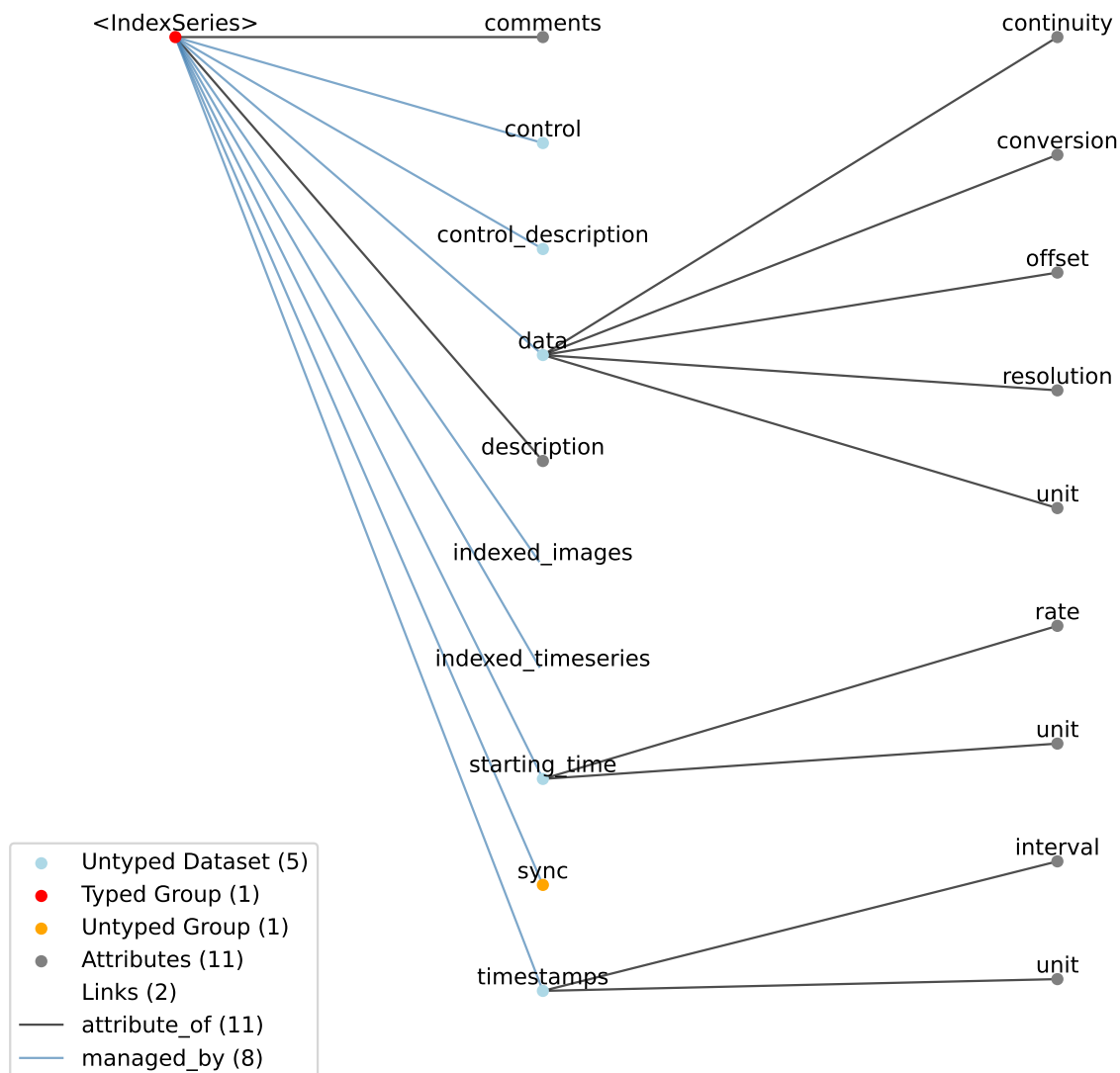


Table 4.19: Datasets, Links, and Attributes contained in <IndexSeries>

Id	Type	Description
<IndexSeries>	Group	Top level Group for <IndexSeries> • Neurodata Type: <i>IndexSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Index of the image (using zero-indexing) in the linked Images object. • Data Type: uint32 • Dimensions: ['num_times'] • Shape: [None] • Name: data
—conversion	At-tribute	This field is unused by IndexSeries. • Data Type: float32 • Required: False • Name: conversion
—resolution	At-tribute	This field is unused by IndexSeries. • Data Type: float32 • Required: False • Name: resolution
—offset	At-tribute	This field is unused by IndexSeries. • Data Type: float32 • Required: False • Name: offset
—unit	At-tribute	This field is unused by IndexSeries and has the value N/A. • Data Type: text • Value: N/A • Name: unit

continues on next page

Table 4.19 – continued from previous page

Id	Type	Description
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
——timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.19 – continued from previous page

Id	Type	Description
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—indexed_timeseries	Link	Link to ImageSeries object containing images that are indexed. Use of this link is discouraged and will be deprecated. Link to an Images type instead. • Target Type <i>ImageSeries</i> • Name: indexed_timeseries
—indexed_images	Link	Link to Images object containing an ordered set of images that are indexed. The Images object must contain a 'ordered_images' dataset specifying the order of the images in the Images type. • Target Type <i>Images</i> • Name: indexed_images

Table 4.20: Groups contained in <IndexSeries>

Id	Type	Description
<IndexSeries>	Group	Top level Group for <IndexSeries> • Neurodata Type: <i>IndexSeries</i> • Extends: <i>TimeSeries</i>
—indexed_timeseries	Link	Link to ImageSeries object containing images that are indexed. Use of this link is discouraged and will be deprecated. Link to an Images type instead. • Target Type <i>ImageSeries</i> • Name: indexed_timeseries

continues on next page

Table 4.20 – continued from previous page

Id	Type	Description
—indexed_images	Link	Link to Images object containing an ordered set of images that are indexed. The Images object must contain a ‘ordered_images’ dataset specifying the order of the images in the Images type. • Target Type <i>Images</i> • Name: indexed_images
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.4.7.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.5 NWB file

Main NWB file specification.

4.5.1 ScratchData

Overview: Any one-off datasets

ScratchData extends NWBData and includes all elements of *NWBData* with the following additions or changes.

- **Extends:** *NWBData*
- **Primitive Type:** Dataset
- **Inherits from:** *NWBData*, *Data*
- **Source filename:** nwb.file.yaml
- **Source Specification:** see [Section 5.6.1](#)

Table 4.21: Datasets, Links, and Attributes contained in <ScratchData>

Id	Type	Description
<ScratchData>	Dataset	Top level Dataset for <ScratchData> • Neurodata Type: ScratchData • Extends: <i>NWBData</i>
—notes	At-tribute	Any notes the user has about the dataset being stored • Data Type: text • Name: notes

4.5.2 NWBFile

Overview: An NWB file storing cellular-based neurophysiology data from a single experimental session.

NWBFile extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Name:** root
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.file.yaml
- **Source Specification:** see [Section 5.6.2](#)

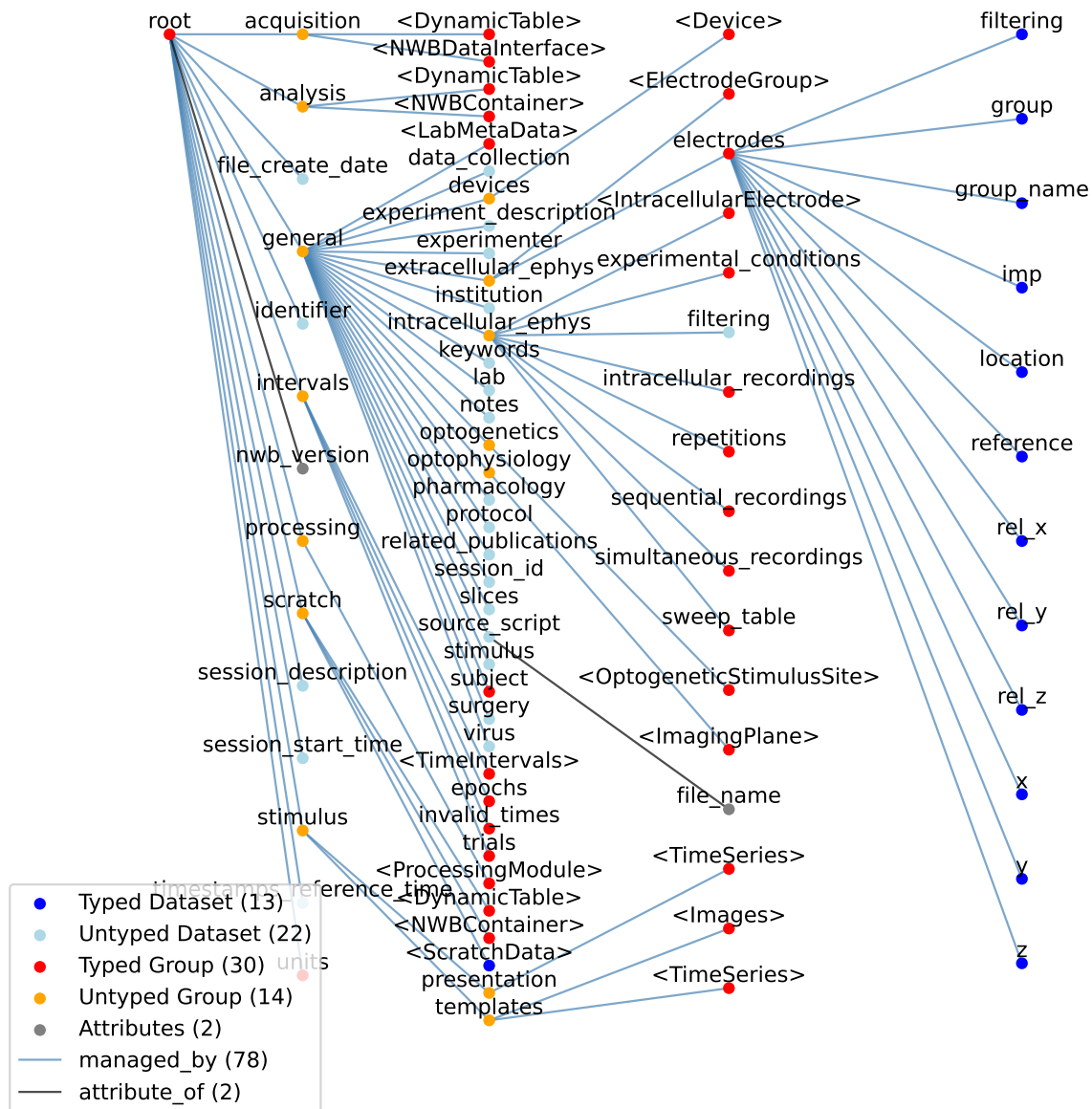


Table 4.22: Datasets, Links, and Attributes contained in <NWBFile>

Id	Type	Description
root	Group	Top level Group for root • Neurodata Type: <i>NWBFile</i> • Extends: <i>NWBContainer</i> • Name: root
—nwb_version	Attribute	File version string. Use semantic versioning, e.g. 1.2.1. This will be the name of the format with trailing major, minor and patch numbers. • Data Type: text • Value: 2.6.0 • Name: nwb_version
—file_create_date	Dataset	A record of the date the file was created and of subsequent modifications. The date is stored in UTC with local timezone offset as ISO 8601 extended formatted strings: 2018-09-28T14:43:54.123+02:00. Dates stored in UTC end in “Z” with no timezone offset. Date accuracy is up to milliseconds. The file can be created after the experiment was run, so this may differ from the experiment start time. Each modification to the nwb file adds a new entry to the array. • Data Type: isodatetime • Dimensions: [‘num_modifications’] • Shape: [None] • Name: file_create_date
—identifier	Dataset	A unique text identifier for the file. For example, concatenated lab name, file creation date/time and experimentalist, or a hash of these and/or other values. The goal is that the string should be unique to all other files. • Data Type: text • Name: identifier
—session_description	Dataset	A description of the experimental session and data in the file. • Data Type: text • Name: session_description
—session_start_time	Dataset	Date and time of the experiment/session start. The date is stored in UTC with local timezone offset as ISO 8601 extended formatted string: 2018-09-28T14:43:54.123+02:00. Dates stored in UTC end in “Z” with no timezone offset. Date accuracy is up to milliseconds. • Data Type: isodatetime • Name: session_start_time
—timestamps_reference_time	Dataset	Date and time corresponding to time zero of all timestamps. The date is stored in UTC with local timezone offset as ISO 8601 extended formatted string: 2018-09-28T14:43:54.123+02:00. Dates stored in UTC end in “Z” with no timezone offset. Date accuracy is up to milliseconds. All times stored in the file use this time as reference (i.e., time zero). • Data Type: isodatetime • Name: timestamps_reference_time

Table 4.23: Groups contained in <NWBFile>

Id	Type	Description
root	Group	Top level Group for root • Neurodata Type: <i>NWBFile</i> • Extends: <i>NWBContainer</i> • Name: root
—acquisition	Group	Data streams recorded from the system, including ephys, ophys, tracking, etc. This group should be read-only after the experiment is completed and timestamps are corrected to a common timebase. The data stored here may be links to raw data stored in external NWB files. This will allow keeping bulky raw data out of the file while preserving the option of keeping some/all in the file. Acquired data includes tracking and experimental data streams (i.e., everything measured from the system). If bulky data is stored in the /acquisition group, the data can exist in a separate NWB file that is linked to by the file being used for processing and analysis. • Name: acquisition
—analysis	Group	Lab-specific and custom scientific analysis of data. There is no defined format for the content of this group - the format is up to the individual user/lab. To facilitate sharing analysis data between labs, the contents here should be stored in standard types (e.g., neurodata_types) and appropriately documented. The file can store lab-specific and custom data analysis without restriction on its form or schema, reducing data formatting restrictions on end users. Such data should be placed in the analysis group. The analysis data should be documented so that it could be shared with other labs. • Name: analysis
—scratch	Group	A place to store one-off analysis results. Data placed here is not intended for sharing. By placing data here, users acknowledge that there is no guarantee that their data meets any standard. • Quantity: 0 or 1 • Name: scratch
—processing	Group	The home for ProcessingModules. These modules perform intermediate analysis of data that is necessary to perform before scientific analysis. Examples include spike clustering, extracting position from tracking data, stitching together image slices. ProcessingModules can be large and express many data sets from relatively complex analysis (e.g., spike detection and clustering) or small, representing extraction of position information from tracking video, or even binary lick/no-lick decisions. Common software tools (e.g., klustakwik, MClust) are expected to read/write data here. ‘Processing’ refers to intermediate analysis of the acquired data to make it more amenable to scientific analysis. • Name: processing

continues on next page

Table 4.23 – continued from previous page

Id	Type	Description
—stimulus	Group	Data pushed into the system (eg, video stimulus, sound, voltage, etc) and secondary representations of that data (eg, measurements of something used as a stimulus). This group should be made read-only after experiment complete and timestamps are corrected to common timebase. Stores both presented stimuli and stimulus templates, the latter in case the same stimulus is presented multiple times, or is pulled from an external stimulus library. Stimuli are here defined as any signal that is pushed into the system as part of the experiment (eg, sound, video, voltage, etc). Many different experiments can use the same stimuli, and stimuli can be re-used during an experiment. The stimulus group is organized so that one version of template stimuli can be stored and these be used multiple times. These templates can exist in the present file or can be linked to a remote library file. • Name: stimulus
—general	Group	Experimental metadata, including protocol, notes and description of hardware device(s). The metadata stored in this section should be used to describe the experiment. Metadata necessary for interpreting the data is stored with the data. General experimental metadata, including animal strain, experimental protocols, experimenter, devices, etc, are stored under ‘general’. Core metadata (e.g., that required to interpret data fields) is stored with the data itself, and implicitly defined by the file specification (e.g., time is in seconds). The strategy used here for storing non-core metadata is to use free-form text fields, such as would appear in sentences or paragraphs from a Methods section. Metadata fields are text to enable them to be more general, for example to represent ranges instead of numerical values. Machine-readable metadata is stored as attributes to these free-form datasets. All entries in the below table are to be included when data is present. Unused groups (e.g., intracellular_ephys in an optophysiology experiment) should not be created unless there is data to store within them. • Name: general
—intervals	Group	Experimental intervals, whether that be logically distinct sub-experiments having a particular scientific goal, trials (see trials subgroup) during an experiment, or epochs (see epochs subgroup) deriving from analysis of data. • Quantity: 0 or 1 • Name: intervals
—units	Group	Data about sorted spike units. • Extends: <i>Units</i> • Quantity: 0 or 1 • Name: units

4.5.2.1 Groups: /acquisition

Data streams recorded from the system, including ephys, ophys, tracking, etc. This group should be read-only after the experiment is completed and timestamps are corrected to a common timebase. The data stored here may be links to raw data stored in external NWB files. This will allow keeping bulky raw data out of the file while preserving the option of keeping some/all in the file. Acquired data includes tracking and experimental data streams (i.e., everything measured from the system). If bulky data is stored in the /acquisition group, the data can exist in a separate NWB file that is linked to by the file being used for processing and analysis.

- **Name:** acquisition

Table 4.24: Groups contained in <acquisition>

Id	Type	Description
acquisition	Group	Top level Group for acquisition • Name: acquisition
—<NWBDataInterface>	Group	Acquired, raw data. • Extends: <i>NWBDataInterface</i> • Quantity: 0 or more
—<DynamicTable>	Group	Tabular data that is relevant to acquisition • Extends: <i>DynamicTable</i> • Quantity: 0 or more

4.5.2.2 Groups: /acquisition/<NWBDataInterface>

Acquired, raw data.

- **Extends:** *NWBDataInterface*
- **Quantity:** 0 or more

4.5.2.3 Groups: /acquisition/<DynamicTable>

Tabular data that is relevant to acquisition

- **Extends:** *DynamicTable*
- **Quantity:** 0 or more

4.5.2.4 Groups: /analysis

Lab-specific and custom scientific analysis of data. There is no defined format for the content of this group - the format is up to the individual user/lab. To facilitate sharing analysis data between labs, the contents here should be stored in standard types (e.g., *neurodata_types*) and appropriately documented. The file can store lab-specific and custom data analysis without restriction on its form or schema, reducing data formatting restrictions on end users. Such data should be placed in the analysis group. The analysis data should be documented so that it could be shared with other labs.

- **Name:** analysis

Table 4.25: Groups contained in <analysis>

Id	Type	Description
analysis	Group	Top level Group for analysis • Name: analysis
—<NWBContainer>	Group	Custom analysis results. • Extends: <i>NWBContainer</i> • Quantity: 0 or more
—<DynamicTable>	Group	Tabular data that is relevant to data stored in analysis • Extends: <i>DynamicTable</i> • Quantity: 0 or more

4.5.2.5 Groups: /analysis/<NWBContainer>

Custom analysis results.

- **Extends:** *NWBContainer*
- **Quantity:** 0 or more

4.5.2.6 Groups: /analysis/<DynamicTable>

Tabular data that is relevant to data stored in analysis

- **Extends:** *DynamicTable*
- **Quantity:** 0 or more

4.5.2.7 Groups: /scratch

A place to store one-off analysis results. Data placed here is not intended for sharing. By placing data here, users acknowledge that there is no guarantee that their data meets any standard.

- **Quantity:** 0 or 1
- **Name:** scratch

Table 4.26: Datasets, Links, and Attributes contained in /scratch

Id	Type	Description
scratch	Group	Top level Group for scratch • Quantity: 0 or 1 • Name: scratch
—<ScratchData>	Dataset	Any one-off datasets • Extends: <i>ScratchData</i> • Quantity: 0 or more

Table 4.27: Groups contained in <scratch>

Id	Type	Description
scratch	Group	Top level Group for scratch • Quantity: 0 or 1 • Name: scratch
—<NWBContainer>	Group	Any one-off containers • Extends: <i>NWBContainer</i> • Quantity: 0 or more
—<DynamicTable>	Group	Any one-off tables • Extends: <i>DynamicTable</i> • Quantity: 0 or more

4.5.2.8 Groups: /scratch/<NWBContainer>

Any one-off containers

- **Extends:** *NWBContainer*
- **Quantity:** 0 or more

4.5.2.9 Groups: /scratch/<DynamicTable>

Any one-off tables

- **Extends:** *DynamicTable*
- **Quantity:** 0 or more

4.5.2.10 Groups: /processing

The home for ProcessingModules. These modules perform intermediate analysis of data that is necessary to perform before scientific analysis. Examples include spike clustering, extracting position from tracking data, stitching together image slices. ProcessingModules can be large and express many data sets from relatively complex analysis (e.g., spike detection and clustering) or small, representing extraction of position information from tracking video, or even binary lick/no-lick decisions. Common software tools (e.g., klustakwik, MClust) are expected to read/write data here. ‘Processing’ refers to intermediate analysis of the acquired data to make it more amenable to scientific analysis.

- **Name:** processing

Table 4.28: Groups contained in <processing>

Id	Type	Description
processing	Group	Top level Group for processing • Name: processing

continues on next page

Table 4.28 – continued from previous page

Id	Type	Description
— <i><ProcessingModule></i>	Group	Intermediate analysis of acquired data. • Extends: <i>ProcessingModule</i> • Quantity: 0 or more

4.5.2.11 Groups: /processing/<ProcessingModule>

Intermediate analysis of acquired data.

- **Extends:** *ProcessingModule*
- **Quantity:** 0 or more

4.5.2.12 Groups: /stimulus

Data pushed into the system (eg, video stimulus, sound, voltage, etc) and secondary representations of that data (eg, measurements of something used as a stimulus). This group should be made read-only after experiment complete and timestamps are corrected to common timebase. Stores both presented stimuli and stimulus templates, the latter in case the same stimulus is presented multiple times, or is pulled from an external stimulus library. Stimuli are here defined as any signal that is pushed into the system as part of the experiment (eg, sound, video, voltage, etc). Many different experiments can use the same stimuli, and stimuli can be re-used during an experiment. The stimulus group is organized so that one version of template stimuli can be stored and these be used multiple times. These templates can exist in the present file or can be linked to a remote library file.

- **Name:** stimulus

Table 4.29: Groups contained in <stimulus>

Id	Type	Description
stimulus	Group	Top level Group for stimulus • Name: stimulus
—presentation	Group	Stimuli presented during the experiment. • Name: presentation
—templates	Group	Template stimuli. Timestamps in templates are based on stimulus design and are relative to the beginning of the stimulus. When templates are used, the stimulus instances must convert presentation times to the experiment's time reference frame. • Name: templates

4.5.2.13 Groups: /stimulus/presentation

Stimuli presented during the experiment.

- **Name:** presentation

Table 4.30: Groups contained in <presentation>

Id	Type	Description
presentation	Group	Top level Group for presentation • Name: presentation
—< <i>TimeSeries</i> >	Group	TimeSeries objects containing data of presented stimuli. • Extends: <i>TimeSeries</i> • Quantity: 0 or more

4.5.2.14 Groups: /stimulus/presentation/<TimeSeries>

TimeSeries objects containing data of presented stimuli.

- **Extends:** *TimeSeries*
- **Quantity:** 0 or more

4.5.2.15 Groups: /stimulus/templates

Template stimuli. Timestamps in templates are based on stimulus design and are relative to the beginning of the stimulus. When templates are used, the stimulus instances must convert presentation times to the experiment's time reference frame.

- **Name:** templates

Table 4.31: Groups contained in <templates>

Id	Type	Description
templates	Group	Top level Group for templates • Name: templates
—< <i>TimeSeries</i> >	Group	TimeSeries objects containing template data of presented stimuli. • Extends: <i>TimeSeries</i> • Quantity: 0 or more
—< <i>Images</i> >	Group	Images objects containing images of presented stimuli. • Extends: <i>Images</i> • Quantity: 0 or more

4.5.2.16 Groups: /stimulus/templates/<TimeSeries>

TimeSeries objects containing template data of presented stimuli.

- **Extends:** *TimeSeries*
- **Quantity:** 0 or more

4.5.2.17 Groups: /stimulus/templates/<Images>

Images objects containing images of presented stimuli.

- **Extends:** *Images*
- **Quantity:** 0 or more

4.5.2.18 Groups: /general

Experimental metadata, including protocol, notes and description of hardware device(s). The metadata stored in this section should be used to describe the experiment. Metadata necessary for interpreting the data is stored with the data. General experimental metadata, including animal strain, experimental protocols, experimenter, devices, etc, are stored under 'general'. Core metadata (e.g., that required to interpret data fields) is stored with the data itself, and implicitly defined by the file specification (e.g., time is in seconds). The strategy used here for storing non-core metadata is to use free-form text fields, such as would appear in sentences or paragraphs from a Methods section. Metadata fields are text to enable them to be more general, for example to represent ranges instead of numerical values. Machine-readable metadata is stored as attributes to these free-form datasets. All entries in the below table are to be included when data is present. Unused groups (e.g., intracellular_ephys in an optophysiology experiment) should not be created unless there is data to store within them.

- **Name:** general

Table 4.32: Datasets, Links, and Attributes contained in /general

Id	Type	Description
general	Group	Top level Group for general • Name: general
—data_collection	Dataset	Notes about data collection and analysis. • Quantity: 0 or 1 • Data Type: text • Name: data_collection
—experiment_description	Dataset	General description of the experiment. • Quantity: 0 or 1 • Data Type: text • Name: experiment_description

continues on next page

Table 4.32 – continued from previous page

Id	Type	Description
—experimenter	Dataset	Name of person(s) who performed the experiment. Can also specify roles of different people involved. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_experimenter'] • Shape: [None] • Name: experimenter
—institution	Dataset	Institution(s) where experiment was performed. • Quantity: 0 or 1 • Data Type: text • Name: institution
—keywords	Dataset	Terms to search over. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_keywords'] • Shape: [None] • Name: keywords
—lab	Dataset	Laboratory where experiment was performed. • Quantity: 0 or 1 • Data Type: text • Name: lab
—notes	Dataset	Notes about the experiment. • Quantity: 0 or 1 • Data Type: text • Name: notes
—pharmacology	Dataset	Description of drugs used, including how and when they were administered. Anesthesia(s), painkiller(s), etc., plus dosage, concentration, etc. • Quantity: 0 or 1 • Data Type: text • Name: pharmacology
—protocol	Dataset	Experimental protocol, if applicable. e.g., include IACUC protocol number. • Quantity: 0 or 1 • Data Type: text • Name: protocol
—related_publications	Dataset	Publication information. PMID, DOI, URL, etc. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_publications'] • Shape: [None] • Name: related_publications

continues on next page

Table 4.32 – continued from previous page

Id	Type	Description
—session_id	Dataset	Lab-specific ID for the session. • Quantity: 0 or 1 • Data Type: text • Name: session_id
—slices	Dataset	Description of slices, including information about preparation thickness, orientation, temperature, and bath solution. • Quantity: 0 or 1 • Data Type: text • Name: slices
—source_script	Dataset	Script file or link to public source code used to create this NWB file. • Quantity: 0 or 1 • Data Type: text • Name: source_script
——file_name	At-tribute	Name of script file. • Data Type: text • Name: file_name
—stimulus	Dataset	Notes about stimuli, such as how and where they were presented. • Quantity: 0 or 1 • Data Type: text • Name: stimulus
—surgery	Dataset	Narrative description about surgery/surgeries, including date(s) and who performed surgery. • Quantity: 0 or 1 • Data Type: text • Name: surgery
—virus	Dataset	Information about virus(es) used in experiments, including virus ID, source, date made, injection location, volume, etc. • Quantity: 0 or 1 • Data Type: text • Name: virus

Table 4.33: Groups contained in <general>

Id	Type	Description
general	Group	Top level Group for general • Name: general
—<LabMetaData>	Group	Place-holder than can be extended so that lab-specific meta-data can be placed in /general. • Extends: <i>LabMetaData</i> • Quantity: 0 or more

continues on next page

Table 4.33 – continued from previous page

Id	Type	Description
—devices	Group	Description of hardware devices used during experiment, e.g., monitors, ADC boards, microscopes, etc. • Quantity: 0 or 1 • Name: devices
—subject	Group	Information about the animal or person from which the data was measured. • Extends: <i>Subject</i> • Quantity: 0 or 1 • Name: subject
—extracellular_ephys	Group	Metadata related to extracellular electrophysiology. • Quantity: 0 or 1 • Name: extracellular_ephys
—intracellular_ephys	Group	Metadata related to intracellular electrophysiology. • Quantity: 0 or 1 • Name: intracellular_ephys
—optogenetics	Group	Metadata describing optogenetic stimulation. • Quantity: 0 or 1 • Name: optogenetics
—optophysiology	Group	Metadata related to optophysiology. • Quantity: 0 or 1 • Name: optophysiology

4.5.2.19 Groups: /general/<LabMetaData>

Place-holder than can be extended so that lab-specific meta-data can be placed in /general.

- **Extends:** *LabMetaData*
- **Quantity:** 0 or more

4.5.2.20 Groups: /general/devices

Description of hardware devices used during experiment, e.g., monitors, ADC boards, microscopes, etc.

- **Quantity:** 0 or 1
- **Name:** devices

Table 4.34: Groups contained in <devices>

Id	Type	Description
devices	Group	Top level Group for devices • Quantity: 0 or 1 • Name: devices
—< <i>Device</i> >	Group	Data acquisition devices. • Extends: <i>Device</i> • Quantity: 0 or more

4.5.2.21 Groups: /general/devices/<Device>

Data acquisition devices.

- **Extends:** *Device*
- **Quantity:** 0 or more

4.5.2.22 Groups: /general/subject

Information about the animal or person from which the data was measured.

- **Extends:** *Subject*
- **Quantity:** 0 or 1
- **Name:** subject

4.5.2.23 Groups: /general/extracellular_ephys

Metadata related to extracellular electrophysiology.

- **Quantity:** 0 or 1
- **Name:** extracellular_ephys

Table 4.35: Groups contained in <extracellular_ephys>

Id	Type	Description
extracellular_ephys	Group	Top level Group for extracellular_ephys • Quantity: 0 or 1 • Name: extracellular_ephys
—< <i>ElectrodeGroup</i> >	Group	Physical group of electrodes. • Extends: <i>ElectrodeGroup</i> • Quantity: 0 or more

continues on next page

Table 4.35 – continued from previous page

Id	Type	Description
—electrodes	Group	A table of all electrodes (i.e. channels) used for recording. • Extends: DynamicTable • Quantity: 0 or 1 • Name: electrodes

4.5.2.24 Groups: /general/extracellular_ephys/<ElectrodeGroup>

Physical group of electrodes.

- **Extends:** [ElectrodeGroup](#)
- **Quantity:** 0 or more

4.5.2.25 Groups: /general/extracellular_ephys/electrodes

A table of all electrodes (i.e. channels) used for recording.

- **Extends:** [DynamicTable](#)
- **Quantity:** 0 or 1
- **Name:** electrodes

Table 4.36: Datasets, Links, and Attributes contained in /general/extracellular_ephys/electrodes

Id	Type	Description
electrodes	Group	Top level Group for electrodes • Extends: DynamicTable • Quantity: 0 or 1 • Name: electrodes
—x	Dataset	x coordinate of the channel location in the brain (+x is posterior). • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: x
—y	Dataset	y coordinate of the channel location in the brain (+y is inferior). • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: y
—z	Dataset	z coordinate of the channel location in the brain (+z is right). • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: z

continues on next page

Table 4.36 – continued from previous page

Id	Type	Description
—imp	Dataset	Impedance of the channel, in ohms. • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: imp
—location	Dataset	Location of the electrode (channel). Specify the area, layer, comments on estimation of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names for anatomical regions when possible. • Extends: VectorData • Data Type: text • Name: location
—filtering	Dataset	Description of hardware filtering, including the filter name and frequency cutoffs. • Extends: VectorData • Quantity: 0 or 1 • Data Type: text • Name: filtering
—group	Dataset	Reference to the ElectrodeGroup this electrode is a part of. • Extends: VectorData • Data Type: object reference to ElectrodeGroup • Name: group
—group_name	Dataset	Name of the ElectrodeGroup this electrode is a part of. • Extends: VectorData • Data Type: text • Name: group_name
—rel_x	Dataset	x coordinate in electrode group • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: rel_x
—rel_y	Dataset	y coordinate in electrode group • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: rel_y
—rel_z	Dataset	z coordinate in electrode group • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Name: rel_z

continues on next page

Table 4.36 – continued from previous page

Id	Type	Description
—reference	Dataset	Description of the reference electrode and/or reference scheme used for this electrode, e.g., “stainless steel skull screw” or “online common average referencing”. • Extends: VectorData • Quantity: 0 or 1 • Data Type: text • Name: reference

4.5.2.26 Groups: /general/intracellular_ephys

Metadata related to intracellular electrophysiology.

- **Quantity:** 0 or 1
- **Name:** intracellular_ephys

Table 4.37: Datasets, Links, and Attributes contained in /general/intracellular_ephys

Id	Type	Description
intracellular_ephys	Group	Top level Group for intracellular_ephys • Quantity: 0 or 1 • Name: intracellular_ephys
—filtering	Dataset	[DEPRECATED] Use IntracellularElectrode.filtering instead. Description of filtering used. Includes filtering type and parameters, frequency fall-off, etc. If this changes between TimeSeries, filter description should be stored as a text attribute for each TimeSeries. • Quantity: 0 or 1 • Data Type: text • Name: filtering

Table 4.38: Groups contained in <intracellular_ephys>

Id	Type	Description
intracellular_ephys	Group	Top level Group for intracellular_ephys • Quantity: 0 or 1 • Name: intracellular_ephys
—< <i>IntracellularElectrode</i> >	Group	An intracellular electrode. • Extends: <i>IntracellularElectrode</i> • Quantity: 0 or more

continues on next page

Table 4.38 – continued from previous page

Id	Type	Description
—sweep_table	Group	[DEPRECATED] Table used to group different PatchClampSeries. SweepTable is being replaced by IntracellularRecordingsTable and SimultaneousRecordingsTable tables. Additional SequentialRecordingsTable, RepetitionsTable and ExperimentalConditions tables provide enhanced support for experiment metadata. • Extends: <i>SweepTable</i> • Quantity: 0 or 1 • Name: sweep_table
—intracellular_recordings	Group	A table to group together a stimulus and response from a single electrode and a single simultaneous recording. Each row in the table represents a single recording consisting typically of a stimulus and a corresponding response. In some cases, however, only a stimulus or a response are recorded as part of an experiment. In this case both, the stimulus and response will point to the same TimeSeries while the idx_start and count of the invalid column will be set to -1, thus, indicating that no values have been recorded for the stimulus or response, respectively. Note, a recording MUST contain at least a stimulus or a response. Typically the stimulus and response are PatchClampSeries. However, the use of AD/DA channels that are not associated to an electrode is also common in intracellular electrophysiology, in which case other TimeSeries may be used. • Extends: <i>IntracellularRecordingsTable</i> • Quantity: 0 or 1 • Name: intracellular_recordings
—simultaneous_recordings	Group	A table for grouping different intracellular recordings from the IntracellularRecordingsTable table together that were recorded simultaneously from different electrodes • Extends: <i>SimultaneousRecordingsTable</i> • Quantity: 0 or 1 • Name: simultaneous_recordings
—sequential_recordings	Group	A table for grouping different sequential recordings from the SimultaneousRecordingsTable table together. This is typically used to group together sequential recordings where the a sequence of stimuli of the same type with varying parameters have been presented in a sequence. • Extends: <i>SequentialRecordingsTable</i> • Quantity: 0 or 1 • Name: sequential_recordings
—repetitions	Group	A table for grouping different sequential intracellular recordings together. With each SequentialRecording typically representing a particular type of stimulus, the RepetitionsTable table is typically used to group sets of stimuli applied in sequence. • Extends: <i>RepetitionsTable</i> • Quantity: 0 or 1 • Name: repetitions

continues on next page

Table 4.38 – continued from previous page

Id	Type	Description
—experimental_conditions	Group	A table for grouping different intracellular recording repetitions together that belong to the same experimental experimental_conditions. • Extends: <i>ExperimentalConditionsTable</i> • Quantity: 0 or 1 • Name: experimental_conditions

4.5.2.27 Groups: /general/intracellular_ephys/<IntracellularElectrode>

An intracellular electrode.

- **Extends:** *IntracellularElectrode*
- **Quantity:** 0 or more

4.5.2.28 Groups: /general/intracellular_ephys/sweep_table

[DEPRECATED] Table used to group different PatchClampSeries. SweepTable is being replaced by IntracellularRecordingsTable and SimultaneousRecordingsTable labels. Additional SequentialRecordingsTable, RepetitionsTable and ExperimentalConditions tables provide enhanced support for experiment metadata.

- **Extends:** *SweepTable*
- **Quantity:** 0 or 1
- **Name:** sweep_table

4.5.2.29 Groups: /general/intracellular_ephys/intracellular_recordings

A table to group together a stimulus and response from a single electrode and a single simultaneous recording. Each row in the table represents a single recording consisting typically of a stimulus and a corresponding response. In some cases, however, only a stimulus or a response are recorded as as part of an experiment. In this case both, the stimulus and response will point to the same TimeSeries while the idx_start and count of the invalid column will be set to -1, thus, indicating that no values have been recorded for the stimulus or response, respectively. Note, a recording **MUST** contain at least a stimulus or a response. Typically the stimulus and response are PatchClampSeries. However, the use of AD/DA channels that are not associated to an electrode is also common in intracellular electrophysiology, in which case other TimeSeries may be used.

- **Extends:** *IntracellularRecordingsTable*
- **Quantity:** 0 or 1
- **Name:** intracellular_recordings

4.5.2.30 Groups: /general/intracellular_ephys/simultaneous_recordings

A table for grouping different intracellular recordings from the IntracellularRecordingsTable table together that were recorded simultaneously from different electrodes

- **Extends:** *SimultaneousRecordingsTable*
- **Quantity:** 0 or 1
- **Name:** simultaneous_recordings

4.5.2.31 Groups: /general/intracellular_ephys/sequential_recordings

A table for grouping different sequential recordings from the SimultaneousRecordingsTable table together. This is typically used to group together sequential recordings where the a sequence of stimuli of the same type with varying parameters have been presented in a sequence.

- **Extends:** *SequentialRecordingsTable*
- **Quantity:** 0 or 1
- **Name:** sequential_recordings

4.5.2.32 Groups: /general/intracellular_ephys/repetitions

A table for grouping different sequential intracellular recordings together. With each SequentialRecording typically representing a particular type of stimulus, the RepetitionsTable table is typically used to group sets of stimuli applied in sequence.

- **Extends:** *RepetitionsTable*
- **Quantity:** 0 or 1
- **Name:** repetitions

4.5.2.33 Groups: /general/intracellular_ephys/experimental_conditions

A table for grouping different intracellular recording repetitions together that belong to the same experimental experimental_conditions.

- **Extends:** *ExperimentalConditionsTable*
- **Quantity:** 0 or 1
- **Name:** experimental_conditions

4.5.2.34 Groups: /general/optogenetics

Metadata describing optogenetic stimulation.

- **Quantity:** 0 or 1
- **Name:** optogenetics

Table 4.39: Groups contained in <optogenetics>

Id	Type	Description
optogenetics	Group	Top level Group for optogenetics • Quantity: 0 or 1 • Name: optogenetics
—< <i>OptogeneticStimulusSite</i> >	Group	An optogenetic stimulation site. • Extends: <i>OptogeneticStimulusSite</i> • Quantity: 0 or more

4.5.2.35 Groups: /general/optogenetics/<OptogeneticStimulusSite>

An optogenetic stimulation site.

- **Extends:** *OptogeneticStimulusSite*
- **Quantity:** 0 or more

4.5.2.36 Groups: /general/optophysiology

Metadata related to optophysiology.

- **Quantity:** 0 or 1
- **Name:** optophysiology

Table 4.40: Groups contained in <optophysiology>

Id	Type	Description
optophysiology	Group	Top level Group for optophysiology • Quantity: 0 or 1 • Name: optophysiology
—< <i>ImagingPlane</i> >	Group	An imaging plane. • Extends: <i>ImagingPlane</i> • Quantity: 0 or more

4.5.2.37 Groups: /general/optophysiology/<ImagingPlane>

An imaging plane.

- **Extends:** *ImagingPlane*
- **Quantity:** 0 or more

4.5.2.38 Groups: /intervals

Experimental intervals, whether that be logically distinct sub-experiments having a particular scientific goal, trials (see trials subgroup) during an experiment, or epochs (see epochs subgroup) deriving from analysis of data.

- **Quantity:** 0 or 1
- **Name:** intervals

Table 4.41: Groups contained in <intervals>

Id	Type	Description
intervals	Group	Top level Group for intervals • Quantity: 0 or 1 • Name: intervals
—epochs	Group	Divisions in time marking experimental stages or sub-divisions of a single recording session. • Extends: <i>TimeIntervals</i> • Quantity: 0 or 1 • Name: epochs
—trials	Group	Repeated experimental events that have a logical grouping. • Extends: <i>TimeIntervals</i> • Quantity: 0 or 1 • Name: trials
—invalid_times	Group	Time intervals that should be removed from analysis. • Extends: <i>TimeIntervals</i> • Quantity: 0 or 1 • Name: invalid_times
—< <i>TimeIntervals</i> >	Group	Optional additional table(s) for describing other experimental time intervals. • Extends: <i>TimeIntervals</i> • Quantity: 0 or more

4.5.2.39 Groups: /intervals/epochs

Divisions in time marking experimental stages or sub-divisions of a single recording session.

- **Extends:** *TimeIntervals*
- **Quantity:** 0 or 1
- **Name:** epochs

4.5.2.40 Groups: /intervals/trials

Repeated experimental events that have a logical grouping.

- **Extends:** *TimeIntervals*
- **Quantity:** 0 or 1
- **Name:** trials

4.5.2.41 Groups: /intervals/invalid_times

Time intervals that should be removed from analysis.

- **Extends:** *TimeIntervals*
- **Quantity:** 0 or 1
- **Name:** invalid_times

4.5.2.42 Groups: /intervals/<TimeIntervals>

Optional additional table(s) for describing other experimental time intervals.

- **Extends:** *TimeIntervals*
- **Quantity:** 0 or more

4.5.2.43 Groups: /units

Data about sorted spike units.

- **Extends:** *Units*
- **Quantity:** 0 or 1
- **Name:** units

4.5.3 LabMetaData

Overview: Lab-specific meta-data.

LabMetaData extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, Container
- **Source filename:** nwb.file.yaml
- **Source Specification:** see [Section 5.6.3](#)

4.5.4 Subject

Overview: Information about the animal or person from which the data was measured.

Subject extends *NWBContainer* and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.file.yaml
- **Source Specification:** see [Section 5.6.4](#)

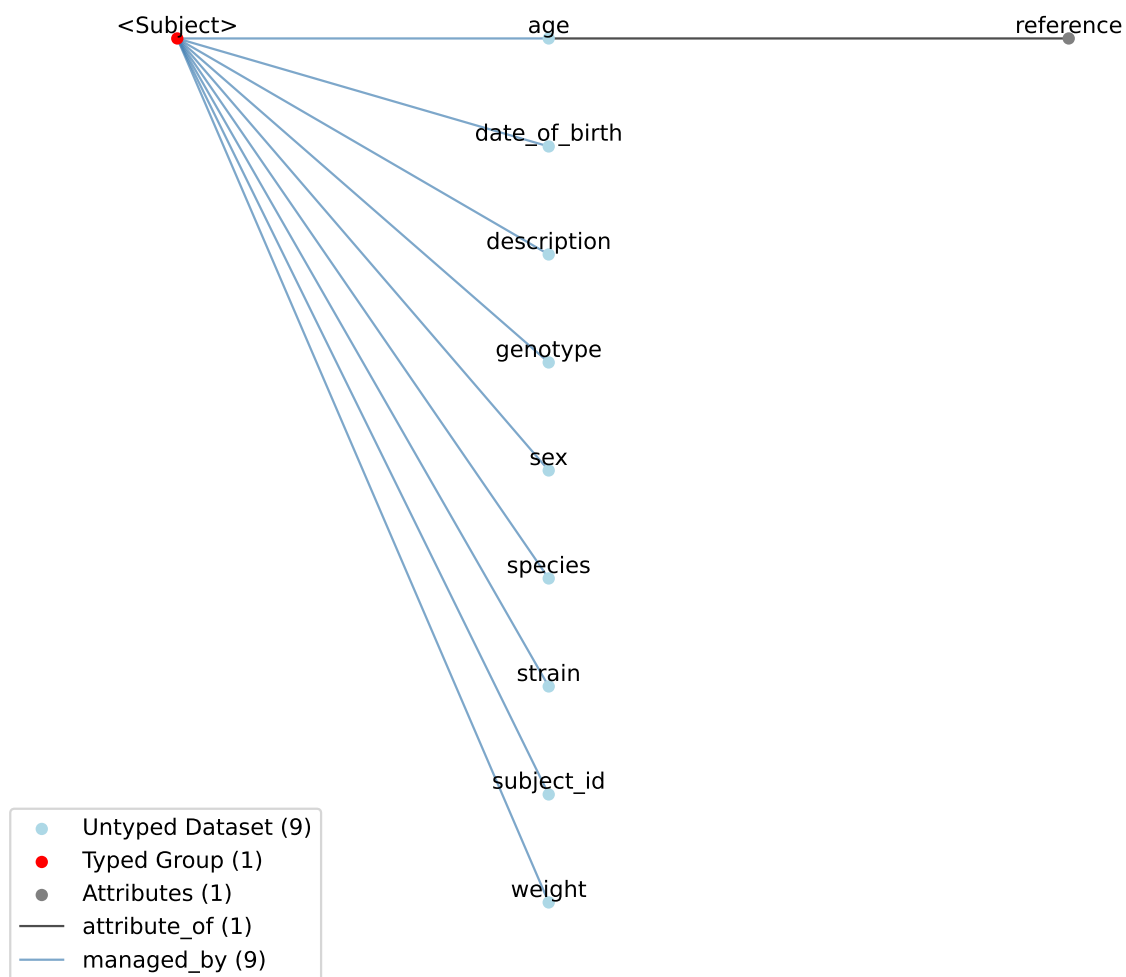


Table 4.42: Datasets, Links, and Attributes contained in <Subject>

Id	Type	Description
<Subject>	Group	Top level Group for <Subject> • Neurodata Type: <i>Subject</i> • Extends: <i>NWBContainer</i>
—age	Dataset	Age of subject. Can be supplied instead of ‘date_of_birth’. • Quantity: 0 or 1 • Data Type: text • Name: age
——reference	At-tribute	Age is with reference to this event. Can be ‘birth’ or ‘gestational’. If reference is omitted, ‘birth’ is implied. • Data Type: text • Required: False • Default Value: birth • Name: reference
—date_of_birth	Dataset	Date of birth of subject. Can be supplied instead of ‘age’. • Quantity: 0 or 1 • Data Type: isodatetime • Name: date_of_birth
—description	Dataset	Description of subject and where subject came from (e.g., breeder, if animal). • Quantity: 0 or 1 • Data Type: text • Name: description
—genotype	Dataset	Genetic strain. If absent, assume Wild Type (WT). • Quantity: 0 or 1 • Data Type: text • Name: genotype
—sex	Dataset	Gender of subject. • Quantity: 0 or 1 • Data Type: text • Name: sex
—species	Dataset	Species of subject. • Quantity: 0 or 1 • Data Type: text • Name: species
—strain	Dataset	Strain of subject. • Quantity: 0 or 1 • Data Type: text • Name: strain

continues on next page

Table 4.42 – continued from previous page

Id	Type	Description
—subject_id	Dataset	ID of animal/person used/participating in experiment (lab convention). • Quantity: 0 or 1 • Data Type: text • Name: subject_id
—weight	Dataset	Weight at time of experiment, at time of surgery and at other important times. • Quantity: 0 or 1 • Data Type: text • Name: weight

4.6 Miscellaneous neurodata_types.

Miscellaneous types.

4.6.1 AbstractFeatureSeries

Overview: Abstract features, such as quantitative descriptions of sensory stimuli. The `TimeSeries::data` field is a 2D array, storing those features (e.g., for visual grating stimulus this might be orientation, spatial frequency and contrast). Null stimuli (eg, uniform gray) can be marked as being an independent feature (eg, 1.0 for gray, 0.0 for actual stimulus) or by storing NaNs for feature values, or through use of the `TimeSeries::control` fields. A set of features is considered to persist until the next set of features is defined. The final set of features stored should be the null set. This is useful when storing the raw stimulus is impractical.

`AbstractFeatureSeries` extends `TimeSeries` and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** `nwb.misc.yaml`
- **Source Specification:** see [Section 5.7.1](#)

Table 4.43: Datasets, Links, and Attributes contained in `<AbstractFeatureSeries>`

Id	Type	Description
<code><AbstractFeatureSeries></code>	Group	Top level Group for <code><AbstractFeatureSeries></code> • Neurodata Type: <i>AbstractFeatureSeries</i> • Extends: <i>TimeSeries</i>
<code>—description</code>	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
<code>—comments</code>	Attribute	Human-readable comments about the <code>TimeSeries</code> . This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments

continues on next page

Table 4.43 – continued from previous page

Id	Type	Description
—data	Dataset	Values of each feature at each time. • Data Type: numeric • Dimensions: [['num_times'], ['num_times', 'num_features']] • Shape: [[None], [None, None]] • Name: data
—unit	At-tribute	Since there can be different units for different features, store the units in 'feature_units'. The default value for this attribute is "see 'feature_units'". • Data Type: text • Required: False • Default Value: see 'feature_units' • Name: unit
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
—offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
—resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution

continues on next page

Table 4.43 – continued from previous page

Id	Type	Description
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—feature_units	Dataset	Units of each feature. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_features'] • Shape: [None] • Name: feature_units
—features	Dataset	Description of the features represented in TimeSeries::data. • Data Type: text • Dimensions: ['num_features'] • Shape: [None] • Name: features
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.43 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description

Table 4.44: Groups contained in <AbstractFeatureSeries>

Id	Type	Description
<AbstractFeatureSeries>	Group	Top level Group for <AbstractFeatureSeries> • Neurodata Type: <i>AbstractFeatureSeries</i> • Extends: <i>TimeSeries</i>

continues on next page

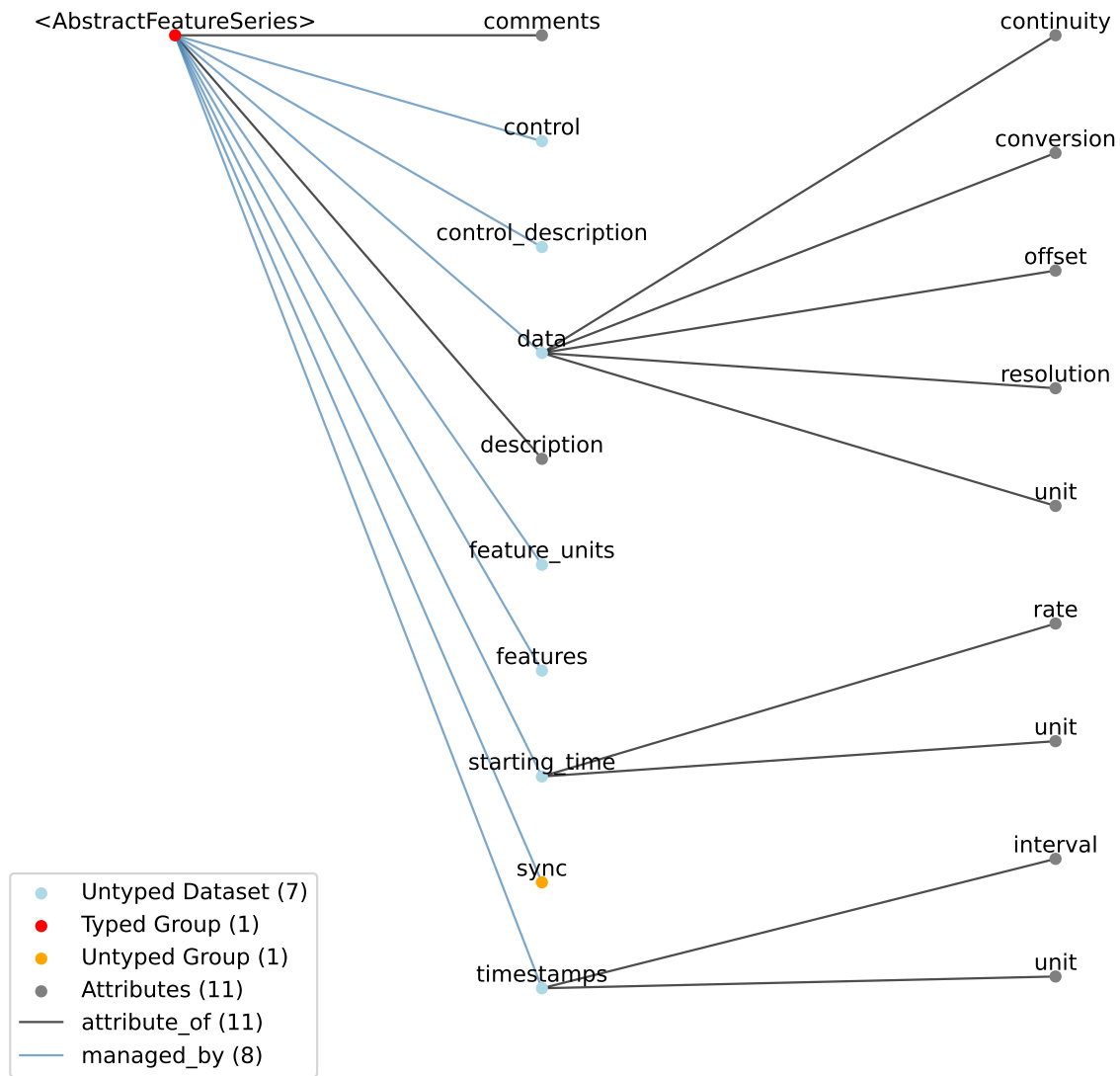
Table 4.44 – continued from previous page

Id	Type	Description
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.6.1.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync



4.6.2 AnnotationSeries

Overview: Stores user annotations made during an experiment. The data[] field stores a text array, and timestamps are stored for each annotation (ie, interval=1). This is largely an alias to a standard TimeSeries storing a text array but that is identifiable as storing annotations in a machine-readable way.

AnnotationSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.misc.yaml
- **Source Specification:** see [Section 5.7.2](#)

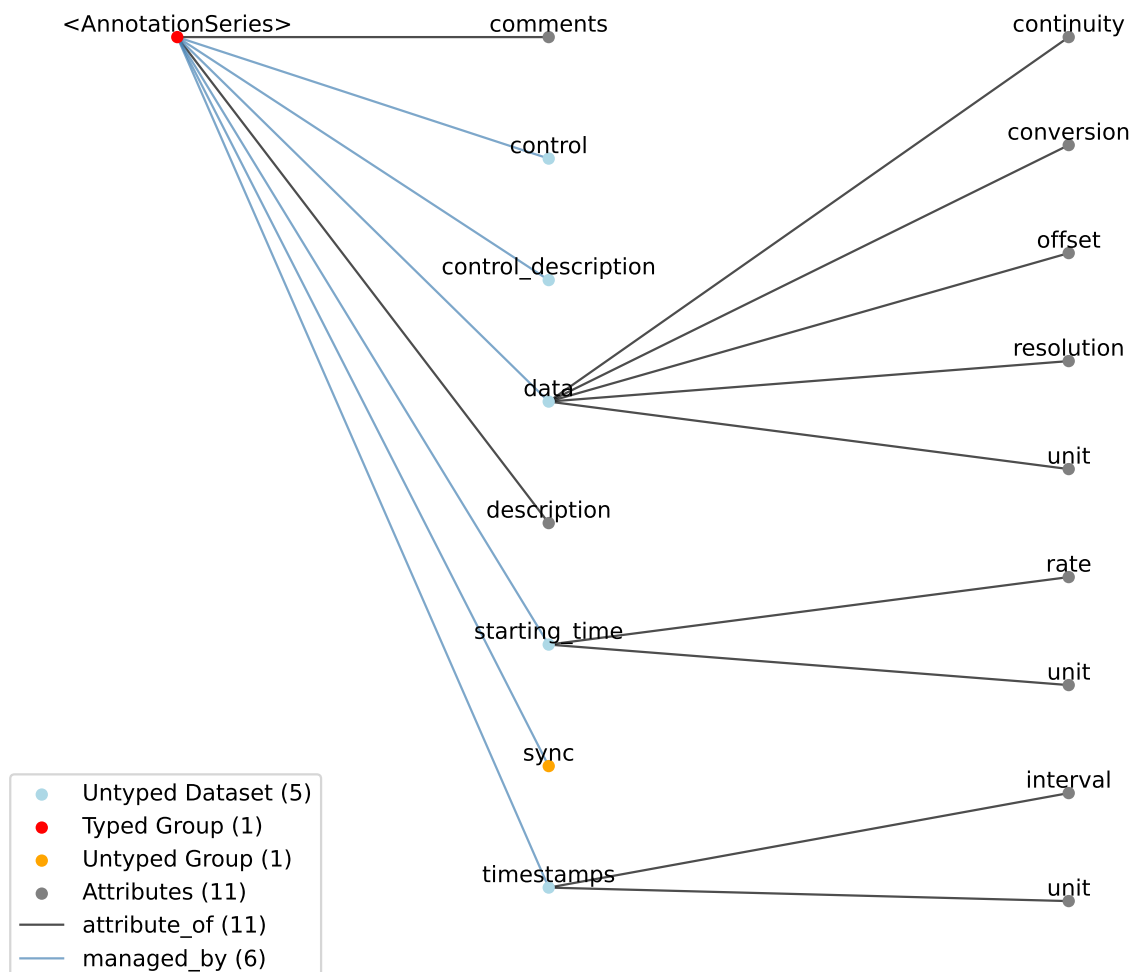


Table 4.45: Datasets, Links, and Attributes contained in <AnnotationSeries>

Id	Type	Description
<AnnotationSeries>	Group	Top level Group for <AnnotationSeries> • Neurodata Type: <i>AnnotationSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Annotations made during an experiment. • Data Type: text • Dimensions: ['num_times'] • Shape: [None] • Name: data
—resolution	At-tribute	Smallest meaningful difference between values in data. Annotations have no units, so the value is fixed to -1.0. • Data Type: float32 • Value: -1.0 • Name: resolution
—unit	At-tribute	Base unit of measurement for working with the data. Annotations have no units, so the value is fixed to 'n/a'. • Data Type: text • Value: n/a • Name: unit

continues on next page

Table 4.45 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate

continues on next page

Table 4.45 – continued from previous page

Id	Type	Description
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description

Table 4.46: Groups contained in <AnnotationSeries>

Id	Type	Description
<AnnotationSeries>	Group	Top level Group for <AnnotationSeries> • Neurodata Type: <i>AnnotationSeries</i> • Extends: <i>TimeSeries</i>
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.6.2.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.6.3 IntervalSeries

Overview: Stores intervals of data. The timestamps field stores the beginning and end of intervals. The data field stores whether the interval just started (>0 value) or ended (<0 value). Different interval types can be represented in the same series by using multiple key values (eg, 1 for feature A, 2 for feature B, 3 for feature C, etc). The field data stores an 8-bit integer. This is largely an alias of a standard TimeSeries but that is identifiable as representing time intervals in a machine-readable way.

IntervalSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.misc.yaml
- **Source Specification:** see [Section 5.7.3](#)

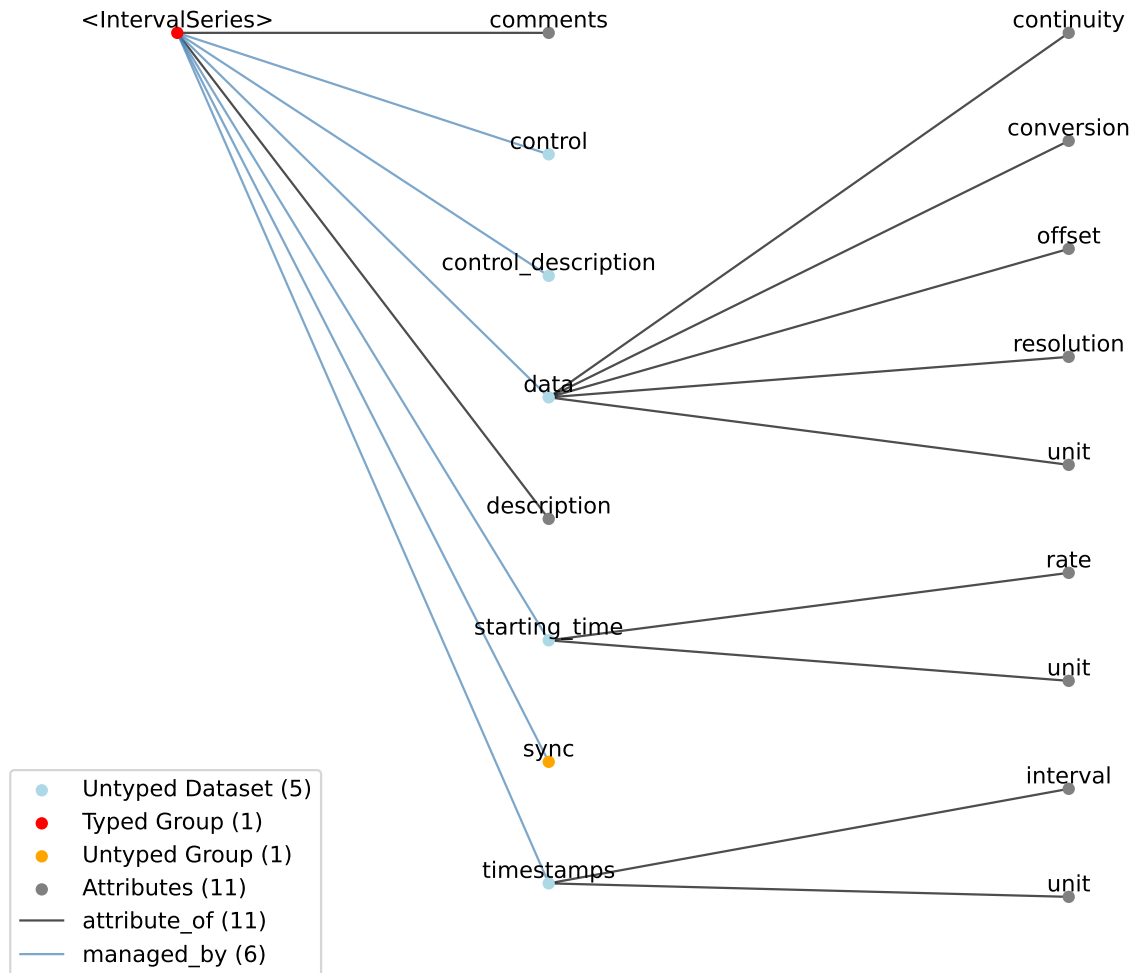


Table 4.47: Datasets, Links, and Attributes contained in <IntervalSeries>

Id	Type	Description
<IntervalSeries>	Group	Top level Group for <IntervalSeries> • Neurodata Type: <i>IntervalSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Use values >0 if interval started, <0 if interval ended. • Data Type: int8 • Dimensions: ['num_times'] • Shape: [None] • Name: data
—resolution	At-tribute	Smallest meaningful difference between values in data. Annotations have no units, so the value is fixed to -1.0. • Data Type: float32 • Value: -1.0 • Name: resolution
—unit	At-tribute	Base unit of measurement for working with the data. Annotations have no units, so the value is fixed to 'n/a'. • Data Type: text • Value: n/a • Name: unit

continues on next page

Table 4.47 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate

continues on next page

Table 4.47 – continued from previous page

Id	Type	Description
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description

Table 4.48: Groups contained in <IntervalSeries>

Id	Type	Description
<IntervalSeries>	Group	Top level Group for <IntervalSeries> • Neurodata Type: <i>IntervalSeries</i> • Extends: <i>TimeSeries</i>
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.6.3.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.6.4 DecompositionSeries

Overview: Spectral analysis of a time series, e.g. of an LFP or a speech signal.

DecompositionSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.misc.yaml
- **Source Specification:** see [Section 5.7.4](#)

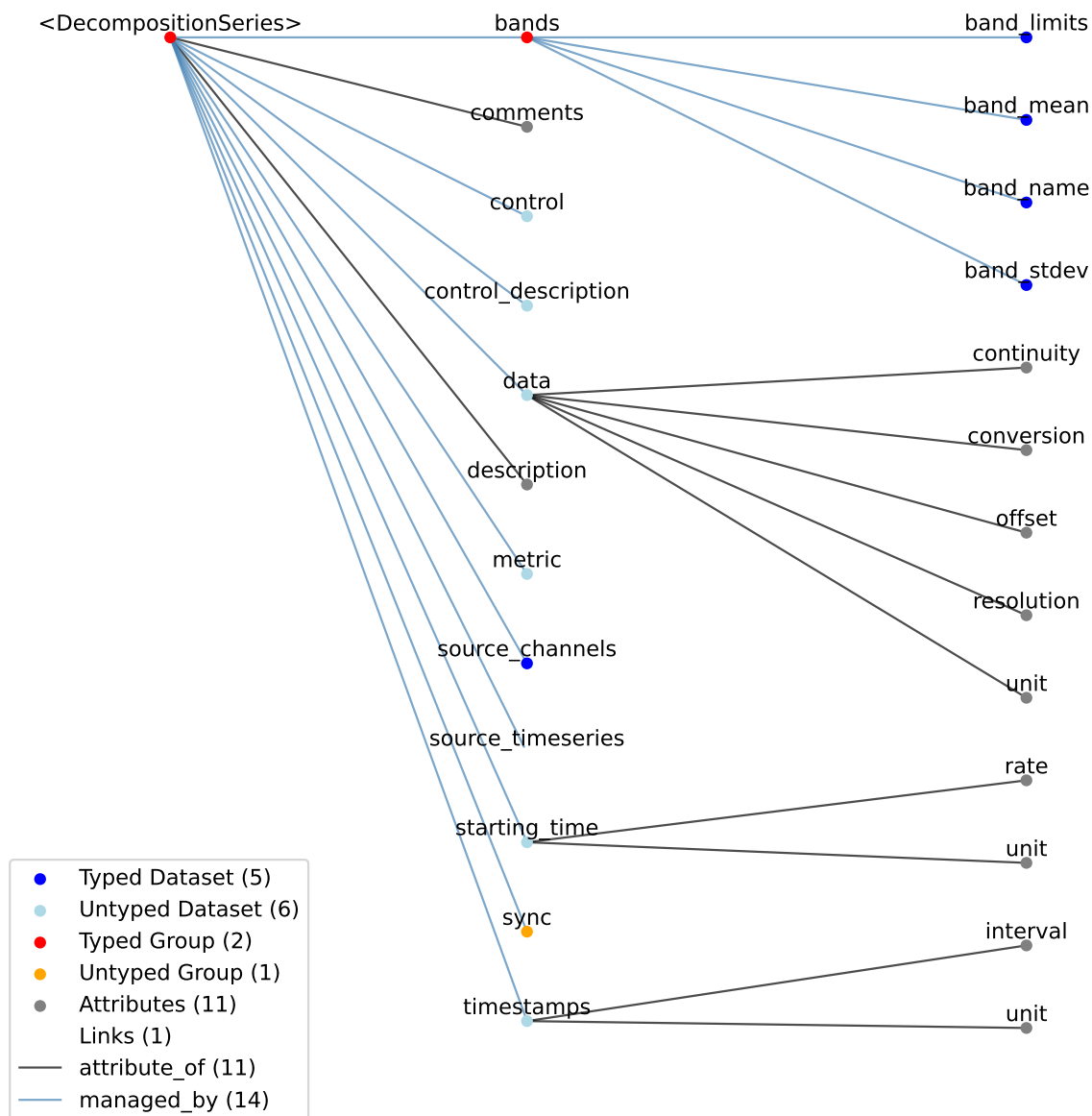


Table 4.49: Datasets, Links, and Attributes contained in <DecompositionSeries>

Id	Type	Description
<DecompositionSeries>	Group	Top level Group for <DecompositionSeries> • Neurodata Type: <i>DecompositionSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Data decomposed into frequency bands. • Data Type: numeric • Dimensions: ['num_times', 'num_channels', 'num_bands'] • Shape: [None, None, None] • Name: data
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply 'data' by 'conversion'. • Data Type: text • Required: False • Default Value: no unit • Name: unit
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion

continues on next page

Table 4.49 – continued from previous page

Id	Type	Description
—offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
—resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—metric	Dataset	The metric used, e.g. phase, amplitude, power. • Data Type: text • Name: metric
—source_channels	Dataset	DynamicTableRegion pointer to the channels that this decomposition series was generated from. • Extends: DynamicTableRegion • Quantity: 0 or 1 • Name: source_channels
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time

continues on next page

Table 4.49 – continued from previous page

Id	Type	Description
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description

continues on next page

Table 4.49 – continued from previous page

Id	Type	Description
—source_timeseries	Link	Link to TimeSeries object that this data was calculated from. Metadata about electrodes and their position can be read from that ElectricalSeries so it is not necessary to store that information here. • Target Type <i>TimeSeries</i> • Name: source_timeseries

Table 4.50: Groups contained in <DecompositionSeries>

Id	Type	Description
<DecompositionSeries>	Group	Top level Group for <DecompositionSeries> • Neurodata Type: <i>DecompositionSeries</i> • Extends: <i>TimeSeries</i>
—source_timeseries	Link	Link to TimeSeries object that this data was calculated from. Metadata about electrodes and their position can be read from that ElectricalSeries so it is not necessary to store that information here. • Target Type <i>TimeSeries</i> • Name: source_timeseries
—bands	Group	Table for describing the bands that this series was generated from. There should be one row in this table for each band. • Extends: <i>DynamicTable</i> • Name: bands
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.6.4.1 Groups: bands

Table for describing the bands that this series was generated from. There should be one row in this table for each band.

- **Extends:** *DynamicTable*
- **Name:** bands

Table 4.51: Datasets, Links, and Attributes contained in bands

Id	Type	Description
bands	Group	Top level Group for bands • Extends: DynamicTable • Name: bands
—band_name	Dataset	Name of the band, e.g. theta. • Extends: VectorData • Data Type: text • Name: band_name
—band_limits	Dataset	Low and high limit of each band in Hz. If it is a Gaussian filter, use 2 SD on either side of the center. • Extends: VectorData • Data Type: float32 • Dimensions: ['num_bands', 'low, high'] • Shape: [None, 2] • Name: band_limits
—band_mean	Dataset	The mean Gaussian filters, in Hz. • Extends: VectorData • Data Type: float32 • Dimensions: ['num_bands'] • Shape: [None] • Name: band_mean
—band_stdev	Dataset	The standard deviation of Gaussian filters, in Hz. • Extends: VectorData • Data Type: float32 • Dimensions: ['num_bands'] • Shape: [None] • Name: band_stdev

4.6.4.2 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.6.5 Units

Overview: Data about spiking units. Event times of observed units (e.g. cell, synapse, etc.) should be concatenated and stored in `spike_times`.

`Units` extends `DynamicTable` and includes all elements of `DynamicTable` with the following additions or changes.

- **Extends:** `DynamicTable`
- **Primitive Type:** Group
- **Default Name:** Units
- **Inherits from:** `DynamicTable`, `Container`
- **Source filename:** `nwb.misc.yaml`
- **Source Specification:** see [Section 5.7.5](#)

Table 4.52: Datasets, Links, and Attributes contained in `<Units>`

Id	Type	Description
<code><Units></code>	Group	Top level Group for <code><Units></code> • Neurodata Type: <code>Units</code> • Extends: <code>DynamicTable</code> • Default Name: Units
<code>—colnames</code>	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: [<code>num_columns</code>] • Shape: [None] • Name: <code>colnames</code>
<code>—description</code>	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: <code>description</code>
<code>—spike_times_index</code>	Dataset	Index into the <code>spike_times</code> dataset. • Extends: <code>VectorIndex</code> • Quantity: 0 or 1 • Name: <code>spike_times_index</code>
<code>—spike_times</code>	Dataset	Spike times for each unit in seconds. • Extends: <code>VectorData</code> • Quantity: 0 or 1 • Data Type: float64 • Name: <code>spike_times</code>

continues on next page

Table 4.52 – continued from previous page

Id	Type	Description
—resolution	Attribute	The smallest possible difference between two spike times. Usually 1 divided by the acquisition sampling rate from which spike times were extracted, but could be larger if the acquisition time series was downsampled or smaller if the acquisition time series was smoothed/interpolated and it is possible for the spike time to be between samples. • Data Type: float64 • Required: False • Name: resolution
—obs_intervals_index	Dataset	Index into the obs_intervals dataset. • Extends: VectorIndex • Quantity: 0 or 1 • Name: obs_intervals_index
—obs_intervals	Dataset	Observation intervals for each unit. • Extends: VectorData • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_intervals', 'start end'] • Shape: [None, 2] • Name: obs_intervals
—electrodes_index	Dataset	Index into electrodes. • Extends: VectorIndex • Quantity: 0 or 1 • Name: electrodes_index
—electrodes	Dataset	Electrode that each spike unit came from, specified using a DynamicTableRegion. • Extends: DynamicTableRegion • Quantity: 0 or 1 • Name: electrodes
—electrode_group	Dataset	Electrode group that each spike unit came from. • Extends: VectorData • Quantity: 0 or 1 • Data Type: object reference to ElectrodeGroup • Name: electrode_group
—waveform_mean	Dataset	Spike waveform mean for each spike unit. • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['num_units', 'num_samples'], ['num_units', 'num_samples', 'num_electrodes']] • Shape: [[None, None], [None, None, None]] • Name: waveform_mean

continues on next page

Table 4.52 – continued from previous page

Id	Type	Description
——sampling_rate	At-tribute	Sampling rate, in hertz. • Data Type: float32 • Required: False • Name: sampling_rate
——unit	At-tribute	Unit of measurement. This value is fixed to ‘volts’. • Data Type: text • Value: volts • Name: unit
—waveform_sd	Dataset	Spike waveform standard deviation for each spike unit. • Extends: VectorData • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['num_units', 'num_samples'], ['num_units', 'num_samples', 'num_electrodes']] • Shape: [[None, None], [None, None, None]] • Name: waveform_sd
——sampling_rate	At-tribute	Sampling rate, in hertz. • Data Type: float32 • Required: False • Name: sampling_rate
——unit	At-tribute	Unit of measurement. This value is fixed to ‘volts’. • Data Type: text • Value: volts • Name: unit

continues on next page

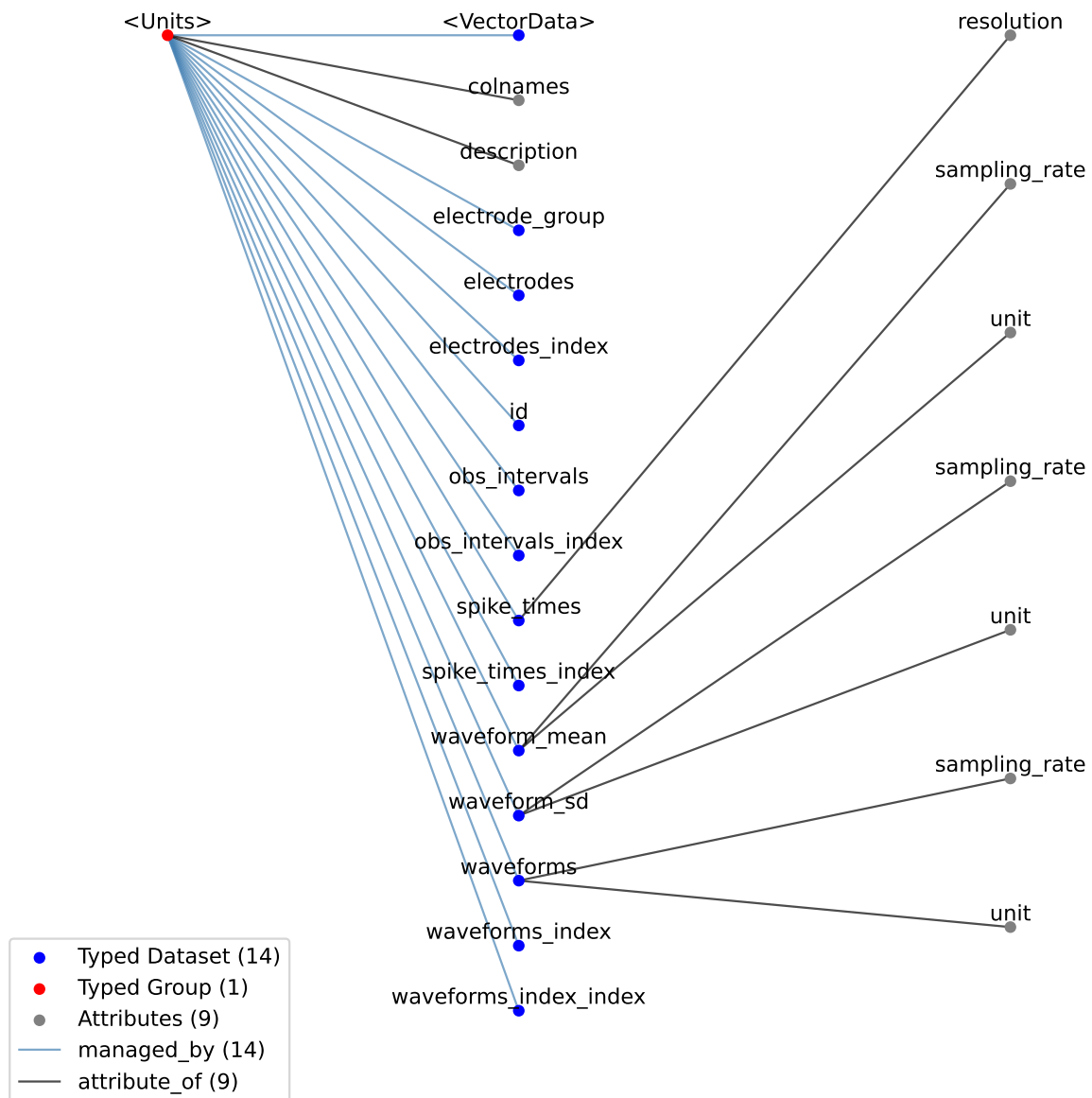
Table 4.52 – continued from previous page

Id	Type	Description
—waveforms	Dataset	Individual waveforms for each spike on each electrode. This is a doubly indexed column. The ‘waveforms_index’ column indexes which waveforms in this column belong to the same spike event for a given unit, where each waveform was recorded from a different electrode. The ‘waveforms_index_index’ column indexes the ‘waveforms_index’ column to indicate which spike events belong to a given unit. For example, if the ‘waveforms_index_index’ column has values [2, 5, 6], then the first 2 elements of the ‘waveforms_index’ column correspond to the 2 spike events of the first unit, the next 3 elements of the ‘waveforms_index’ column correspond to the 3 spike events of the second unit, and the next 1 element of the ‘waveforms_index’ column corresponds to the 1 spike event of the third unit. If the ‘waveforms_index’ column has values [3, 6, 8, 10, 12, 13], then the first 3 elements of the ‘waveforms’ column contain the 3 spike waveforms that were recorded from 3 different electrodes for the first spike time of the first unit. See https://nwb-schema.readthedocs.io/en/stable/format_description.html#doubly-ragged-arrays for a graphical representation of this example. When there is only one electrode for each unit (i.e., each spike time is associated with a single waveform), then the ‘waveforms_index’ column will have values 1, 2, ..., N, where N is the number of spike events. The number of electrodes for each spike event should be the same within a given unit. The ‘electrodes’ column should be used to indicate which electrodes are associated with each unit, and the order of the waveforms within a given unit x spike event should be in the same order as the electrodes referenced in the ‘electrodes’ column of this table. The number of samples for each waveform must be the same. • Extends: VectorData • Quantity: 0 or 1 • Data Type: numeric • Dimensions: ['num_waveforms', 'num_samples'] • Shape: [None, None] • Name: waveforms
——sampling_rate	Attribute	Sampling rate, in hertz. • Data Type: float32 • Required: False • Name: sampling_rate
——unit	Attribute	Unit of measurement. This value is fixed to ‘volts’. • Data Type: text • Value: volts • Name: unit
—waveforms_index	Dataset	Index into the waveforms dataset. One value for every spike event. See ‘waveforms’ for more detail. • Extends: VectorIndex • Quantity: 0 or 1 • Name: waveforms_index

continues on next page

Table 4.52 – continued from previous page

Id	Type	Description
—waveforms_index_index	Dataset	Index into the waveforms_index dataset. One value for every unit (row in the table). See ‘waveforms’ for more detail. • Extends: VectorIndex • Quantity: 0 or 1 • Name: waveforms_index_index
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: [‘num_rows’] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more



4.7 Behavior

This source module contains `neurodata_types` for behavior data.

4.7.1 SpatialSeries

Overview: Direction, e.g., of gaze or travel, or position. The `TimeSeries::data` field is a 2D array storing position or direction relative to some reference frame. Array structure: [num measurements] [num dimensions]. Each `SpatialSeries` has a text dataset `reference_frame` that indicates the zero-position, or the zero-axes for direction. For example, if representing gaze direction, ‘straight-ahead’ might be a specific pixel on the monitor, or some other point in space. For position data, the 0,0 point might be the top-left corner of an enclosure, as viewed from the tracking camera. The unit of data will indicate how to interpret `SpatialSeries` values.

`SpatialSeries` extends `TimeSeries` and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** `nwb.behavior.yaml`
- **Source Specification:** see [Section 5.8.1](#)

Table 4.53: Datasets, Links, and Attributes contained in <SpatialSeries>

Id	Type	Description
<SpatialSeries>	Group	Top level Group for <SpatialSeries> • Neurodata Type: <i>SpatialSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments

continues on next page

Table 4.53 – continued from previous page

Id	Type	Description
—data	Dataset	1-D or 2-D array storing position or direction relative to some reference frame. • Data Type: numeric • Dimensions: [['num_times'], ['num_times', 'x'], ['num_times', 'x,y'], ['num_times', 'x,y,z']] • Shape: [[None], [None, 1], [None, 2], [None, 3]] • Name: data
—unit	At-tribute	Base unit of measurement for working with the data. The default value is 'meters'. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply 'data' by 'conversion' and add 'offset'. • Data Type: text • Required: False • Default Value: meters • Name: unit
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
—offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
—resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution

continues on next page

Table 4.53 – continued from previous page

Id	Type	Description
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—reference_frame	Dataset	Description defining what exactly ‘straight-ahead’ means. • Quantity: 0 or 1 • Data Type: text • Name: reference_frame
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
—interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval

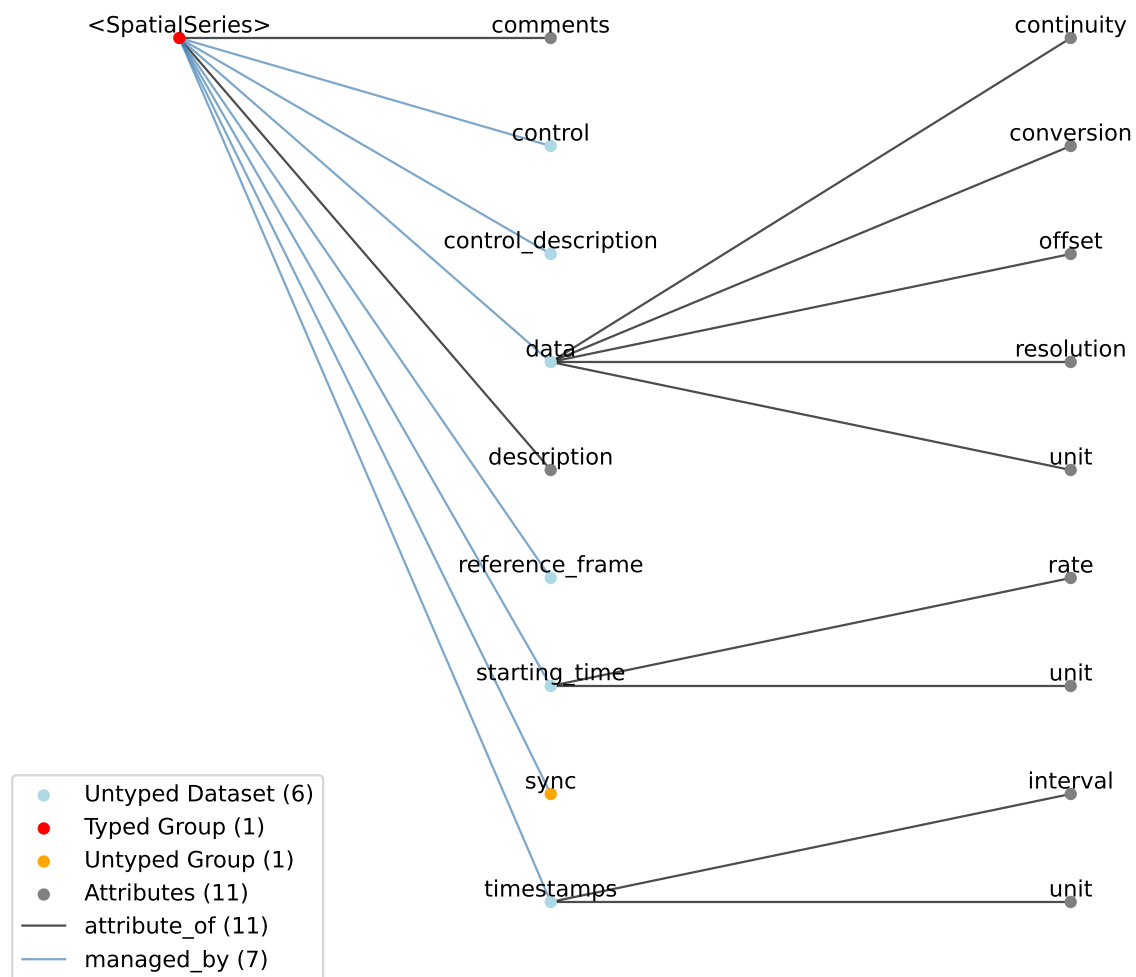
continues on next page

Table 4.53 – continued from previous page

Id	Type	Description
—unit	Attribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description

Table 4.54: Groups contained in <SpatialSeries>

Id	Type	Description
<SpatialSeries>	Group	Top level Group for <SpatialSeries> • Neurodata Type: <i>SpatialSeries</i> • Extends: <i>TimeSeries</i>
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync



4.7.1.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.7.2 BehavioralEpochs

Overview: TimeSeries for storing behavioral epochs. The objective of this and the other two Behavioral interfaces (e.g. BehavioralEvents and BehavioralTimeSeries) is to provide generic hooks for software tools/scripts. This allows a tool/script to take the output one specific interface (e.g., UnitTimes) and plot that data relative to another data modality (e.g., behavioral events) without having to define all possible modalities in advance. Declaring one of these interfaces means that one or more TimeSeries of the specified type is published. These TimeSeries should reside in a group having the same name as the interface. For example, if a BehavioralTimeSeries interface is declared, the module will have one or more TimeSeries defined in the module sub-group 'BehavioralTimeSeries'. BehavioralEpochs should use IntervalSeries. BehavioralEvents is used for irregular events. BehavioralTimeSeries is for continuous data.

BehavioralEpochs extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** BehavioralEpochs
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.behavior.yaml
- **Source Specification:** see [Section 5.8.2](#)

Table 4.55: Groups contained in <BehavioralEpochs>

Id	Type	Description
<BehavioralEpochs>	Group	Top level Group for <BehavioralEpochs> • Neurodata Type: <i>BehavioralEpochs</i> • Extends: <i>NWBDataInterface</i> • Default Name: BehavioralEpochs
—< <i>IntervalSeries</i> >	Group	IntervalSeries object containing start and stop times of epochs. • Extends: <i>IntervalSeries</i> • Quantity: 0 or more

4.7.2.1 Groups: <IntervalSeries>

IntervalSeries object containing start and stop times of epochs.

- **Extends:** *IntervalSeries*
- **Quantity:** 0 or more

4.7.3 BehavioralEvents

Overview: TimeSeries for storing behavioral events. See description of [BehavioralEpochs](#BehavioralEpochs) for more details.

BehavioralEvents extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** BehavioralEvents
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, Container
- **Source filename:** nwb.behavior.yaml
- **Source Specification:** see [Section 5.8.3](#)

Table 4.56: Groups contained in <BehavioralEvents>

Id	Type	Description
<BehavioralEvents>	Group	Top level Group for <BehavioralEvents> • Neurodata Type: <i>BehavioralEvents</i> • Extends: <i>NWBDataInterface</i> • Default Name: BehavioralEvents
—<TimeSeries>	Group	TimeSeries object containing behavioral events. • Extends: <i>TimeSeries</i> • Quantity: 0 or more

4.7.3.1 Groups: <TimeSeries>

TimeSeries object containing behavioral events.

- **Extends:** *TimeSeries*
- **Quantity:** 0 or more

4.7.4 BehavioralTimeSeries

Overview: TimeSeries for storing Behavioral time series data. See description of [BehavioralEpochs](#BehavioralEpochs) for more details.

BehavioralTimeSeries extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** BehavioralTimeSeries
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, Container
- **Source filename:** nwb.behavior.yaml
- **Source Specification:** see [Section 5.8.4](#)

Table 4.57: Groups contained in <BehavioralTimeSeries>

Id	Type	Description
<BehavioralTimeSeries>	Group	Top level Group for <BehavioralTimeSeries> • Neurodata Type: <i>BehavioralTimeSeries</i> • Extends: <i>NWBDataInterface</i> • Default Name: BehavioralTimeSeries
—<TimeSeries>	Group	TimeSeries object containing continuous behavioral data. • Extends: <i>TimeSeries</i> • Quantity: 0 or more

4.7.4.1 Groups: <TimeSeries>

TimeSeries object containing continuous behavioral data.

- **Extends:** *TimeSeries*
- **Quantity:** 0 or more

4.7.5 PupilTracking

Overview: Eye-tracking data, representing pupil size.

PupilTracking extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** PupilTracking
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.behavior.yaml
- **Source Specification:** see [Section 5.8.5](#)

Table 4.58: Groups contained in <PupilTracking>

Id	Type	Description
<PupilTracking>	Group	Top level Group for <PupilTracking> • Neurodata Type: <i>PupilTracking</i> • Extends: <i>NWBDataInterface</i> • Default Name: PupilTracking
—<TimeSeries>	Group	TimeSeries object containing time series data on pupil size. • Extends: <i>TimeSeries</i> • Quantity: 1 or more

4.7.5.1 Groups: <TimeSeries>

TimeSeries object containing time series data on pupil size.

- **Extends:** *TimeSeries*
- **Quantity:** 1 or more

4.7.6 EyeTracking

Overview: Eye-tracking data, representing direction of gaze.

EyeTracking extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** EyeTracking
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.behavior.yaml
- **Source Specification:** see [Section 5.8.6](#)

Table 4.59: Groups contained in <EyeTracking>

Id	Type	Description
<EyeTracking>	Group	Top level Group for <EyeTracking> • Neurodata Type: <i>EyeTracking</i> • Extends: <i>NWBDataInterface</i> • Default Name: EyeTracking
—< <i>SpatialSeries</i> >	Group	SpatialSeries object containing data measuring direction of gaze. • Extends: <i>SpatialSeries</i> • Quantity: 0 or more

4.7.6.1 Groups: <SpatialSeries>

SpatialSeries object containing data measuring direction of gaze.

- **Extends:** *SpatialSeries*
- **Quantity:** 0 or more

4.7.7 CompassDirection

Overview: With a `CompassDirection` interface, a module publishes a `SpatialSeries` object representing a floating point value for theta. The `SpatialSeries::reference_frame` field should indicate what direction corresponds to 0 and which is the direction of rotation (this should be clockwise). The `si_unit` for the `SpatialSeries` should be radians or degrees.

`CompassDirection` extends `NWBDataInterface` and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** `CompassDirection`
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, `Container`
- **Source filename:** `nwb.behavior.yaml`
- **Source Specification:** see [Section 5.8.7](#)

Table 4.60: Groups contained in `<CompassDirection>`

Id	Type	Description
<code><CompassDirection></code>	Group	Top level Group for <code><CompassDirection></code> • Neurodata Type: <i>CompassDirection</i> • Extends: <i>NWBDataInterface</i> • Default Name: <code>CompassDirection</code>
— <code><SpatialSeries></code>	Group	<code>SpatialSeries</code> object containing direction of gaze travel. • Extends: <i>SpatialSeries</i> • Quantity: 0 or more

4.7.7.1 Groups: `<SpatialSeries>`

`SpatialSeries` object containing direction of gaze travel.

- **Extends:** *SpatialSeries*
- **Quantity:** 0 or more

4.7.8 Position

Overview: Position data, whether along the x, x/y or x/y/z axis.

Position extends *NWBDataInterface* and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** Position
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.behavior.yaml
- **Source Specification:** see [Section 5.8.8](#)

Table 4.61: Groups contained in <Position>

Id	Type	Description
<Position>	Group	Top level Group for <Position> • Neurodata Type: <i>Position</i> • Extends: <i>NWBDataInterface</i> • Default Name: Position
—< <i>SpatialSeries</i> >	Group	SpatialSeries object containing position data. • Extends: <i>SpatialSeries</i> • Quantity: 1 or more

4.7.8.1 Groups: <SpatialSeries>

SpatialSeries object containing position data.

- **Extends:** *SpatialSeries*
- **Quantity:** 1 or more

4.8 Extracellular electrophysiology

This source module contains neurodata_types for extracellular electrophysiology data.

4.8.1 ElectricalSeries

Overview: A time series of acquired voltage data from extracellular recordings. The data field is an int or float array storing data in volts. The first dimension should always represent time. The second dimension, if present, should represent channels.

ElectricalSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Subtypes:** *SpikeEventSeries*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.1](#)

Table 4.62: Datasets, Links, and Attributes contained in <ElectricalSeries>

Id	Type	Description
<ElectricalSeries>	Group	Top level Group for <ElectricalSeries> • Neurodata Type: <i>ElectricalSeries</i> • Extends: <i>TimeSeries</i>
—filtering	At-tribute	Filtering applied to all channels of the data. For example, if this ElectricalSeries represents high-pass-filtered data (also known as AP Band), then this value could be “High-pass 4-pole Bessel filter at 500 Hz”. If this ElectricalSeries represents low-pass-filtered LFP data and the type of filter is unknown, then this value could be “Low-pass filter at 300 Hz”. If a non-standard filter type is used, provide as much detail about the filter properties as possible. • Data Type: text • Required: False • Name: filtering
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description

continues on next page

Table 4.62 – continued from previous page

Id	Type	Description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Recorded voltage data. • Data Type: numeric • Dimensions: [['num_times'], ['num_times', 'num_channels'], ['num_times', 'num_channels', 'num_samples']] • Shape: [[None], [None, None], [None, None, None]] • Name: data
—unit	At-tribute	Base unit of measurement for working with the data. This value is fixed to 'volts'. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply 'data' by 'conversion', followed by 'channel_conversion' (if present), and then add 'offset'. • Data Type: text • Value: volts • Name: unit
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
—offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset

continues on next page

Table 4.62 – continued from previous page

Id	Type	Description
—resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—electrodes	Dataset	DynamicTableRegion pointer to the electrodes that this time series was generated from. • Extends: DynamicTableRegion • Name: electrodes
—channel_conversion	Dataset	Channel-specific conversion factor. Multiply the data in the ‘data’ dataset by these values along the channel axis (as indicated by axis attribute) AND by the global conversion factor in the ‘conversion’ attribute of ‘data’ to get the data values in Volts, i.e, data in Volts = data * data.conversion * channel_conversion. This approach allows for both global and per-channel data conversion factors needed to support the storage of electrical recordings as native values generated by data acquisition systems. If this dataset is not present, then there is no channel-specific conversion factor, i.e. it is 1 for all channels. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [‘num_channels’] • Shape: [None] • Name: channel_conversion
—axis	At-tribute	The zero-indexed axis of the ‘data’ dataset that the channel-specific conversion factor corresponds to. This value is fixed to 1. • Data Type: int32 • Value: 1 • Name: axis

continues on next page

Table 4.62 – continued from previous page

Id	Type	Description
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
—interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
—unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control

continues on next page

Table 4.62 – continued from previous page

Id	Type	Description
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description

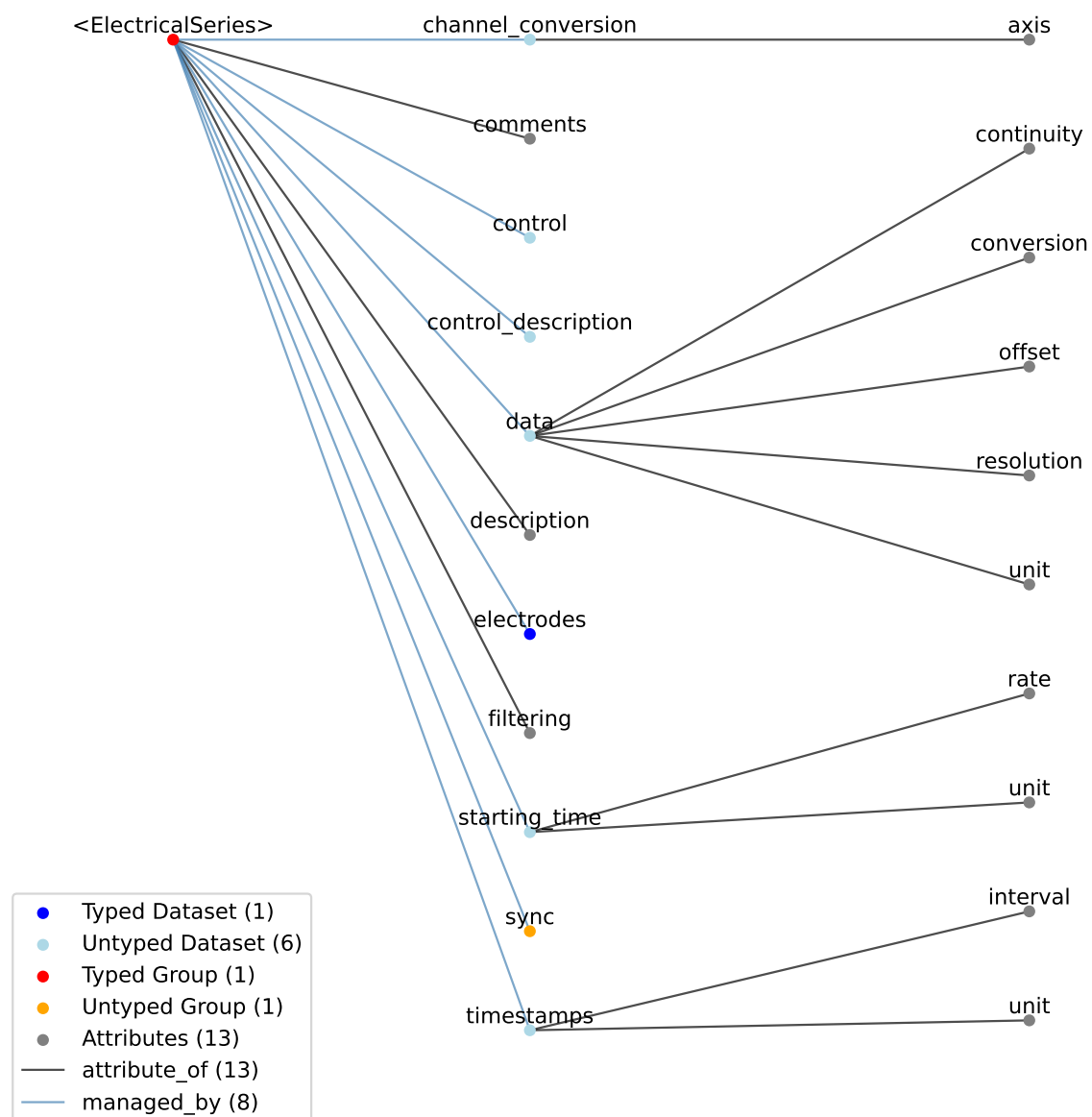
Table 4.63: Groups contained in <ElectricalSeries>

Id	Type	Description
<ElectricalSeries>	Group	Top level Group for <ElectricalSeries> • Neurodata Type: <i>ElectricalSeries</i> • Extends: <i>TimeSeries</i>
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.8.1.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync



4.8.2 SpikeEventSeries

Overview: Stores snapshots/snippets of recorded spike events (i.e., threshold crossings). This may also be raw data, as reported by ephys hardware. If so, the TimeSeries::description field should describe how events were detected. All SpikeEventSeries should reside in a module (under EventWaveform interface) even if the spikes were reported and stored by hardware. All events span the same recording channels and store snapshots of equal duration. TimeSeries::data array structure: [num events] [num channels] [num samples] (or [num events] [num samples] for single electrode).

SpikeEventSeries extends ElectricalSeries and includes all elements of *ElectricalSeries* with the following additions or changes.

- **Extends:** *ElectricalSeries*
- **Primitive Type:** Group
- **Inherits from:** *ElectricalSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.2](#)

Table 4.64: Datasets, Links, and Attributes contained in <SpikeEventSeries>

Id	Type	Description
<SpikeEventSeries>	Group	Top level Group for <SpikeEventSeries> • Neurodata Type: <i>SpikeEventSeries</i> • Extends: <i>ElectricalSeries</i>
—filtering	Attribute	Filtering applied to all channels of the data. For example, if this ElectricalSeries represents high-pass-filtered data (also known as AP Band), then this value could be “High-pass 4-pole Bessel filter at 500 Hz”. If this ElectricalSeries represents low-pass-filtered LFP data and the type of filter is unknown, then this value could be “Low-pass filter at 300 Hz”. If a non-standard filter type is used, provide as much detail about the filter properties as possible. • Data Type: text • Required: False • Name: filtering
—description	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	Attribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments

continues on next page

Table 4.64 – continued from previous page

Id	Type	Description
—data	Dataset	Spike waveforms. • Data Type: numeric • Dimensions: [['num_events', 'num_samples'], ['num_events', 'num_channels', 'num_samples']] • Shape: [[None, None], [None, None, None]] • Name: data
——unit	At-tribute	Unit of measurement for waveforms, which is fixed to 'volts'. • Data Type: text • Value: volts • Name: unit
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution

continues on next page

Table 4.64 – continued from previous page

Id	Type	Description
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. Timestamps are required for the events. Unlike for TimeSeries, timestamps are required for SpikeEventSeries and are thus re-specified here. • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
——electrodes	Dataset	DynamicTableRegion pointer to the electrodes that this time series was generated from. • Extends: DynamicTableRegion • Name: electrodes

continues on next page

Table 4.64 – continued from previous page

Id	Type	Description
—channel_conversion	Dataset	Channel-specific conversion factor. Multiply the data in the ‘data’ dataset by these values along the channel axis (as indicated by axis attribute) AND by the global conversion factor in the ‘conversion’ attribute of ‘data’ to get the data values in Volts, i.e. data in Volts = data * data.conversion * channel_conversion. This approach allows for both global and per-channel data conversion factors needed to support the storage of electrical recordings as native values generated by data acquisition systems. If this dataset is not present, then there is no channel-specific conversion factor, i.e. it is 1 for all channels. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [‘num_channels’] • Shape: [None] • Name: channel_conversion
—axis	Attribute	The zero-indexed axis of the ‘data’ dataset that the channel-specific conversion factor corresponds to. This value is fixed to 1. • Data Type: int32 • Value: 1 • Name: axis
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	Attribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	Attribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control

continues on next page

Table 4.64 – continued from previous page

Id	Type	Description
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description

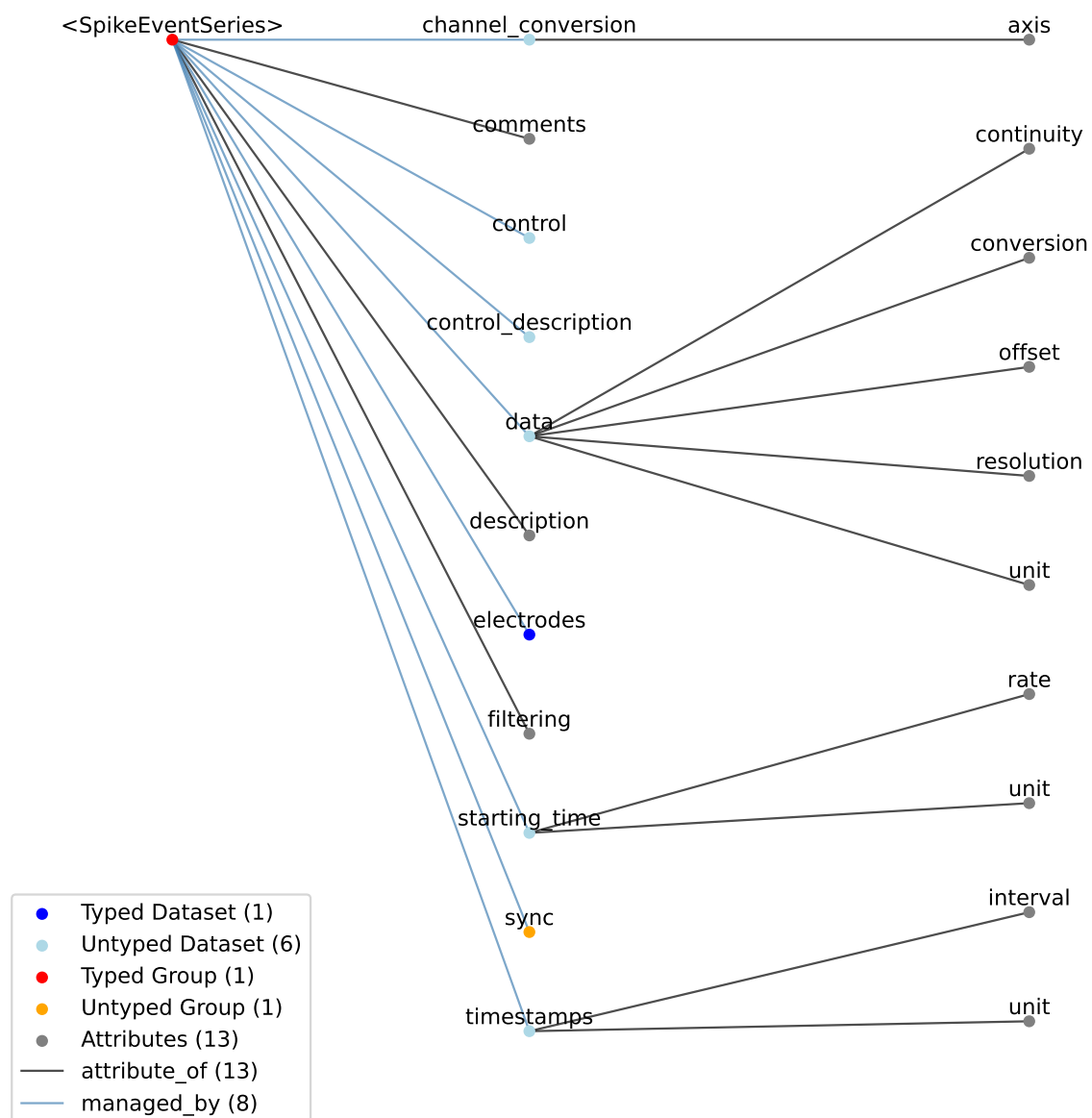
Table 4.65: Groups contained in <SpikeEventSeries>

Id	Type	Description
<SpikeEventSeries>	Group	Top level Group for <SpikeEventSeries> • Neurodata Type: <i>SpikeEventSeries</i> • Extends: <i>ElectricalSeries</i>
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.8.2.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync



4.8.3 FeatureExtraction

Overview: Features, such as PC1 and PC2, that are extracted from signals stored in a SpikeEventSeries or other source. FeatureExtraction extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** FeatureExtraction
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, Container
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see Section 5.9.3

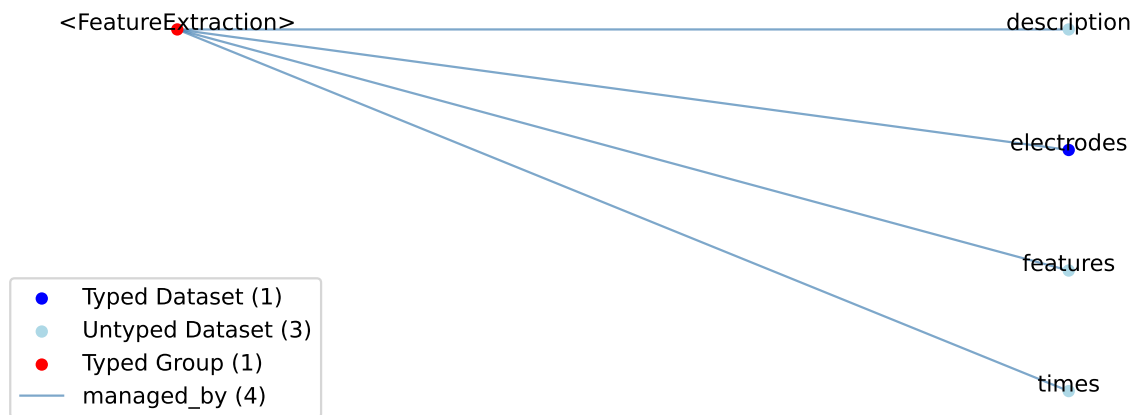


Table 4.66: Datasets, Links, and Attributes contained in <FeatureExtraction>

Id	Type	Description
<FeatureExtraction>	Group	Top level Group for <FeatureExtraction> • Neurodata Type: <i>FeatureExtraction</i> • Extends: <i>NWBDataInterface</i> • Default Name: FeatureExtraction
—description	Dataset	Description of features (eg, “PC1”) for each of the extracted features. • Data Type: text • Dimensions: [‘num_features’] • Shape: [None] • Name: description
—features	Dataset	Multi-dimensional array of features extracted from each event. • Data Type: float32 • Dimensions: [‘num_events’, ‘num_channels’, ‘num_features’] • Shape: [None, None, None] • Name: features

continues on next page

Table 4.66 – continued from previous page

Id	Type	Description
—times	Dataset	Times of events that features correspond to (can be a link). • Data Type: float64 • Dimensions: ['num_events'] • Shape: [None] • Name: times
—electrodes	Dataset	DynamicTableRegion pointer to the electrodes that this time series was generated from. • Extends: DynamicTableRegion • Name: electrodes

4.8.4 EventDetection

Overview: Detected spike events from voltage trace(s).

EventDetection extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** EventDetection
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.4](#)

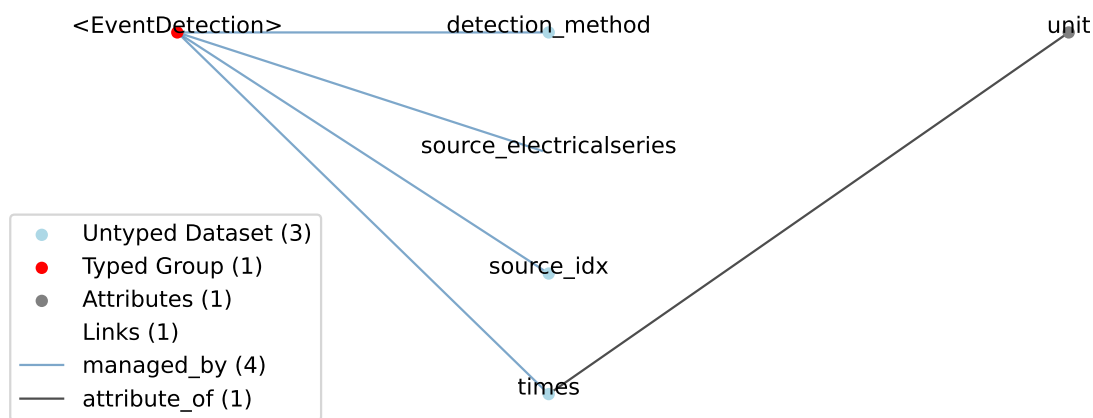


Table 4.67: Datasets, Links, and Attributes contained in `<EventDetection>`

Id	Type	Description
<code><EventDetection></code>	Group	Top level Group for <code><EventDetection></code> • Neurodata Type: <i>EventDetection</i> • Extends: <i>NWBDataInterface</i> • Default Name: EventDetection
<code>—detection_method</code>	Dataset	Description of how events were detected, such as voltage threshold, or dV/dT threshold, as well as relevant values. • Data Type: text • Name: detection_method

continues on next page

Table 4.67 – continued from previous page

Id	Type	Description
—source_idx	Dataset	Indices (zero-based) into source <code>ElectricalSeries::data</code> array corresponding to time of event. “description” should define what is meant by time of event (e.g., .25 ms before action potential peak, zero-crossing time, etc). The index points to each event from the raw data. • Data Type: int32 • Dimensions: ['num_events'] • Shape: [None] • Name: source_idx
—times	Dataset	Timestamps of events, in seconds. • Data Type: float64 • Dimensions: ['num_events'] • Shape: [None] • Name: times
——unit	Attribute	Unit of measurement for event times, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—source_electricalseries	Link	Link to the <code>ElectricalSeries</code> that this data was calculated from. Metadata about electrodes and their position can be read from that <code>ElectricalSeries</code> so it’s not necessary to include that information here. • Target Type <i>ElectricalSeries</i> • Name: source_electricalseries

Table 4.68: Groups contained in <EventDetection>

Id	Type	Description
<EventDetection>	Group	Top level Group for <EventDetection> • Neurodata Type: <i>EventDetection</i> • Extends: <i>NWBDataInterface</i> • Default Name: EventDetection
—source_electricalseries	Link	Link to the <code>ElectricalSeries</code> that this data was calculated from. Metadata about electrodes and their position can be read from that <code>ElectricalSeries</code> so it’s not necessary to include that information here. • Target Type <i>ElectricalSeries</i> • Name: source_electricalseries

4.8.5 EventWaveform

Overview: Represents either the waveforms of detected events, as extracted from a raw data trace in /acquisition, or the event waveforms that were stored during experiment acquisition.

EventWaveform extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** EventWaveform
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, Container
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.5](#)

Table 4.69: Groups contained in <EventWaveform>

Id	Type	Description
<EventWaveform>	Group	Top level Group for <EventWaveform> • Neurodata Type: <i>EventWaveform</i> • Extends: <i>NWBDataInterface</i> • Default Name: EventWaveform
—< <i>SpikeEventSeries</i> >	Group	SpikeEventSeries object(s) containing detected spike event waveforms. • Extends: <i>SpikeEventSeries</i> • Quantity: 0 or more

4.8.5.1 Groups: <SpikeEventSeries>

SpikeEventSeries object(s) containing detected spike event waveforms.

- **Extends:** *SpikeEventSeries*
- **Quantity:** 0 or more

4.8.6 FilteredEphys

Overview: Electrophysiology data from one or more channels that has been subjected to filtering. Examples of filtered data include Theta and Gamma (LFP has its own interface). FilteredEphys modules publish an ElectricalSeries for each filtered channel or set of channels. The name of each ElectricalSeries is arbitrary but should be informative. The source of the filtered data, whether this is from analysis of another time series or as acquired by hardware, should be noted in each's TimeSeries::description field. There is no assumed 1::1 correspondence between filtered ephys signals and electrodes, as a single signal can apply to many nearby electrodes, and one electrode may have different filtered (e.g., theta and/or gamma) signals represented. Filter properties should be noted in the ElectricalSeries 'filtering' attribute.

FilteredEphys extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** FilteredEphys
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.6](#)

Table 4.70: Groups contained in <FilteredEphys>

Id	Type	Description
<FilteredEphys>	Group	Top level Group for <FilteredEphys> • Neurodata Type: <i>FilteredEphys</i> • Extends: <i>NWBDataInterface</i> • Default Name: FilteredEphys
—< <i>ElectricalSeries</i> >	Group	ElectricalSeries object(s) containing filtered electrophysiology data. • Extends: <i>ElectricalSeries</i> • Quantity: 1 or more

4.8.6.1 Groups: <ElectricalSeries>

ElectricalSeries object(s) containing filtered electrophysiology data.

- **Extends:** *ElectricalSeries*
- **Quantity:** 1 or more

4.8.7 LFP

Overview: LFP data from one or more channels. The electrode map in each published ElectricalSeries will identify which channels are providing LFP data. Filter properties should be noted in the ElectricalSeries ‘filtering’ attribute.

LFP extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** LFP
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, Container
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.7](#)

Table 4.71: Groups contained in <LFP>

Id	Type	Description
<LFP>	Group	Top level Group for <LFP> • Neurodata Type: <i>LFP</i> • Extends: <i>NWBDataInterface</i> • Default Name: LFP
—< <i>ElectricalSeries</i> >	Group	ElectricalSeries object(s) containing LFP data for one or more channels. • Extends: <i>ElectricalSeries</i> • Quantity: 1 or more

4.8.7.1 Groups: <ElectricalSeries>

ElectricalSeries object(s) containing LFP data for one or more channels.

- **Extends:** *ElectricalSeries*
- **Quantity:** 1 or more

4.8.8 ElectrodeGroup

Overview: A physical grouping of electrodes, e.g. a shank of an array.

ElectrodeGroup extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.8](#)

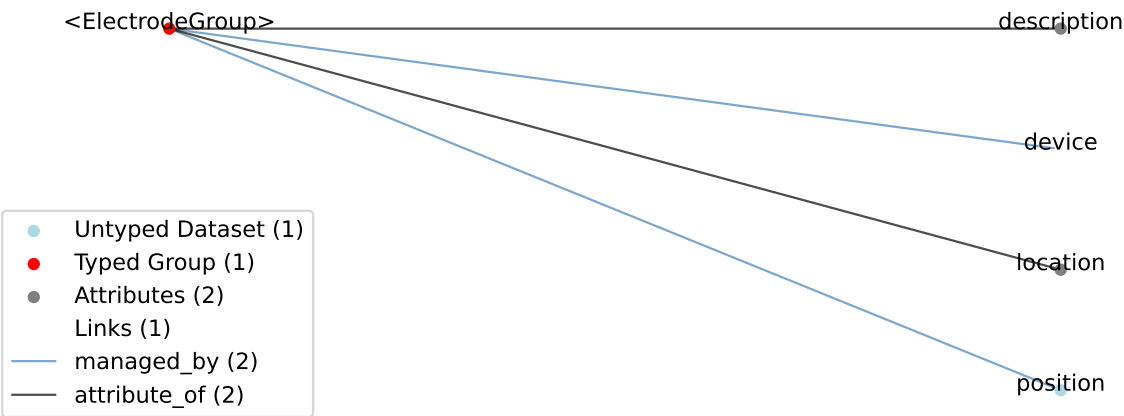


Table 4.72: Datasets, Links, and Attributes contained in <ElectrodeGroup>

Id	Type	Description
<ElectrodeGroup>	Group	Top level Group for <ElectrodeGroup> • Neurodata Type: <i>ElectrodeGroup</i> • Extends: <i>NWBContainer</i>
—description	At-tribute	Description of this electrode group. • Data Type: text • Name: description
—location	At-tribute	Location of electrode group. Specify the area, layer, comments on estimation of area/layer, etc. Use standard atlas names for anatomical regions when possible. • Data Type: text • Name: location

continues on next page

Table 4.72 – continued from previous page

Id	Type	Description
—position	Dataset	stereotaxic or common framework coordinates • Quantity: 0 or 1 • Data Type: Compound data type with the following elements: – x: x coordinate (<i>dtype= float32</i>) – y: y coordinate (<i>dtype= float32</i>) – z: z coordinate (<i>dtype= float32</i>) • Name: position
—device	Link	Link to the device that was used to record from this electrode group. • Target Type <i>Device</i> • Name: device

Table 4.73: Groups contained in <ElectrodeGroup>

Id	Type	Description
<ElectrodeGroup>	Group	Top level Group for <ElectrodeGroup> • Neurodata Type: <i>ElectrodeGroup</i> • Extends: <i>NWBContainer</i>
—device	Link	Link to the device that was used to record from this electrode group. • Target Type <i>Device</i> • Name: device

4.8.9 ClusterWaveforms

Overview: DEPRECATED The mean waveform shape, including standard deviation, of the different clusters. Ideally, the waveform analysis should be performed on data that is only high-pass filtered. This is a separate module because it is expected to require updating. For example, IMEC probes may require different storage requirements to store/display mean waveforms, requiring a new interface or an extension of this one.

ClusterWaveforms extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** ClusterWaveforms
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.9](#)

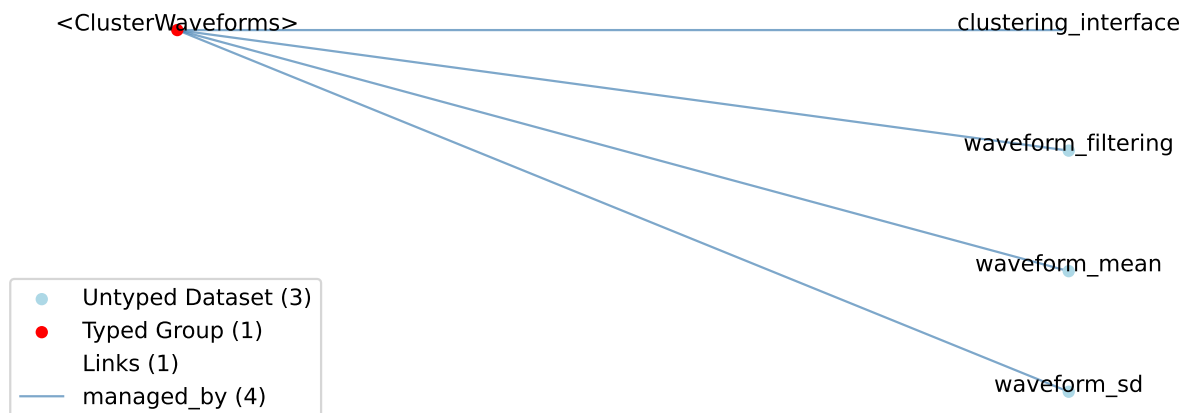


Table 4.74: Datasets, Links, and Attributes contained in <ClusterWaveforms>

Id	Type	Description
<ClusterWaveforms>	Group	Top level Group for <ClusterWaveforms> • Neurodata Type: <i>ClusterWaveforms</i> • Extends: <i>NWBDataInterface</i> • Default Name: ClusterWaveforms
—waveform_filtering	Dataset	Filtering applied to data before generating mean/sd • Data Type: text • Name: waveform_filtering

continues on next page

Table 4.74 – continued from previous page

Id	Type	Description
—waveform_mean	Dataset	The mean waveform for each cluster, using the same indices for each wave as cluster numbers in the associated Clustering module (i.e, cluster 3 is in array slot [3]). Waveforms corresponding to gaps in cluster sequence should be empty (e.g., zero- filled) • Data Type: float32 • Dimensions: ['num_clusters', 'num_samples'] • Shape: [None, None] • Name: waveform_mean
—waveform_sd	Dataset	Stdev of waveforms for each cluster, using the same indices as in mean • Data Type: float32 • Dimensions: ['num_clusters', 'num_samples'] • Shape: [None, None] • Name: waveform_sd
—clustering_interface	Link	Link to Clustering interface that was the source of the clustered data • Target Type <i>Clustering</i> • Name: clustering_interface

Table 4.75: Groups contained in <ClusterWaveforms>

Id	Type	Description
<ClusterWaveforms>	Group	Top level Group for <ClusterWaveforms> • Neurodata Type: <i>ClusterWaveforms</i> • Extends: <i>NWBDataInterface</i> • Default Name: ClusterWaveforms
—clustering_interface	Link	Link to Clustering interface that was the source of the clustered data • Target Type <i>Clustering</i> • Name: clustering_interface

4.8.10 Clustering

Overview: DEPRECATED Clustered spike data, whether from automatic clustering tools (e.g., klustakwik) or as a result of manual sorting.

Clustering extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** Clustering
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ecephys.yaml
- **Source Specification:** see [Section 5.9.10](#)

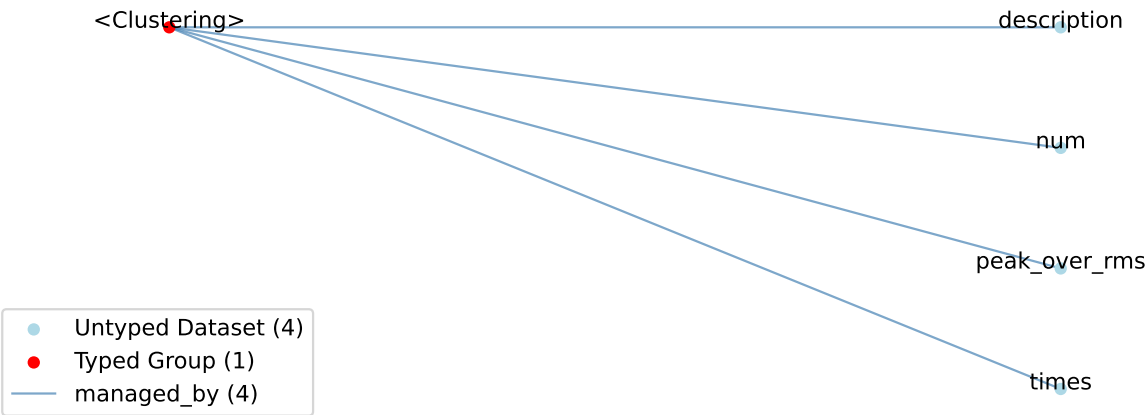


Table 4.76: Datasets, Links, and Attributes contained in <Clustering>

Id	Type	Description
<Clustering>	Group	Top level Group for <Clustering> • Neurodata Type: <i>Clustering</i> • Extends: <i>NWBDataInterface</i> • Default Name: Clustering
—description	Dataset	Description of clusters or clustering, (e.g. cluster 0 is noise, clusters curated using Klusters, etc) • Data Type: text • Name: description
—num	Dataset	Cluster number of each event • Data Type: int32 • Dimensions: ['num_events'] • Shape: [None] • Name: num

continues on next page

Table 4.76 – continued from previous page

Id	Type	Description
—peak_over_rms	Dataset	Maximum ratio of waveform peak to RMS on any channel in the cluster (provides a basic clustering metric). • Data Type: float32 • Dimensions: ['num_clusters'] • Shape: [None] • Name: peak_over_rms
—times	Dataset	Times of clustered events, in seconds. This may be a link to times field in associated FeatureExtraction module. • Data Type: float64 • Dimensions: ['num_events'] • Shape: [None] • Name: times

4.9 Intracellular electrophysiology

This source module contains neurodata_types for intracellular electrophysiology data.

4.9.1 PatchClampSeries

Overview: An abstract base class for patch-clamp data - stimulus or response, current or voltage.

PatchClampSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Subtypes:** *CurrentClampSeries*, *CurrentClampStimulusSeries*, *VoltageClampStimulusSeries*, *Ize-roClampSeries*, *VoltageClampSeries*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see [Section 5.10.1](#)

Table 4.77: Datasets, Links, and Attributes contained in <PatchClampSeries>

Id	Type	Description
<PatchClampSeries>	Group	Top level Group for <PatchClampSeries> • Neurodata Type: <i>PatchClampSeries</i> • Extends: <i>TimeSeries</i>
—stimulus_description	At-tribute	Protocol/stimulus name for this patch-clamp dataset. • Data Type: text • Name: stimulus_description
—sweep_number	At-tribute	Sweep number, allows to group different PatchClampSeries together. • Data Type: uint32 • Required: False • Name: sweep_number
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description

continues on next page

Table 4.77 – continued from previous page

Id	Type	Description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Recorded voltage or current. • Data Type: numeric • Dimensions: ['num_times'] • Shape: [None] • Name: data
—unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply 'data' by 'conversion' and add 'offset'. • Data Type: text • Name: unit
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
—offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset

continues on next page

Table 4.77 – continued from previous page

Id	Type	Description
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—gain	Dataset	Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp). • Quantity: 0 or 1 • Data Type: float32 • Name: gain
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.77 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
—interval	Attribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
—unit	Attribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode

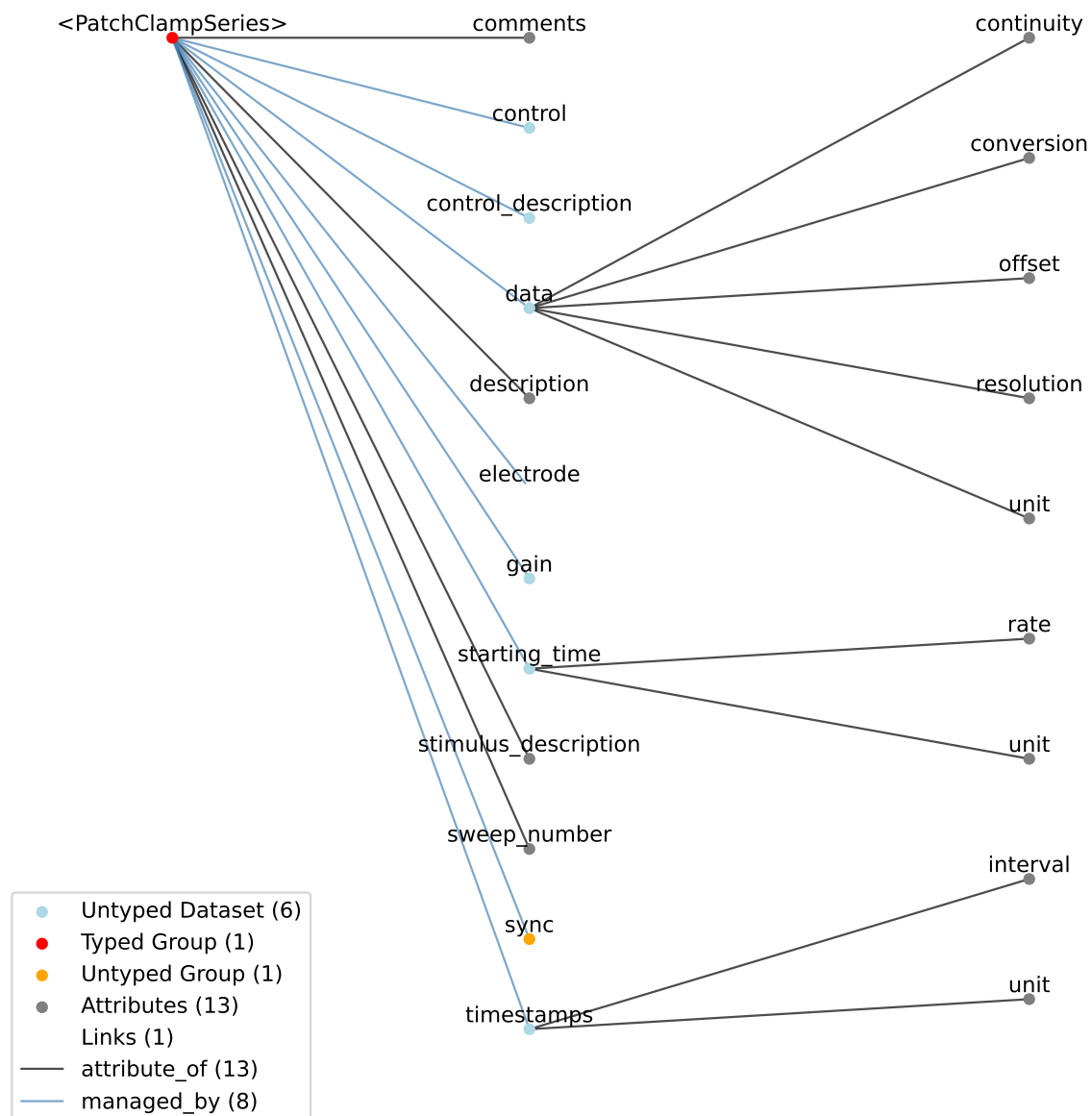


Table 4.78: Groups contained in <PatchClampSeries>

Id	Type	Description
<PatchClampSeries>	Group	Top level Group for <PatchClampSeries> • Neurodata Type: <i>PatchClampSeries</i> • Extends: <i>TimeSeries</i>
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.9.1.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.9.2 CurrentClampSeries

Overview: Voltage data from an intracellular current-clamp recording. A corresponding CurrentClampStimulusSeries (stored separately as a stimulus) is used to store the current injected.

CurrentClampSeries extends PatchClampSeries and includes all elements of *PatchClampSeries* with the following additions or changes.

- **Extends:** *PatchClampSeries*
- **Primitive Type:** Group
- **Inherits from:** *PatchClampSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, Container
- **Subtypes:** *IZeroClampSeries*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see [Section 5.10.2](#)

Table 4.79: Datasets, Links, and Attributes contained in <CurrentClampSeries>

Id	Type	Description
<CurrentClampSeries>	Group	Top level Group for <CurrentClampSeries> • Neurodata Type: <i>CurrentClampSeries</i> • Extends: <i>PatchClampSeries</i>
—stimulus_description	Attribute	Protocol/stimulus name for this patch-clamp dataset. • Data Type: text • Name: stimulus_description
—sweep_number	Attribute	Sweep number, allows to group different PatchClampSeries together. • Data Type: uint32 • Required: False • Name: sweep_number
—description	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	Attribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Recorded voltage. • Name: data

continues on next page

Table 4.79 – continued from previous page

Id	Type	Description
——unit	At-tribute	Base unit of measurement for working with the data. which is fixed to ‘volts’. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Value: volts • Name: unit
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution

continues on next page

Table 4.79 – continued from previous page

Id	Type	Description
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—bias_current	Dataset	Bias current, in amps. • Quantity: 0 or 1 • Data Type: float32 • Name: bias_current
—bridge_balance	Dataset	Bridge balance, in ohms. • Quantity: 0 or 1 • Data Type: float32 • Name: bridge_balance
—capacitance_compensation	Dataset	Capacitance compensation, in farads. • Quantity: 0 or 1 • Data Type: float32 • Name: capacitance_compensation
—gain	Dataset	Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp). • Quantity: 0 or 1 • Data Type: float32 • Name: gain
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.79 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
—interval	At-tribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
—unit	At-tribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode

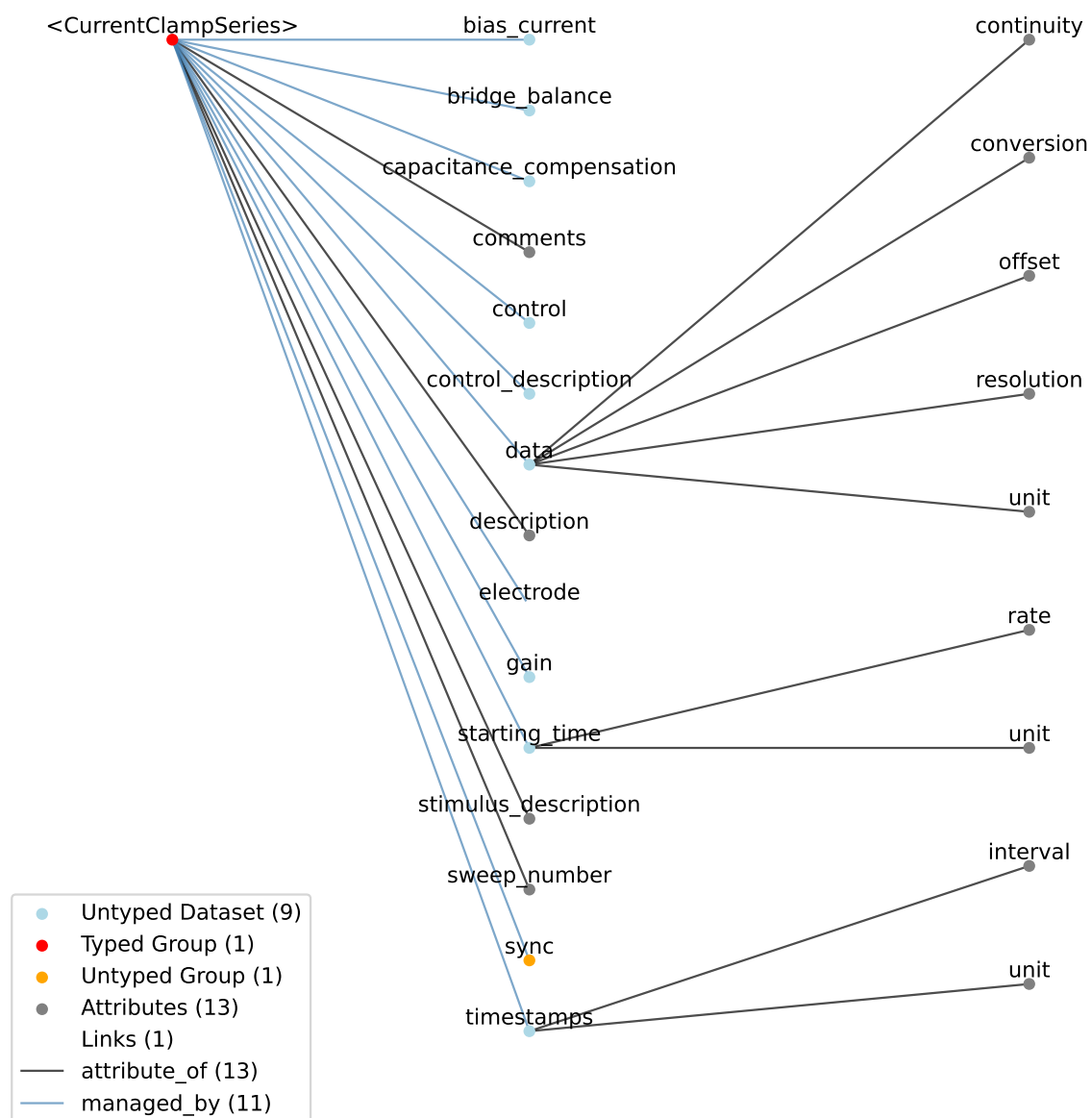


Table 4.80: Groups contained in <CurrentClampSeries>

Id	Type	Description
<CurrentClampSeries>	Group	Top level Group for <CurrentClampSeries> • Neurodata Type: <i>CurrentClampSeries</i> • Extends: <i>PatchClampSeries</i>
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.9.2.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.9.3 IZeroClampSeries

Overview: Voltage data from an intracellular recording when all current and amplifier settings are off (i.e., CurrentClampSeries fields will be zero). There is no CurrentClampStimulusSeries associated with an IZero series because the amplifier is disconnected and no stimulus can reach the cell.

IZeroClampSeries extends CurrentClampSeries and includes all elements of *CurrentClampSeries* with the following additions or changes.

- **Extends:** *CurrentClampSeries*
- **Primitive Type:** Group
- **Inherits from:** *CurrentClampSeries*, *PatchClampSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.3

Table 4.81: Datasets, Links, and Attributes contained in <IZeroClampSeries>

Id	Type	Description
<IZeroClampSeries>	Group	Top level Group for <IZeroClampSeries> • Neurodata Type: <i>IZeroClampSeries</i> • Extends: <i>CurrentClampSeries</i>
—stimulus_description	Attribute	An IZeroClampSeries has no stimulus, so this attribute is automatically set to “N/A” • Data Type: text • Value: N/A • Name: stimulus_description
—sweep_number	Attribute	Sweep number, allows to group different PatchClampSeries together. • Data Type: uint32 • Required: False • Name: sweep_number
—description	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	Attribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments

continues on next page

Table 4.81 – continued from previous page

Id	Type	Description
—bias_current	Dataset	Bias current, in amps, fixed to 0.0. • Data Type: float32 • Name: bias_current
—bridge_balance	Dataset	Bridge balance, in ohms, fixed to 0.0. • Data Type: float32 • Name: bridge_balance
—capacitance_compensation	Dataset	Capacitance compensation, in farads, fixed to 0.0. • Data Type: float32 • Name: capacitance_compensation
—data	Dataset	Recorded voltage. • Name: data
——unit	Attribute	Base unit of measurement for working with the data. which is fixed to ‘volts’. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Value: volts • Name: unit
——conversion	Attribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	Attribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset

continues on next page

Table 4.81 – continued from previous page

Id	Type	Description
—resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
—continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—gain	Dataset	Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp). • Quantity: 0 or 1 • Data Type: float32 • Name: gain
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.81 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
—interval	Attribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
—unit	Attribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode

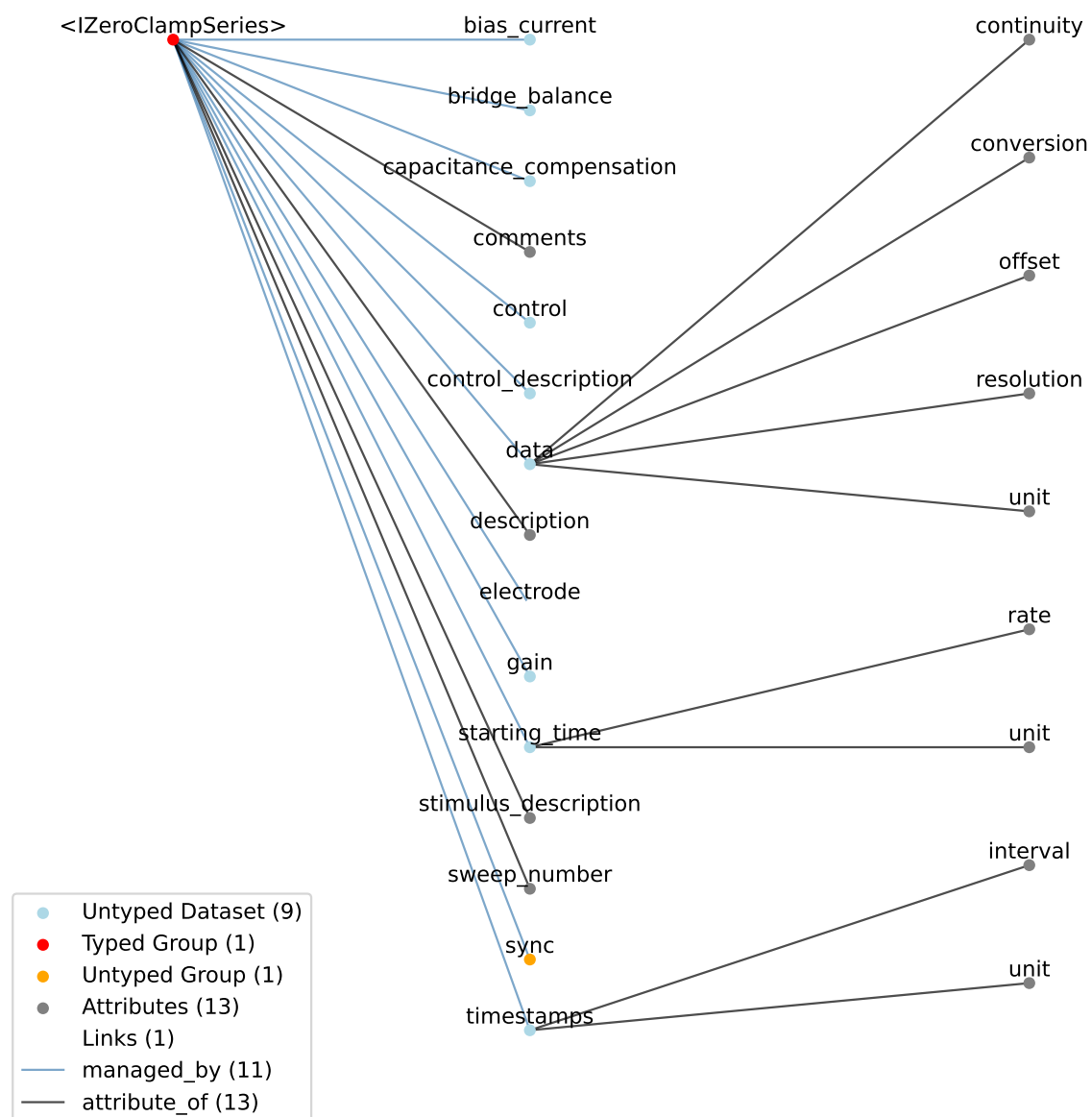


Table 4.82: Groups contained in <IZeroClampSeries>

Id	Type	Description
<IZeroClampSeries>	Group	Top level Group for <IZeroClampSeries> • Neurodata Type: <i>IZeroClampSeries</i> • Extends: <i>CurrentClampSeries</i>
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.9.3.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.9.4 CurrentClampStimulusSeries

Overview: Stimulus current applied during current clamp recording.

CurrentClampStimulusSeries extends PatchClampSeries and includes all elements of *PatchClampSeries* with the following additions or changes.

- **Extends:** *PatchClampSeries*
- **Primitive Type:** Group
- **Inherits from:** *PatchClampSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.4

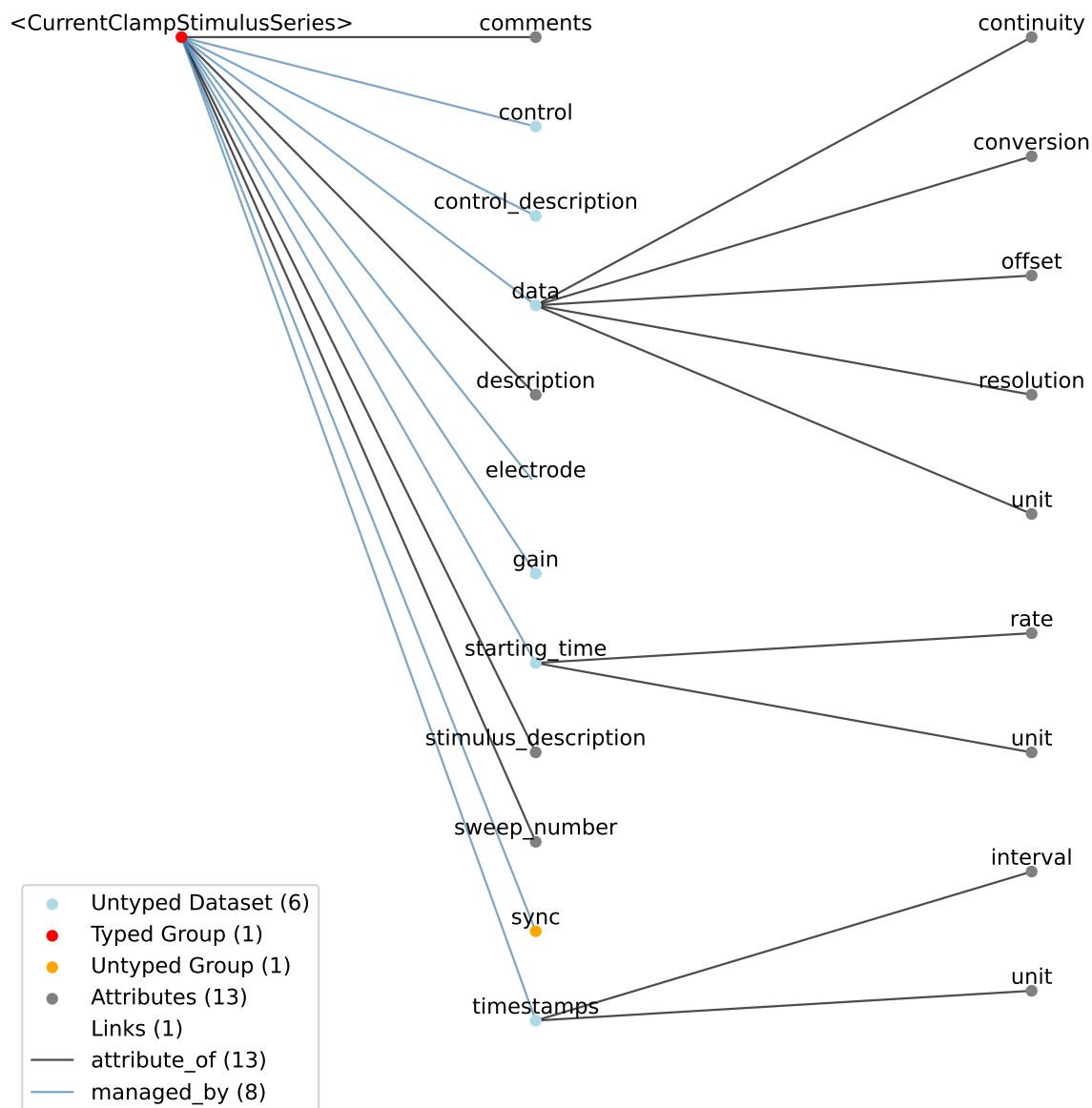


Table 4.83: Datasets, Links, and Attributes contained in <CurrentClampStimulusSeries>

Id	Type	Description
<CurrentClampStimulusSeries>	Group	Top level Group for <CurrentClampStimulusSeries> • Neurodata Type: <i>CurrentClampStimulusSeries</i> • Extends: <i>PatchClampSeries</i>
—stimulus_description	Attribute	Protocol/stimulus name for this patch-clamp dataset. • Data Type: text • Name: stimulus_description
—sweep_number	Attribute	Sweep number, allows to group different PatchClampSeries together. • Data Type: uint32 • Required: False • Name: sweep_number
—description	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	Attribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Stimulus current applied. • Name: data
—unit	Attribute	Base unit of measurement for working with the data. which is fixed to ‘amperes’. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Value: amperes • Name: unit

continues on next page

Table 4.83 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity

continues on next page

Table 4.83 – continued from previous page

Id	Type	Description
—gain	Dataset	Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp). • Quantity: 0 or 1 • Data Type: float32 • Name: gain
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	Attribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	Attribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
—interval	Attribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
—unit	Attribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control

continues on next page

Table 4.83 – continued from previous page

Id	Type	Description
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode

Table 4.84: Groups contained in <CurrentClampStimulusSeries>

Id	Type	Description
<CurrentClampStimulusSeries>	Group	Top level Group for <CurrentClampStimulusSeries> • Neurodata Type: <i>CurrentClampStimulusSeries</i> • Extends: <i>PatchClampSeries</i>
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.9.4.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.9.5 VoltageClampSeries

Overview: Current data from an intracellular voltage-clamp recording. A corresponding VoltageClampStimulusSeries (stored separately as a stimulus) is used to store the voltage injected.

VoltageClampSeries extends PatchClampSeries and includes all elements of *PatchClampSeries* with the following additions or changes.

- **Extends:** *PatchClampSeries*
- **Primitive Type:** Group
- **Inherits from:** *PatchClampSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see [Section 5.10.5](#)

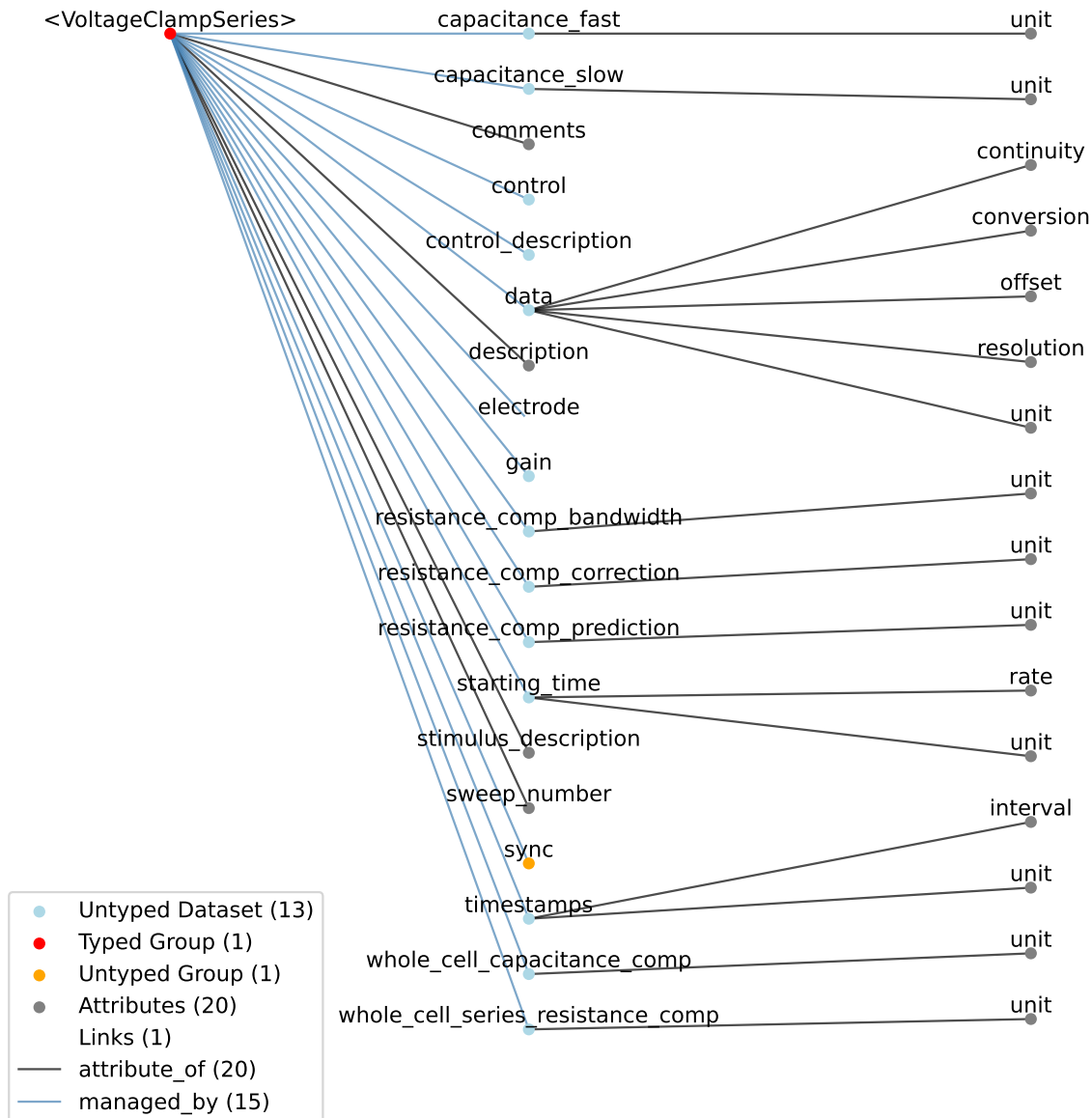


Table 4.85: Datasets, Links, and Attributes contained in <VoltageClampSeries>

Id	Type	Description
<VoltageClampSeries>	Group	Top level Group for <VoltageClampSeries> • Neurodata Type: <i>VoltageClampSeries</i> • Extends: <i>PatchClampSeries</i>
—stimulus_description	At-tribute	Protocol/stimulus name for this patch-clamp dataset. • Data Type: text • Name: stimulus_description
—sweep_number	At-tribute	Sweep number, allows to group different PatchClampSeries together. • Data Type: uint32 • Required: False • Name: sweep_number
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Recorded current. • Name: data
—unit	At-tribute	Base unit of measurement for working with the data. which is fixed to ‘amperes’. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Value: amperes • Name: unit

continues on next page

Table 4.85 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity

continues on next page

Table 4.85 – continued from previous page

Id	Type	Description
—capacitance_fast	Dataset	Fast capacitance, in farads. • Quantity: 0 or 1 • Data Type: float32 • Name: capacitance_fast
——unit	At-tribute	Unit of measurement for capacitance_fast, which is fixed to ‘farads’. • Data Type: text • Value: farads • Name: unit
—capacitance_slow	Dataset	Slow capacitance, in farads. • Quantity: 0 or 1 • Data Type: float32 • Name: capacitance_slow
——unit	At-tribute	Unit of measurement for capacitance_fast, which is fixed to ‘farads’. • Data Type: text • Value: farads • Name: unit
—resistance_comp_bandwidth	Dataset	Resistance compensation bandwidth, in hertz. • Quantity: 0 or 1 • Data Type: float32 • Name: resistance_comp_bandwidth
——unit	At-tribute	Unit of measurement for resistance_comp_bandwidth, which is fixed to ‘hertz’. • Data Type: text • Value: hertz • Name: unit
—resistance_comp_correction	Dataset	Resistance compensation correction, in percent. • Quantity: 0 or 1 • Data Type: float32 • Name: resistance_comp_correction
——unit	At-tribute	Unit of measurement for resistance_comp_correction, which is fixed to ‘percent’. • Data Type: text • Value: percent • Name: unit
—resistance_comp_prediction	Dataset	Resistance compensation prediction, in percent. • Quantity: 0 or 1 • Data Type: float32 • Name: resistance_comp_prediction

continues on next page

Table 4.85 – continued from previous page

Id	Type	Description
——unit	At-tribute	Unit of measurement for resistance_comp_prediction, which is fixed to ‘percent’. • Data Type: text • Value: percent • Name: unit
—whole_cell_capacitance_co	Dataset	Whole cell capacitance compensation, in farads. • Quantity: 0 or 1 • Data Type: float32 • Name: whole_cell_capacitance_comp
——unit	At-tribute	Unit of measurement for whole_cell_capacitance_comp, which is fixed to ‘farads’. • Data Type: text • Value: farads • Name: unit
—whole_cell_series_resistanc	Dataset	Whole cell series resistance compensation, in ohms. • Quantity: 0 or 1 • Data Type: float32 • Name: whole_cell_series_resistance_comp
——unit	At-tribute	Unit of measurement for whole_cell_series_resistance_comp, which is fixed to ‘ohms’. • Data Type: text • Value: ohms • Name: unit
—gain	Dataset	Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp). • Quantity: 0 or 1 • Data Type: float32 • Name: gain
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.85 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
—interval	Attribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
—unit	Attribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode

Table 4.86: Groups contained in <VoltageClampSeries>

Id	Type	Description
<VoltageClampSeries>	Group	Top level Group for <VoltageClampSeries> • Neurodata Type: <i>VoltageClampSeries</i> • Extends: <i>PatchClampSeries</i>
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.9.5.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.9.6 VoltageClampStimulusSeries

Overview: Stimulus voltage applied during a voltage clamp recording.

VoltageClampStimulusSeries extends PatchClampSeries and includes all elements of *PatchClampSeries* with the following additions or changes.

- **Extends:** *PatchClampSeries*
- **Primitive Type:** Group
- **Inherits from:** *PatchClampSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.6

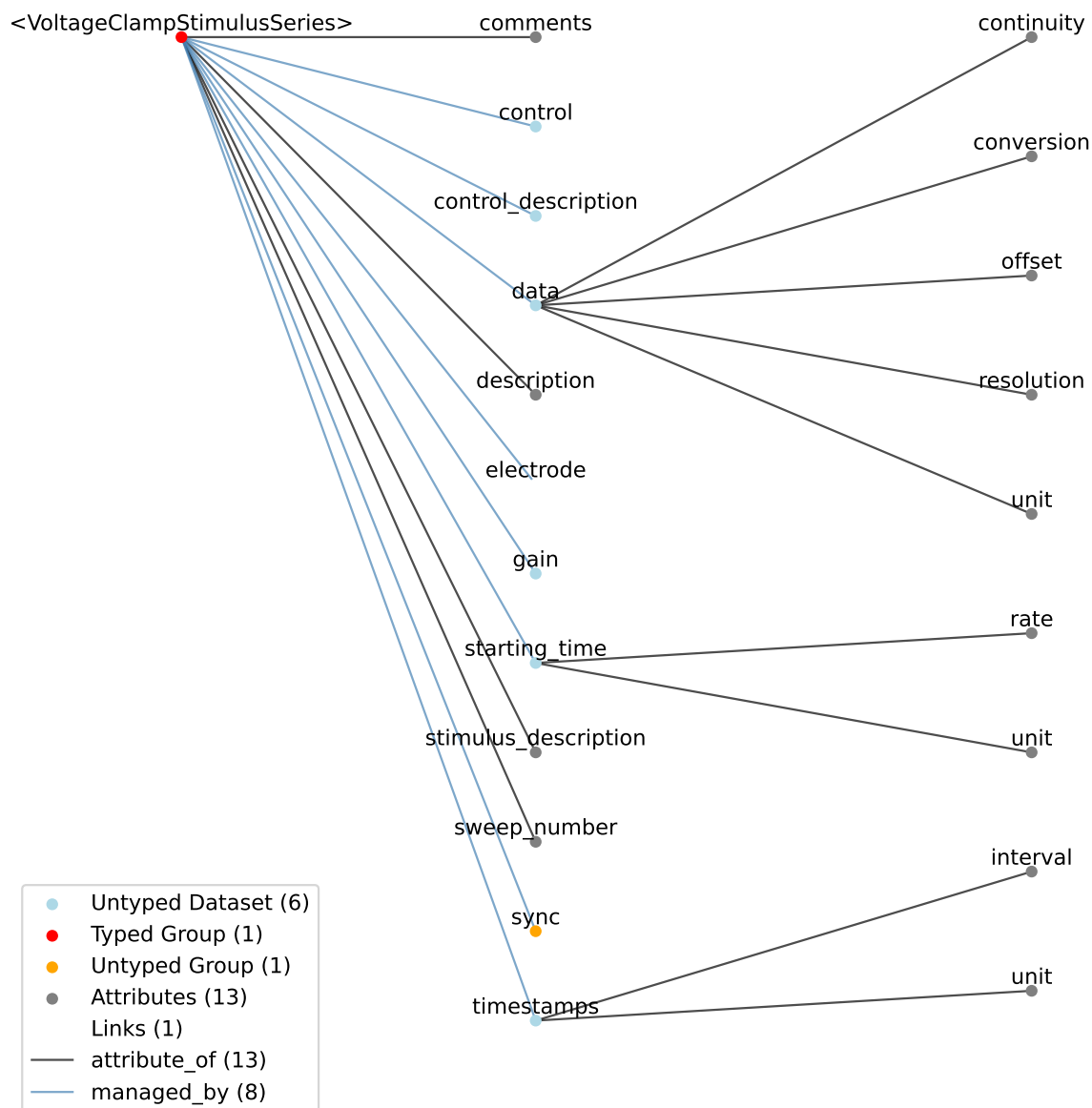


Table 4.87: Datasets, Links, and Attributes contained in <VoltageClampStimulusSeries>

Id	Type	Description
<VoltageClampStimulusSeries>	Group	Top level Group for <VoltageClampStimulusSeries> • Neurodata Type: <i>VoltageClampStimulusSeries</i> • Extends: <i>PatchClampSeries</i>
—stimulus_description	Attribute	Protocol/stimulus name for this patch-clamp dataset. • Data Type: text • Name: stimulus_description
—sweep_number	Attribute	Sweep number, allows to group different PatchClampSeries together. • Data Type: uint32 • Required: False • Name: sweep_number
—description	Attribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	Attribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Stimulus voltage applied. • Name: data
—unit	Attribute	Base unit of measurement for working with the data. which is fixed to ‘volts’. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Value: volts • Name: unit

continues on next page

Table 4.87 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity

continues on next page

Table 4.87 – continued from previous page

Id	Type	Description
—gain	Dataset	Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp). • Quantity: 0 or 1 • Data Type: float32 • Name: gain
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	Attribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
—unit	Attribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
—interval	Attribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
—unit	Attribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control

continues on next page

Table 4.87 – continued from previous page

Id	Type	Description
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode

Table 4.88: Groups contained in <VoltageClampStimulusSeries>

Id	Type	Description
<VoltageClampStimulusSeries>	Group	Top level Group for <VoltageClampStimulusSeries> • Neurodata Type: <i>VoltageClampStimulusSeries</i> • Extends: <i>PatchClampSeries</i>
—electrode	Link	Link to IntracellularElectrode object that describes the electrode that was used to apply or record this data. • Target Type <i>IntracellularElectrode</i> • Name: electrode
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.9.6.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.9.7 IntracellularElectrode

Overview: An intracellular electrode and its metadata.

IntracellularElectrode extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see [Section 5.10.7](#)

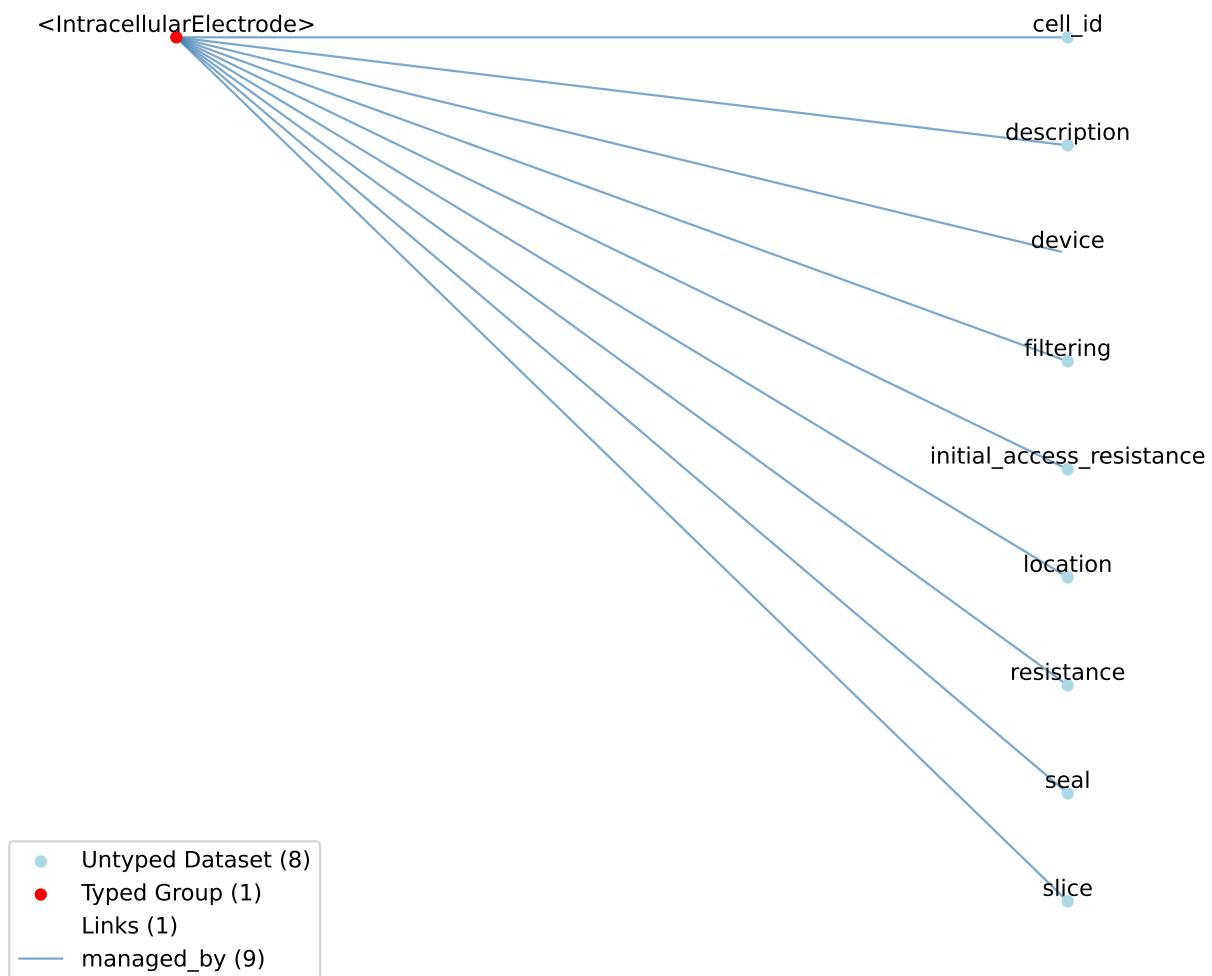


Table 4.89: Datasets, Links, and Attributes contained in <IntracellularElectrode>

Id	Type	Description
<IntracellularElectrode>	Group	Top level Group for <IntracellularElectrode> • Neurodata Type: <i>IntracellularElectrode</i> • Extends: <i>NWBContainer</i>
—cell_id	Dataset	unique ID of the cell • Quantity: 0 or 1 • Data Type: text • Name: cell_id
—description	Dataset	Description of electrode (e.g., whole-cell, sharp, etc.). • Data Type: text • Name: description
—filtering	Dataset	Electrode specific filtering. • Quantity: 0 or 1 • Data Type: text • Name: filtering
—initial_access_resistance	Dataset	Initial access resistance. • Quantity: 0 or 1 • Data Type: text • Name: initial_access_resistance
—location	Dataset	Location of the electrode. Specify the area, layer, comments on estimation of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names for anatomical regions when possible. • Quantity: 0 or 1 • Data Type: text • Name: location
—resistance	Dataset	Electrode resistance, in ohms. • Quantity: 0 or 1 • Data Type: text • Name: resistance
—seal	Dataset	Information about seal used for recording. • Quantity: 0 or 1 • Data Type: text • Name: seal
—slice	Dataset	Information about slice used for recording. • Quantity: 0 or 1 • Data Type: text • Name: slice
—device	Link	Device that was used to record from this electrode. • Target Type <i>Device</i> • Name: device

Table 4.90: Groups contained in <IntracellularElectrode>

Id	Type	Description
<IntracellularElectrode>	Group	Top level Group for <IntracellularElectrode> • Neurodata Type: <i>IntracellularElectrode</i> • Extends: <i>NWBContainer</i>
—device	Link	Device that was used to record from this electrode. • Target Type <i>Device</i> • Name: device

4.9.8 SweepTable

Overview: [DEPRECATED] Table used to group different PatchClampSeries. SweepTable is being replaced by IntracellularRecordingsTable and SimultaneousRecordingsTable tables. Additional SequentialRecordingsTable, RepetitionsTable, and ExperimentalConditions tables provide enhanced support for experiment metadata.

SweepTable extends DynamicTable and includes all elements of DynamicTable with the following additions or changes.

- **Extends:** DynamicTable
- **Primitive Type:** Group
- **Inherits from:** DynamicTable, Container
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.8

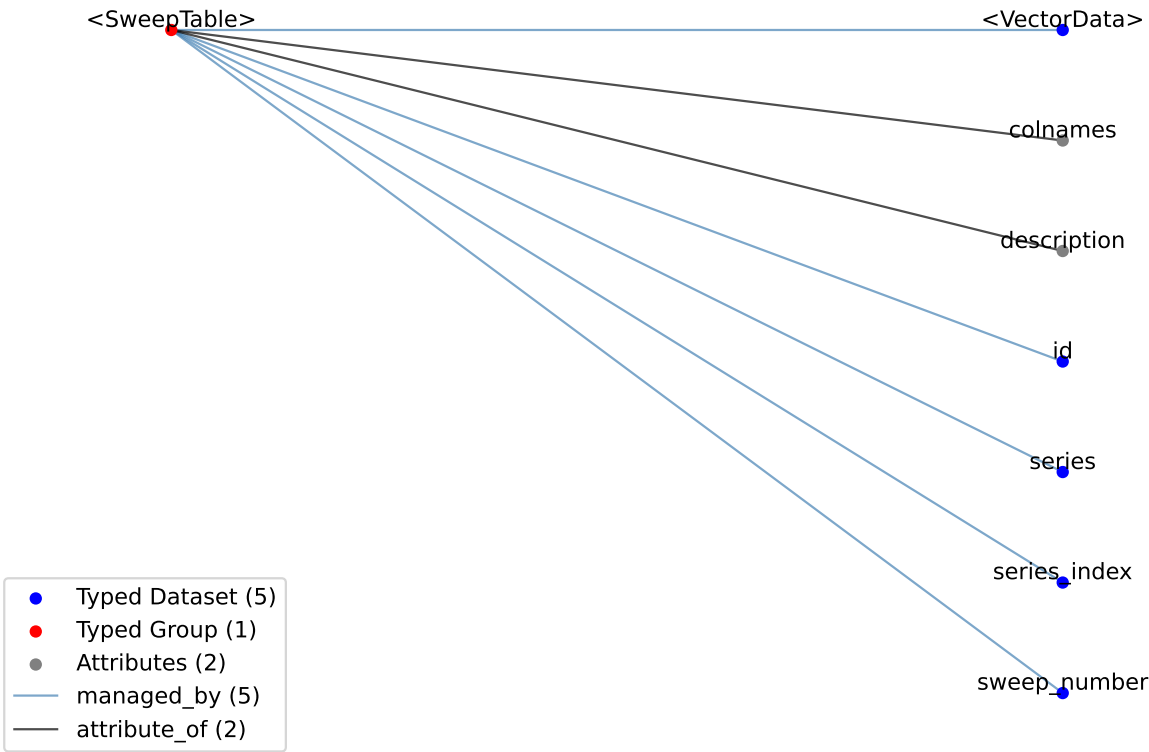


Table 4.91: Datasets, Links, and Attributes contained in <SweepTable>

Id	Type	Description
<SweepTable>	Group	Top level Group for <SweepTable> • Neurodata Type: <i>SweepTable</i> • Extends: <i>DynamicTable</i>

continues on next page

Table 4.91 – continued from previous page

Id	Type	Description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: description
—sweep_number	Dataset	Sweep number of the PatchClampSeries in that row. • Extends: VectorData • Data Type: uint32 • Name: sweep_number
—series	Dataset	The PatchClampSeries with the sweep number in that row. • Extends: VectorData • Data Type: object reference to <i>PatchClampSeries</i> • Name: series
—series_index	Dataset	Index for series. • Extends: VectorIndex • Name: series_index
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more

4.9.9 IntracellularElectrodesTable

Overview: Table for storing intracellular electrode related metadata.

IntracellularElectrodesTable extends DynamicTable and includes all elements of DynamicTable with the following additions or changes.

- **Extends:** DynamicTable
- **Primitive Type:** Group
- **Inherits from:** DynamicTable, Container
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.9

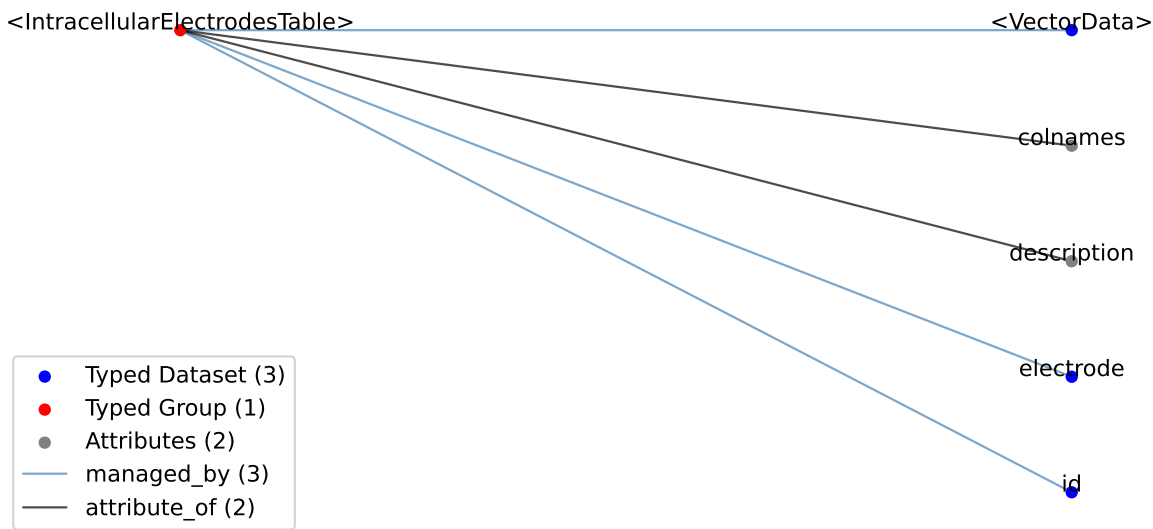


Table 4.92: Datasets, Links, and Attributes contained in <IntracellularElectrodesTable>

Id	Type	Description
<IntracellularElectrodesTable>	Group	Top level Group for <IntracellularElectrodesTable> • Neurodata Type: <i>IntracellularElectrodesTable</i> • Extends: DynamicTable
—description	Attribute	Description of what is in this dynamic table. • Data Type: text • Value: Table for storing intracellular electrode related metadata. • Name: description
—colnames	Attribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames

continues on next page

Table 4.92 – continued from previous page

Id	Type	Description
—electrode	Dataset	Column for storing the reference to the intracellular electrode. • Extends: VectorData • Data Type: object reference to <i>IntracellularElectrode</i> • Name: electrode
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more

4.9.10 IntracellularStimuliTable

Overview: Table for storing intracellular stimulus related metadata.

IntracellularStimuliTable extends DynamicTable and includes all elements of DynamicTable with the following additions or changes.

- **Extends:** DynamicTable
- **Primitive Type:** Group
- **Inherits from:** DynamicTable, Container
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.10

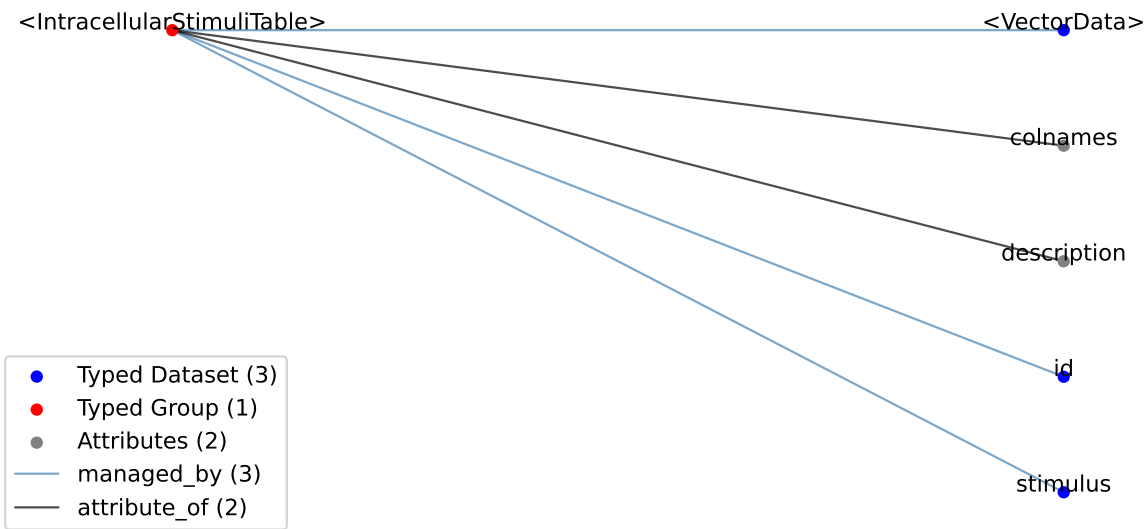


Table 4.93: Datasets, Links, and Attributes contained in <IntracellularStimuliTable>

Id	Type	Description
<IntracellularStimuliTable>	Group	Top level Group for <IntracellularStimuliTable> • Neurodata Type: <i>IntracellularStimuliTable</i> • Extends: DynamicTable
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Value: Table for storing intracellular stimulus related metadata. • Name: description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames

continues on next page

Table 4.93 – continued from previous page

Id	Type	Description
—stimulus	Dataset	Column storing the reference to the recorded stimulus for the recording (rows). • Extends: <i>TimeSeriesReferenceVectorData</i> • Name: stimulus
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: <i>ElementIdentifiers</i> • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< <i>VectorData</i> >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: <i>VectorData</i> • Quantity: 0 or more

4.9.11 IntracellularResponsesTable

Overview: Table for storing intracellular response related metadata.

IntracellularResponsesTable extends DynamicTable and includes all elements of DynamicTable with the following additions or changes.

- **Extends:** DynamicTable
- **Primitive Type:** Group
- **Inherits from:** DynamicTable, Container
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.11

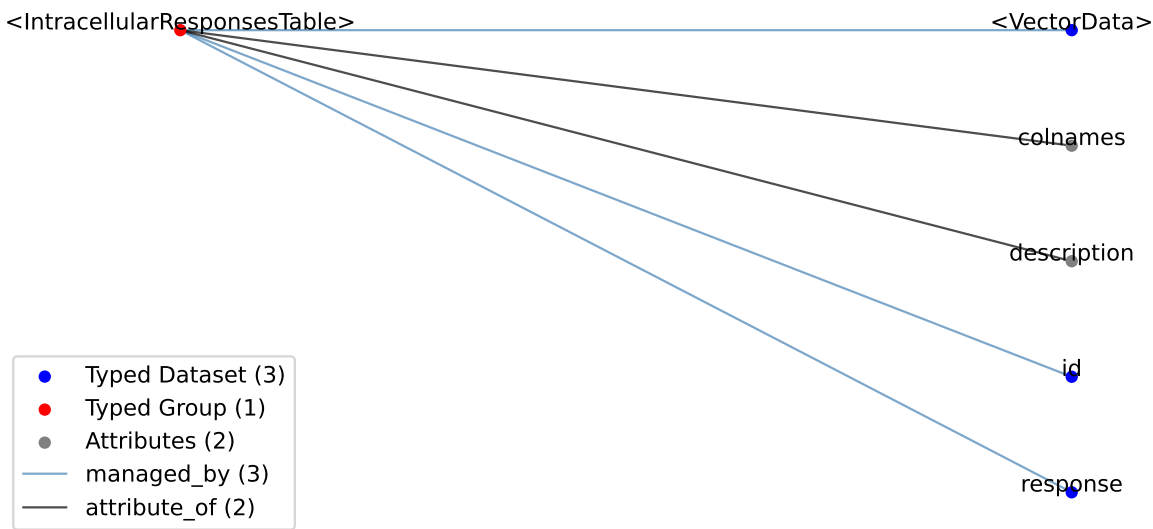


Table 4.94: Datasets, Links, and Attributes contained in <IntracellularResponsesTable>

Id	Type	Description
<IntracellularResponses-Table>	Group	Top level Group for <IntracellularResponsesTable> • Neurodata Type: <i>IntracellularResponsesTable</i> • Extends: DynamicTable
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Value: Table for storing intracellular response related metadata. • Name: description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames

continues on next page

Table 4.94 – continued from previous page

Id	Type	Description
—response	Dataset	Column storing the reference to the recorded response for the recording (rows) • Extends: <i>TimeSeriesReferenceVectorData</i> • Name: response
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: <i>ElementIdentifiers</i> • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< <i>VectorData</i> >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: <i>VectorData</i> • Quantity: 0 or more

4.9.12 IntracellularRecordingsTable

Overview: A table to group together a stimulus and response from a single electrode and a single simultaneous recording. Each row in the table represents a single recording consisting typically of a stimulus and a corresponding response. In some cases, however, only a stimulus or a response is recorded as part of an experiment. In this case, both the stimulus and response will point to the same TimeSeries while the `idx_start` and `count` of the `invalid` column will be set to -1, thus, indicating that no values have been recorded for the stimulus or response, respectively. Note, a recording **MUST** contain at least a stimulus or a response. Typically the stimulus and response are `PatchClampSeries`. However, the use of AD/DA channels that are not associated to an electrode is also common in intracellular electrophysiology, in which case other TimeSeries may be used.

`IntracellularRecordingsTable` extends `AlignedDynamicTable` and includes all elements of `AlignedDynamicTable` with the following additions or changes.

- **Extends:** `AlignedDynamicTable`
- **Primitive Type:** Group
- **Name:** `intracellular_recordings`
- **Inherits from:** `AlignedDynamicTable`, `DynamicTable`, `Container`
- **Source filename:** `nwb.icephys.yaml`
- **Source Specification:** see [Section 5.10.12](#)

Table 4.95: Datasets, Links, and Attributes contained in `<IntracellularRecordingsTable>`

Id	Type	Description
<code>intracellular_recordings</code>	Group	Top level Group for <code>intracellular_recordings</code> • Neurodata Type: <i><code>IntracellularRecordingsTable</code></i> • Extends: <code>AlignedDynamicTable</code> • Name: <code>intracellular_recordings</code>
<code>—description</code>	Attribute	Description of the contents of this table. Inherited from <code>AlignedDynamicTable</code> and overwritten here to fix the value of the attribute. • Data Type: text • Value: A table to group together a stimulus and response from a single electrode and a single simultaneous recording and for storing metadata about the intracellular recording. • Name: <code>description</code>
<code>—categories</code>	Attribute	The names of the categories in this <code>AlignedDynamicTable</code> . Each category is represented by one <code>DynamicTable</code> stored in the parent group. This attribute should be used to specify an order of categories and the category names must match the names of the corresponding <code>DynamicTable</code> in the group. • Data Type: text • Dimensions: [<code>'num_categories'</code>] • Shape: [None] • Name: <code>categories</code>

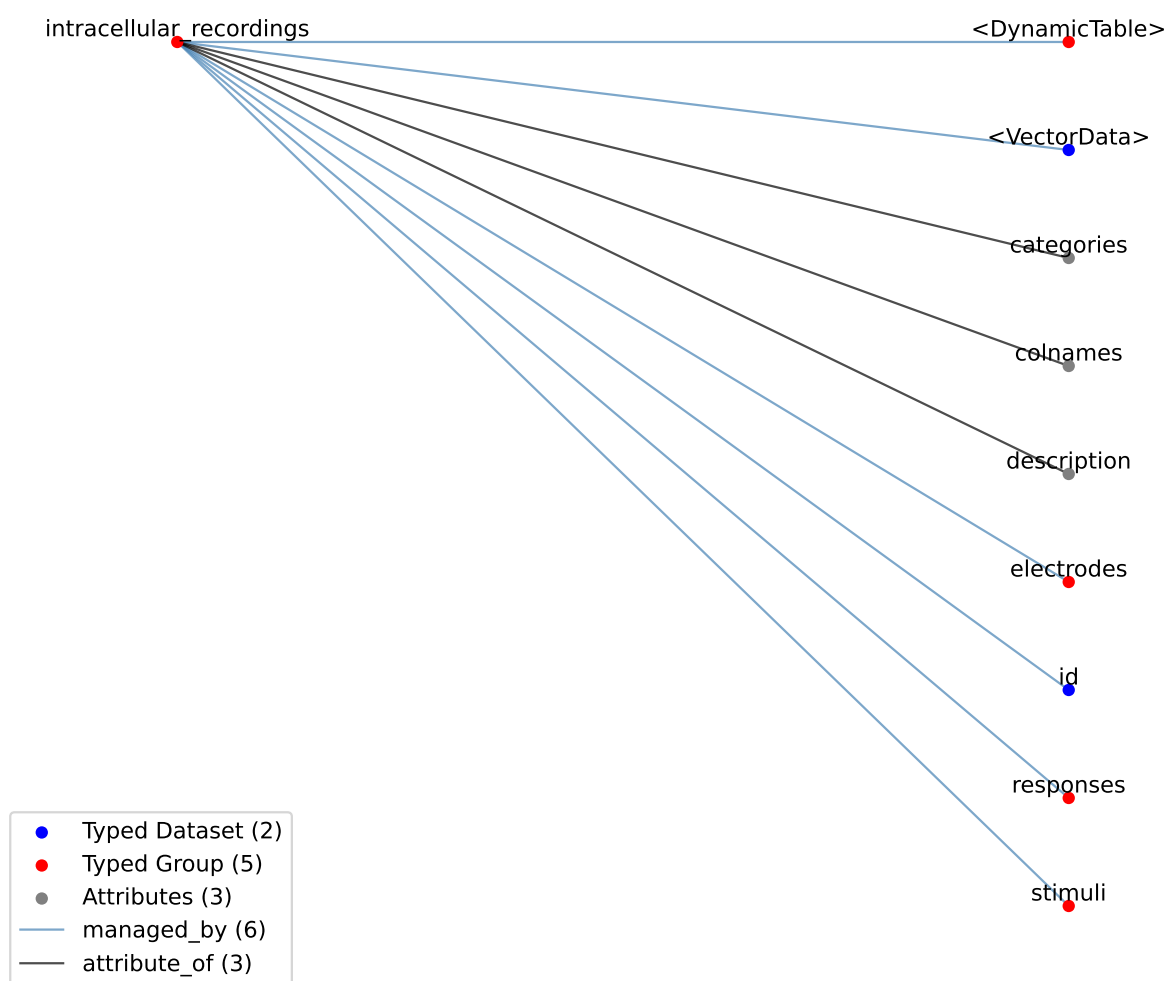
continues on next page

Table 4.95 – continued from previous page

Id	Type	Description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more

Table 4.96: Groups contained in <IntracellularRecordingsTable>

Id	Type	Description
intracellular_recordings	Group	Top level Group for intracellular_recordings • Neurodata Type: <i>IntracellularRecordingsTable</i> • Extends: <i>AlignedDynamicTable</i> • Name: intracellular_recordings
—electrodes	Group	Table for storing intracellular electrode related metadata. • Extends: <i>IntracellularElectrodesTable</i> • Name: electrodes
—stimuli	Group	Table for storing intracellular stimulus related metadata. • Extends: <i>IntracellularStimuliTable</i> • Name: stimuli
—responses	Group	Table for storing intracellular response related metadata. • Extends: <i>IntracellularResponsesTable</i> • Name: responses
—< DynamicTable >	Group	A DynamicTable representing a particular category for columns in the AlignedDynamicTable parent container. The table MUST be aligned with (i.e., have the same number of rows) as all other DynamicTables stored in the AlignedDynamicTable parent container. The name of the category is given by the name of the DynamicTable and its description by the description attribute of the DynamicTable. • Extends: DynamicTable • Quantity: 0 or more



4.9.12.1 Groups: electrodes

Table for storing intracellular electrode related metadata.

- **Extends:** *IntracellularElectrodesTable*
- **Name:** electrodes

4.9.12.2 Groups: stimuli

Table for storing intracellular stimulus related metadata.

- **Extends:** *IntracellularStimuliTable*
- **Name:** stimuli

4.9.12.3 Groups: responses

Table for storing intracellular response related metadata.

- **Extends:** *IntracellularResponsesTable*
- **Name:** responses

4.9.12.4 Groups: <DynamicTable>

A DynamicTable representing a particular category for columns in the AlignedDynamicTable parent container. The table MUST be aligned with (i.e., have the same number of rows) as all other DynamicTables stored in the AlignedDynamicTable parent container. The name of the category is given by the name of the DynamicTable and its description by the description attribute of the DynamicTable.

- **Extends:** *DynamicTable*
- **Quantity:** 0 or more

4.9.13 SimultaneousRecordingsTable

Overview: A table for grouping different intracellular recordings from the IntracellularRecordingsTable table together that were recorded simultaneously from different electrodes.

SimultaneousRecordingsTable extends DynamicTable and includes all elements of DynamicTable with the following additions or changes.

- **Extends:** DynamicTable
- **Primitive Type:** Group
- **Name:** simultaneous_recordings
- **Inherits from:** DynamicTable, Container
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.13

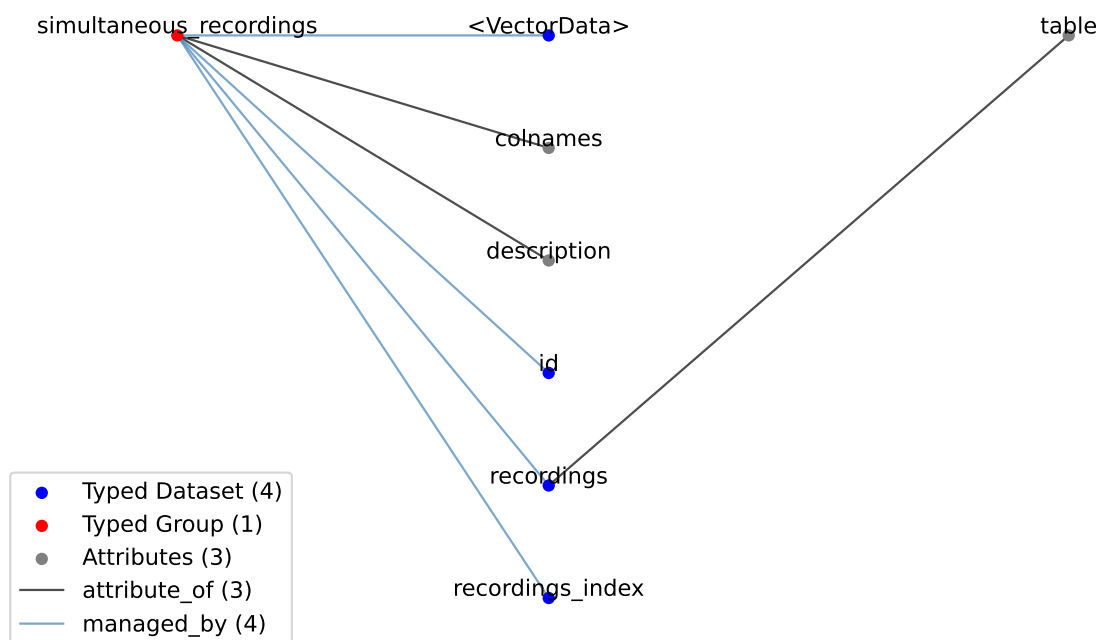


Table 4.97: Datasets, Links, and Attributes contained in <SimultaneousRecordingsTable>

Id	Type	Description
simultaneous_recordings	Group	Top level Group for simultaneous_recordings • Neurodata Type: <i>SimultaneousRecordingsTable</i> • Extends: DynamicTable • Name: simultaneous_recordings

continues on next page

Table 4.97 – continued from previous page

Id	Type	Description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: description
—recordings	Dataset	A reference to one or more rows in the IntracellularRecordingsTable table. • Extends: DynamicTableRegion • Name: recordings
—table	At-tribute	Reference to the IntracellularRecordingsTable table that this table region applies to. This specializes the attribute inherited from DynamicTableRegion to fix the type of table that can be referenced here. • Data Type: object reference to IntracellularRecordingsTable • Name: table
—recordings_index	Dataset	Index dataset for the recordings column. • Extends: VectorIndex • Name: recordings_index
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more

4.9.14 SequentialRecordingsTable

Overview: A table for grouping different sequential recordings from the SimultaneousRecordingsTable table together. This is typically used to group together sequential recordings where a sequence of stimuli of the same type with varying parameters have been presented in a sequence.

SequentialRecordingsTable extends DynamicTable and includes all elements of [DynamicTable](#) with the following additions or changes.

- **Extends:** [DynamicTable](#)
- **Primitive Type:** Group
- **Name:** sequential_recordings
- **Inherits from:** [DynamicTable](#), [Container](#)
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see [Section 5.10.14](#)

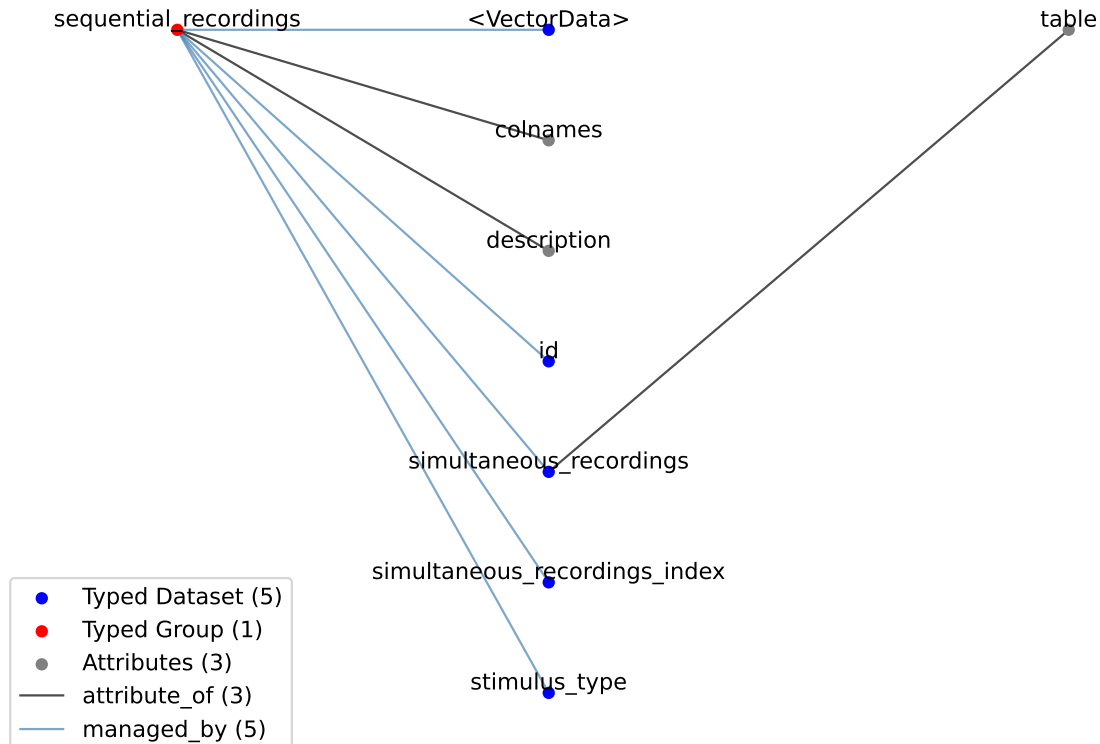


Table 4.98: Datasets, Links, and Attributes contained in <SequentialRecordingsTable>

Id	Type	Description
sequential_recordings	Group	Top level Group for sequential_recordings • Neurodata Type: <i>SequentialRecordingsTable</i> • Extends: <i>DynamicTable</i> • Name: sequential_recordings
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: description
—simultaneous_recordings	Dataset	A reference to one or more rows in the SimultaneousRecordingsTable table. • Extends: <i>DynamicTableRegion</i> • Name: simultaneous_recordings
—table	At-tribute	Reference to the SimultaneousRecordingsTable table that this table region applies to. This specializes the attribute inherited from DynamicTableRegion to fix the type of table that can be referenced here. • Data Type: object reference to <i>SimultaneousRecordingsTable</i> • Name: table
—simultaneous_recordings_index	Dataset	Index dataset for the simultaneous_recordings column. • Extends: <i>VectorIndex</i> • Name: simultaneous_recordings_index
—stimulus_type	Dataset	The type of stimulus used for the sequential recording. • Extends: <i>VectorData</i> • Data Type: text • Name: stimulus_type
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: <i>ElementIdentifiers</i> • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< <i>VectorData</i> >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: <i>VectorData</i> • Quantity: 0 or more

4.9.15 RepetitionsTable

Overview: A table for grouping different sequential intracellular recordings together. With each SequentialRecording typically representing a particular type of stimulus, the RepetitionsTable table is typically used to group sets of stimuli applied in sequence.

RepetitionsTable extends DynamicTable and includes all elements of [DynamicTable](#) with the following additions or changes.

- **Extends:** [DynamicTable](#)
- **Primitive Type:** Group
- **Name:** repetitions
- **Inherits from:** [DynamicTable](#), [Container](#)
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see [Section 5.10.15](#)

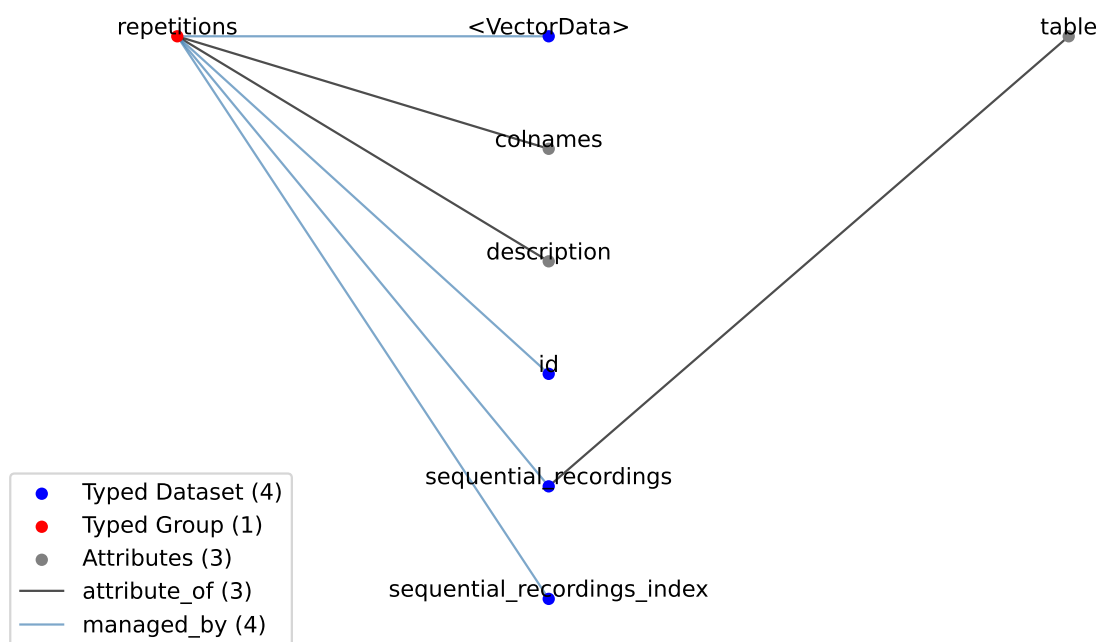


Table 4.99: Datasets, Links, and Attributes contained in <RepetitionsTable>

Id	Type	Description
repetitions	Group	Top level Group for repetitions • Neurodata Type: <i>RepetitionsTable</i> • Extends: DynamicTable • Name: repetitions

continues on next page

Table 4.99 – continued from previous page

Id	Type	Description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: description
—sequential_recordings	Dataset	A reference to one or more rows in the SequentialRecordingsTable table. • Extends: DynamicTableRegion • Name: sequential_recordings
—table	At-tribute	Reference to the SequentialRecordingsTable table that this table region applies to. This specializes the attribute inherited from DynamicTableRegion to fix the type of table that can be referenced here. • Data Type: object reference to SequentialRecordingsTable • Name: table
—sequential_recordings_index	Dataset	Index dataset for the sequential_recordings column. • Extends: VectorIndex • Name: sequential_recordings_index
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more

4.9.16 ExperimentalConditionsTable

Overview: A table for grouping different intracellular recording repetitions together that belong to the same experimental condition.

ExperimentalConditionsTable extends DynamicTable and includes all elements of DynamicTable with the following additions or changes.

- **Extends:** DynamicTable
- **Primitive Type:** Group
- **Name:** experimental_conditions
- **Inherits from:** DynamicTable, Container
- **Source filename:** nwb.icephys.yaml
- **Source Specification:** see Section 5.10.16

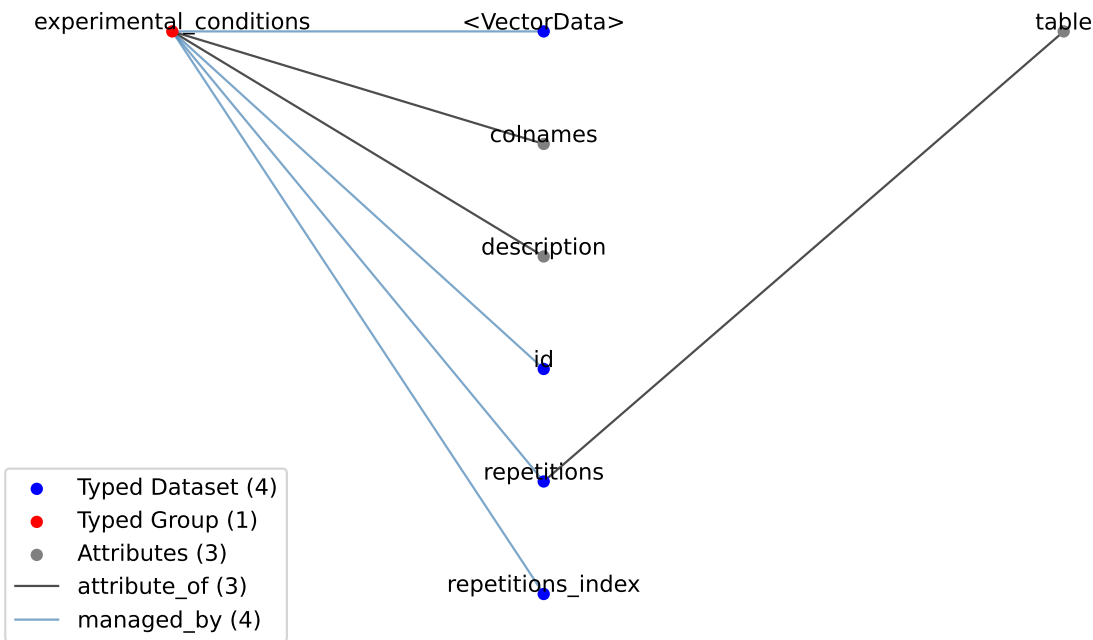


Table 4.100: Datasets, Links, and Attributes contained in <ExperimentalConditionsTable>

Id	Type	Description
experimental_conditions	Group	Top level Group for experimental_conditions • Neurodata Type: <i>ExperimentalConditionsTable</i> • Extends: DynamicTable • Name: experimental_conditions

continues on next page

Table 4.100 – continued from previous page

Id	Type	Description
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: description
—repetitions	Dataset	A reference to one or more rows in the RepetitionsTable table. • Extends: DynamicTableRegion • Name: repetitions
—table	At-tribute	Reference to the RepetitionsTable table that this table region applies to. This specializes the attribute inherited from DynamicTableRegion to fix the type of table that can be referenced here. • Data Type: object reference to RepetitionsTable • Name: table
—repetitions_index	Dataset	Index dataset for the repetitions column. • Extends: VectorIndex • Name: repetitions_index
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more

4.10 Optogenetics

This source module contains neurodata_types for opto-genetics data.

4.10.1 OptogeneticSeries

Overview: An optogenetic stimulus.

OptogeneticSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ogen.yaml
- **Source Specification:** see [Section 5.11.1](#)

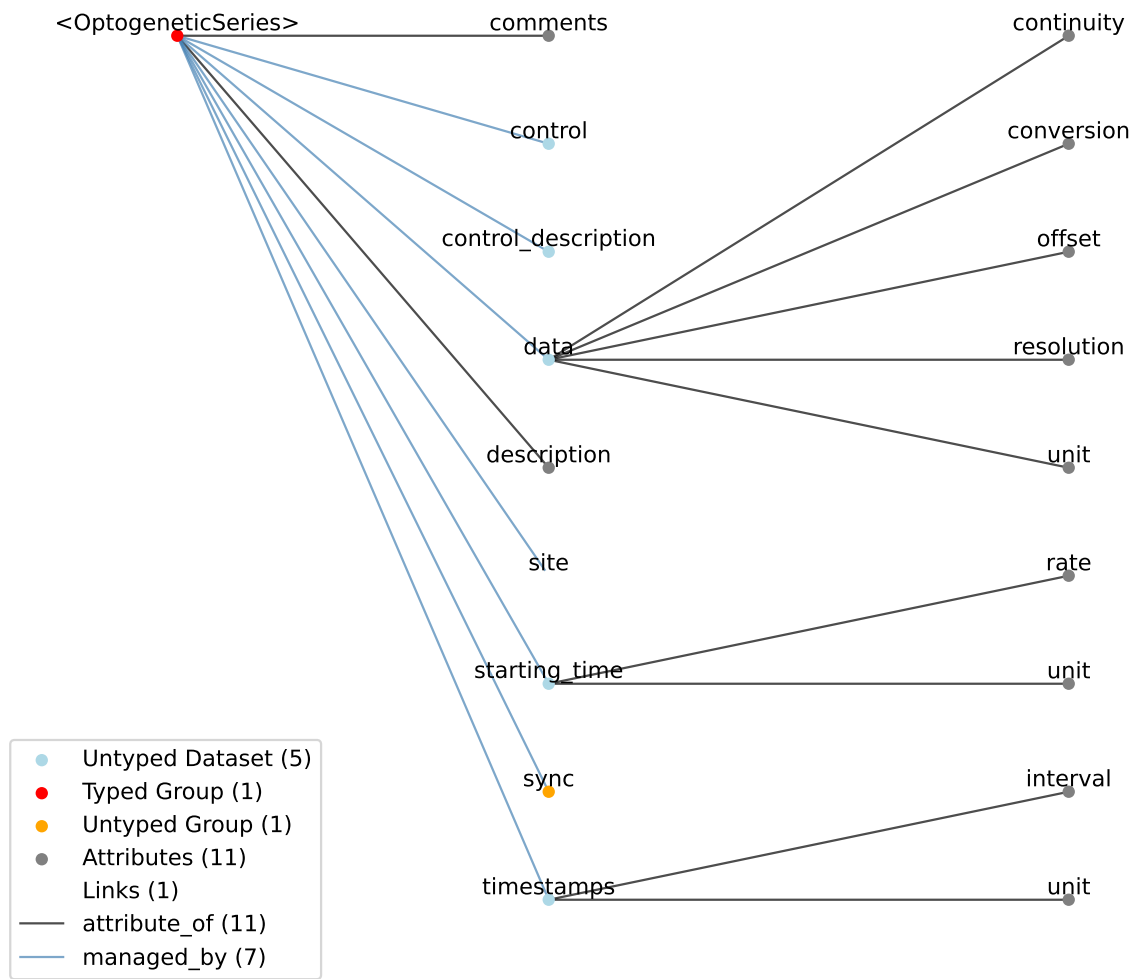


Table 4.101: Datasets, Links, and Attributes contained in <OptogeneticSeries>

Id	Type	Description
<OptogeneticSeries>	Group	Top level Group for <OptogeneticSeries> • Neurodata Type: <i>OptogeneticSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Applied power for optogenetic stimulus, in watts. • Data Type: numeric • Dimensions: ['num_times'] • Shape: [None] • Name: data
—unit	At-tribute	Unit of measurement for data, which is fixed to 'watts'. • Data Type: text • Value: watts • Name: unit
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion

continues on next page

Table 4.101 – continued from previous page

Id	Type	Description
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.101 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
—interval	Attribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
—unit	Attribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—site	Link	Link to OptogeneticStimulusSite object that describes the site to which this stimulus was applied. • Target Type <i>OptogeneticStimulusSite</i> • Name: site

Table 4.102: Groups contained in <OptogeneticSeries>

Id	Type	Description
<OptogeneticSeries>	Group	Top level Group for <OptogeneticSeries> • Neurodata Type: <i>OptogeneticSeries</i> • Extends: <i>TimeSeries</i>
—site	Link	Link to OptogeneticStimulusSite object that describes the site to which this stimulus was applied. • Target Type <i>OptogeneticStimulusSite</i> • Name: site
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.10.1.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.10.2 OptogeneticStimulusSite

Overview: A site of optogenetic stimulation.

OptogeneticStimulusSite extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.ogen.yaml
- **Source Specification:** see Section 5.11.2

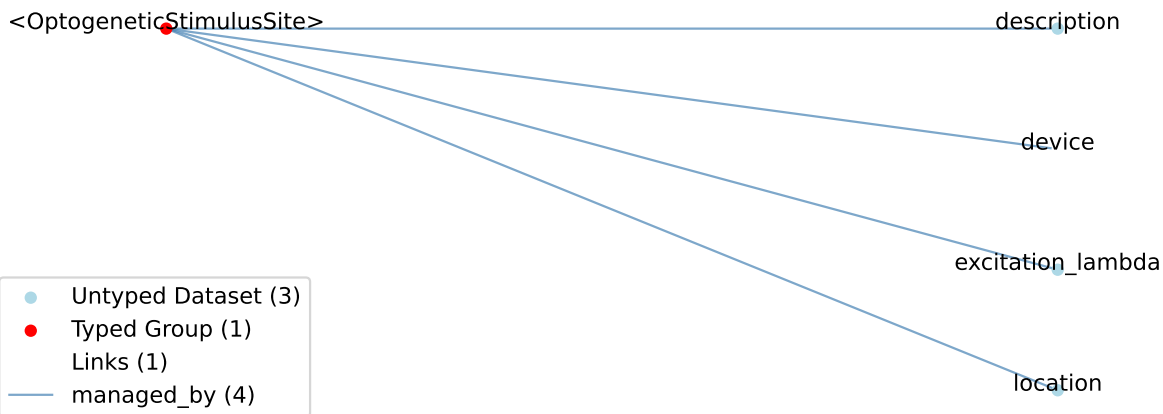


Table 4.103: Datasets, Links, and Attributes contained in <OptogeneticStimulusSite>

Id	Type	Description
<OptogeneticStimulusSite>	Group	Top level Group for <OptogeneticStimulusSite> • Neurodata Type: <i>OptogeneticStimulusSite</i> • Extends: <i>NWBContainer</i>
—description	Dataset	Description of stimulation site. • Data Type: text • Name: description
—excitation_lambda	Dataset	Excitation wavelength, in nm. • Data Type: float32 • Name: excitation_lambda
—location	Dataset	Location of the stimulation site. Specify the area, layer, comments on estimation of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names for anatomical regions when possible. • Data Type: text • Name: location

continues on next page

Table 4.103 – continued from previous page

Id	Type	Description
—device	Link	Device that generated the stimulus. • Target Type <i>Device</i> • Name: device

Table 4.104: Groups contained in <OptogeneticStimulusSite>

Id	Type	Description
<OptogeneticStimulusSite>	Group	Top level Group for <OptogeneticStimulusSite> • Neurodata Type: <i>OptogeneticStimulusSite</i> • Extends: <i>NWBContainer</i>
—device	Link	Device that generated the stimulus. • Target Type <i>Device</i> • Name: device

4.11 Optical physiology

This source module contains neurodata_types for optical physiology data.

4.11.1 OnePhotonSeries

Overview: Image stack recorded over time from 1-photon microscope.

OnePhotonSeries extends ImageSeries and includes all elements of *ImageSeries* with the following additions or changes.

- **Extends:** *ImageSeries*
- **Primitive Type:** Group
- **Inherits from:** *ImageSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.1](#)

Table 4.105: Datasets, Links, and Attributes contained in <OnePhotonSeries>

Id	Type	Description
<OnePhotonSeries>	Group	Top level Group for <OnePhotonSeries> • Neurodata Type: <i>OnePhotonSeries</i> • Extends: <i>ImageSeries</i>
—pmt_gain	At-tribute	Photomultiplier gain. • Data Type: float32 • Required: False • Name: pmt_gain
—scan_line_rate	At-tribute	Lines imaged per second. This is also stored in /general/optophysiology but is kept here as it is useful information for analysis, and so good to be stored w/ the actual data. • Data Type: float32 • Required: False • Name: scan_line_rate
—exposure_time	At-tribute	Exposure time of the sample; often the inverse of the frequency. • Data Type: float32 • Required: False • Name: exposure_time
—binning	At-tribute	Amount of pixels combined into ‘bins’; could be 1, 2, 4, 8, etc. • Data Type: uint8 • Required: False • Name: binning

continues on next page

Table 4.105 – continued from previous page

Id	Type	Description
—power	At-tribute	Power of the excitation in mW, if known. • Data Type: float32 • Required: False • Name: power
—intensity	At-tribute	Intensity of the excitation in mW/mm ² , if known. • Data Type: float32 • Required: False • Name: intensity
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Binary data representing images across frames. If data are stored in an external file, this should be an empty 3D array. • Data Type: numeric • Dimensions: [['frame', 'x', 'y'], ['frame', 'x', 'y', 'z']] • Shape: [[None, None, None], [None, None, None, None]] • Name: data
—conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion

continues on next page

Table 4.105 – continued from previous page

Id	Type	Description
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Name: unit
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
——dimension	Dataset	Number of pixels on x, y, (and z) axes. • Quantity: 0 or 1 • Data Type: int32 • Dimensions: [‘rank’] • Shape: [None] • Name: dimension

continues on next page

Table 4.105 – continued from previous page

Id	Type	Description
—external_file	Dataset	Paths to one or more external file(s). The field is only present if format='external'. This is only relevant if the image series is stored in the file system as one or more image file(s). This field should NOT be used if the image is stored in another NWB file and that file is linked to this file. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_files'] • Shape: [None] • Name: external_file
—starting_frame	Attribute	Each external image may contain one or more consecutive frames of the full ImageSeries. This attribute serves as an index to indicate which frames each file contains, to facilitate random access. The 'starting_frame' attribute, hence, contains a list of frame numbers within the full ImageSeries of the first frame of each file listed in the parent 'external_file' dataset. Zero-based indexing is used (hence, the first element will always be zero). For example, if the 'external_file' dataset has three paths to files and the first file has 5 frames, the second file has 10 frames, and the third file has 20 frames, then this attribute will have values [0, 5, 15]. If there is a single external file that holds all of the frames of the ImageSeries (and so there is a single element in the 'external_file' dataset), then this attribute should have value [0]. • Data Type: int32 • Dimensions: ['num_files'] • Shape: [None] • Name: starting_frame
—format	Dataset	Format of image. If this is 'external', then the attribute 'external_file' contains the path information to the image files. If this is 'raw', then the raw (single-channel) binary data is stored in the 'data' dataset. If this attribute is not present, then the default format='raw' case is assumed. • Quantity: 0 or 1 • Data Type: text • Name: format
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
—rate	Attribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate

continues on next page

Table 4.105 – continued from previous page

Id	Type	Description
—unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
—interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
—unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: [‘num_times’] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: [‘num_control_values’] • Shape: [None] • Name: control_description
—imaging_plane	Link	Link to ImagingPlane object from which this TimeSeries data was generated. • Target Type <i>ImagingPlane</i> • Name: imaging_plane

continues on next page

Table 4.105 – continued from previous page

Id	Type	Description
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

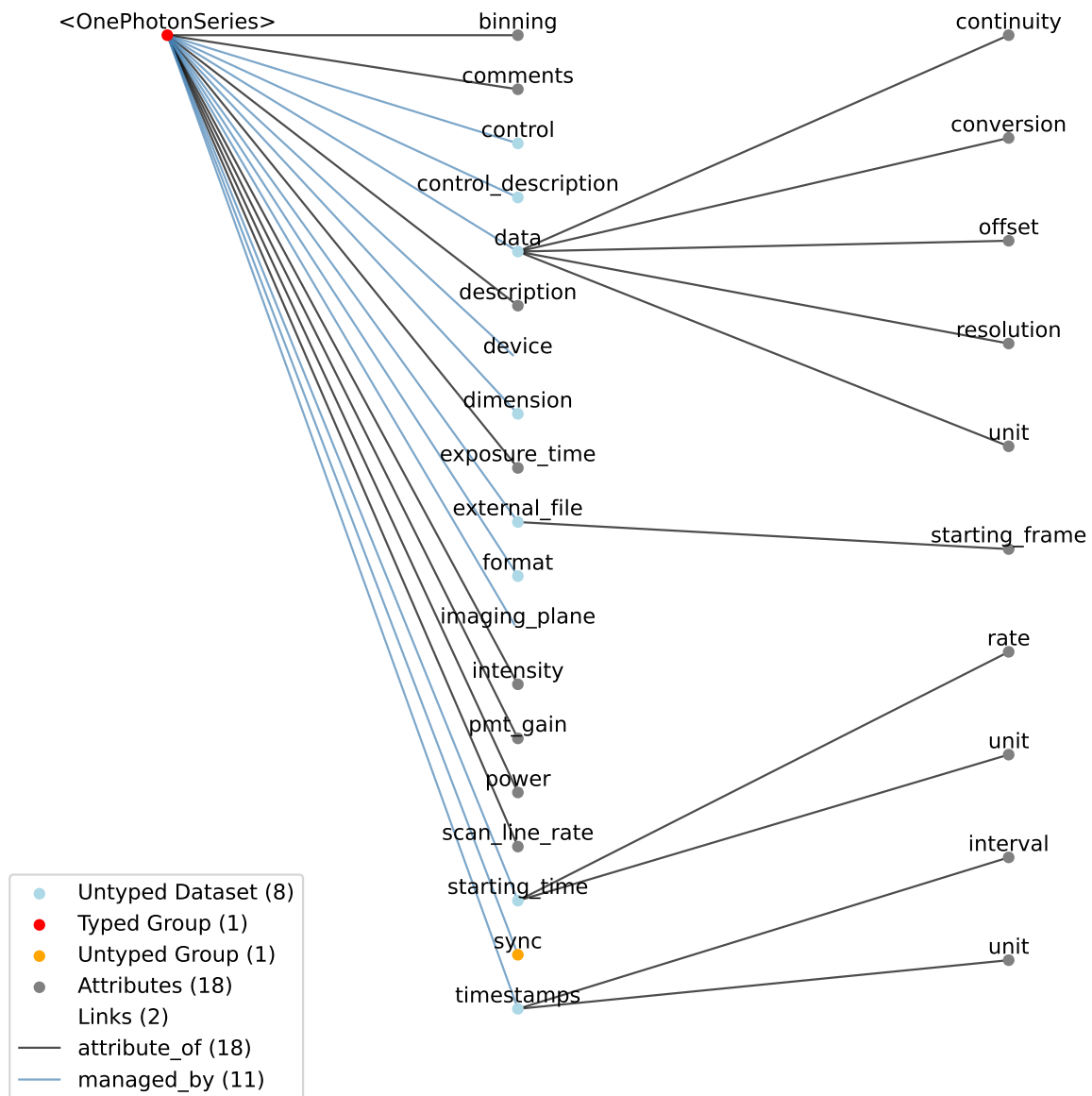
Table 4.106: Groups contained in <OnePhotonSeries>

Id	Type	Description
<OnePhotonSeries>	Group	Top level Group for <OnePhotonSeries> • Neurodata Type: <i>OnePhotonSeries</i> • Extends: <i>ImageSeries</i>
—imaging_plane	Link	Link to ImagingPlane object from which this TimeSeries data was generated. • Target Type <i>ImagingPlane</i> • Name: imaging_plane
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.11.1.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync



4.11.2 TwoPhotonSeries

Overview: Image stack recorded over time from 2-photon microscope.

TwoPhotonSeries extends ImageSeries and includes all elements of *ImageSeries* with the following additions or changes.

- **Extends:** *ImageSeries*
- **Primitive Type:** Group
- **Inherits from:** *ImageSeries*, *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.2](#)

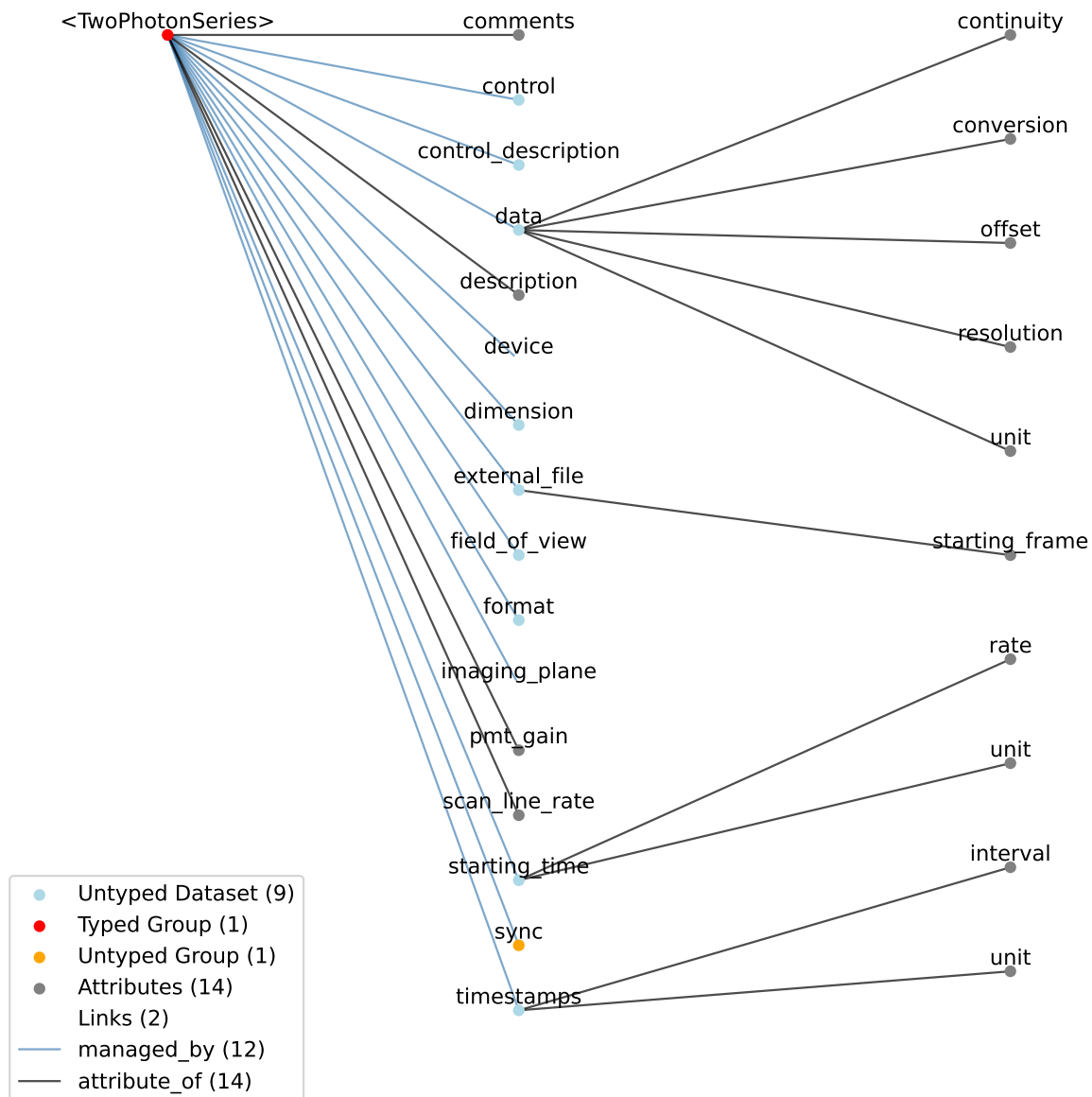


Table 4.107: Datasets, Links, and Attributes contained in <TwoPhotonSeries>

Id	Type	Description
<TwoPhotonSeries>	Group	Top level Group for <TwoPhotonSeries> • Neurodata Type: <i>TwoPhotonSeries</i> • Extends: <i>ImageSeries</i>
—pmt_gain	At-tribute	Photomultiplier gain. • Data Type: float32 • Required: False • Name: pmt_gain
—scan_line_rate	At-tribute	Lines imaged per second. This is also stored in /general/optophysiology but is kept here as it is useful information for analysis, and so good to be stored w/ the actual data. • Data Type: float32 • Required: False • Name: scan_line_rate
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—field_of_view	Dataset	Width, height and depth of image, or imaged area, in meters. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['width height'], ['width height depth']] • Shape: [[2], [3]] • Name: field_of_view
—data	Dataset	Binary data representing images across frames. If data are stored in an external file, this should be an empty 3D array. • Data Type: numeric • Dimensions: [['frame', 'x', 'y'], ['frame', 'x', 'y', 'z']] • Shape: [[None, None, None], [None, None, None, None]] • Name: data

continues on next page

Table 4.107 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the ‘conversion’ multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by ‘conversion’ to finalize its coercion to the specified ‘unit’. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Name: unit

continues on next page

Table 4.107 – continued from previous page

Id	Type	Description
—continuity	Attribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—dimension	Dataset	Number of pixels on x, y, (and z) axes. • Quantity: 0 or 1 • Data Type: int32 • Dimensions: ['rank'] • Shape: [None] • Name: dimension
—external_file	Dataset	Paths to one or more external file(s). The field is only present if format='external'. This is only relevant if the image series is stored in the file system as one or more image file(s). This field should NOT be used if the image is stored in another NWB file and that file is linked to this file. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_files'] • Shape: [None] • Name: external_file
—starting_frame	Attribute	Each external image may contain one or more consecutive frames of the full ImageSeries. This attribute serves as an index to indicate which frames each file contains, to facilitate random access. The 'starting_frame' attribute, hence, contains a list of frame numbers within the full ImageSeries of the first frame of each file listed in the parent 'external_file' dataset. Zero-based indexing is used (hence, the first element will always be zero). For example, if the 'external_file' dataset has three paths to files and the first file has 5 frames, the second file has 10 frames, and the third file has 20 frames, then this attribute will have values [0, 5, 15]. If there is a single external file that holds all of the frames of the ImageSeries (and so there is a single element in the 'external_file' dataset), then this attribute should have value [0]. • Data Type: int32 • Dimensions: ['num_files'] • Shape: [None] • Name: starting_frame

continues on next page

Table 4.107 – continued from previous page

Id	Type	Description
—format	Dataset	Format of image. If this is ‘external’, then the attribute ‘external_file’ contains the path information to the image files. If this is ‘raw’, then the raw (single-channel) binary data is stored in the ‘data’ dataset. If this attribute is not present, then the default format=‘raw’ case is assumed. • Quantity: 0 or 1 • Data Type: text • Name: format
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: [‘num_times’] • Shape: [None] • Name: timestamps
——interval	At-tribute	Value is ‘1’ • Data Type: int32 • Value: 1 • Name: interval
——unit	At-tribute	Unit of measurement for timestamps, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.107 – continued from previous page

Id	Type	Description
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description
—imaging_plane	Link	Link to ImagingPlane object from which this TimeSeries data was generated. • Target Type <i>ImagingPlane</i> • Name: imaging_plane
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

Table 4.108: Groups contained in <TwoPhotonSeries>

Id	Type	Description
<TwoPhotonSeries>	Group	Top level Group for <TwoPhotonSeries> • Neurodata Type: <i>TwoPhotonSeries</i> • Extends: <i>ImageSeries</i>
—imaging_plane	Link	Link to ImagingPlane object from which this TimeSeries data was generated. • Target Type <i>ImagingPlane</i> • Name: imaging_plane
—device	Link	Link to the Device object that was used to capture these images. • Target Type <i>Device</i> • Name: device

continues on next page

Table 4.108 – continued from previous page

Id	Type	Description
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.11.2.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.11.3 RoiResponseSeries

Overview: ROI responses over an imaging plane. The first dimension represents time. The second dimension, if present, represents ROIs.

RoiResponseSeries extends TimeSeries and includes all elements of *TimeSeries* with the following additions or changes.

- **Extends:** *TimeSeries*
- **Primitive Type:** Group
- **Inherits from:** *TimeSeries*, *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.3](#)

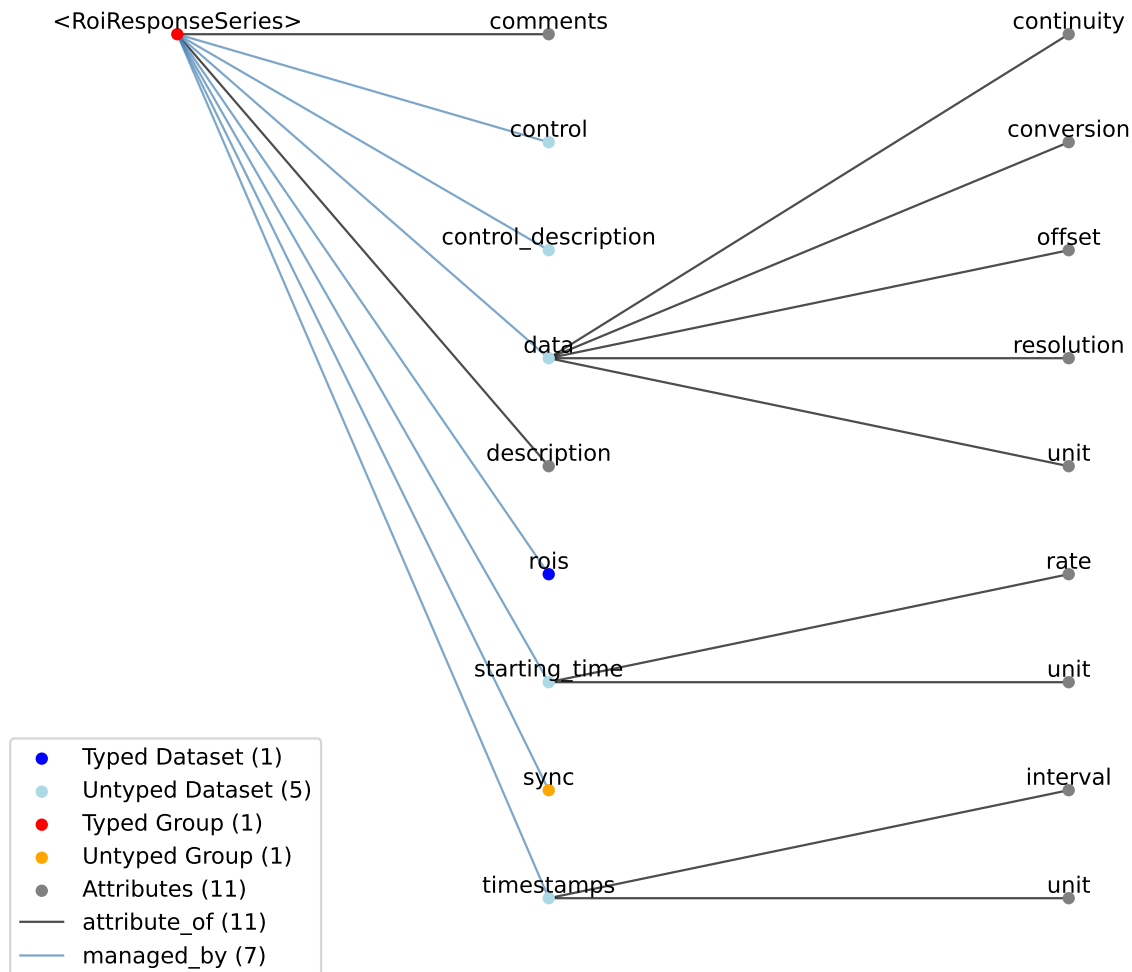


Table 4.109: Datasets, Links, and Attributes contained in <RoiResponseSeries>

Id	Type	Description
<RoiResponseSeries>	Group	Top level Group for <RoiResponseSeries> • Neurodata Type: <i>RoiResponseSeries</i> • Extends: <i>TimeSeries</i>
—description	At-tribute	Description of the time series. • Data Type: text • Required: False • Default Value: no description • Name: description
—comments	At-tribute	Human-readable comments about the TimeSeries. This second descriptive field can be used to store additional information, or descriptive information if the primary description field is populated with a computer-readable string. • Data Type: text • Required: False • Default Value: no comments • Name: comments
—data	Dataset	Signals from ROIs. • Data Type: numeric • Dimensions: [['num_times'], ['num_times', 'num_ROIs']] • Shape: [[None], [None, None]] • Name: data
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified 'unit'. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by 'conversion' to convert the data to the specified 'unit'. e.g. if the data acquisition system stores values in this object as signed 16-bit integers (int16 range -32,768 to 32,767) that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition values to recorded volts is $2.5/32768/8000 = 9.5367e-9$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——offset	At-tribute	Scalar to add to the data after scaling by 'conversion' to finalize its coercion to the specified 'unit'. Two common examples of this include (a) data stored in an unsigned type that requires a shift after scaling to re-center the data, and (b) specialized recording devices that naturally cause a scalar offset with respect to the true units. • Data Type: float32 • Required: False • Default Value: 0.0 • Name: offset

continues on next page

Table 4.109 – continued from previous page

Id	Type	Description
——resolution	At-tribute	Smallest meaningful difference between values in data, stored in the specified by unit, e.g., the change in value of the least significant bit, or a larger number if signal noise is known to be present. If unknown, use -1.0. • Data Type: float32 • Required: False • Default Value: -1.0 • Name: resolution
——unit	At-tribute	Base unit of measurement for working with the data. Actual stored values are not necessarily stored in these units. To access the data in these units, multiply ‘data’ by ‘conversion’ and add ‘offset’. • Data Type: text • Name: unit
——continuity	At-tribute	Optionally describe the continuity of the data. Can be “continuous”, “instantaneous”, or “step”. For example, a voltage trace would be “continuous”, because samples are recorded from a continuous process. An array of lick times would be “instantaneous”, because the data represents distinct moments in time. Times of image presentations would be “step” because the picture remains the same until the next timepoint. This field is optional, but is useful in providing information about the underlying data. It may inform the way this data is interpreted, the way it is visualized, and what analysis methods are applicable. • Data Type: text • Required: False • Name: continuity
—rois	Dataset	DynamicTableRegion referencing into an ROITable containing information on the ROIs stored in this timeseries. • Extends: DynamicTableRegion • Name: rois
—starting_time	Dataset	Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute. • Quantity: 0 or 1 • Data Type: float64 • Name: starting_time
——rate	At-tribute	Sampling rate, in Hz. • Data Type: float32 • Name: rate
——unit	At-tribute	Unit of measurement for time, which is fixed to ‘seconds’. • Data Type: text • Value: seconds • Name: unit

continues on next page

Table 4.109 – continued from previous page

Id	Type	Description
—timestamps	Dataset	Timestamps for samples stored in data, in seconds, relative to the common experiment master-clock stored in NWB-File.timestamps_reference_time. • Quantity: 0 or 1 • Data Type: float64 • Dimensions: ['num_times'] • Shape: [None] • Name: timestamps
—interval	At-tribute	Value is '1' • Data Type: int32 • Value: 1 • Name: interval
—unit	At-tribute	Unit of measurement for timestamps, which is fixed to 'seconds'. • Data Type: text • Value: seconds • Name: unit
—control	Dataset	Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should be the same size as the first dimension of data. • Quantity: 0 or 1 • Data Type: uint8 • Dimensions: ['num_times'] • Shape: [None] • Name: control
—control_description	Dataset	Description of each control value. Must be present if control is present. If present, control_description[0] should describe time points where control == 0. • Quantity: 0 or 1 • Data Type: text • Dimensions: ['num_control_values'] • Shape: [None] • Name: control_description

Table 4.110: Groups contained in <RoiResponseSeries>

Id	Type	Description
<RoiResponseSeries>	Group	Top level Group for <RoiResponseSeries> • Neurodata Type: <i>RoiResponseSeries</i> • Extends: <i>TimeSeries</i>

continues on next page

Table 4.110 – continued from previous page

Id	Type	Description
—sync	Group	Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes. • Quantity: 0 or 1 • Name: sync

4.11.3.1 Groups: sync

Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common timebase. The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of ‘sync’ are mostly for archival purposes.

- **Quantity:** 0 or 1
- **Name:** sync

4.11.4 DfOverF

Overview: dF/F information about a region of interest (ROI). Storage hierarchy of dF/F should be the same as for segmentation (i.e., same names for ROIs and for image planes).

DfOverF extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** DfOverF
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.4](#)

Table 4.111: Groups contained in <DfOverF>

Id	Type	Description
<DfOverF>	Group	Top level Group for <DfOverF> • Neurodata Type: <i>DfOverF</i> • Extends: <i>NWBDataInterface</i> • Default Name: DfOverF
—< <i>RoiResponseSeries</i> >	Group	RoiResponseSeries object(s) containing dF/F for a ROI. • Extends: <i>RoiResponseSeries</i> • Quantity: 1 or more

4.11.4.1 Groups: <RoiResponseSeries>

RoiResponseSeries object(s) containing dF/F for a ROI.

- **Extends:** *RoiResponseSeries*
- **Quantity:** 1 or more

4.11.5 Fluorescence

Overview: Fluorescence information about a region of interest (ROI). Storage hierarchy of fluorescence should be the same as for segmentation (ie, same names for ROIs and for image planes).

Fluorescence extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** Fluorescence
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.5](#)

Table 4.112: Groups contained in <Fluorescence>

Id	Type	Description
<Fluorescence>	Group	Top level Group for <Fluorescence> • Neurodata Type: <i>Fluorescence</i> • Extends: <i>NWBDataInterface</i> • Default Name: Fluorescence
—< <i>RoiResponseSeries</i> >	Group	RoiResponseSeries object(s) containing fluorescence data for a ROI. • Extends: <i>RoiResponseSeries</i> • Quantity: 1 or more

4.11.5.1 Groups: <RoiResponseSeries>

RoiResponseSeries object(s) containing fluorescence data for a ROI.

- **Extends:** *RoiResponseSeries*
- **Quantity:** 1 or more

4.11.6 ImageSegmentation

Overview: Stores pixels in an image that represent different regions of interest (ROIs) or masks. All segmentation for a given imaging plane is stored together, with storage for multiple imaging planes (masks) supported. Each ROI is stored in its own subgroup, with the ROI group containing both a 2D mask and a list of pixels that make up this mask. Segments can also be used for masking neuropil. If segmentation is allowed to change with time, a new imaging plane (or module) is required and ROI names should remain consistent between them.

ImageSegmentation extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** ImageSegmentation
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see Section 5.12.6

Table 4.113: Groups contained in <ImageSegmentation>

Id	Type	Description
<ImageSegmentation>	Group	Top level Group for <ImageSegmentation> • Neurodata Type: <i>ImageSegmentation</i> • Extends: <i>NWBDataInterface</i> • Default Name: ImageSegmentation
—< <i>PlaneSegmentation</i> >	Group	Results from image segmentation of a specific imaging plane. • Extends: <i>PlaneSegmentation</i> • Quantity: 1 or more

4.11.6.1 Groups: <PlaneSegmentation>

Results from image segmentation of a specific imaging plane.

- **Extends:** *PlaneSegmentation*
- **Quantity:** 1 or more

4.11.7 PlaneSegmentation

Overview: Results from image segmentation of a specific imaging plane.

PlaneSegmentation extends DynamicTable and includes all elements of [DynamicTable](#) with the following additions or changes.

- **Extends:** [DynamicTable](#)
- **Primitive Type:** Group
- **Inherits from:** [DynamicTable](#), [Container](#)
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.7](#)

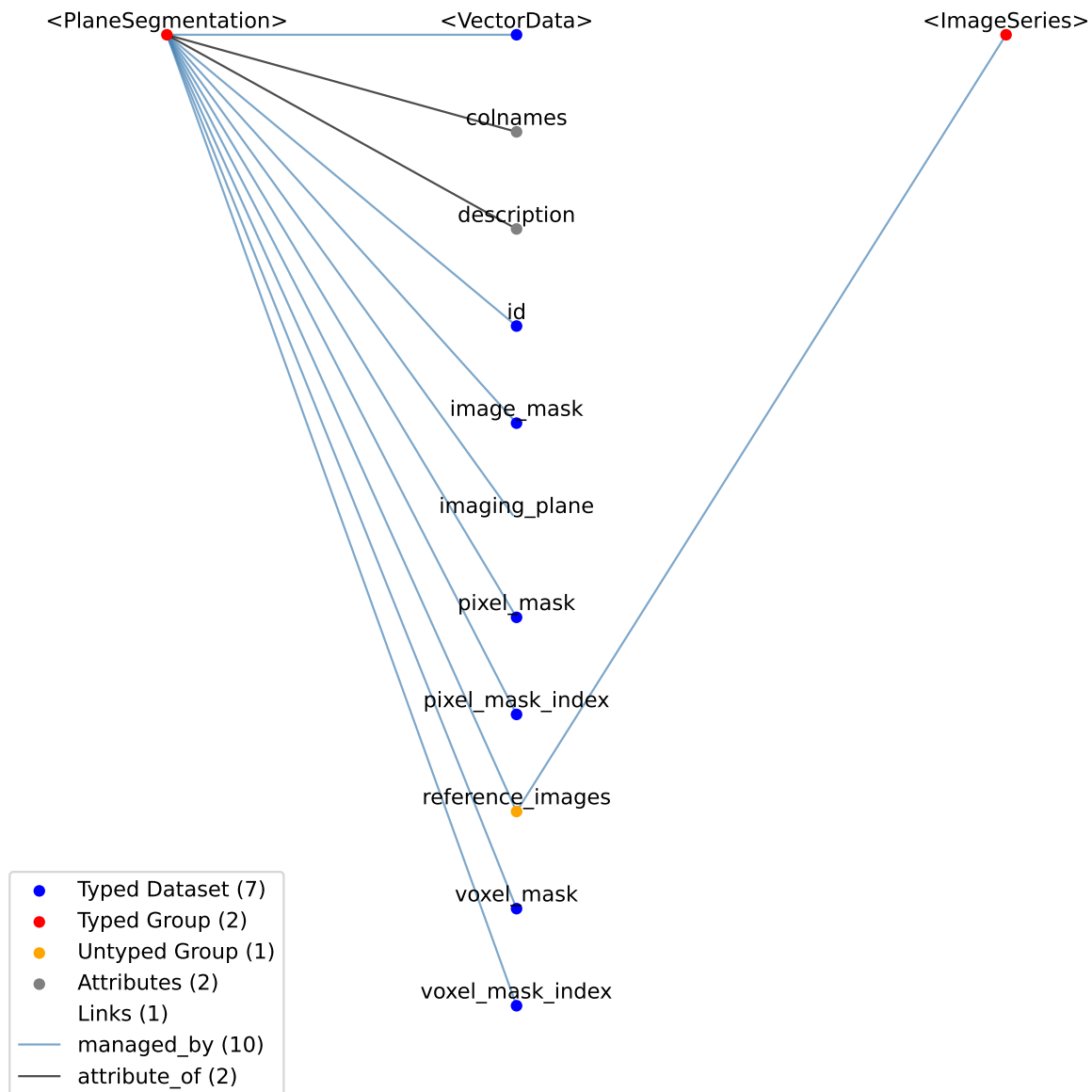


Table 4.114: Datasets, Links, and Attributes contained in <PlaneSegmentation>

Id	Type	Description
<PlaneSegmentation>	Group	Top level Group for <PlaneSegmentation> • Neurodata Type: <i>PlaneSegmentation</i> • Extends: <i>DynamicTable</i>
—colnames	At-tribute	The names of the columns in this table. This should be used to specify an order to the columns. • Data Type: text • Dimensions: ['num_columns'] • Shape: [None] • Name: colnames
—description	At-tribute	Description of what is in this dynamic table. • Data Type: text • Name: description
—image_mask	Dataset	ROI masks for each ROI. Each image mask is the size of the original imaging plane (or volume) and members of the ROI are finite non-zero. • Extends: <i>VectorData</i> • Quantity: 0 or 1 • Dimensions: [['num_roi', 'num_x', 'num_y'], ['num_roi', 'num_x', 'num_y', 'num_z']] • Shape: [[None, None, None], [None, None, None, None]] • Name: image_mask
—pixel_mask_index	Dataset	Index into pixel_mask. • Extends: <i>VectorIndex</i> • Quantity: 0 or 1 • Name: pixel_mask_index
—pixel_mask	Dataset	Pixel masks for each ROI: a list of indices and weights for the ROI. Pixel masks are concatenated and parsing of this dataset is maintained by the PlaneSegmentation • Extends: <i>VectorData</i> • Quantity: 0 or 1 • Data Type: Compound data type with the following elements: – x: Pixel x-coordinate. (<i>dtype</i>= uint32) – y: Pixel y-coordinate. (<i>dtype</i>= uint32) – weight: Weight of the pixel. (<i>dtype</i>= float32) • Name: pixel_mask
—voxel_mask_index	Dataset	Index into voxel_mask. • Extends: <i>VectorIndex</i> • Quantity: 0 or 1 • Name: voxel_mask_index

continues on next page

Table 4.114 – continued from previous page

Id	Type	Description
—voxel_mask	Dataset	Voxel masks for each ROI: a list of indices and weights for the ROI. Voxel masks are concatenated and parsing of this dataset is maintained by the PlaneSegmentation • Extends: VectorData • Quantity: 0 or 1 • Data Type: Compound data type with the following elements: – x: Voxel x-coordinate. (<i>dtype</i>= uint32) – y: Voxel y-coordinate. (<i>dtype</i>= uint32) – z: Voxel z-coordinate. (<i>dtype</i>= uint32) – weight: Weight of the voxel. (<i>dtype</i>= float32) • Name: voxel_mask
—id	Dataset	Array of unique identifiers for the rows of this dynamic table. • Extends: ElementIdentifiers • Data Type: int • Dimensions: ['num_rows'] • Shape: [None] • Name: id
—< VectorData >	Dataset	Vector columns, including index columns, of this dynamic table. • Extends: VectorData • Quantity: 0 or more
—imaging_plane	Link	Link to ImagingPlane object from which this data was generated. • Target Type ImagingPlane • Name: imaging_plane

Table 4.115: Groups contained in <PlaneSegmentation>

Id	Type	Description
<PlaneSegmentation>	Group	Top level Group for <PlaneSegmentation> • Neurodata Type: PlaneSegmentation • Extends: DynamicTable
—imaging_plane	Link	Link to ImagingPlane object from which this data was generated. • Target Type ImagingPlane • Name: imaging_plane
—reference_images	Group	Image stacks that the segmentation masks apply to. • Name: reference_images

4.11.7.1 Groups: `reference_images`

Image stacks that the segmentation masks apply to.

- **Name:** `reference_images`

Table 4.116: Groups contained in `<reference_images>`

Id	Type	Description
<code>reference_images</code>	Group	Top level Group for <code>reference_images</code> • Name: <code>reference_images</code>
<code>—<ImageSeries></code>	Group	One or more image stacks that the masks apply to (can be one-element stack). • Extends: <i>ImageSeries</i> • Quantity: 0 or more

4.11.7.2 Groups: `reference_images/<ImageSeries>`

One or more image stacks that the masks apply to (can be one-element stack).

- **Extends:** *ImageSeries*
- **Quantity:** 0 or more

4.11.8 ImagingPlane

Overview: An imaging plane and its metadata.

ImagingPlane extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.8](#)

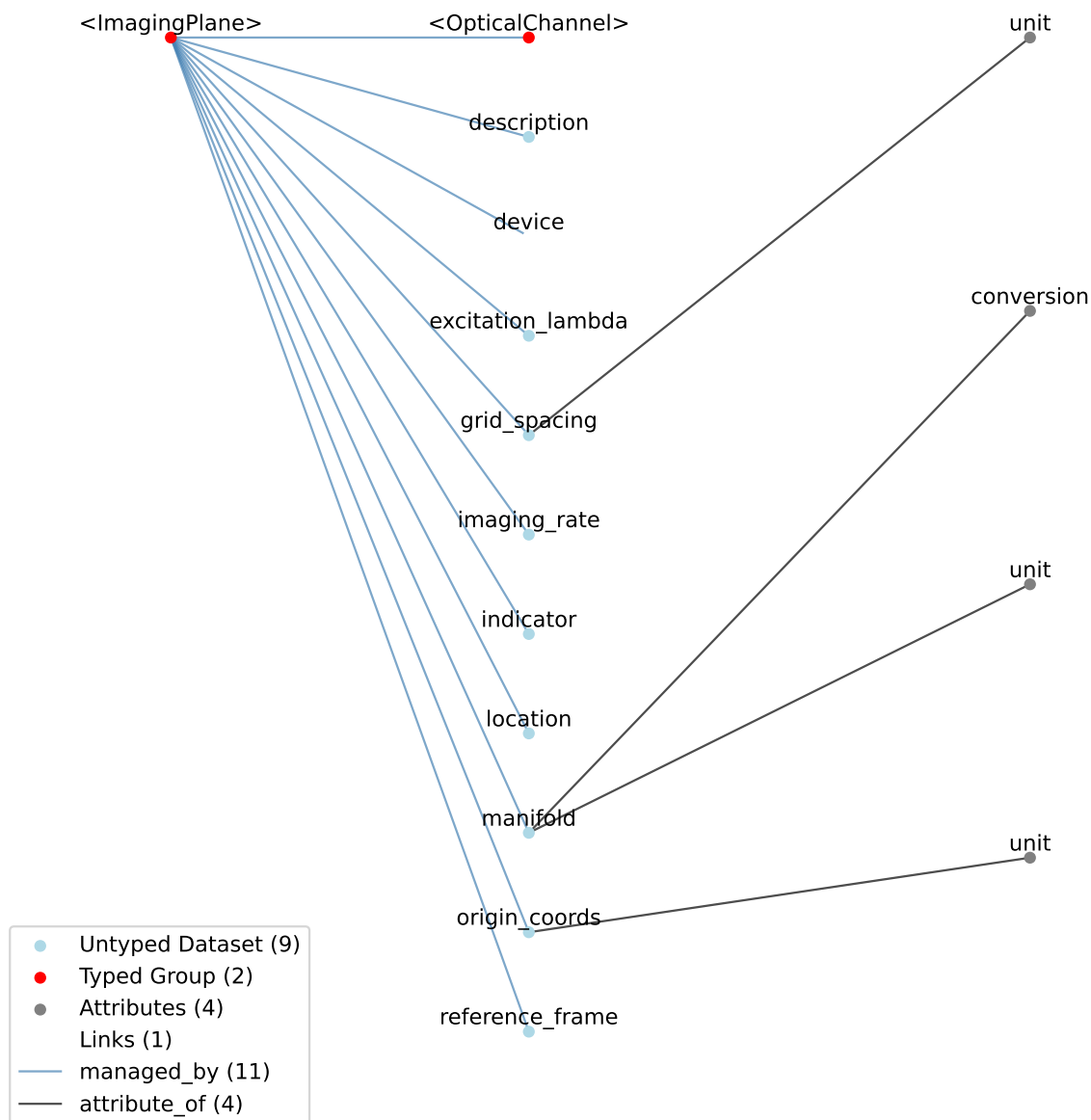


Table 4.117: Datasets, Links, and Attributes contained in <ImagingPlane>

Id	Type	Description
<ImagingPlane>	Group	Top level Group for <ImagingPlane> • Neurodata Type: <i>ImagingPlane</i> • Extends: <i>NWBContainer</i>
—description	Dataset	Description of the imaging plane. • Quantity: 0 or 1 • Data Type: text • Name: description
—excitation_lambda	Dataset	Excitation wavelength, in nm. • Data Type: float32 • Name: excitation_lambda
—imaging_rate	Dataset	Rate that images are acquired, in Hz. If the corresponding TimeSeries is present, the rate should be stored there instead. • Quantity: 0 or 1 • Data Type: float32 • Name: imaging_rate
—indicator	Dataset	Calcium indicator. • Data Type: text • Name: indicator
—location	Dataset	Location of the imaging plane. Specify the area, layer, comments on estimation of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names for anatomical regions when possible. • Data Type: text • Name: location
—manifold	Dataset	DEPRECATED Physical position of each pixel. ‘xyz’ represents the position of the pixel relative to the defined coordinate space. Deprecated in favor of origin_coords and grid_spacing. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['height', 'width', 'x, y, z'], ['height', 'width', 'depth', 'x, y, z']] • Shape: [[None, None, 3], [None, None, None, 3]] • Name: manifold

continues on next page

Table 4.117 – continued from previous page

Id	Type	Description
——conversion	At-tribute	Scalar to multiply each element in data to convert it to the specified ‘unit’. If the data are stored in acquisition system units or other units that require a conversion to be interpretable, multiply the data by ‘conversion’ to convert the data to the specified ‘unit’. e.g. if the data acquisition system stores values in this object as pixels from $x = -500$ to 499, $y = -500$ to 499 that correspond to a 2 m x 2 m range, then the ‘conversion’ multiplier to get from raw data acquisition pixel units to meters is $2/1000$. • Data Type: float32 • Required: False • Default Value: 1.0 • Name: conversion
——unit	At-tribute	Base unit of measurement for working with the data. The default value is ‘meters’. • Data Type: text • Required: False • Default Value: meters • Name: unit
—origin_coords	Dataset	Physical location of the first element of the imaging plane (0, 0) for 2-D data or (0, 0, 0) for 3-D data. See also reference_frame for what the physical location is relative to (e.g., bregma). • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['x', 'y'], ['x', 'y', 'z']] • Shape: [[2], [3]] • Name: origin_coords
——unit	At-tribute	Measurement units for origin_coords. The default value is ‘meters’. • Data Type: text • Required: False • Default Value: meters • Name: unit
—grid_spacing	Dataset	Space between pixels in (x, y) or voxels in (x, y, z) directions, in the specified unit. Assumes imaging plane is a regular grid. See also reference_frame to interpret the grid. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: [['x', 'y'], ['x', 'y', 'z']] • Shape: [[2], [3]] • Name: grid_spacing
——unit	At-tribute	Measurement units for grid_spacing. The default value is ‘meters’. • Data Type: text • Required: False • Default Value: meters • Name: unit

continues on next page

Table 4.117 – continued from previous page

Id	Type	Description
—reference_frame	Dataset	Describes reference frame of origin_coords and grid_spacing. For example, this can be a text description of the anatomical location and orientation of the grid defined by origin_coords and grid_spacing or the vectors needed to transform or rotate the grid to a common anatomical axis (e.g., AP/DV/ML). This field is necessary to interpret origin_coords and grid_spacing. If origin_coords and grid_spacing are not present, then this field is not required. For example, if the microscope takes 10 x 10 x 2 images, where the first value of the data matrix (index (0, 0, 0)) corresponds to (-1.2, -0.6, -2) mm relative to bregma, the spacing between pixels is 0.2 mm in x, 0.2 mm in y and 0.5 mm in z, and larger numbers in x means more anterior, larger numbers in y means more rightward, and larger numbers in z means more ventral, then enter the following – origin_coords = (-1.2, -0.6, -2) grid_spacing = (0.2, 0.2, 0.5) reference_frame = “Origin coordinates are relative to bregma. First dimension corresponds to anterior-posterior axis (larger index = more anterior). Second dimension corresponds to medial-lateral axis (larger index = more rightward). Third dimension corresponds to dorsal-ventral axis (larger index = more ventral).” • Quantity: 0 or 1 • Data Type: text • Name: reference_frame
—device	Link	Link to the Device object that was used to record from this electrode. • Target Type <i>Device</i> • Name: device

Table 4.118: Groups contained in <ImagingPlane>

Id	Type	Description
<ImagingPlane>	Group	Top level Group for <ImagingPlane> • Neurodata Type: <i>ImagingPlane</i> • Extends: <i>NWBContainer</i>
—device	Link	Link to the Device object that was used to record from this electrode. • Target Type <i>Device</i> • Name: device
—< <i>OpticalChannel</i> >	Group	An optical channel used to record from an imaging plane. • Extends: <i>OpticalChannel</i> • Quantity: 1 or more

4.11.8.1 Groups: <OpticalChannel>

An optical channel used to record from an imaging plane.

- **Extends:** *OpticalChannel*
- **Quantity:** 1 or more

4.11.9 OpticalChannel

Overview: An optical channel used to record from an imaging plane.

OpticalChannel extends NWBContainer and includes all elements of *NWBContainer* with the following additions or changes.

- **Extends:** *NWBContainer*
- **Primitive Type:** Group
- **Inherits from:** *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.9](#)

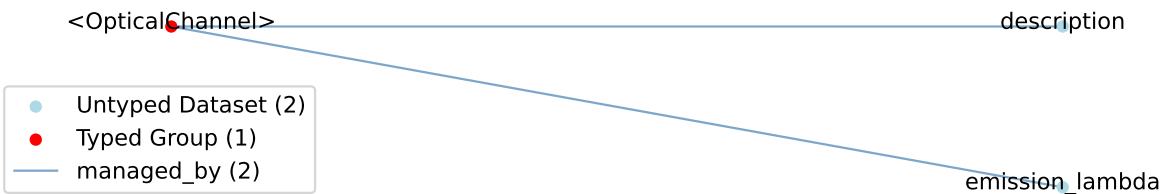


Table 4.119: Datasets, Links, and Attributes contained in <OpticalChannel>

Id	Type	Description
<OpticalChannel>	Group	Top level Group for <OpticalChannel> • Neurodata Type: <i>OpticalChannel</i> • Extends: <i>NWBContainer</i>
—description	Dataset	Description or other notes about the channel. • Data Type: text • Name: description
—emission_lambda	Dataset	Emission wavelength for channel, in nm. • Data Type: float32 • Name: emission_lambda

4.11.10 MotionCorrection

Overview: An image stack where all frames are shifted (registered) to a common coordinate system, to account for movement and drift between frames. Note: each frame at each point in time is assumed to be 2-D (has only x & y dimensions).

MotionCorrection extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** MotionCorrection
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see [Section 5.12.10](#)

Table 4.120: Groups contained in <MotionCorrection>

Id	Type	Description
<MotionCorrection>	Group	Top level Group for <MotionCorrection> • Neurodata Type: <i>MotionCorrection</i> • Extends: <i>NWBDataInterface</i> • Default Name: MotionCorrection
—<CorrectedImageStack>	Group	Results from motion correction of an image stack. • Extends: <i>CorrectedImageStack</i> • Quantity: 1 or more

4.11.10.1 Groups: <CorrectedImageStack>

Results from motion correction of an image stack.

- **Extends:** *CorrectedImageStack*
- **Quantity:** 1 or more

4.11.11 CorrectedImageStack

Overview: Results from motion correction of an image stack.

CorrectedImageStack extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.ophys.yaml
- **Source Specification:** see Section 5.12.11

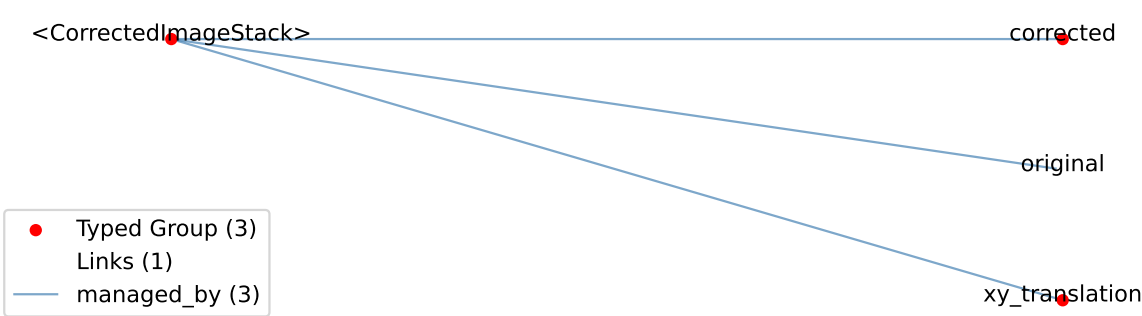


Table 4.121: Datasets, Links, and Attributes contained in <CorrectedImageStack>

Id	Type	Description
<CorrectedImageStack>	Group	Top level Group for <CorrectedImageStack> • Neurodata Type: <i>CorrectedImageStack</i> • Extends: <i>NWBDataInterface</i>
—original	Link	Link to ImageSeries object that is being registered. • Target Type <i>ImageSeries</i> • Name: original

Table 4.122: Groups contained in <CorrectedImageStack>

Id	Type	Description
<CorrectedImageStack>	Group	Top level Group for <CorrectedImageStack> • Neurodata Type: <i>CorrectedImageStack</i> • Extends: <i>NWBDataInterface</i>
—original	Link	Link to ImageSeries object that is being registered. • Target Type <i>ImageSeries</i> • Name: original

continues on next page

Table 4.122 – continued from previous page

Id	Type	Description
—corrected	Group	Image stack with frames shifted to the common coordinates. • Extends: <i>ImageSeries</i> • Name: corrected
—xy_translation	Group	Stores the x,y delta necessary to align each frame to the common coordinates, for example, to align each frame to a reference image. • Extends: <i>TimeSeries</i> • Name: xy_translation

4.11.11.1 Groups: corrected

Image stack with frames shifted to the common coordinates.

- **Extends:** *ImageSeries*
- **Name:** corrected

4.11.11.2 Groups: xy_translation

Stores the x,y delta necessary to align each frame to the common coordinates, for example, to align each frame to a reference image.

- **Extends:** *TimeSeries*
- **Name:** xy_translation

4.12 Retinotopy

This source module contains neurodata_type for retinotopy data.

4.12.1 ImagingRetinotopy

Overview: Intrinsic signal optical imaging or widefield imaging for measuring retinotopy. Stores orthogonal maps (e.g., altitude/azimuth; radius/theta) of responses to specific stimuli and a combined polarity map from which to identify visual areas. This group does not store the raw responses imaged during retinotopic mapping or the stimuli presented, but rather the resulting phase and power maps after applying a Fourier transform on the averaged responses. Note: for data consistency, all images and arrays are stored in the format [row][column] and [row, col], which equates to [y][x]. Field of view and dimension arrays may appear backward (i.e., y before x).

ImagingRetinotopy extends NWBDataInterface and includes all elements of *NWBDataInterface* with the following additions or changes.

- **Extends:** *NWBDataInterface*
- **Primitive Type:** Group
- **Default Name:** ImagingRetinotopy
- **Inherits from:** *NWBDataInterface*, *NWBContainer*, *Container*
- **Source filename:** nwb.retinotopy.yaml
- **Source Specification:** see [Section 5.13.1](#)

Table 4.123: Datasets, Links, and Attributes contained in <ImagingRetinotopy>

Id	Type	Description
<ImagingRetinotopy>	Group	Top level Group for <ImagingRetinotopy> • Neurodata Type: <i>ImagingRetinotopy</i> • Extends: <i>NWBDataInterface</i> • Default Name: ImagingRetinotopy
—axis_1_phase_map	Dataset	Phase response to stimulus on the first measured axis. • Data Type: float32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: axis_1_phase_map
——dimension	At-tribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension

continues on next page

Table 4.123 – continued from previous page

Id	Type	Description
——field_of_view	At-tribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
——unit	At-tribute	Unit that axis data is stored in (e.g., degrees). • Data Type: text • Name: unit
—axis_1_power_map	Dataset	Power response on the first measured axis. Response is scaled so 0.0 is no power in the response and 1.0 is maximum relative power. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: axis_1_power_map
——dimension	At-tribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension
——field_of_view	At-tribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
——unit	At-tribute	Unit that axis data is stored in (e.g., degrees). • Data Type: text • Name: unit
—axis_2_phase_map	Dataset	Phase response to stimulus on the second measured axis. • Data Type: float32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: axis_2_phase_map
——dimension	At-tribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension

continues on next page

Table 4.123 – continued from previous page

Id	Type	Description
——field_of_view	At-tribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
——unit	At-tribute	Unit that axis data is stored in (e.g., degrees). • Data Type: text • Name: unit
—axis_2_power_map	Dataset	Power response on the second measured axis. Response is scaled so 0.0 is no power in the response and 1.0 is maximum relative power. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: axis_2_power_map
——dimension	At-tribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension
——field_of_view	At-tribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
——unit	At-tribute	Unit that axis data is stored in (e.g., degrees). • Data Type: text • Name: unit
—axis_descriptions	Dataset	Two-element array describing the contents of the two response axis fields. Description should be something like ['altitude', 'azimuth'] or ['radius', 'theta']. • Data Type: text • Dimensions: ['axis_1', 'axis_2'] • Shape: [2] • Name: axis_descriptions

continues on next page

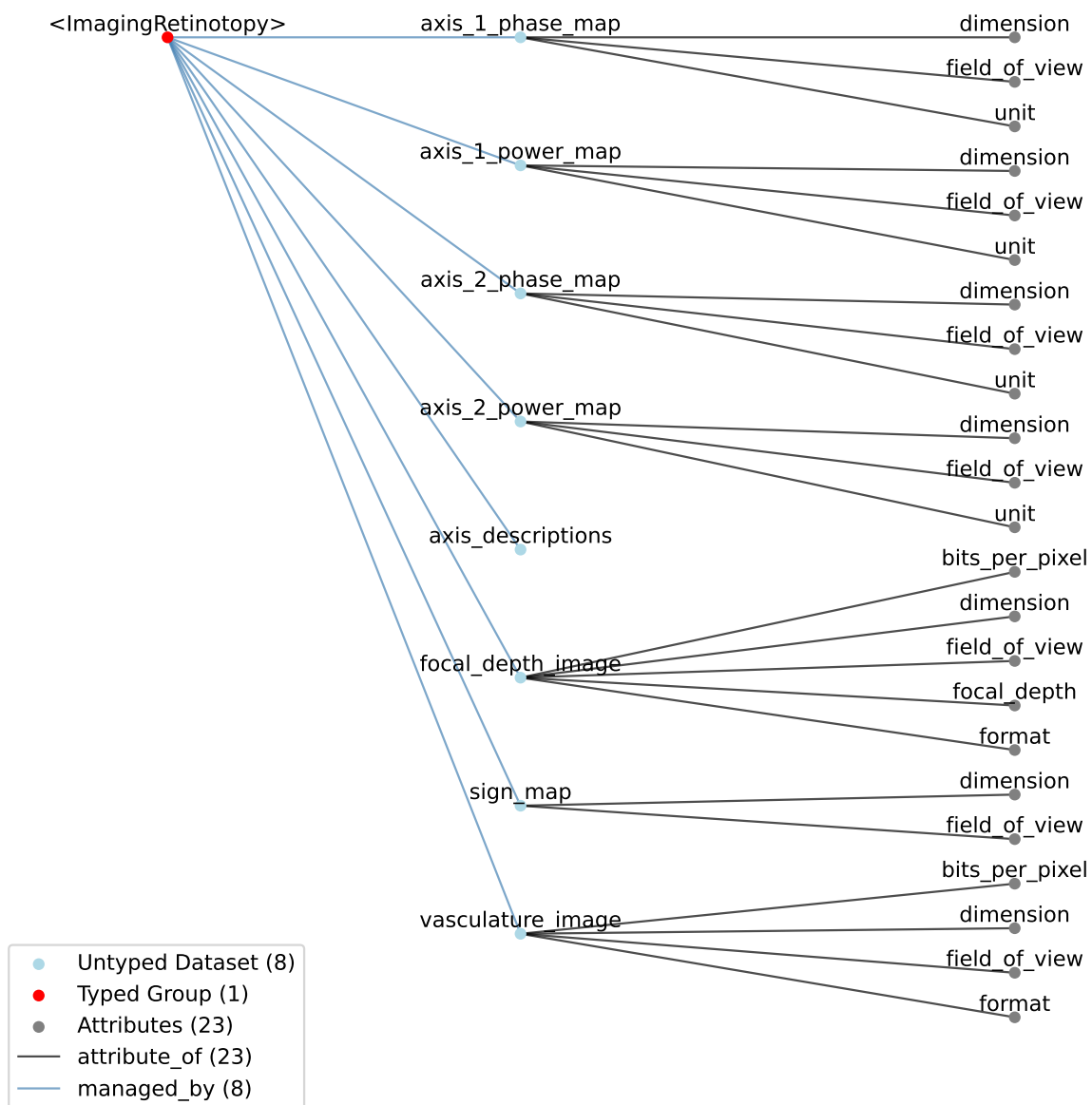
Table 4.123 – continued from previous page

Id	Type	Description
—focal_depth_image	Dataset	Gray-scale image taken with same settings/parameters (e.g., focal depth, wavelength) as data collection. Array format: [rows][columns]. • Quantity: 0 or 1 • Data Type: uint16 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: focal_depth_image
—bits_per_pixel	Attribute	Number of bits used to represent each value. This is necessary to determine maximum (white) pixel value. • Data Type: int32 • Name: bits_per_pixel
—dimension	Attribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension
—field_of_view	Attribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
—focal_depth	Attribute	Focal depth offset, in meters. • Data Type: float32 • Name: focal_depth
—format	Attribute	Format of image. Right now only 'raw' is supported. • Data Type: text • Name: format
—sign_map	Dataset	Sine of the angle between the direction of the gradient in axis_1 and axis_2. • Quantity: 0 or 1 • Data Type: float32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: sign_map

continues on next page

Table 4.123 – continued from previous page

Id	Type	Description
——dimension	At-tribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension
——field_of_view	At-tribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
—vasculature_image	Dataset	Gray-scale anatomical image of cortical surface. Array structure: [rows][columns] • Data Type: uint16 • Dimensions: ['num_rows', 'num_cols'] • Shape: [None, None] • Name: vasculature_image
——bits_per_pixel	At-tribute	Number of bits used to represent each value. This is necessary to determine maximum (white) pixel value • Data Type: int32 • Name: bits_per_pixel
——dimension	At-tribute	Number of rows and columns in the image. Additional Information: row, column representation is equivalent to height, width. • Data Type: int32 • Dimensions: ['num_rows', 'num_cols'] • Shape: [2] • Name: dimension
——field_of_view	At-tribute	Size of viewing area, in meters. • Data Type: float32 • Dimensions: ['height', 'width'] • Shape: [2] • Name: field_of_view
——format	At-tribute	Format of image. Right now only 'raw' is supported. • Data Type: text • Name: format



SCHEMA SOURCES

Source Specification: see [Section 5.1](#)

5.1 Namespace – NWB core

Description: see [Section 3.1](#)

YAML Specification:

```
1 author:
2 - Andrew Tritt
3 - Oliver Ruebel
4 - Ryan Ly
5 - Ben Dichter
6 - Keith Godfrey
7 - Jeff Teeters
8 contact:
9 - ajtritt@lbl.gov
10 - oruebel@lbl.gov
11 - rly@lbl.gov
12 - bdichter@lbl.gov
13 - keithg@alleninstitute.org
14 - jteeters@berkeley.edu
15 doc: NWB namespace
16 full_name: NWB core
17 name: core
18 schema:
19 - namespace: hdmf-common
20 - doc: This source module contains base data types used throughout the NWB data format.
21   source: nwb.base.yaml
22   title: Base data types
23 - doc: This source module contains neurodata_types for device data.
24   source: nwb.device.yaml
25   title: Devices
26 - doc: This source module contains neurodata_types for epoch data.
27   source: nwb.epoch.yaml
28   title: Epochs
29 - doc: This source module contains neurodata_types for image data.
30   source: nwb.image.yaml
31   title: Image data
```

(continues on next page)

(continued from previous page)

```

32 - doc: Main NWB file specification.
33   source: nwb.file.yaml
34   title: NWB file
35 - doc: Miscellaneous types.
36   source: nwb.misc.yaml
37   title: Miscellaneous neurodata_types.
38 - doc: This source module contains neurodata_types for behavior data.
39   source: nwb.behavior.yaml
40   title: Behavior
41 - doc: This source module contains neurodata_types for extracellular electrophysiology
42   data.
43   source: nwb.ecephys.yaml
44   title: Extracellular electrophysiology
45 - doc: This source module contains neurodata_types for intracellular electrophysiology
46   data.
47   source: nwb.icephys.yaml
48   title: Intracellular electrophysiology
49 - doc: This source module contains neurodata_types for opto-genetics data.
50   source: nwb.ogen.yaml
51   title: Optogenetics
52 - doc: This source module contains neurodata_types for optical physiology data.
53   source: nwb.ophys.yaml
54   title: Optical physiology
55 - doc: This source module contains neurodata_type for retinotopy data.
56   source: nwb.retinotopy.yaml
57   title: Retinotopy
58 version: 2.6.0-alpha

```

5.2 Base data types

This source module contains base data types used throughout the NWB data format.

5.2.1 NWBData

Extends: [Data](#)

Description: see [Section 4.1.1](#)

YAML Specification:

```

1 doc: An abstract data type for a dataset.
2 neurodata_type_def: NWBData
3 neurodata_type_inc: Data

```

5.2.2 TimeSeriesReferenceVectorData

Extends: `VectorData`

Description: see [Section 4.1.2](#)

YAML Specification:

```

1 attributes:
2 - doc: Description of what these vectors represent.
3   dtype: text
4   name: description
5 default_name: timeseries
6 doc: Column storing references to a TimeSeries (rows). For each TimeSeries this
   ↳ VectorData
7   column stores the start_index and count to indicate the range in time to be selected
8   as well as an object reference to the TimeSeries.
9 dtype:
10 - doc: Start index into the TimeSeries 'data' and 'timestamp' datasets of the referenced
11   TimeSeries. The first dimension of those arrays is always time.
12   dtype: int32
13   name: idx_start
14 - doc: Number of data samples available in this time series, during this epoch
15   dtype: int32
16   name: count
17 - doc: The TimeSeries that this index applies to
18   dtype:
19     reftype: object
20     target_type: TimeSeries
21   name: timeseries
22 neurodata_type_def: TimeSeriesReferenceVectorData
23 neurodata_type_inc: VectorData

```

5.2.3 Image

Extends: *NWBData*

Description: see [Section 4.1.3](#)

YAML Specification:

```
1 attributes:
2 - doc: Pixel resolution of the image, in pixels per centimeter.
3   dtype: float32
4   name: resolution
5   required: false
6 - doc: Description of the image.
7   dtype: text
8   name: description
9   required: false
10 dims:
11 - - x
12   - y
13 - - x
14   - y
15   - r, g, b
16 - - x
17   - y
18   - r, g, b, a
19 doc: An abstract data type for an image. Shape can be 2-D (x, y), or 3-D where the
20   third dimension can have three or four elements, e.g. (x, y, (r, g, b)) or (x, y,
21   (r, g, b, a)).
22 dtype: numeric
23 neurodata_type_def: Image
24 neurodata_type_inc: NWBData
25 shape:
26 - -
27   -
28 - -
29   -
30   - 3
31 - -
32   -
33   - 4
```

5.2.4 ImageReferences

Extends: *NWBData*

Description: see [Section 4.1.4](#)

YAML Specification:

```
1 dims:  
2 - num_images  
3 doc: Ordered dataset of references to Image objects.  
4 dtype:  
5   reftype: object  
6   target_type: Image  
7 neurodata_type_def: ImageReferences  
8 neurodata_type_inc: NWBData  
9 shape:  
10 -
```

5.2.5 NWBContainer

Extends: [Container](#)

Description: see [Section 4.1.5](#)

YAML Specification:

```
1 doc: An abstract data type for a generic container storing collections of data and
2   metadata. Base type for all data and metadata containers.
3 neurodata_type_def: NWBContainer
4 neurodata_type_inc: Container
```

5.2.6 NWBDataInterface

Extends: *NWBContainer*

Description: see [Section 4.1.6](#)

YAML Specification:

```
1 doc: An abstract data type for a generic container storing collections of data, as
2   opposed to metadata.
3 neurodata_type_def: NWBDataInterface
4 neurodata_type_inc: NWBContainer
```

5.2.7 TimeSeries

Extends: *NWBDataInterface*

Description: see [Section 4.1.7](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - default_value: 1.0
17     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
18         If the data are stored in acquisition system units or other units that require
19         a conversion to be interpretable, multiply the data by 'conversion' to convert
20         the data to the specified 'unit'. e.g. if the data acquisition system stores
21         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
22         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
23         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
24         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
25     dtype: float32
26     name: conversion
27     required: false
28   - default_value: 0.0
29     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
30         to the specified 'unit'. Two common examples of this include (a) data stored
31         in an unsigned type that requires a shift after scaling to re-center the data,
32         and (b) specialized recording devices that naturally cause a scalar offset with
33         respect to the true units.
34     dtype: float32
35     name: offset
36     required: false
37   - default_value: -1.0
38     doc: Smallest meaningful difference between values in data, stored in the specified
39         by unit, e.g., the change in value of the least significant bit, or a larger
40         number if signal noise is known to be present. If unknown, use -1.0.
41     dtype: float32
42     name: resolution
43     required: false
44   - doc: Base unit of measurement for working with the data. Actual stored values
45       are not necessarily stored in these units. To access the data in these units,
46       multiply 'data' by 'conversion' and add 'offset'.

```

(continues on next page)

(continued from previous page)

```

47     dtype: text
48     name: unit
49     - doc: Optionally describe the continuity of the data. Can be "continuous",
↳ "instantaneous",
50       or "step". For example, a voltage trace would be "continuous", because samples
51       are recorded from a continuous process. An array of lick times would be
↳ "instantaneous",
52       because the data represents distinct moments in time. Times of image presentations
53       would be "step" because the picture remains the same until the next timepoint.
54       This field is optional, but is useful in providing information about the underlying
55       data. It may inform the way this data is interpreted, the way it is visualized,
56       and what analysis methods are applicable.
57     dtype: text
58     name: continuity
59     required: false
60     dims:
61     - - num_times
62     - - num_times
63     - num_DIM2
64     - - num_times
65     - num_DIM2
66     - num_DIM3
67     - - num_times
68     - num_DIM2
69     - num_DIM3
70     - num_DIM4
71     doc: Data values. Data can be in 1-D, 2-D, 3-D, or 4-D. The first dimension should
72         always represent time. This can also be used to store binary data (e.g., image
73         frames). This can also be a link to data stored in an external file.
74     name: data
75     shape:
76     - -
77     - -
78     -
79     - -
80     -
81     -
82     - -
83     -
84     -
85     -
86     - attributes:
87     - doc: Sampling rate, in Hz.
88       dtype: float32
89       name: rate
90     - doc: Unit of measurement for time, which is fixed to 'seconds'.
91       dtype: text
92       name: unit
93       value: seconds
94     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
95         the timestamp of the first sample can be specified and all subsequent ones calculated
96         from the sampling rate attribute.

```

(continues on next page)

(continued from previous page)

```

97 dtype: float64
98 name: starting_time
99 quantity: '?'
100 - attributes:
101   - doc: Value is '1'
102     dtype: int32
103     name: interval
104     value: 1
105   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
106     dtype: text
107     name: unit
108     value: seconds
109 dims:
110   - num_times
111 doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
112     master-clock stored in NWBFile.timestamps_reference_time.
113 dtype: float64
114 name: timestamps
115 quantity: '?'
116 shape:
117   -
118 - dims:
119   - num_times
120 doc: Numerical labels that apply to each time point in data for the purpose of querying
121     and slicing data by these values. If present, the length of this array should
122     be the same size as the first dimension of data.
123 dtype: uint8
124 name: control
125 quantity: '?'
126 shape:
127   -
128 - dims:
129   - num_control_values
130 doc: Description of each control value. Must be present if control is present. If
131     present, control_description[0] should describe time points where control == 0.
132 dtype: text
133 name: control_description
134 quantity: '?'
135 shape:
136   -
137 doc: General purpose time series.
138 groups:
139   - doc: Lab-specific time and sync information as provided directly from hardware devices
140     and that is necessary for aligning all acquired time information to a common_
↪ timebase.
141     The timestamp array stores time in the common timebase. This group will usually
142     only be populated in TimeSeries that are stored external to the NWB file, in files
143     storing raw data. Once timestamp data is calculated, the contents of 'sync' are
144     mostly for archival purposes.
145 name: sync
146 quantity: '?'

```

(continues on next page)

(continued from previous page)

147 `neurodata_type_def`: TimeSeries
148 `neurodata_type_inc`: NWBDataInterface

5.2.8 ProcessingModule

Extends: *NWBContainer*

Description: see [Section 4.1.8](#)

YAML Specification:

```
1 attributes:
2 - doc: Description of this collection of processed data.
3   dtype: text
4   name: description
5 doc: A collection of processed data.
6 groups:
7 - doc: Data objects stored in this collection.
8   neurodata_type_inc: NWBDataInterface
9   quantity: '*'
10 - doc: Tables stored in this collection.
11   neurodata_type_inc: DynamicTable
12   quantity: '*'
13 neurodata_type_def: ProcessingModule
14 neurodata_type_inc: NWBContainer
```

5.2.9 Images

Extends: *NWBDataInterface*

Description: see [Section 4.1.9](#)

YAML Specification:

```

1 attributes:
2 - doc: Description of this collection of images.
3   dtype: text
4   name: description
5 datasets:
6 - doc: Images stored in this collection.
7   neurodata_type_inc: Image
8   quantity: +
9 - doc: Ordered dataset of references to Image objects stored in the parent group.
10    Each Image object in the Images group should be stored once and only once, so
11    the dataset should have the same length as the number of images.
12   name: order_of_images
13   neurodata_type_inc: ImageReferences
14   quantity: '?'
15 default_name: Images
16 doc: A collection of images with an optional way to specify the order of the images
17    using the "order_of_images" dataset. An order must be specified if the images are
18    referenced by index, e.g., from an IndexSeries.
19 neurodata_type_def: Images
20 neurodata_type_inc: NWBDataInterface

```

5.3 Devices

This source module contains `neurodata_types` for device data.

5.3.1 Device

Extends: *NWBContainer*

Description: see [Section 4.2.1](#)

YAML Specification:

```
1 attributes:
2 - doc: Description of the device (e.g., model, firmware version, processing software
3   version, etc.) as free-form text.
4   dtype: text
5   name: description
6   required: false
7 - doc: The name of the manufacturer of the device.
8   dtype: text
9   name: manufacturer
10  required: false
11 doc: Metadata about a data acquisition device, e.g., recording system, electrode,
12   microscope.
13 neurodata_type_def: Device
14 neurodata_type_inc: NWBContainer
```

5.4 Epochs

This source module contains `neurodata_types` for epoch data.

5.4.1 TimeIntervals

Extends: `DynamicTable`

Description: see [Section 4.3.1](#)

YAML Specification:

```

1  attributes:
2  - dims:
3    - num_columns
4    doc: The names of the columns in this table. This should be used to specify an order
5      to the columns.
6    dtype: text
7    name: colnames
8    shape:
9      -
10   - doc: Description of what is in this dynamic table.
11     dtype: text
12     name: description
13 datasets:
14 - doc: Start time of epoch, in seconds.
15   dtype: float32
16   name: start_time
17   neurodata_type_inc: VectorData
18 - doc: Stop time of epoch, in seconds.
19   dtype: float32
20   name: stop_time
21   neurodata_type_inc: VectorData
22 - doc: User-defined tags that identify or categorize events.
23   dtype: text
24   name: tags
25   neurodata_type_inc: VectorData
26   quantity: '?'
27 - doc: Index for tags.
28   name: tags_index
29   neurodata_type_inc: VectorIndex
30   quantity: '?'
31 - doc: An index into a TimeSeries object.
32   name: timeseries
33   neurodata_type_inc: TimeSeriesReferenceVectorData
34   quantity: '?'
35 - doc: Index for timeseries.
36   name: timeseries_index
37   neurodata_type_inc: VectorIndex
38   quantity: '?'
39 - dims:
40   - num_rows
41   doc: Array of unique identifiers for the rows of this dynamic table.

```

(continues on next page)

(continued from previous page)

```
42 dtype: int
43 name: id
44 neurodata_type_inc: ElementIdentifiers
45 shape:
46 -
47 - doc: Vector columns, including index columns, of this dynamic table.
48   neurodata_type_inc: VectorData
49   quantity: '*'
50 doc: A container for aggregating epoch data and the TimeSeries that each epoch applies
51     to.
52 neurodata_type_def: TimeIntervals
53 neurodata_type_inc: DynamicTable
```

5.5 Image data

This source module contains neurodata_types for image data.

5.5.1 GrayscaleImage

Extends: *Image*

Description: see [Section 4.4.1](#)

YAML Specification:

```

1 attributes:
2 - doc: Pixel resolution of the image, in pixels per centimeter.
3   dtype: float32
4   name: resolution
5   required: false
6 - doc: Description of the image.
7   dtype: text
8   name: description
9   required: false
10 dims:
11 - x
12 - y
13 doc: A grayscale image.
14 dtype: numeric
15 neurodata_type_def: GrayscaleImage
16 neurodata_type_inc: Image
17 shape:
18 -
19 -

```

5.5.2 RGBImage

Extends: *Image*

Description: see [Section 4.4.2](#)

YAML Specification:

```
1 attributes:
2 - doc: Pixel resolution of the image, in pixels per centimeter.
3   dtype: float32
4   name: resolution
5   required: false
6 - doc: Description of the image.
7   dtype: text
8   name: description
9   required: false
10 dims:
11 - x
12 - y
13 - r, g, b
14 doc: A color image.
15 dtype: numeric
16 neurodata_type_def: RGBImage
17 neurodata_type_inc: Image
18 shape:
19 -
20 -
21 - 3
```

5.5.3 RGBImage

Extends: *Image*

Description: see [Section 4.4.3](#)

YAML Specification:

```

1 attributes:
2 - doc: Pixel resolution of the image, in pixels per centimeter.
3   dtype: float32
4   name: resolution
5   required: false
6 - doc: Description of the image.
7   dtype: text
8   name: description
9   required: false
10 dims:
11 - x
12 - y
13 - r, g, b, a
14 doc: A color image with transparency.
15 dtype: numeric
16 neurodata_type_def: RGBImage
17 neurodata_type_inc: Image
18 shape:
19 -
20 -
21 - 4

```

5.5.4 ImageSeries

Extends: *TimeSeries*

Description: see [Section 4.4.4](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - default_value: 1.0
17     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
18         If the data are stored in acquisition system units or other units that require
19         a conversion to be interpretable, multiply the data by 'conversion' to convert
20         the data to the specified 'unit'. e.g. if the data acquisition system stores
21         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
22         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
23         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
24         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
25     dtype: float32
26     name: conversion
27     required: false
28   - default_value: 0.0
29     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
30         to the specified 'unit'. Two common examples of this include (a) data stored
31         in an unsigned type that requires a shift after scaling to re-center the data,
32         and (b) specialized recording devices that naturally cause a scalar offset with
33         respect to the true units.
34     dtype: float32
35     name: offset
36     required: false
37   - default_value: -1.0
38     doc: Smallest meaningful difference between values in data, stored in the specified
39         by unit, e.g., the change in value of the least significant bit, or a larger
40         number if signal noise is known to be present. If unknown, use -1.0.
41     dtype: float32
42     name: resolution
43     required: false
44   - doc: Base unit of measurement for working with the data. Actual stored values
45       are not necessarily stored in these units. To access the data in these units,
46       multiply 'data' by 'conversion' and add 'offset'.

```

(continues on next page)

(continued from previous page)

```

47     dtype: text
48     name: unit
49     - doc: Optionally describe the continuity of the data. Can be "continuous",
    ↪ "instantaneous",
50       or "step". For example, a voltage trace would be "continuous", because samples
51       are recorded from a continuous process. An array of lick times would be
    ↪ "instantaneous",
52       because the data represents distinct moments in time. Times of image presentations
53       would be "step" because the picture remains the same until the next timepoint.
54       This field is optional, but is useful in providing information about the underlying
55       data. It may inform the way this data is interpreted, the way it is visualized,
56       and what analysis methods are applicable.
57     dtype: text
58     name: continuity
59     required: false
60     dims:
61     - - frame
62       - x
63       - y
64     - - frame
65       - x
66       - y
67       - z
68     doc: Binary data representing images across frames. If data are stored in an external
69         file, this should be an empty 3D array.
70     dtype: numeric
71     name: data
72     shape:
73     - -
74       -
75       -
76     - -
77       -
78       -
79     -
80     - dims:
81       - rank
82     doc: Number of pixels on x, y, (and z) axes.
83     dtype: int32
84     name: dimension
85     quantity: '?'
86     shape:
87     -
88     - attributes:
89       - dims:
90         - num_files
91     doc: Each external image may contain one or more consecutive frames of the full
92         ImageSeries. This attribute serves as an index to indicate which frames each
93         file contains, to facilitate random access. The 'starting_frame' attribute, hence,
94         contains a list of frame numbers within the full ImageSeries of the first frame
95         of each file listed in the parent 'external_file' dataset. Zero-based indexing
96         is used (hence, the first element will always be zero). For example, if the

```

(continues on next page)

(continued from previous page)

```

97     'external_file' dataset has three paths to files and the first file has 5 frames,
98     the second file has 10 frames, and the third file has 20 frames, then this_
↪ attribute
99     will have values [0, 5, 15]. If there is a single external file that holds all
100    of the frames of the ImageSeries (and so there is a single element in the
↪ 'external_file'
101    dataset), then this attribute should have value [0].
102    dtype: int32
103    name: starting_frame
104    shape:
105    -
106    dims:
107    - num_files
108    doc: Paths to one or more external file(s). The field is only present if format=
↪ 'external'.
109    This is only relevant if the image series is stored in the file system as one
110    or more image file(s). This field should NOT be used if the image is stored in
111    another NWB file and that file is linked to this file.
112    dtype: text
113    name: external_file
114    quantity: '?'
115    shape:
116    -
117    - default_value: raw
118    doc: Format of image. If this is 'external', then the attribute 'external_file'
119    contains the path information to the image files. If this is 'raw', then the raw
120    (single-channel) binary data is stored in the 'data' dataset. If this attribute
121    is not present, then the default format='raw' case is assumed.
122    dtype: text
123    name: format
124    quantity: '?'
125    - attributes:
126    - doc: Sampling rate, in Hz.
127      dtype: float32
128      name: rate
129    - doc: Unit of measurement for time, which is fixed to 'seconds'.
130      dtype: text
131      name: unit
132      value: seconds
133    doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
134    the timestamp of the first sample can be specified and all subsequent ones calculated
135    from the sampling rate attribute.
136    dtype: float64
137    name: starting_time
138    quantity: '?'
139    - attributes:
140    - doc: Value is '1'
141      dtype: int32
142      name: interval
143      value: 1
144    - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
145      dtype: text

```

(continues on next page)

(continued from previous page)

```

146     name: unit
147     value: seconds
148     dims:
149     - num_times
150     doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
151         master-clock stored in NWBFile.timestamps_reference_time.
152     dtype: float64
153     name: timestamps
154     quantity: '?'
155     shape:
156     -
157 - dims:
158     - num_times
159     doc: Numerical labels that apply to each time point in data for the purpose of querying
160         and slicing data by these values. If present, the length of this array should
161         be the same size as the first dimension of data.
162     dtype: uint8
163     name: control
164     quantity: '?'
165     shape:
166     -
167 - dims:
168     - num_control_values
169     doc: Description of each control value. Must be present if control is present. If
170         present, control_description[0] should describe time points where control == 0.
171     dtype: text
172     name: control_description
173     quantity: '?'
174     shape:
175     -
176     doc: General image data that is common between acquisition and stimulus time series.
177         Sometimes the image data is stored in the file in a raw format while other times
178         it will be stored as a series of external image files in the host file system. The
179         data field will either be binary data, if the data is stored in the NWB file, or
180         empty, if the data is stored in an external image stack. [frame][x][y] or_
↪ [frame][x][y][z].
181     groups:
182     - doc: Lab-specific time and sync information as provided directly from hardware devices
183         and that is necessary for aligning all acquired time information to a common_
↪ timebase.
184         The timestamp array stores time in the common timebase. This group will usually
185         only be populated in TimeSeries that are stored external to the NWB file, in files
186         storing raw data. Once timestamp data is calculated, the contents of 'sync' are
187         mostly for archival purposes.
188     name: sync
189     quantity: '?'
190     links:
191     - doc: Link to the Device object that was used to capture these images.
192     name: device
193     quantity: '?'
194     target_type: Device

```

(continues on next page)

(continued from previous page)

195 `neurodata_type_def`: ImageSeries
196 `neurodata_type_inc`: TimeSeries

5.5.5 ImageMaskSeries

Extends: *ImageSeries*

Description: see [Section 4.4.5](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - default_value: 1.0
17     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
18         If the data are stored in acquisition system units or other units that require
19         a conversion to be interpretable, multiply the data by 'conversion' to convert
20         the data to the specified 'unit'. e.g. if the data acquisition system stores
21         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
22         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
23         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
24         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
25     dtype: float32
26     name: conversion
27     required: false
28   - default_value: 0.0
29     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
30         to the specified 'unit'. Two common examples of this include (a) data stored
31         in an unsigned type that requires a shift after scaling to re-center the data,
32         and (b) specialized recording devices that naturally cause a scalar offset with
33         respect to the true units.
34     dtype: float32
35     name: offset
36     required: false
37   - default_value: -1.0
38     doc: Smallest meaningful difference between values in data, stored in the specified
39         by unit, e.g., the change in value of the least significant bit, or a larger
40         number if signal noise is known to be present. If unknown, use -1.0.
41     dtype: float32
42     name: resolution
43     required: false
44   - doc: Base unit of measurement for working with the data. Actual stored values
45       are not necessarily stored in these units. To access the data in these units,
46       multiply 'data' by 'conversion' and add 'offset'.

```

(continues on next page)

(continued from previous page)

```

47     dtype: text
48     name: unit
49     - doc: Optionally describe the continuity of the data. Can be "continuous",
    ↪ "instantaneous",
50       or "step". For example, a voltage trace would be "continuous", because samples
51       are recorded from a continuous process. An array of lick times would be
    ↪ "instantaneous",
52       because the data represents distinct moments in time. Times of image presentations
53       would be "step" because the picture remains the same until the next timepoint.
54       This field is optional, but is useful in providing information about the underlying
55       data. It may inform the way this data is interpreted, the way it is visualized,
56       and what analysis methods are applicable.
57     dtype: text
58     name: continuity
59     required: false
60     dims:
61     - - frame
62       - x
63       - y
64     - - frame
65       - x
66       - y
67       - z
68     doc: Binary data representing images across frames. If data are stored in an external
69         file, this should be an empty 3D array.
70     dtype: numeric
71     name: data
72     shape:
73     - -
74       -
75       -
76     - -
77       -
78       -
79     -
80     - dims:
81       - rank
82     doc: Number of pixels on x, y, (and z) axes.
83     dtype: int32
84     name: dimension
85     quantity: '?'
86     shape:
87     -
88     - attributes:
89       - dims:
90         - num_files
91     doc: Each external image may contain one or more consecutive frames of the full
92         ImageSeries. This attribute serves as an index to indicate which frames each
93         file contains, to facilitate random access. The 'starting_frame' attribute, hence,
94         contains a list of frame numbers within the full ImageSeries of the first frame
95         of each file listed in the parent 'external_file' dataset. Zero-based indexing
96         is used (hence, the first element will always be zero). For example, if the

```

(continues on next page)

(continued from previous page)

```

97     'external_file' dataset has three paths to files and the first file has 5 frames,
98     the second file has 10 frames, and the third file has 20 frames, then this
↪ attribute
99     will have values [0, 5, 15]. If there is a single external file that holds all
100    of the frames of the ImageSeries (and so there is a single element in the
↪ 'external_file'
101    dataset), then this attribute should have value [0].
102    dtype: int32
103    name: starting_frame
104    shape:
105    -
106    dims:
107    - num_files
108    doc: Paths to one or more external file(s). The field is only present if format=
↪ 'external'.
109    This is only relevant if the image series is stored in the file system as one
110    or more image file(s). This field should NOT be used if the image is stored in
111    another NWB file and that file is linked to this file.
112    dtype: text
113    name: external_file
114    quantity: '?'
115    shape:
116    -
117    - default_value: raw
118    doc: Format of image. If this is 'external', then the attribute 'external_file'
119    contains the path information to the image files. If this is 'raw', then the raw
120    (single-channel) binary data is stored in the 'data' dataset. If this attribute
121    is not present, then the default format='raw' case is assumed.
122    dtype: text
123    name: format
124    quantity: '?'
125    - attributes:
126    - doc: Sampling rate, in Hz.
127      dtype: float32
128      name: rate
129    - doc: Unit of measurement for time, which is fixed to 'seconds'.
130      dtype: text
131      name: unit
132      value: seconds
133    doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
134    the timestamp of the first sample can be specified and all subsequent ones calculated
135    from the sampling rate attribute.
136    dtype: float64
137    name: starting_time
138    quantity: '?'
139    - attributes:
140    - doc: Value is '1'
141      dtype: int32
142      name: interval
143      value: 1
144    - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
145      dtype: text

```

(continues on next page)

(continued from previous page)

```

146     name: unit
147     value: seconds
148     dims:
149     - num_times
150     doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
151         master-clock stored in NWBFile.timestamps_reference_time.
152     dtype: float64
153     name: timestamps
154     quantity: '?'
155     shape:
156     -
157 - dims:
158     - num_times
159     doc: Numerical labels that apply to each time point in data for the purpose of querying
160         and slicing data by these values. If present, the length of this array should
161         be the same size as the first dimension of data.
162     dtype: uint8
163     name: control
164     quantity: '?'
165     shape:
166     -
167 - dims:
168     - num_control_values
169     doc: Description of each control value. Must be present if control is present. If
170         present, control_description[0] should describe time points where control == 0.
171     dtype: text
172     name: control_description
173     quantity: '?'
174     shape:
175     -
176     doc: An alpha mask that is applied to a presented visual stimulus. The 'data' array
177         contains an array of mask values that are applied to the displayed image. Mask values
178         are stored as RGBA. Mask can vary with time. The timestamps array indicates the
179         starting time of a mask, and that mask pattern continues until it's explicitly changed.
180 groups:
181 - doc: Lab-specific time and sync information as provided directly from hardware devices
182     and that is necessary for aligning all acquired time information to a common_
↪ timebase.
183     The timestamp array stores time in the common timebase. This group will usually
184     only be populated in TimeSeries that are stored external to the NWB file, in files
185     storing raw data. Once timestamp data is calculated, the contents of 'sync' are
186     mostly for archival purposes.
187     name: sync
188     quantity: '?'
189 links:
190 - doc: Link to ImageSeries object that this image mask is applied to.
191     name: masked_imageseries
192     target_type: ImageSeries
193 - doc: Link to the Device object that was used to capture these images.
194     name: device
195     quantity: '?'

```

(continues on next page)

(continued from previous page)

```
196     target_type: Device
197 neurodata_type_def: ImageMaskSeries
198 neurodata_type_inc: ImageSeries
```

5.5.6 OpticalSeries

Extends: *ImageSeries*

Description: see [Section 4.4.6](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - doc: Distance from camera/monitor to target/eye.
16   dtype: float32
17   name: distance
18   quantity: '?'
19 - dims:
20   - width, height
21   - width, height, depth
22   doc: Width, height and depth of image, or imaged area, in meters.
23   dtype: float32
24   name: field_of_view
25   quantity: '?'
26   shape:
27   - 2
28   - 3
29 - attributes:
30 - default_value: 1.0
31   doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
32       If the data are stored in acquisition system units or other units that require
33       a conversion to be interpretable, multiply the data by 'conversion' to convert
34       the data to the specified 'unit'. e.g. if the data acquisition system stores
35       values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
36       that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
37       gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
38       values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
39   dtype: float32
40   name: conversion
41   required: false
42 - default_value: 0.0
43   doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
44       to the specified 'unit'. Two common examples of this include (a) data stored
45       in an unsigned type that requires a shift after scaling to re-center the data,
46       and (b) specialized recording devices that naturally cause a scalar offset with

```

(continues on next page)

(continued from previous page)

```

    respect to the true units.
dtype: float32
name: offset
required: false
- default_value: -1.0
doc: Smallest meaningful difference between values in data, stored in the specified
    by unit, e.g., the change in value of the least significant bit, or a larger
    number if signal noise is known to be present. If unknown, use -1.0.
dtype: float32
name: resolution
required: false
- doc: Base unit of measurement for working with the data. Actual stored values
    are not necessarily stored in these units. To access the data in these units,
    multiply 'data' by 'conversion' and add 'offset'.
dtype: text
name: unit
- doc: Optionally describe the continuity of the data. Can be "continuous",
    ↪ "instantaneous",
    or "step". For example, a voltage trace would be "continuous", because samples
    are recorded from a continuous process. An array of lick times would be
    ↪ "instantaneous",
    because the data represents distinct moments in time. Times of image presentations
    would be "step" because the picture remains the same until the next timepoint.
    This field is optional, but is useful in providing information about the underlying
    data. It may inform the way this data is interpreted, the way it is visualized,
    and what analysis methods are applicable.
dtype: text
name: continuity
required: false
dims:
- - frame
  - x
  - y
- - frame
  - x
  - y
  - r, g, b
doc: Images presented to subject, either grayscale or RGB
dtype: numeric
name: data
shape:
- -
- -
- -
- -
- -
- 3
- doc: Description of image relative to some reference frame (e.g., which way is up).
    Must also specify frame of reference.
dtype: text
name: orientation

```

(continues on next page)

(continued from previous page)

```

97  quantity: '?'
98  - dims:
99    - rank
100  doc: Number of pixels on x, y, (and z) axes.
101  dtype: int32
102  name: dimension
103  quantity: '?'
104  shape:
105    -
106  - attributes:
107    - dims:
108      - num_files
109      doc: Each external image may contain one or more consecutive frames of the full
110        ImageSeries. This attribute serves as an index to indicate which frames each
111        file contains, to facilitate random access. The 'starting_frame' attribute, hence,
112        contains a list of frame numbers within the full ImageSeries of the first frame
113        of each file listed in the parent 'external_file' dataset. Zero-based indexing
114        is used (hence, the first element will always be zero). For example, if the
115        'external_file' dataset has three paths to files and the first file has 5 frames,
116        the second file has 10 frames, and the third file has 20 frames, then this_
117  ↪ attribute
118    will have values [0, 5, 15]. If there is a single external file that holds all
119    of the frames of the ImageSeries (and so there is a single element in the
120  ↪ 'external_file'
121    dataset), then this attribute should have value [0].
122    dtype: int32
123    name: starting_frame
124    shape:
125      -
126    dims:
127      - num_files
128      doc: Paths to one or more external file(s). The field is only present if format=
129  ↪ 'external'.
130      This is only relevant if the image series is stored in the file system as one
131      or more image file(s). This field should NOT be used if the image is stored in
132      another NWB file and that file is linked to this file.
133      dtype: text
134      name: external_file
135      quantity: '?'
136      shape:
137        -
138      - default_value: raw
139      doc: Format of image. If this is 'external', then the attribute 'external_file'
140        contains the path information to the image files. If this is 'raw', then the raw
141        (single-channel) binary data is stored in the 'data' dataset. If this attribute
142        is not present, then the default format='raw' case is assumed.
143      dtype: text
144      name: format
145      quantity: '?'
146  - attributes:
147    - doc: Sampling rate, in Hz.
148      dtype: float32

```

(continues on next page)

(continued from previous page)

```

146     name: rate
147 - doc: Unit of measurement for time, which is fixed to 'seconds'.
148     dtype: text
149     name: unit
150     value: seconds
151 doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
152     the timestamp of the first sample can be specified and all subsequent ones calculated
153     from the sampling rate attribute.
154 dtype: float64
155 name: starting_time
156 quantity: '?'
157 - attributes:
158 - doc: Value is '1'
159     dtype: int32
160     name: interval
161     value: 1
162 - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
163     dtype: text
164     name: unit
165     value: seconds
166 dims:
167 - num_times
168 doc: Timestamps for samples stored in data, in seconds, relative to the common_
169     ↪ experiment
170     master-clock stored in NWBFile.timestamps_reference_time.
171 dtype: float64
172 name: timestamps
173 quantity: '?'
174 shape:
175 -
176 - dims:
177 - num_times
178 doc: Numerical labels that apply to each time point in data for the purpose of querying
179     and slicing data by these values. If present, the length of this array should
180     be the same size as the first dimension of data.
181 dtype: uint8
182 name: control
183 quantity: '?'
184 shape:
185 -
186 - dims:
187 - num_control_values
188 doc: Description of each control value. Must be present if control is present. If
189     present, control_description[0] should describe time points where control == 0.
190 dtype: text
191 name: control_description
192 quantity: '?'
193 shape:
194 -
195 doc: Image data that is presented or recorded. A stimulus template movie will be stored
196     only as an image. When the image is presented as stimulus, additional data is required,
197     such as field of view (e.g., how much of the visual field the image covers, or how

```

(continues on next page)

(continued from previous page)

what is the area of the target being imaged). If the `OpticalSeries` represents acquired imaging data, orientation is also important.

groups:

- **doc:** Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common ↵
↵timebase.

The timestamp array stores time in the common timebase. This group will usually only be populated in `TimeSeries` that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

name: sync

quantity: '?'

links:

- **doc:** Link to the Device object that was used to capture these images.

name: device

quantity: '?'

target_type: Device

neurodata_type_def: `OpticalSeries`

neurodata_type_inc: `ImageSeries`

5.5.7 IndexSeries

Extends: *TimeSeries*

Description: see [Section 4.4.7](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - doc: This field is unused by IndexSeries.
17     dtype: float32
18     name: conversion
19     required: false
20   - doc: This field is unused by IndexSeries.
21     dtype: float32
22     name: resolution
23     required: false
24   - doc: This field is unused by IndexSeries.
25     dtype: float32
26     name: offset
27     required: false
28   - doc: This field is unused by IndexSeries and has the value N/A.
29     dtype: text
30     name: unit
31     value: N/A
32   - doc: Optionally describe the continuity of the data. Can be "continuous",
33     ↳ "instantaneous",
34       or "step". For example, a voltage trace would be "continuous", because samples
35       are recorded from a continuous process. An array of lick times would be
36     ↳ "instantaneous",
37       because the data represents distinct moments in time. Times of image presentations
38       would be "step" because the picture remains the same until the next timepoint.
39       This field is optional, but is useful in providing information about the underlying
40       data. It may inform the way this data is interpreted, the way it is visualized,
41       and what analysis methods are applicable.
42     dtype: text
43     name: continuity
44     required: false
45 dims:
46 - num_times

```

(continues on next page)

(continued from previous page)

```

45  doc: Index of the image (using zero-indexing) in the linked Images object.
46  dtype: uint32
47  name: data
48  shape:
49  -
50  - attributes:
51    - doc: Sampling rate, in Hz.
52      dtype: float32
53      name: rate
54    - doc: Unit of measurement for time, which is fixed to 'seconds'.
55      dtype: text
56      name: unit
57      value: seconds
58  doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
59      the timestamp of the first sample can be specified and all subsequent ones calculated
60      from the sampling rate attribute.
61  dtype: float64
62  name: starting_time
63  quantity: '?'
64  - attributes:
65    - doc: Value is '1'
66      dtype: int32
67      name: interval
68      value: 1
69    - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
70      dtype: text
71      name: unit
72      value: seconds
73  dims:
74  - num_times
75  doc: Timestamps for samples stored in data, in seconds, relative to the common_
76  ↪ experiment
77      master-clock stored in NWBFile.timestamps_reference_time.
78  dtype: float64
79  name: timestamps
80  quantity: '?'
81  shape:
82  -
83  - dims:
84    - num_times
85  doc: Numerical labels that apply to each time point in data for the purpose of querying
86      and slicing data by these values. If present, the length of this array should
87      be the same size as the first dimension of data.
88  dtype: uint8
89  name: control
90  quantity: '?'
91  shape:
92  -
93  - dims:
94    - num_control_values
95  doc: Description of each control value. Must be present if control is present. If
      present, control_description[0] should describe time points where control == 0.

```

(continues on next page)

(continued from previous page)

```

96 dtype: text
97 name: control_description
98 quantity: '?'
99 shape:
100 -
101 doc: Stores indices to image frames stored in an ImageSeries. The purpose of the
    ↳ IndexSeries
102 is to allow a static image stack to be stored in an Images object, and the images
103 in the stack to be referenced out-of-order. This can be for the display of individual
104 images, or of movie segments (as a movie is simply a series of images). The data
105 field stores the index of the frame in the referenced Images object, and the timestamps
106 array indicates when that image was displayed.
107 groups:
108 - doc: Lab-specific time and sync information as provided directly from hardware devices
109   and that is necessary for aligning all acquired time information to a common
    ↳ timebase.
110   The timestamp array stores time in the common timebase. This group will usually
111   only be populated in TimeSeries that are stored external to the NWB file, in files
112   storing raw data. Once timestamp data is calculated, the contents of 'sync' are
113   mostly for archival purposes.
114   name: sync
115   quantity: '?'
116 links:
117 - doc: Link to ImageSeries object containing images that are indexed. Use of this
118   link is discouraged and will be deprecated. Link to an Images type instead.
119   name: indexed_timeseries
120   quantity: '?'
121   target_type: ImageSeries
122 - doc: Link to Images object containing an ordered set of images that are indexed.
123   The Images object must contain a 'ordered_images' dataset specifying the order
124   of the images in the Images type.
125   name: indexed_images
126   quantity: '?'
127   target_type: Images
128 neurodata_type_def: IndexSeries
129 neurodata_type_inc: TimeSeries

```

5.6 NWB file

Main NWB file specification.

5.6.1 ScratchData

Extends: *NWBData*

Description: see [Section 4.5.1](#)

YAML Specification:

```
1 attributes:
2 - doc: Any notes the user has about the dataset being stored
3   dtype: text
4   name: notes
5 doc: Any one-off datasets
6 neurodata_type_def: ScratchData
7 neurodata_type_inc: NWBData
```

5.6.2 NWBFile

Extends: *NWBContainer*

Description: see Section 4.5.2

YAML Specification:

```

1 attributes:
2 - doc: File version string. Use semantic versioning, e.g. 1.2.1. This will be the
3   name of the format with trailing major, minor and patch numbers.
4   dtype: text
5   name: nwb_version
6   value: 2.6.0
7 datasets:
8 - dims:
9   - num_modifications
10  doc: 'A record of the date the file was created and of subsequent modifications.
11    The date is stored in UTC with local timezone offset as ISO 8601 extended formatted
12    strings: 2018-09-28T14:43:54.123+02:00. Dates stored in UTC end in "Z" with no
13    timezone offset. Date accuracy is up to milliseconds. The file can be created
14    after the experiment was run, so this may differ from the experiment start time.
15    Each modification to the nwb file adds a new entry to the array.'
16  dtype: isodatettime
17  name: file_create_date
18  shape:
19    -
20 - doc: A unique text identifier for the file. For example, concatenated lab name,
21   file creation date/time and experimentalist, or a hash of these and/or other values.
22   The goal is that the string should be unique to all other files.
23  dtype: text
24  name: identifier
25 - doc: A description of the experimental session and data in the file.
26  dtype: text
27  name: session_description
28 - doc: 'Date and time of the experiment/session start. The date is stored in UTC with
29   local timezone offset as ISO 8601 extended formatted string: 2018-09-
30   ↪123+02:00.
31   Dates stored in UTC end in "Z" with no timezone offset. Date accuracy is up to
32   milliseconds.'
33  dtype: isodatettime
34  name: session_start_time
35 - doc: 'Date and time corresponding to time zero of all timestamps. The date is stored
36   in UTC with local timezone offset as ISO 8601 extended formatted string: 2018-09-
37   ↪28T14:43:54.123+02:00.
38   Dates stored in UTC end in "Z" with no timezone offset. Date accuracy is up to
39   milliseconds. All times stored in the file use this time as reference (i.e., time
40   zero).'
41  dtype: isodatettime
42  name: timestamps_reference_time
43 doc: An NWB file storing cellular-based neurophysiology data from a single experimental
44 session.
45 groups:
46 - doc: Data streams recorded from the system, including ephys, ophys, tracking, etc.

```

(continues on next page)

(continued from previous page)

This group should be read-only after the experiment is completed and timestamps are corrected to a common timebase. The data stored here may be links to raw data stored in external NWB files. This will allow keeping bulky raw data out of the file while preserving the option of keeping some/all in the file. Acquired data includes tracking and experimental data streams (i.e., everything measured from the system). If bulky data is stored in the /acquisition group, the data can exist in a separate NWB file that is linked to by the file being used for processing and analysis.

groups:

- **doc:** Acquired, raw data.
neurodata_type_inc: NWBDataInterface
quantity: '*'
- **doc:** Tabular data that is relevant to acquisition
neurodata_type_inc: DynamicTable
quantity: '*'

name: acquisition

- **doc:** Lab-specific and custom scientific analysis of data. There is no defined format for the content of this group - the format is up to the individual user/lab. To facilitate sharing analysis data between labs, the contents here should be stored in standard types (e.g., neurodata_types) and appropriately documented. The file can store lab-specific and custom data analysis without restriction on its form or schema, reducing data formatting restrictions on end users. Such data should be placed in the analysis group. The analysis data should be documented so that it could be shared with other labs.

groups:

- **doc:** Custom analysis results.
neurodata_type_inc: NWBContainer
quantity: '*'
- **doc:** Tabular data that is relevant to data stored in analysis
neurodata_type_inc: DynamicTable
quantity: '*'

name: analysis**datasets:**

- **doc:** Any one-off datasets
neurodata_type_inc: ScratchData
quantity: '*'

doc: A place to store one-off analysis results. Data placed here is not intended for sharing. By placing data here, users acknowledge that there is no guarantee that their data meets any standard.

groups:

- **doc:** Any one-off containers
neurodata_type_inc: NWBContainer
quantity: '*'
- **doc:** Any one-off tables
neurodata_type_inc: DynamicTable
quantity: '*'

name: scratch**quantity:** '?'

- **doc:** The home for ProcessingModules. These modules perform intermediate analysis of data that is necessary to perform before scientific analysis. Examples include spike clustering, extracting position from tracking data, stitching together image slices. ProcessingModules can be large and express many data sets from relatively

(continues on next page)

(continued from previous page)

```

97     complex analysis (e.g., spike detection and clustering) or small, representing
98     extraction of position information from tracking video, or even binary lick/no-lick
99     decisions. Common software tools (e.g., klustakwik, MClust) are expected to read/
100     ↪write
101     data here. 'Processing' refers to intermediate analysis of the acquired data
102     to make it more amenable to scientific analysis.
103     groups:
104     - doc: Intermediate analysis of acquired data.
105       neurodata_type_inc: ProcessingModule
106       quantity: '*'
107       name: processing
108     - doc: Data pushed into the system (eg, video stimulus, sound, voltage, etc) and ↪
109     ↪secondary
110     representations of that data (eg, measurements of something used as a stimulus).
111     This group should be made read-only after experiment complete and timestamps are
112     corrected to common timebase. Stores both presented stimuli and stimulus templates,
113     the latter in case the same stimulus is presented multiple times, or is pulled
114     from an external stimulus library. Stimuli are here defined as any signal that
115     is pushed into the system as part of the experiment (eg, sound, video, voltage,
116     etc). Many different experiments can use the same stimuli, and stimuli can be
117     re-used during an experiment. The stimulus group is organized so that one version
118     of template stimuli can be stored and these be used multiple times. These templates
119     can exist in the present file or can be linked to a remote library file.
120     groups:
121     - doc: Stimuli presented during the experiment.
122       groups:
123       - doc: TimeSeries objects containing data of presented stimuli.
124         neurodata_type_inc: TimeSeries
125         quantity: '*'
126         name: presentation
127       - doc: Template stimuli. Timestamps in templates are based on stimulus design and
128         are relative to the beginning of the stimulus. When templates are used, the
129         stimulus instances must convert presentation times to the experiment's time
130         reference frame.
131       groups:
132       - doc: TimeSeries objects containing template data of presented stimuli.
133         neurodata_type_inc: TimeSeries
134         quantity: '*'
135       - doc: Images objects containing images of presented stimuli.
136         neurodata_type_inc: Images
137         quantity: '*'
138         name: templates
139       name: stimulus
140     - datasets:
141     - doc: Notes about data collection and analysis.
142       dtype: text
143       name: data_collection
144       quantity: '?'
145     - doc: General description of the experiment.
146       dtype: text
147       name: experiment_description
148       quantity: '?'

```

(continues on next page)

(continued from previous page)

```

147 - dims:
148   - num_experimenters
149   doc: Name of person(s) who performed the experiment. Can also specify roles of
150       different people involved.
151   dtype: text
152   name: experimenter
153   quantity: '?'
154   shape:
155   -
156 - doc: Institution(s) where experiment was performed.
157   dtype: text
158   name: institution
159   quantity: '?'
160 - dims:
161   - num_keywords
162   doc: Terms to search over.
163   dtype: text
164   name: keywords
165   quantity: '?'
166   shape:
167   -
168 - doc: Laboratory where experiment was performed.
169   dtype: text
170   name: lab
171   quantity: '?'
172 - doc: Notes about the experiment.
173   dtype: text
174   name: notes
175   quantity: '?'
176 - doc: Description of drugs used, including how and when they were administered.
177       Anesthesia(s), painkiller(s), etc., plus dosage, concentration, etc.
178   dtype: text
179   name: pharmacology
180   quantity: '?'
181 - doc: Experimental protocol, if applicable. e.g., include IACUC protocol number.
182   dtype: text
183   name: protocol
184   quantity: '?'
185 - dims:
186   - num_publications
187   doc: Publication information. PMID, DOI, URL, etc.
188   dtype: text
189   name: related_publications
190   quantity: '?'
191   shape:
192   -
193 - doc: Lab-specific ID for the session.
194   dtype: text
195   name: session_id
196   quantity: '?'
197 - doc: Description of slices, including information about preparation thickness,
198       orientation, temperature, and bath solution.

```

(continues on next page)

(continued from previous page)

```

199     dtype: text
200     name: slices
201     quantity: '?'
202 -   attributes:
203     - doc: Name of script file.
204       dtype: text
205       name: file_name
206     doc: Script file or link to public source code used to create this NWB file.
207     dtype: text
208     name: source_script
209     quantity: '?'
210 -   doc: Notes about stimuli, such as how and where they were presented.
211     dtype: text
212     name: stimulus
213     quantity: '?'
214 -   doc: Narrative description about surgery/surgeries, including date(s) and who
215         performed surgery.
216     dtype: text
217     name: surgery
218     quantity: '?'
219 -   doc: Information about virus(es) used in experiments, including virus ID, source,
220         date made, injection location, volume, etc.
221     dtype: text
222     name: virus
223     quantity: '?'
224 doc: Experimental metadata, including protocol, notes and description of hardware
225     device(s). The metadata stored in this section should be used to describe the
226     experiment. Metadata necessary for interpreting the data is stored with the data.
227     General experimental metadata, including animal strain, experimental protocols,
228     experimenter, devices, etc, are stored under 'general'. Core metadata (e.g., that
229     required to interpret data fields) is stored with the data itself, and implicitly
230     defined by the file specification (e.g., time is in seconds). The strategy used
231     here for storing non-core metadata is to use free-form text fields, such as would
232     appear in sentences or paragraphs from a Methods section. Metadata fields are
233     text to enable them to be more general, for example to represent ranges instead
234     of numerical values. Machine-readable metadata is stored as attributes to these
235     free-form datasets. All entries in the below table are to be included when data
236     is present. Unused groups (e.g., intracellular_ephys in an optophysiology experiment)
237     should not be created unless there is data to store within them.
238 groups:
239 -   doc: Place-holder than can be extended so that lab-specific meta-data can be placed
240         in /general.
241     neurodata_type_inc: LabMetaData
242     quantity: '*'
243 -   doc: Description of hardware devices used during experiment, e.g., monitors, ADC
244         boards, microscopes, etc.
245     groups:
246     - doc: Data acquisition devices.
247       neurodata_type_inc: Device
248       quantity: '*'
249     name: devices
250     quantity: '?'

```

(continues on next page)

(continued from previous page)

```

251 - doc: Information about the animal or person from which the data was measured.
252   name: subject
253   neurodata_type_inc: Subject
254   quantity: '?'
255 - doc: Metadata related to extracellular electrophysiology.
256   groups:
257   - doc: Physical group of electrodes.
258     neurodata_type_inc: ElectrodeGroup
259     quantity: '*'
260   - datasets:
261     - doc: x coordinate of the channel location in the brain (+x is posterior).
262       dtype: float32
263       name: x
264       neurodata_type_inc: VectorData
265       quantity: '?'
266     - doc: y coordinate of the channel location in the brain (+y is inferior).
267       dtype: float32
268       name: y
269       neurodata_type_inc: VectorData
270       quantity: '?'
271     - doc: z coordinate of the channel location in the brain (+z is right).
272       dtype: float32
273       name: z
274       neurodata_type_inc: VectorData
275       quantity: '?'
276     - doc: Impedance of the channel, in ohms.
277       dtype: float32
278       name: imp
279       neurodata_type_inc: VectorData
280       quantity: '?'
281     - doc: Location of the electrode (channel). Specify the area, layer, comments
282       on estimation of area/layer, stereotaxic coordinates if in vivo, etc. Use
283       standard atlas names for anatomical regions when possible.
284       dtype: text
285       name: location
286       neurodata_type_inc: VectorData
287     - doc: Description of hardware filtering, including the filter name and frequency
288       cutoffs.
289       dtype: text
290       name: filtering
291       neurodata_type_inc: VectorData
292       quantity: '?'
293     - doc: Reference to the ElectrodeGroup this electrode is a part of.
294       dtype:
295         reftype: object
296         target_type: ElectrodeGroup
297       name: group
298       neurodata_type_inc: VectorData
299     - doc: Name of the ElectrodeGroup this electrode is a part of.
300       dtype: text
301       name: group_name
302       neurodata_type_inc: VectorData

```

(continues on next page)

(continued from previous page)

```

303     - doc: x coordinate in electrode group
304       dtype: float32
305       name: rel_x
306       neurodata_type_inc: VectorData
307       quantity: '?'
308     - doc: y coordinate in electrode group
309       dtype: float32
310       name: rel_y
311       neurodata_type_inc: VectorData
312       quantity: '?'
313     - doc: z coordinate in electrode group
314       dtype: float32
315       name: rel_z
316       neurodata_type_inc: VectorData
317       quantity: '?'
318     - doc: Description of the reference electrode and/or reference scheme used for
319       this electrode, e.g., "stainless steel skull screw" or "online common average
320       referencing".
321       dtype: text
322       name: reference
323       neurodata_type_inc: VectorData
324       quantity: '?'
325     doc: A table of all electrodes (i.e. channels) used for recording.
326     name: electrodes
327     neurodata_type_inc: DynamicTable
328     quantity: '?'
329     name: extracellular_ephys
330     quantity: '?'
331   - datasets:
332     - doc: '[DEPRECATED] Use IntracellularElectrode.filtering instead. Description
333       of filtering used. Includes filtering type and parameters, frequency fall-off,
334       etc. If this changes between TimeSeries, filter description should be stored
335       as a text attribute for each TimeSeries.'
336       dtype: text
337       name: filtering
338       quantity: '?'
339     doc: Metadata related to intracellular electrophysiology.
340     groups:
341     - doc: An intracellular electrode.
342       neurodata_type_inc: IntracellularElectrode
343       quantity: '*'
344     - doc: '[DEPRECATED] Table used to group different PatchClampSeries. SweepTable
345       is being replaced by IntracellularRecordingsTable and SimultaneousRecordingsTable
346       tabs. Additional SequentialRecordingsTable, RepetitionsTable and
347       ↪ ExperimentalConditions
348       tables provide enhanced support for experiment metadata.'
349       name: sweep_table
350       neurodata_type_inc: SweepTable
351       quantity: '?'
352     - doc: A table to group together a stimulus and response from a single electrode
353       and a single simultaneous recording. Each row in the table represents a single
       recording consisting typically of a stimulus and a corresponding response.

```

(continues on next page)

(continued from previous page)

In some cases, however, only a stimulus or a response are recorded as part of an experiment. In this case both, the stimulus and response will point to the same TimeSeries while the `idx_start` and `count` of the `invalid` column will be set to -1, thus, indicating that no values have been recorded for the stimulus or response, respectively. Note, a recording MUST contain at least a stimulus or a response. Typically the stimulus and response are `PatchClampSeries`.

However, the use of AD/DA channels that are not associated to an electrode is also common in intracellular electrophysiology, in which case other TimeSeries may be used.

name: `intracellular_recordings`
neurodata_type_inc: `IntracellularRecordingsTable`
quantity: `'?'`

- **doc:** A table for grouping different intracellular recordings from the `IntracellularRecordingsTable` table together that were recorded simultaneously from different electrodes

name: `simultaneous_recordings`
neurodata_type_inc: `SimultaneousRecordingsTable`
quantity: `'?'`

- **doc:** A table for grouping different sequential recordings from the `SimultaneousRecordingsTable` table together. This is typically used to group together sequential recordings where the a sequence of stimuli of the same type with varying parameters have been presented in a sequence.

name: `sequential_recordings`
neurodata_type_inc: `SequentialRecordingsTable`
quantity: `'?'`

- **doc:** A table for grouping different sequential intracellular recordings together. With each `SequentialRecording` typically representing a particular type of stimulus, the `RepetitionsTable` table is typically used to group sets of stimuli applied in sequence.

name: `repetitions`
neurodata_type_inc: `RepetitionsTable`
quantity: `'?'`

- **doc:** A table for grouping different intracellular recording repetitions together that belong to the same experimental `experimental_conditions`.

name: `experimental_conditions`
neurodata_type_inc: `ExperimentalConditionsTable`
quantity: `'?'`

name: `intracellular_ephys`
quantity: `'?'`

- **doc:** Metadata describing optogenetic stimulation.

groups:

- **doc:** An optogenetic stimulation site.
neurodata_type_inc: `OptogeneticStimulusSite`
quantity: `'*'`

name: `optogenetics`
quantity: `'?'`

- **doc:** Metadata related to optophysiology.

groups:

- **doc:** An imaging plane.
neurodata_type_inc: `ImagingPlane`

(continues on next page)

(continued from previous page)

```

403     quantity: '*'
404     name: optophysiology
405     quantity: '?'
406     name: general
407 - doc: Experimental intervals, whether that be logically distinct sub-experiments
408     having a particular scientific goal, trials (see trials subgroup) during an
409     experiment,
410     or epochs (see epochs subgroup) deriving from analysis of data.
411 groups:
412 - doc: Divisions in time marking experimental stages or sub-divisions of a single
413     recording session.
414     name: epochs
415     neurodata_type_inc: TimeIntervals
416     quantity: '?'
417 - doc: Repeated experimental events that have a logical grouping.
418     name: trials
419     neurodata_type_inc: TimeIntervals
420     quantity: '?'
421 - doc: Time intervals that should be removed from analysis.
422     name: invalid_times
423     neurodata_type_inc: TimeIntervals
424     quantity: '?'
425 - doc: Optional additional table(s) for describing other experimental time intervals.
426     neurodata_type_inc: TimeIntervals
427     quantity: '*'
428     name: intervals
429     quantity: '?'
430 - doc: Data about sorted spike units.
431     name: units
432     neurodata_type_inc: Units
433     quantity: '?'
434 name: root
435 neurodata_type_def: NWBFile
436 neurodata_type_inc: NWBContainer

```

5.6.3 LabMetaData

Extends: *NWBContainer*

Description: see [Section 4.5.3](#)

YAML Specification:

```
1 doc: Lab-specific meta-data.  
2 neurodata_type_def: LabMetaData  
3 neurodata_type_inc: NWBContainer
```

5.6.4 Subject

Extends: *NWBContainer*

Description: see [Section 4.5.4](#)

YAML Specification:

```

1 datasets:
2   - attributes:
3     - default_value: birth
4       doc: Age is with reference to this event. Can be 'birth' or 'gestational'. If
5         reference is omitted, 'birth' is implied.
6       dtype: text
7       name: reference
8       required: false
9     doc: Age of subject. Can be supplied instead of 'date_of_birth'.
10    dtype: text
11    name: age
12    quantity: '?'
13  - doc: Date of birth of subject. Can be supplied instead of 'age'.
14    dtype: isodatetime
15    name: date_of_birth
16    quantity: '?'
17  - doc: Description of subject and where subject came from (e.g., breeder, if animal).
18    dtype: text
19    name: description
20    quantity: '?'
21  - doc: Genetic strain. If absent, assume Wild Type (WT).
22    dtype: text
23    name: genotype
24    quantity: '?'
25  - doc: Gender of subject.
26    dtype: text
27    name: sex
28    quantity: '?'
29  - doc: Species of subject.
30    dtype: text
31    name: species
32    quantity: '?'
33  - doc: Strain of subject.
34    dtype: text
35    name: strain
36    quantity: '?'
37  - doc: ID of animal/person used/participating in experiment (lab convention).
38    dtype: text
39    name: subject_id
40    quantity: '?'
41  - doc: Weight at time of experiment, at time of surgery and at other important times.
42    dtype: text
43    name: weight
44    quantity: '?'
45  doc: Information about the animal or person from which the data was measured.
46  neurodata_type_def: Subject
47  neurodata_type_inc: NWBContainer

```

5.7 Miscellaneous neurodata_types.

Miscellaneous types.

5.7.1 AbstractFeatureSeries

Extends: *TimeSeries*

Description: see [Section 4.6.1](#)

YAML Specification:

```

1  attributes:
2  - default_value: no description
3    doc: Description of the time series.
4    dtype: text
5    name: description
6    required: false
7  - default_value: no comments
8    doc: Human-readable comments about the TimeSeries. This second descriptive field
9        can be used to store additional information, or descriptive information if the
10       primary description field is populated with a computer-readable string.
11    dtype: text
12    name: comments
13    required: false
14 datasets:
15 - attributes:
16   - default_value: see 'feature_units'
17     doc: Since there can be different units for different features, store the units
18         in 'feature_units'. The default value for this attribute is "see 'feature_units'".
19     dtype: text
20     name: unit
21     required: false
22   - default_value: 1.0
23     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
24         If the data are stored in acquisition system units or other units that require
25         a conversion to be interpretable, multiply the data by 'conversion' to convert
26         the data to the specified 'unit'. e.g. if the data acquisition system stores
27         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
28         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
29         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
30         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
31     dtype: float32
32     name: conversion
33     required: false
34   - default_value: 0.0
35     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
36         to the specified 'unit'. Two common examples of this include (a) data stored
37         in an unsigned type that requires a shift after scaling to re-center the data,
38         and (b) specialized recording devices that naturally cause a scalar offset with
39         respect to the true units.
40     dtype: float32
41     name: offset

```

(continues on next page)

(continued from previous page)

```

42     required: false
43 -   default_value: -1.0
44     doc: Smallest meaningful difference between values in data, stored in the specified
45         by unit, e.g., the change in value of the least significant bit, or a larger
46         number if signal noise is known to be present. If unknown, use -1.0.
47     dtype: float32
48     name: resolution
49     required: false
50 -   doc: Optionally describe the continuity of the data. Can be "continuous",
51     ↪ "instantaneous",
52     or "step". For example, a voltage trace would be "continuous", because samples
53     are recorded from a continuous process. An array of lick times would be
54     ↪ "instantaneous",
55     because the data represents distinct moments in time. Times of image presentations
56     would be "step" because the picture remains the same until the next timepoint.
57     This field is optional, but is useful in providing information about the underlying
58     data. It may inform the way this data is interpreted, the way it is visualized,
59     and what analysis methods are applicable.
60     dtype: text
61     name: continuity
62     required: false
63     dims:
64     - - num_times
65     - - num_times
66     - - num_features
67     doc: Values of each feature at each time.
68     dtype: numeric
69     name: data
70     shape:
71     - -
72     - -
73     - -
74 -   dims:
75     - num_features
76     doc: Units of each feature.
77     dtype: text
78     name: feature_units
79     quantity: '?'
80     shape:
81     -
82 -   dims:
83     - num_features
84     doc: Description of the features represented in TimeSeries::data.
85     dtype: text
86     name: features
87     shape:
88     -
89 -   attributes:
90     - doc: Sampling rate, in Hz.
91       dtype: float32
92       name: rate
93     - doc: Unit of measurement for time, which is fixed to 'seconds'.

```

(continues on next page)

(continued from previous page)

```

92     dtype: text
93     name: unit
94     value: seconds
95     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
96         the timestamp of the first sample can be specified and all subsequent ones calculated
97         from the sampling rate attribute.
98     dtype: float64
99     name: starting_time
100    quantity: '?'
101    - attributes:
102      - doc: Value is '1'
103        dtype: int32
104        name: interval
105        value: 1
106      - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
107        dtype: text
108        name: unit
109        value: seconds
110    dims:
111      - num_times
112    doc: Timestamps for samples stored in data, in seconds, relative to the common_
113    ↪ experiment
114       master-clock stored in NWBFile.timestamps_reference_time.
115    dtype: float64
116    name: timestamps
117    quantity: '?'
118    shape:
119      -
120    - dims:
121      - num_times
122    doc: Numerical labels that apply to each time point in data for the purpose of querying
123        and slicing data by these values. If present, the length of this array should
124        be the same size as the first dimension of data.
125    dtype: uint8
126    name: control
127    quantity: '?'
128    shape:
129      -
130    - dims:
131      - num_control_values
132    doc: Description of each control value. Must be present if control is present. If
133        present, control_description[0] should describe time points where control == 0.
134    dtype: text
135    name: control_description
136    quantity: '?'
137    shape:
138      -
139    doc: Abstract features, such as quantitative descriptions of sensory stimuli. The
140        TimeSeries::data field is a 2D array, storing those features (e.g., for visual grating
141        stimulus this might be orientation, spatial frequency and contrast). Null stimuli
142        (eg, uniform gray) can be marked as being an independent feature (eg, 1.0 for gray,
143        0.0 for actual stimulus) or by storing NaNs for feature values, or through use of

```

(continues on next page)

(continued from previous page)

the `TimeSeries::control` fields. A set of features is considered to persist until the next set of features is defined. The final set of features stored should be the null set. This is useful when storing the raw stimulus is impractical.

groups:

- **doc:** Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common ↵
↵timebase.

The timestamp array stores time in the common timebase. This group will usually only be populated in `TimeSeries` that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

name: sync

quantity: '?'

neurodata_type_def: `AbstractFeatureSeries`

neurodata_type_inc: `TimeSeries`

5.7.2 AnnotationSeries

Extends: *TimeSeries*

Description: see [Section 4.6.2](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - doc: Smallest meaningful difference between values in data. Annotations have no
17       units, so the value is fixed to -1.0.
18     dtype: float32
19     name: resolution
20     value: -1.0
21   - doc: Base unit of measurement for working with the data. Annotations have no units,
22       so the value is fixed to 'n/a'.
23     dtype: text
24     name: unit
25     value: n/a
26   - default_value: 1.0
27     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
28         If the data are stored in acquisition system units or other units that require
29         a conversion to be interpretable, multiply the data by 'conversion' to convert
30         the data to the specified 'unit'. e.g. if the data acquisition system stores
31         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
32         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
33         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
34         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
35     dtype: float32
36     name: conversion
37     required: false
38   - default_value: 0.0
39     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
40         to the specified 'unit'. Two common examples of this include (a) data stored
41         in an unsigned type that requires a shift after scaling to re-center the data,
42         and (b) specialized recording devices that naturally cause a scalar offset with
43         respect to the true units.
44     dtype: float32
45     name: offset
46     required: false

```

(continues on next page)

(continued from previous page)

```

47 - doc: Optionally describe the continuity of the data. Can be "continuous",
↳ "instantaneous",
48     or "step". For example, a voltage trace would be "continuous", because samples
49     are recorded from a continuous process. An array of lick times would be
↳ "instantaneous",
50     because the data represents distinct moments in time. Times of image presentations
51     would be "step" because the picture remains the same until the next timepoint.
52     This field is optional, but is useful in providing information about the underlying
53     data. It may inform the way this data is interpreted, the way it is visualized,
54     and what analysis methods are applicable.
55     dtype: text
56     name: continuity
57     required: false
58     dims:
59     - num_times
60     doc: Annotations made during an experiment.
61     dtype: text
62     name: data
63     shape:
64     -
65 - attributes:
66 - doc: Sampling rate, in Hz.
67     dtype: float32
68     name: rate
69 - doc: Unit of measurement for time, which is fixed to 'seconds'.
70     dtype: text
71     name: unit
72     value: seconds
73     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
74     the timestamp of the first sample can be specified and all subsequent ones calculated
75     from the sampling rate attribute.
76     dtype: float64
77     name: starting_time
78     quantity: '?'
79 - attributes:
80 - doc: Value is '1'
81     dtype: int32
82     name: interval
83     value: 1
84 - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
85     dtype: text
86     name: unit
87     value: seconds
88     dims:
89     - num_times
90     doc: Timestamps for samples stored in data, in seconds, relative to the common
↳ experiment
91     master-clock stored in NWBFile.timestamps_reference_time.
92     dtype: float64
93     name: timestamps
94     quantity: '?'
95     shape:

```

(continues on next page)

(continued from previous page)

```

96 -
97 - dims:
98   - num_times
99   doc: Numerical labels that apply to each time point in data for the purpose of querying
100     and slicing data by these values. If present, the length of this array should
101     be the same size as the first dimension of data.
102   dtype: uint8
103   name: control
104   quantity: '?'
105   shape:
106 -
107 - dims:
108   - num_control_values
109   doc: Description of each control value. Must be present if control is present. If
110     present, control_description[0] should describe time points where control == 0.
111   dtype: text
112   name: control_description
113   quantity: '?'
114   shape:
115 -
116 doc: Stores user annotations made during an experiment. The data[] field stores a
117   text array, and timestamps are stored for each annotation (ie, interval=1). This
118   is largely an alias to a standard TimeSeries storing a text array but that is
119   ↪ identifiable
120   as storing annotations in a machine-readable way.
121 groups:
122 - doc: Lab-specific time and sync information as provided directly from hardware devices
123   and that is necessary for aligning all acquired time information to a common
124   ↪ timebase.
125   The timestamp array stores time in the common timebase. This group will usually
126   only be populated in TimeSeries that are stored external to the NWB file, in files
127   storing raw data. Once timestamp data is calculated, the contents of 'sync' are
128   mostly for archival purposes.
129   name: sync
130   quantity: '?'
131 neurodata_type_def: AnnotationSeries
132 neurodata_type_inc: TimeSeries

```

5.7.3 IntervalSeries

Extends: *TimeSeries*

Description: see [Section 4.6.3](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - doc: Smallest meaningful difference between values in data. Annotations have no
17       units, so the value is fixed to -1.0.
18     dtype: float32
19     name: resolution
20     value: -1.0
21   - doc: Base unit of measurement for working with the data. Annotations have no units,
22       so the value is fixed to 'n/a'.
23     dtype: text
24     name: unit
25     value: n/a
26   - default_value: 1.0
27     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
28         If the data are stored in acquisition system units or other units that require
29         a conversion to be interpretable, multiply the data by 'conversion' to convert
30         the data to the specified 'unit'. e.g. if the data acquisition system stores
31         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
32         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
33         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
34         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
35     dtype: float32
36     name: conversion
37     required: false
38   - default_value: 0.0
39     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
40         to the specified 'unit'. Two common examples of this include (a) data stored
41         in an unsigned type that requires a shift after scaling to re-center the data,
42         and (b) specialized recording devices that naturally cause a scalar offset with
43         respect to the true units.
44     dtype: float32
45     name: offset
46     required: false

```

(continues on next page)

(continued from previous page)

```

47 - doc: Optionally describe the continuity of the data. Can be "continuous",
↳ "instantaneous",
48     or "step". For example, a voltage trace would be "continuous", because samples
49     are recorded from a continuous process. An array of lick times would be
↳ "instantaneous",
50     because the data represents distinct moments in time. Times of image presentations
51     would be "step" because the picture remains the same until the next timepoint.
52     This field is optional, but is useful in providing information about the underlying
53     data. It may inform the way this data is interpreted, the way it is visualized,
54     and what analysis methods are applicable.
55     dtype: text
56     name: continuity
57     required: false
58 dims:
59 - num_times
60 doc: Use values >0 if interval started, <0 if interval ended.
61 dtype: int8
62 name: data
63 shape:
64 -
65 - attributes:
66 - doc: Sampling rate, in Hz.
67   dtype: float32
68   name: rate
69 - doc: Unit of measurement for time, which is fixed to 'seconds'.
70   dtype: text
71   name: unit
72   value: seconds
73 doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
74     the timestamp of the first sample can be specified and all subsequent ones calculated
75     from the sampling rate attribute.
76 dtype: float64
77 name: starting_time
78 quantity: '?'
79 - attributes:
80 - doc: Value is '1'
81   dtype: int32
82   name: interval
83   value: 1
84 - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
85   dtype: text
86   name: unit
87   value: seconds
88 dims:
89 - num_times
90 doc: Timestamps for samples stored in data, in seconds, relative to the common_
↳ experiment
91     master-clock stored in NWBFile.timestamps_reference_time.
92 dtype: float64
93 name: timestamps
94 quantity: '?'
95 shape:

```

(continues on next page)

(continued from previous page)

```

96 -
97 - dims:
98   - num_times
99   doc: Numerical labels that apply to each time point in data for the purpose of querying
100     and slicing data by these values. If present, the length of this array should
101     be the same size as the first dimension of data.
102   dtype: uint8
103   name: control
104   quantity: '?'
105   shape:
106 -
107 - dims:
108   - num_control_values
109   doc: Description of each control value. Must be present if control is present. If
110     present, control_description[0] should describe time points where control == 0.
111   dtype: text
112   name: control_description
113   quantity: '?'
114   shape:
115 -
116 doc: Stores intervals of data. The timestamps field stores the beginning and end of
117   intervals. The data field stores whether the interval just started (>0 value) or
118   ended (<0 value). Different interval types can be represented in the same series
119   by using multiple key values (eg, 1 for feature A, 2 for feature B, 3 for feature
120   C, etc). The field data stores an 8-bit integer. This is largely an alias of a standard
121   TimeSeries but that is identifiable as representing time intervals in a machine-
122   ↪readable
123   way.
124 groups:
125 - doc: Lab-specific time and sync information as provided directly from hardware devices
126   and that is necessary for aligning all acquired time information to a common_
127   ↪timebase.
128   The timestamp array stores time in the common timebase. This group will usually
129   only be populated in TimeSeries that are stored external to the NWB file, in files
130   storing raw data. Once timestamp data is calculated, the contents of 'sync' are
131   mostly for archival purposes.
132   name: sync
133   quantity: '?'
134 neurodata_type_def: IntervalSeries
135 neurodata_type_inc: TimeSeries

```

5.7.4 DecompositionSeries

Extends: *TimeSeries*

Description: see [Section 4.6.4](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - default_value: no unit
17     doc: Base unit of measurement for working with the data. Actual stored values
18         are not necessarily stored in these units. To access the data in these units,
19         multiply 'data' by 'conversion'.
20     dtype: text
21     name: unit
22     required: false
23   - default_value: 1.0
24     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
25         If the data are stored in acquisition system units or other units that require
26         a conversion to be interpretable, multiply the data by 'conversion' to convert
27         the data to the specified 'unit'. e.g. if the data acquisition system stores
28         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
29         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
30         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
31         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
32     dtype: float32
33     name: conversion
34     required: false
35   - default_value: 0.0
36     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
37         to the specified 'unit'. Two common examples of this include (a) data stored
38         in an unsigned type that requires a shift after scaling to re-center the data,
39         and (b) specialized recording devices that naturally cause a scalar offset with
40         respect to the true units.
41     dtype: float32
42     name: offset
43     required: false
44   - default_value: -1.0
45     doc: Smallest meaningful difference between values in data, stored in the specified
46         by unit, e.g., the change in value of the least significant bit, or a larger

```

(continues on next page)

(continued from previous page)

```

47     number if signal noise is known to be present. If unknown, use -1.0.
48     dtype: float32
49     name: resolution
50     required: false
51     - doc: Optionally describe the continuity of the data. Can be "continuous",
    ↪ "instantaneous",
52       or "step". For example, a voltage trace would be "continuous", because samples
53       are recorded from a continuous process. An array of lick times would be
    ↪ "instantaneous",
54       because the data represents distinct moments in time. Times of image presentations
55       would be "step" because the picture remains the same until the next timepoint.
56       This field is optional, but is useful in providing information about the underlying
57       data. It may inform the way this data is interpreted, the way it is visualized,
58       and what analysis methods are applicable.
59     dtype: text
60     name: continuity
61     required: false
62     dims:
63     - num_times
64     - num_channels
65     - num_bands
66     doc: Data decomposed into frequency bands.
67     dtype: numeric
68     name: data
69     shape:
70     -
71     -
72     -
73     - doc: The metric used, e.g. phase, amplitude, power.
74     dtype: text
75     name: metric
76     - doc: DynamicTableRegion pointer to the channels that this decomposition series was
77       generated from.
78     name: source_channels
79     neurodata_type_inc: DynamicTableRegion
80     quantity: '?'
81     - attributes:
82     - doc: Sampling rate, in Hz.
83       dtype: float32
84       name: rate
85     - doc: Unit of measurement for time, which is fixed to 'seconds'.
86       dtype: text
87       name: unit
88       value: seconds
89     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
90       the timestamp of the first sample can be specified and all subsequent ones calculated
91       from the sampling rate attribute.
92     dtype: float64
93     name: starting_time
94     quantity: '?'
95     - attributes:
96     - doc: Value is '1'

```

(continues on next page)

(continued from previous page)

```

97     dtype: int32
98     name: interval
99     value: 1
100 - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
101     dtype: text
102     name: unit
103     value: seconds
104     dims:
105     - num_times
106     doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
107         master-clock stored in NWBFile.timestamps_reference_time.
108     dtype: float64
109     name: timestamps
110     quantity: '?'
111     shape:
112     -
113 - dims:
114     - num_times
115     doc: Numerical labels that apply to each time point in data for the purpose of querying
116         and slicing data by these values. If present, the length of this array should
117         be the same size as the first dimension of data.
118     dtype: uint8
119     name: control
120     quantity: '?'
121     shape:
122     -
123 - dims:
124     - num_control_values
125     doc: Description of each control value. Must be present if control is present. If
126         present, control_description[0] should describe time points where control == 0.
127     dtype: text
128     name: control_description
129     quantity: '?'
130     shape:
131     -
132     doc: Spectral analysis of a time series, e.g. of an LFP or a speech signal.
133     groups:
134     - datasets:
135         - doc: Name of the band, e.g. theta.
136           dtype: text
137           name: band_name
138           neurodata_type_inc: VectorData
139         - dims:
140             - num_bands
141             - low, high
142           doc: Low and high limit of each band in Hz. If it is a Gaussian filter, use 2
143               SD on either side of the center.
144           dtype: float32
145           name: band_limits
146           neurodata_type_inc: VectorData
147           shape:

```

(continues on next page)

(continued from previous page)

```

148 -
149 - 2
150 - dims:
151 -   num_bands
152   doc: The mean Gaussian filters, in Hz.
153   dtype: float32
154   name: band_mean
155   neurodata_type_inc: VectorData
156   shape:
157 -
158 - dims:
159 -   num_bands
160   doc: The standard deviation of Gaussian filters, in Hz.
161   dtype: float32
162   name: band_stdev
163   neurodata_type_inc: VectorData
164   shape:
165 -
166   doc: Table for describing the bands that this series was generated from. There should
167   be one row in this table for each band.
168   name: bands
169   neurodata_type_inc: DynamicTable
170 - doc: Lab-specific time and sync information as provided directly from hardware devices
171   and that is necessary for aligning all acquired time information to a common
172   ↳timebase.
173   The timestamp array stores time in the common timebase. This group will usually
174   only be populated in TimeSeries that are stored external to the NWB file, in files
175   storing raw data. Once timestamp data is calculated, the contents of 'sync' are
176   mostly for archival purposes.
177   name: sync
178   quantity: '?'
179 links:
180 - doc: Link to TimeSeries object that this data was calculated from. Metadata about
181   electrodes and their position can be read from that ElectricalSeries so it is
182   not necessary to store that information here.
183   name: source_timeseries
184   quantity: '?'
185   target_type: TimeSeries
186   neurodata_type_def: DecompositionSeries
187   neurodata_type_inc: TimeSeries

```

5.7.5 Units

Extends: [DynamicTable](#)

Description: see [Section 4.6.5](#)

YAML Specification:

```

1 attributes:
2 - dims:
3   - num_columns
4   doc: The names of the columns in this table. This should be used to specify an order
5     to the columns.
6   dtype: text
7   name: colnames
8   shape:
9     -
10  - doc: Description of what is in this dynamic table.
11    dtype: text
12    name: description
13 datasets:
14 - doc: Index into the spike_times dataset.
15   name: spike_times_index
16   neurodata_type_inc: VectorIndex
17   quantity: '?'
18 - attributes:
19   - doc: The smallest possible difference between two spike times. Usually 1 divided
20     by the acquisition sampling rate from which spike times were extracted, but
21     could be larger if the acquisition time series was downsampled or smaller if
22     the acquisition time series was smoothed/interpolated and it is possible for
23     the spike time to be between samples.
24     dtype: float64
25     name: resolution
26     required: false
27   doc: Spike times for each unit in seconds.
28   dtype: float64
29   name: spike_times
30   neurodata_type_inc: VectorData
31   quantity: '?'
32 - doc: Index into the obs_intervals dataset.
33   name: obs_intervals_index
34   neurodata_type_inc: VectorIndex
35   quantity: '?'
36 - dims:
37   - num_intervals
38   - start|end
39   doc: Observation intervals for each unit.
40   dtype: float64
41   name: obs_intervals
42   neurodata_type_inc: VectorData
43   quantity: '?'
44   shape:
45     -
46     - 2

```

(continues on next page)

(continued from previous page)

```

47 - doc: Index into electrodes.
48   name: electrodes_index
49   neurodata_type_inc: VectorIndex
50   quantity: '?'
51 - doc: Electrode that each spike unit came from, specified using a DynamicTableRegion.
52   name: electrodes
53   neurodata_type_inc: DynamicTableRegion
54   quantity: '?'
55 - doc: Electrode group that each spike unit came from.
56   dtype:
57     reftype: object
58     target_type: ElectrodeGroup
59   name: electrode_group
60   neurodata_type_inc: VectorData
61   quantity: '?'
62 - attributes:
63   - doc: Sampling rate, in hertz.
64     dtype: float32
65     name: sampling_rate
66     required: false
67   - doc: Unit of measurement. This value is fixed to 'volts'.
68     dtype: text
69     name: unit
70     value: volts
71   dims:
72     - num_units
73     - num_samples
74     - num_units
75     - num_samples
76     - num_electrodes
77   doc: Spike waveform mean for each spike unit.
78   dtype: float32
79   name: waveform_mean
80   neurodata_type_inc: VectorData
81   quantity: '?'
82   shape:
83     -
84     -
85     -
86     -
87     -
88 - attributes:
89   - doc: Sampling rate, in hertz.
90     dtype: float32
91     name: sampling_rate
92     required: false
93   - doc: Unit of measurement. This value is fixed to 'volts'.
94     dtype: text
95     name: unit
96     value: volts
97   dims:
98     - num_units

```

(continues on next page)

(continued from previous page)

```

99     - num_samples
100     - num_units
101     - num_samples
102     - num_electrodes
103 doc: Spike waveform standard deviation for each spike unit.
104 dtype: float32
105 name: waveform_sd
106 neurodata_type_inc: VectorData
107 quantity: '?'
108 shape:
109     - -
110     -
111     - -
112     -
113     -
114 - attributes:
115     - doc: Sampling rate, in hertz.
116       dtype: float32
117       name: sampling_rate
118       required: false
119     - doc: Unit of measurement. This value is fixed to 'volts'.
120       dtype: text
121       name: unit
122       value: volts
123 dims:
124     - num_waveforms
125     - num_samples
126 doc: Individual waveforms for each spike on each electrode. This is a doubly indexed
127     column. The 'waveforms_index' column indexes which waveforms in this column belong
128     to the same spike event for a given unit, where each waveform was recorded from
129     a different electrode. The 'waveforms_index_index' column indexes the 'waveforms_
130     ↪index'
131     column to indicate which spike events belong to a given unit. For example, if
132     the 'waveforms_index_index' column has values [2, 5, 6], then the first 2 elements
133     of the 'waveforms_index' column correspond to the 2 spike events of the first
134     unit, the next 3 elements of the 'waveforms_index' column correspond to the 3
135     spike events of the second unit, and the next 1 element of the 'waveforms_index'
136     column corresponds to the 1 spike event of the third unit. If the 'waveforms_index'
137     column has values [3, 6, 8, 10, 12, 13], then the first 3 elements of the 'waveforms'
138     column contain the 3 spike waveforms that were recorded from 3 different electrodes
139     for the first spike time of the first unit. See
140     ↪arrays https://nwb-schema.readthedocs.io/en/stable/format\_description.html#doubly-ragged-
141     for a graphical representation of this example. When there is only one electrode
142     for each unit (i.e., each spike time is associated with a single waveform), then
143     the 'waveforms_index' column will have values 1, 2, ..., N, where N is the number
144     of spike events. The number of electrodes for each spike event should be the same
145     within a given unit. The 'electrodes' column should be used to indicate which
146     electrodes are associated with each unit, and the order of the waveforms within
147     a given unit x spike event should be in the same order as the electrodes referenced
148     in the 'electrodes' column of this table. The number of samples for each waveform
    must be the same.

```

(continues on next page)

(continued from previous page)

```

149 dtype: numeric
150 name: waveforms
151 neurodata_type_inc: VectorData
152 quantity: '?'
153 shape:
154 -
155 -
156 - doc: Index into the waveforms dataset. One value for every spike event. See 'waveforms'
157   for more detail.
158   name: waveforms_index
159   neurodata_type_inc: VectorIndex
160   quantity: '?'
161 - doc: Index into the waveforms_index dataset. One value for every unit (row in the
162   table). See 'waveforms' for more detail.
163   name: waveforms_index_index
164   neurodata_type_inc: VectorIndex
165   quantity: '?'
166 - dims:
167   - num_rows
168   doc: Array of unique identifiers for the rows of this dynamic table.
169   dtype: int
170   name: id
171   neurodata_type_inc: ElementIdentifiers
172   shape:
173   -
174 - doc: Vector columns, including index columns, of this dynamic table.
175   neurodata_type_inc: VectorData
176   quantity: '*'
177 default_name: Units
178 doc: Data about spiking units. Event times of observed units (e.g. cell, synapse,
179   etc.) should be concatenated and stored in spike_times.
180 neurodata_type_def: Units
181 neurodata_type_inc: DynamicTable

```

5.8 Behavior

This source module contains `neurodata_types` for behavior data.

5.8.1 SpatialSeries

Extends: *TimeSeries*

Description: see [Section 4.7.1](#)

YAML Specification:

```

1  attributes:
2  - default_value: no description
3    doc: Description of the time series.
4    dtype: text
5    name: description
6    required: false
7  - default_value: no comments
8    doc: Human-readable comments about the TimeSeries. This second descriptive field
9        can be used to store additional information, or descriptive information if the
10       primary description field is populated with a computer-readable string.
11    dtype: text
12    name: comments
13    required: false
14 datasets:
15 - attributes:
16   - default_value: meters
17     doc: Base unit of measurement for working with the data. The default value is
18         'meters'. Actual stored values are not necessarily stored in these units. To
19         access the data in these units, multiply 'data' by 'conversion' and add 'offset'.
20     dtype: text
21     name: unit
22     required: false
23   - default_value: 1.0
24     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
25         If the data are stored in acquisition system units or other units that require
26         a conversion to be interpretable, multiply the data by 'conversion' to convert
27         the data to the specified 'unit'. e.g. if the data acquisition system stores
28         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
29         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
30         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
31         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
32     dtype: float32
33     name: conversion
34     required: false
35   - default_value: 0.0
36     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
37         to the specified 'unit'. Two common examples of this include (a) data stored
38         in an unsigned type that requires a shift after scaling to re-center the data,
39         and (b) specialized recording devices that naturally cause a scalar offset with
40         respect to the true units.
41     dtype: float32

```

(continues on next page)

(continued from previous page)

```

42     name: offset
43     required: false
44 -   default_value: -1.0
45     doc: Smallest meaningful difference between values in data, stored in the specified
46         by unit, e.g., the change in value of the least significant bit, or a larger
47         number if signal noise is known to be present. If unknown, use -1.0.
48     dtype: float32
49     name: resolution
50     required: false
51 -   doc: Optionally describe the continuity of the data. Can be "continuous",
52     ↪ "instantaneous",
53     or "step". For example, a voltage trace would be "continuous", because samples
54     are recorded from a continuous process. An array of lick times would be
55     ↪ "instantaneous",
56     because the data represents distinct moments in time. Times of image presentations
57     would be "step" because the picture remains the same until the next timepoint.
58     This field is optional, but is useful in providing information about the underlying
59     data. It may inform the way this data is interpreted, the way it is visualized,
60     and what analysis methods are applicable.
61     dtype: text
62     name: continuity
63     required: false
64     dims:
65     - - num_times
66     - - num_times
67     - x
68     - - num_times
69     - x,y
70     - - num_times
71     - x,y,z
72     doc: 1-D or 2-D array storing position or direction relative to some reference frame.
73     dtype: numeric
74     name: data
75     shape:
76     - -
77     - -
78     - 1
79     - -
80     - 2
81     - -
82     - 3
83 -   doc: Description defining what exactly 'straight-ahead' means.
84     dtype: text
85     name: reference_frame
86     quantity: '?'
87 -   attributes:
88     - doc: Sampling rate, in Hz.
89       dtype: float32
90       name: rate
91     - doc: Unit of measurement for time, which is fixed to 'seconds'.
92       dtype: text
93       name: unit

```

(continues on next page)

(continued from previous page)

```

92     value: seconds
93     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
94         the timestamp of the first sample can be specified and all subsequent ones calculated
95         from the sampling rate attribute.
96     dtype: float64
97     name: starting_time
98     quantity: '?'
99 -   attributes:
100     - doc: Value is '1'
101       dtype: int32
102       name: interval
103       value: 1
104     - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
105       dtype: text
106       name: unit
107       value: seconds
108     dims:
109     - num_times
110     doc: Timestamps for samples stored in data, in seconds, relative to the common_
111         ↪ experiment
112         master-clock stored in NWBFile.timestamps_reference_time.
113     dtype: float64
114     name: timestamps
115     quantity: '?'
116     shape:
117     -
118   - dims:
119     - num_times
120     doc: Numerical labels that apply to each time point in data for the purpose of querying
121         and slicing data by these values. If present, the length of this array should
122         be the same size as the first dimension of data.
123     dtype: uint8
124     name: control
125     quantity: '?'
126     shape:
127     -
128   - dims:
129     - num_control_values
130     doc: Description of each control value. Must be present if control is present. If
131         present, control_description[0] should describe time points where control == 0.
132     dtype: text
133     name: control_description
134     quantity: '?'
135     shape:
136     -
137   doc: "Direction, e.g., of gaze or travel, or position. The TimeSeries::data field
138       is a 2D array storing position or direction relative to some reference frame. Array
139       structure: [num measurements] [num dimensions]. Each SpatialSeries has a text dataset
140       reference_frame that indicates the zero-position, or the zero-axes for direction.
141       For example, if representing gaze direction, 'straight-ahead' might be a specific
142       pixel on the monitor, or some other point in space. For position data, the 0,0 point
143       might be the top-left corner of an enclosure, as viewed from the tracking camera.

```

(continues on next page)

(continued from previous page)

The unit of data will indicate how to interpret SpatialSeries values."

groups:

- **doc:** Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common ↵
↵timebase.

The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

name: sync

quantity: '?'

neurodata_type_def: SpatialSeries

neurodata_type_inc: TimeSeries

5.8.2 BehavioralEpochs

Extends: *NWBDataInterface*

Description: see [Section 4.7.2](#)

YAML Specification:

```
1 default_name: BehavioralEpochs
2 doc: TimeSeries for storing behavioral epochs. The objective of this and the other
3     two Behavioral interfaces (e.g. BehavioralEvents and BehavioralTimeSeries) is to
4     provide generic hooks for software tools/scripts. This allows a tool/script to take
5     the output one specific interface (e.g., UnitTimes) and plot that data relative
6     to another data modality (e.g., behavioral events) without having to define all
7     possible modalities in advance. Declaring one of these interfaces means that one
8     or more TimeSeries of the specified type is published. These TimeSeries should reside
9     in a group having the same name as the interface. For example, if a
10    BehavioralTimeSeries
11    interface is declared, the module will have one or more TimeSeries defined in the
12    module sub-group 'BehavioralTimeSeries'. BehavioralEpochs should use IntervalSeries.
13    BehavioralEvents is used for irregular events. BehavioralTimeSeries is for continuous
14    data.
15 groups:
16 - doc: IntervalSeries object containing start and stop times of epochs.
17   neurodata_type_inc: IntervalSeries
18   quantity: '*'
19 neurodata_type_def: BehavioralEpochs
20 neurodata_type_inc: NWBDataInterface
```

5.8.3 BehavioralEvents

Extends: *NWBDataInterface*

Description: see [Section 4.7.3](#)

YAML Specification:

```

1 default_name: BehavioralEvents
2 doc: TimeSeries for storing behavioral events. See description of <a href="
   ↪#BehavioralEpochs">BehavioralEpochs</a>
3   for more details.
4 groups:
5 - doc: TimeSeries object containing behavioral events.
6   neurodata_type_inc: TimeSeries
7   quantity: '*'
8 neurodata_type_def: BehavioralEvents
9 neurodata_type_inc: NWBDataInterface

```

5.8.4 BehavioralTimeSeries

Extends: *NWBDataInterface*

Description: see [Section 4.7.4](#)

YAML Specification:

```
1 default_name: BehavioralTimeSeries
2 doc: TimeSeries for storing Behavoioral time series data. See description of <a href="
   ↪ #BehavioralEpochs">BehavioralEpochs</a>
3   for more details.
4 groups:
5 - doc: TimeSeries object containing continuous behavioral data.
6   neurodata_type_inc: TimeSeries
7   quantity: '*'
8 neurodata_type_def: BehavioralTimeSeries
9 neurodata_type_inc: NWBDataInterface
```

5.8.5 PupilTracking

Extends: *NWBDataInterface*

Description: see [Section 4.7.5](#)

YAML Specification:

```
1 default_name: PupilTracking
2 doc: Eye-tracking data, representing pupil size.
3 groups:
4 - doc: TimeSeries object containing time series data on pupil size.
5   neurodata_type_inc: TimeSeries
6   quantity: +
7 neurodata_type_def: PupilTracking
8 neurodata_type_inc: NWBDataInterface
```

5.8.6 EyeTracking

Extends: *NWBDataInterface*

Description: see [Section 4.7.6](#)

YAML Specification:

```

1 default_name: EyeTracking
2 doc: Eye-tracking data, representing direction of gaze.
3 groups:
4 - doc: SpatialSeries object containing data measuring direction of gaze.
5   neurodata_type_inc: SpatialSeries
6   quantity: '*'
7 neurodata_type_def: EyeTracking
8 neurodata_type_inc: NWBDataInterface

```

5.8.7 CompassDirection

Extends: *NWBDataInterface*

Description: see [Section 4.7.7](#)

YAML Specification:

```

1 default_name: CompassDirection
2 doc: With a CompassDirection interface, a module publishes a SpatialSeries object
3   representing a floating point value for theta. The SpatialSeries::reference_frame
4   field should indicate what direction corresponds to 0 and which is the direction
5   of rotation (this should be clockwise). The si_unit for the SpatialSeries should
6   be radians or degrees.
7 groups:
8 - doc: SpatialSeries object containing direction of gaze travel.
9   neurodata_type_inc: SpatialSeries
10  quantity: '*'
11 neurodata_type_def: CompassDirection
12 neurodata_type_inc: NWBDataInterface

```

5.8.8 Position

Extends: *NWBDataInterface*

Description: see [Section 4.7.8](#)

YAML Specification:

```

1 default_name: Position
2 doc: Position data, whether along the x, x/y or x/y/z axis.
3 groups:
4 - doc: SpatialSeries object containing position data.
5   neurodata_type_inc: SpatialSeries
6   quantity: +
7 neurodata_type_def: Position
8 neurodata_type_inc: NWBDataInterface

```

5.9 Extracellular electrophysiology

This source module contains neurodata_types for extracellular electrophysiology data.

5.9.1 ElectricalSeries

Extends: *TimeSeries*

Description: see [Section 4.8.1](#)

YAML Specification:

```

1 attributes:
2 - doc: Filtering applied to all channels of the data. For example, if this_
   ↪ ElectricalSeries
3   represents high-pass-filtered data (also known as AP Band), then this value could
4   be "High-pass 4-pole Bessel filter at 500 Hz". If this ElectricalSeries represents
5   low-pass-filtered LFP data and the type of filter is unknown, then this value
6   could be "Low-pass filter at 300 Hz". If a non-standard filter type is used, provide
7   as much detail about the filter properties as possible.
8   dtype: text
9   name: filtering
10  required: false
11 - default_value: no description
12   doc: Description of the time series.
13   dtype: text
14   name: description
15   required: false
16 - default_value: no comments
17   doc: Human-readable comments about the TimeSeries. This second descriptive field
18   can be used to store additional information, or descriptive information if the
19   primary description field is populated with a computer-readable string.
20   dtype: text
21   name: comments
22   required: false
23 datasets:
24 - attributes:
25   - doc: Base unit of measurement for working with the data. This value is fixed to
26     'volts'. Actual stored values are not necessarily stored in these units. To
27     access the data in these units, multiply 'data' by 'conversion', followed by
28     'channel_conversion' (if present), and then add 'offset'.
29     dtype: text
30     name: unit
31     value: volts
32   - default_value: 1.0
33     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
34     If the data are stored in acquisition system units or other units that require
35     a conversion to be interpretable, multiply the data by 'conversion' to convert
36     the data to the specified 'unit'. e.g. if the data acquisition system stores
37     values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
38     that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
39     gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
40     values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .

```

(continues on next page)

(continued from previous page)

```

41 dtype: float32
42 name: conversion
43 required: false
44 - default_value: 0.0
45 doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
46     to the specified 'unit'. Two common examples of this include (a) data stored
47     in an unsigned type that requires a shift after scaling to re-center the data,
48     and (b) specialized recording devices that naturally cause a scalar offset with
49     respect to the true units.
50 dtype: float32
51 name: offset
52 required: false
53 - default_value: -1.0
54 doc: Smallest meaningful difference between values in data, stored in the specified
55     by unit, e.g., the change in value of the least significant bit, or a larger
56     number if signal noise is known to be present. If unknown, use -1.0.
57 dtype: float32
58 name: resolution
59 required: false
60 - doc: Optionally describe the continuity of the data. Can be "continuous",
61     ↪ "instantaneous",
62     or "step". For example, a voltage trace would be "continuous", because samples
63     are recorded from a continuous process. An array of lick times would be
64     ↪ "instantaneous",
65     because the data represents distinct moments in time. Times of image presentations
66     would be "step" because the picture remains the same until the next timepoint.
67     This field is optional, but is useful in providing information about the underlying
68     data. It may inform the way this data is interpreted, the way it is visualized,
69     and what analysis methods are applicable.
70 dtype: text
71 name: continuity
72 required: false
73 dims:
74 - - num_times
75 - - num_times
76 - - num_channels
77 - - num_times
78 - - num_channels
79 - - num_samples
80 doc: Recorded voltage data.
81 dtype: numeric
82 name: data
83 shape:
84 - -
85 - -
86 - -
87 - -
88 - doc: DynamicTableRegion pointer to the electrodes that this time series was generated
89     from.
90 name: electrodes

```

(continues on next page)

(continued from previous page)

```

91 neurodata_type_inc: DynamicTableRegion
92 - attributes:
93   - doc: The zero-indexed axis of the 'data' dataset that the channel-specific conversion
94     factor corresponds to. This value is fixed to 1.
95     dtype: int32
96     name: axis
97     value: 1
98   dims:
99   - num_channels
100 doc: Channel-specific conversion factor. Multiply the data in the 'data' dataset
101   by these values along the channel axis (as indicated by axis attribute) AND by
102   the global conversion factor in the 'conversion' attribute of 'data' to get the
103   data values in Volts, i.e, data in Volts = data * data.conversion * channel_
104   ↪ conversion.
105   This approach allows for both global and per-channel data conversion factors needed
106   to support the storage of electrical recordings as native values generated by
107   data acquisition systems. If this dataset is not present, then there is no channel-
108   ↪ specific
109   conversion factor, i.e. it is 1 for all channels.
110   dtype: float32
111   name: channel_conversion
112   quantity: '?'
113   shape:
114   -
115 - attributes:
116   - doc: Sampling rate, in Hz.
117     dtype: float32
118     name: rate
119   - doc: Unit of measurement for time, which is fixed to 'seconds'.
120     dtype: text
121     name: unit
122     value: seconds
123 doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
124   the timestamp of the first sample can be specified and all subsequent ones calculated
125   from the sampling rate attribute.
126   dtype: float64
127   name: starting_time
128   quantity: '?'
129 - attributes:
130   - doc: Value is '1'
131     dtype: int32
132     name: interval
133     value: 1
134   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
135     dtype: text
136     name: unit
137     value: seconds
138   dims:
139   - num_times
140 doc: Timestamps for samples stored in data, in seconds, relative to the common_
141   ↪ experiment
142   master-clock stored in NWBFile.timestamps_reference_time.

```

(continues on next page)

(continued from previous page)

```

140 dtype: float64
141 name: timestamps
142 quantity: '?'
143 shape:
144 -
145 - dims:
146   - num_times
147   doc: Numerical labels that apply to each time point in data for the purpose of querying
148       and slicing data by these values. If present, the length of this array should
149       be the same size as the first dimension of data.
150   dtype: uint8
151   name: control
152   quantity: '?'
153   shape:
154   -
155   - dims:
156     - num_control_values
157     doc: Description of each control value. Must be present if control is present. If
158         present, control_description[0] should describe time points where control == 0.
159     dtype: text
160     name: control_description
161     quantity: '?'
162     shape:
163     -
164   doc: A time series of acquired voltage data from extracellular recordings. The data
165       field is an int or float array storing data in volts. The first dimension should
166       always represent time. The second dimension, if present, should represent channels.
167 groups:
168 - doc: Lab-specific time and sync information as provided directly from hardware devices
169   and that is necessary for aligning all acquired time information to a common
170   ↳ timebase.
171   The timestamp array stores time in the common timebase. This group will usually
172   only be populated in TimeSeries that are stored external to the NWB file, in files
173   storing raw data. Once timestamp data is calculated, the contents of 'sync' are
174   mostly for archival purposes.
175   name: sync
176   quantity: '?'
177 neurodata_type_def: ElectricalSeries
178 neurodata_type_inc: TimeSeries

```

5.9.2 SpikeEventSeries

Extends: *ElectricalSeries*

Description: see [Section 4.8.2](#)

YAML Specification:

```

1 attributes:
2 - doc: Filtering applied to all channels of the data. For example, if this_
   ↪ ElectricalSeries
3     represents high-pass-filtered data (also known as AP Band), then this value could
4     be "High-pass 4-pole Bessel filter at 500 Hz". If this ElectricalSeries represents
5     low-pass-filtered LFP data and the type of filter is unknown, then this value
6     could be "Low-pass filter at 300 Hz". If a non-standard filter type is used, provide
7     as much detail about the filter properties as possible.
8     dtype: text
9     name: filtering
10    required: false
11 - default_value: no description
12    doc: Description of the time series.
13    dtype: text
14    name: description
15    required: false
16 - default_value: no comments
17    doc: Human-readable comments about the TimeSeries. This second descriptive field
18    can be used to store additional information, or descriptive information if the
19    primary description field is populated with a computer-readable string.
20    dtype: text
21    name: comments
22    required: false
23 datasets:
24 - attributes:
25   - doc: Unit of measurement for waveforms, which is fixed to 'volts'.
26     dtype: text
27     name: unit
28     value: volts
29   - default_value: 1.0
30     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
31     If the data are stored in acquisition system units or other units that require
32     a conversion to be interpretable, multiply the data by 'conversion' to convert
33     the data to the specified 'unit'. e.g. if the data acquisition system stores
34     values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
35     that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
36     gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
37     values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
38     dtype: float32
39     name: conversion
40     required: false
41   - default_value: 0.0
42     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
43     to the specified 'unit'. Two common examples of this include (a) data stored
44     in an unsigned type that requires a shift after scaling to re-center the data,
45     and (b) specialized recording devices that naturally cause a scalar offset with

```

(continues on next page)

(continued from previous page)

```

46     respect to the true units.
47     dtype: float32
48     name: offset
49     required: false
50 -   default_value: -1.0
51     doc: Smallest meaningful difference between values in data, stored in the specified
52         by unit, e.g., the change in value of the least significant bit, or a larger
53         number if signal noise is known to be present. If unknown, use -1.0.
54     dtype: float32
55     name: resolution
56     required: false
57 -   doc: Optionally describe the continuity of the data. Can be "continuous",
↪ "instantaneous",
58     or "step". For example, a voltage trace would be "continuous", because samples
59     are recorded from a continuous process. An array of lick times would be
↪ "instantaneous",
60     because the data represents distinct moments in time. Times of image presentations
61     would be "step" because the picture remains the same until the next timepoint.
62     This field is optional, but is useful in providing information about the underlying
63     data. It may inform the way this data is interpreted, the way it is visualized,
64     and what analysis methods are applicable.
65     dtype: text
66     name: continuity
67     required: false
68     dims:
69     - - num_events
70       - num_samples
71     - - num_events
72       - num_channels
73       - num_samples
74     doc: Spike waveforms.
75     dtype: numeric
76     name: data
77     shape:
78     - -
79     - -
80     - -
81     - -
82     - -
83 -   attributes:
84     -   doc: Value is '1'
85         dtype: int32
86         name: interval
87         value: 1
88     -   doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
89         dtype: text
90         name: unit
91         value: seconds
92     dims:
93     - num_times
94     doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment

```

(continues on next page)

(continued from previous page)

master-clock stored in NWBFile.timestamps_reference_time. Timestamps are required for the events. Unlike for TimeSeries, timestamps are required for SpikeEventSeries and are thus re-specified here.

dtype: float64

name: timestamps

shape:

-

- **doc:** DynamicTableRegion pointer to the electrodes that this time series was generated from.

name: electrodes

neurodata_type_inc: DynamicTableRegion

- **attributes:**

- **doc:** The zero-indexed axis of the 'data' dataset that the channel-specific conversion factor corresponds to. This value is fixed to 1.

dtype: int32

name: axis

value: 1

dims:

- num_channels

doc: Channel-specific conversion factor. Multiply the data in the 'data' dataset by these values along the channel axis (as indicated by axis attribute) AND by the global conversion factor in the 'conversion' attribute of 'data' to get the data values in Volts, i.e, data in Volts = data * data.conversion * channel_

↪ conversion.

This approach allows for both global and per-channel data conversion factors needed to support the storage of electrical recordings as native values generated by data acquisition systems. If this dataset is not present, then there is no channel-

↪ specific

conversion factor, i.e. it is 1 for all channels.

dtype: float32

name: channel_conversion

quantity: '?'

shape:

-

- **attributes:**

- **doc:** Sampling rate, in Hz.

dtype: float32

name: rate

- **doc:** Unit of measurement for time, which is fixed to 'seconds'.

dtype: text

name: unit

value: seconds

doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced, the timestamp of the first sample can be specified and all subsequent ones calculated from the sampling rate attribute.

dtype: float64

name: starting_time

quantity: '?'

- **dims:**

- num_times

doc: Numerical labels that apply to each time point in data for the purpose of querying and slicing data by these values. If present, the length of this array should

(continues on next page)

(continued from previous page)

```

145     be the same size as the first dimension of data.
146     dtype: uint8
147     name: control
148     quantity: '?'
149     shape:
150     -
151 - dims:
152   - num_control_values
153   doc: Description of each control value. Must be present if control is present. If
154       present, control_description[0] should describe time points where control == 0.
155   dtype: text
156   name: control_description
157   quantity: '?'
158   shape:
159   -
160 doc: 'Stores snapshots/snippets of recorded spike events (i.e., threshold crossings).
161     This may also be raw data, as reported by ephys hardware. If so, the
162     ↳TimeSeries::description
163     field should describe how events were detected. All SpikeEventSeries should reside
164     in a module (under EventWaveform interface) even if the spikes were reported and
165     stored by hardware. All events span the same recording channels and store snapshots
166     of equal duration. TimeSeries::data array structure: [num events] [num channels]
167     [num samples] (or [num events] [num samples] for single electrode).'
```

groups:

```

168 - doc: Lab-specific time and sync information as provided directly from hardware devices
169     and that is necessary for aligning all acquired time information to a common
170     ↳timebase.
171     The timestamp array stores time in the common timebase. This group will usually
172     only be populated in TimeSeries that are stored external to the NWB file, in files
173     storing raw data. Once timestamp data is calculated, the contents of 'sync' are
174     mostly for archival purposes.
175     name: sync
176     quantity: '?'
177 neurodata_type_def: SpikeEventSeries
178 neurodata_type_inc: ElectricalSeries
```

5.9.3 FeatureExtraction

Extends: *NWBDataInterface*

Description: see Section 4.8.3

YAML Specification:

```

1 datasets:
2   - dims:
3     - num_features
4     doc: Description of features (eg, 'PC1') for each of the extracted features.
5     dtype: text
6     name: description
7     shape:
8     -
9   - dims:
10    - num_events
11    - num_channels
12    - num_features
13    doc: Multi-dimensional array of features extracted from each event.
14    dtype: float32
15    name: features
16    shape:
17    -
18    -
19    -
20  - dims:
21    - num_events
22    doc: Times of events that features correspond to (can be a link).
23    dtype: float64
24    name: times
25    shape:
26    -
27  - doc: DynamicTableRegion pointer to the electrodes that this time series was generated
28    from.
29    name: electrodes
30    neurodata_type_inc: DynamicTableRegion
31  default_name: FeatureExtraction
32  doc: Features, such as PC1 and PC2, that are extracted from signals stored in a
33    ↳ SpikeEventSeries
34    or other source.
35  neurodata_type_def: FeatureExtraction
36  neurodata_type_inc: NWBDataInterface

```

5.9.4 EventDetection

Extends: *NWBDataInterface*

Description: see [Section 4.8.4](#)

YAML Specification:

```

1 datasets:
2   - doc: Description of how events were detected, such as voltage threshold, or dV/dT
3     threshold, as well as relevant values.
4     dtype: text
5     name: detection_method
6   - dims:
7     - num_events
8     doc: Indices (zero-based) into source ElectricalSeries::data array corresponding
9       to time of event. 'description' should define what is meant by time of event
10      (e.g., .25 ms before action potential peak, zero-crossing time, etc). The index
11      points to each event from the raw data.
12     dtype: int32
13     name: source_idx
14     shape:
15       -
16   - attributes:
17     - doc: Unit of measurement for event times, which is fixed to 'seconds'.
18       dtype: text
19       name: unit
20       value: seconds
21     dims:
22     - num_events
23     doc: Timestamps of events, in seconds.
24     dtype: float64
25     name: times
26     shape:
27       -
28   default_name: EventDetection
29   doc: Detected spike events from voltage trace(s).
30   links:
31   - doc: Link to the ElectricalSeries that this data was calculated from. Metadata about
32     electrodes and their position can be read from that ElectricalSeries so it's not
33     necessary to include that information here.
34     name: source_electricalseries
35     target_type: ElectricalSeries
36   neurodata_type_def: EventDetection
37   neurodata_type_inc: NWBDataInterface

```

5.9.5 EventWaveform

Extends: *NWBDataInterface*

Description: see [Section 4.8.5](#)

YAML Specification:

```

1 default_name: EventWaveform
2 doc: Represents either the waveforms of detected events, as extracted from a raw data
3   trace in /acquisition, or the event waveforms that were stored during experiment
4   acquisition.
5 groups:
6 - doc: SpikeEventSeries object(s) containing detected spike event waveforms.
7   neurodata_type_inc: SpikeEventSeries
8   quantity: '*'
9 neurodata_type_def: EventWaveform
10 neurodata_type_inc: NWBDataInterface

```

5.9.6 FilteredEphys

Extends: *NWBDataInterface*

Description: see [Section 4.8.6](#)

YAML Specification:

```
1 default_name: FilteredEphys
2 doc: Electrophysiology data from one or more channels that has been subjected to
   ↳ filtering.
3   Examples of filtered data include Theta and Gamma (LFP has its own interface).
   ↳ FilteredEphys
4   modules publish an ElectricalSeries for each filtered channel or set of channels.
5   The name of each ElectricalSeries is arbitrary but should be informative. The source
6   of the filtered data, whether this is from analysis of another time series or as
7   acquired by hardware, should be noted in each's TimeSeries::description field. There
8   is no assumed 1::1 correspondence between filtered ephys signals and electrodes,
9   as a single signal can apply to many nearby electrodes, and one electrode may have
10  different filtered (e.g., theta and/or gamma) signals represented. Filter properties
11  should be noted in the ElectricalSeries 'filtering' attribute.
12 groups:
13 - doc: ElectricalSeries object(s) containing filtered electrophysiology data.
14   neurodata_type_inc: ElectricalSeries
15   quantity: +
16 neurodata_type_def: FilteredEphys
17 neurodata_type_inc: NWBDataInterface
```

5.9.7 LFP

Extends: *NWBDataInterface*

Description: see [Section 4.8.7](#)

YAML Specification:

```

1 default_name: LFP
2 doc: LFP data from one or more channels. The electrode map in each published_
   ↳ElectricalSeries
3   will identify which channels are providing LFP data. Filter properties should be
4   noted in the ElectricalSeries 'filtering' attribute.
5 groups:
6 - doc: ElectricalSeries object(s) containing LFP data for one or more channels.
7   neurodata_type_inc: ElectricalSeries
8   quantity: +
9 neurodata_type_def: LFP
10 neurodata_type_inc: NWBDataInterface

```

5.9.8 ElectrodeGroup

Extends: *NWBContainer*

Description: see [Section 4.8.8](#)

YAML Specification:

```

1 attributes:
2 - doc: Description of this electrode group.
3   dtype: text
4   name: description
5 - doc: Location of electrode group. Specify the area, layer, comments on estimation
6   of area/layer, etc. Use standard atlas names for anatomical regions when possible.
7   dtype: text
8   name: location
9 datasets:
10 - doc: stereotaxic or common framework coordinates
11   dtype:
12     - doc: x coordinate
13       dtype: float32
14       name: x
15     - doc: y coordinate
16       dtype: float32
17       name: y
18     - doc: z coordinate
19       dtype: float32
20       name: z
21   name: position
22   quantity: '?'
23 doc: A physical grouping of electrodes, e.g. a shank of an array.
24 links:
25 - doc: Link to the device that was used to record from this electrode group.
26   name: device
27   target_type: Device
28 neurodata_type_def: ElectrodeGroup
29 neurodata_type_inc: NWBContainer

```

5.9.9 ClusterWaveforms

Extends: *NWBDataInterface*

Description: see Section 4.8.9

YAML Specification:

```

1 datasets:
2   - doc: Filtering applied to data before generating mean/sd
3     dtype: text
4     name: waveform_filtering
5   - dims:
6     - num_clusters
7     - num_samples
8     doc: The mean waveform for each cluster, using the same indices for each wave as
9         cluster numbers in the associated Clustering module (i.e, cluster 3 is in array
10        slot [3]). Waveforms corresponding to gaps in cluster sequence should be empty
11        (e.g., zero- filled)
12     dtype: float32
13     name: waveform_mean
14     shape:
15       -
16       -
17   - dims:
18     - num_clusters
19     - num_samples
20     doc: Stdev of waveforms for each cluster, using the same indices as in mean
21     dtype: float32
22     name: waveform_sd
23     shape:
24       -
25       -
26 default_name: ClusterWaveforms
27 doc: DEPRECATED The mean waveform shape, including standard deviation, of the different
28     clusters. Ideally, the waveform analysis should be performed on data that is only
29     high-pass filtered. This is a separate module because it is expected to require
30     updating. For example, IMEC probes may require different storage requirements to
31     store/display mean waveforms, requiring a new interface or an extension of this
32     one.
33 links:
34   - doc: Link to Clustering interface that was the source of the clustered data
35     name: clustering_interface
36     target_type: Clustering
37 neurodata_type_def: ClusterWaveforms
38 neurodata_type_inc: NWBDataInterface

```

5.9.10 Clustering

Extends: *NWBDataInterface*

Description: see [Section 4.8.10](#)

YAML Specification:

```

1 datasets:
2 - doc: Description of clusters or clustering, (e.g. cluster 0 is noise, clusters curated
3   using Klusters, etc)
4   dtype: text
5   name: description
6 - dims:
7   - num_events
8   doc: Cluster number of each event
9   dtype: int32
10  name: num
11  shape:
12    -
13 - dims:
14   - num_clusters
15   doc: Maximum ratio of waveform peak to RMS on any channel in the cluster (provides
16     a basic clustering metric).
17   dtype: float32
18   name: peak_over_rms
19   shape:
20     -
21 - dims:
22   - num_events
23   doc: Times of clustered events, in seconds. This may be a link to times field in
24     associated FeatureExtraction module.
25   dtype: float64
26   name: times
27   shape:
28     -
29 default_name: Clustering
30 doc: DEPRECATED Clustered spike data, whether from automatic clustering tools (e.g.,
31   klustakwik) or as a result of manual sorting.
32 neurodata_type_def: Clustering
33 neurodata_type_inc: NWBDataInterface

```

5.10 Intracellular electrophysiology

This source module contains neurodata_types for intracellular electrophysiology data.

5.10.1 PatchClampSeries

Extends: *TimeSeries*

Description: see [Section 4.9.1](#)

YAML Specification:

```

1  attributes:
2  - doc: Protocol/stimulus name for this patch-clamp dataset.
3    dtype: text
4    name: stimulus_description
5  - doc: Sweep number, allows to group different PatchClampSeries together.
6    dtype: uint32
7    name: sweep_number
8    required: false
9  - default_value: no description
10   doc: Description of the time series.
11   dtype: text
12   name: description
13   required: false
14 - default_value: no comments
15   doc: Human-readable comments about the TimeSeries. This second descriptive field
16       can be used to store additional information, or descriptive information if the
17       primary description field is populated with a computer-readable string.
18   dtype: text
19   name: comments
20   required: false
21 datasets:
22 - attributes:
23   - doc: Base unit of measurement for working with the data. Actual stored values
24       are not necessarily stored in these units. To access the data in these units,
25       multiply 'data' by 'conversion' and add 'offset'.
26     dtype: text
27     name: unit
28   - default_value: 1.0
29     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
30         If the data are stored in acquisition system units or other units that require
31         a conversion to be interpretable, multiply the data by 'conversion' to convert
32         the data to the specified 'unit'. e.g. if the data acquisition system stores
33         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
34         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
35         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
36         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
37     dtype: float32
38     name: conversion
39     required: false
40   - default_value: 0.0
41     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion

```

(continues on next page)

(continued from previous page)

```

42     to the specified 'unit'. Two common examples of this include (a) data stored
43     in an unsigned type that requires a shift after scaling to re-center the data,
44     and (b) specialized recording devices that naturally cause a scalar offset with
45     respect to the true units.
46     dtype: float32
47     name: offset
48     required: false
49 -   default_value: -1.0
50     doc: Smallest meaningful difference between values in data, stored in the specified
51         by unit, e.g., the change in value of the least significant bit, or a larger
52         number if signal noise is known to be present. If unknown, use -1.0.
53     dtype: float32
54     name: resolution
55     required: false
56 -   doc: Optionally describe the continuity of the data. Can be "continuous",
57     ↪ "instantaneous",
58     or "step". For example, a voltage trace would be "continuous", because samples
59     are recorded from a continuous process. An array of lick times would be
60     ↪ "instantaneous",
61     because the data represents distinct moments in time. Times of image presentations
62     would be "step" because the picture remains the same until the next timepoint.
63     This field is optional, but is useful in providing information about the underlying
64     data. It may inform the way this data is interpreted, the way it is visualized,
65     and what analysis methods are applicable.
66     dtype: text
67     name: continuity
68     required: false
69     dims:
70     - num_times
71     doc: Recorded voltage or current.
72     dtype: numeric
73     name: data
74     shape:
75     -
76     doc: Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp).
77     dtype: float32
78     name: gain
79     quantity: '?'
80 -   attributes:
81     - doc: Sampling rate, in Hz.
82       dtype: float32
83       name: rate
84     - doc: Unit of measurement for time, which is fixed to 'seconds'.
85       dtype: text
86       name: unit
87       value: seconds
88     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
89         the timestamp of the first sample can be specified and all subsequent ones calculated
90         from the sampling rate attribute.
91     dtype: float64
92     name: starting_time
93     quantity: '?'

```

(continues on next page)

(continued from previous page)

```

92 - attributes:
93   - doc: Value is '1'
94     dtype: int32
95     name: interval
96     value: 1
97   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
98     dtype: text
99     name: unit
100    value: seconds
101  dims:
102    - num_times
103    doc: Timestamps for samples stored in data, in seconds, relative to the common_
104    ↪ experiment
105       master-clock stored in NWBFile.timestamps_reference_time.
106    dtype: float64
107    name: timestamps
108    quantity: '?'
109    shape:
110  - dims:
111    - num_times
112    doc: Numerical labels that apply to each time point in data for the purpose of querying
113       and slicing data by these values. If present, the length of this array should
114       be the same size as the first dimension of data.
115    dtype: uint8
116    name: control
117    quantity: '?'
118    shape:
119  -
120  - dims:
121    - num_control_values
122    doc: Description of each control value. Must be present if control is present. If
123       present, control_description[0] should describe time points where control == 0.
124    dtype: text
125    name: control_description
126    quantity: '?'
127    shape:
128  -
129  doc: An abstract base class for patch-clamp data - stimulus or response, current or
130     voltage.
131  groups:
132  - doc: Lab-specific time and sync information as provided directly from hardware devices
133     and that is necessary for aligning all acquired time information to a common_
134     ↪ timebase.
135       The timestamp array stores time in the common timebase. This group will usually
136       only be populated in TimeSeries that are stored external to the NWB file, in files
137       storing raw data. Once timestamp data is calculated, the contents of 'sync' are
138       mostly for archival purposes.
139     name: sync
140     quantity: '?'
141  links:
142  - doc: Link to IntracellularElectrode object that describes the electrode that was

```

(continues on next page)

(continued from previous page)

```
142     used to apply or record this data.  
143     name: electrode  
144     target_type: IntracellularElectrode  
145     neurodata_type_def: PatchClampSeries  
146     neurodata_type_inc: TimeSeries
```

5.10.2 CurrentClampSeries

Extends: *PatchClampSeries*

Description: see [Section 4.9.2](#)

YAML Specification:

```

1 attributes:
2 - doc: Protocol/stimulus name for this patch-clamp dataset.
3   dtype: text
4   name: stimulus_description
5 - doc: Sweep number, allows to group different PatchClampSeries together.
6   dtype: uint32
7   name: sweep_number
8   required: false
9 - default_value: no description
10  doc: Description of the time series.
11  dtype: text
12  name: description
13  required: false
14 - default_value: no comments
15  doc: Human-readable comments about the TimeSeries. This second descriptive field
16      can be used to store additional information, or descriptive information if the
17      primary description field is populated with a computer-readable string.
18  dtype: text
19  name: comments
20  required: false
21 datasets:
22 - attributes:
23   - doc: Base unit of measurement for working with the data. which is fixed to 'volts'.
24       Actual stored values are not necessarily stored in these units. To access the
25       data in these units, multiply 'data' by 'conversion' and add 'offset'.
26     dtype: text
27     name: unit
28     value: volts
29   - default_value: 1.0
30     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
31         If the data are stored in acquisition system units or other units that require
32         a conversion to be interpretable, multiply the data by 'conversion' to convert
33         the data to the specified 'unit'. e.g. if the data acquisition system stores
34         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
35         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
36         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
37         values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
38     dtype: float32
39     name: conversion
40     required: false
41   - default_value: 0.0
42     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
43         to the specified 'unit'. Two common examples of this include (a) data stored
44         in an unsigned type that requires a shift after scaling to re-center the data,
45         and (b) specialized recording devices that naturally cause a scalar offset with
46         respect to the true units.

```

(continues on next page)

(continued from previous page)

```

47     dtype: float32
48     name: offset
49     required: false
50 -   default_value: -1.0
51     doc: Smallest meaningful difference between values in data, stored in the specified
52         by unit, e.g., the change in value of the least significant bit, or a larger
53         number if signal noise is known to be present. If unknown, use -1.0.
54     dtype: float32
55     name: resolution
56     required: false
57 -   doc: Optionally describe the continuity of the data. Can be "continuous",
58     ↪ "instantaneous",
59     or "step". For example, a voltage trace would be "continuous", because samples
60     are recorded from a continuous process. An array of lick times would be
61     ↪ "instantaneous",
62     because the data represents distinct moments in time. Times of image presentations
63     would be "step" because the picture remains the same until the next timepoint.
64     This field is optional, but is useful in providing information about the underlying
65     data. It may inform the way this data is interpreted, the way it is visualized,
66     and what analysis methods are applicable.
67     dtype: text
68     name: continuity
69     required: false
70     doc: Recorded voltage.
71     name: data
72 -   doc: Bias current, in amps.
73     dtype: float32
74     name: bias_current
75     quantity: '?'
76 -   doc: Bridge balance, in ohms.
77     dtype: float32
78     name: bridge_balance
79     quantity: '?'
80 -   doc: Capacitance compensation, in farads.
81     dtype: float32
82     name: capacitance_compensation
83     quantity: '?'
84 -   doc: Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp).
85     dtype: float32
86     name: gain
87     quantity: '?'
88 -   attributes:
89     -   doc: Sampling rate, in Hz.
90         dtype: float32
91         name: rate
92     -   doc: Unit of measurement for time, which is fixed to 'seconds'.
93         dtype: text
94         name: unit
95         value: seconds
96     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
    the timestamp of the first sample can be specified and all subsequent ones calculated
    from the sampling rate attribute.

```

(continues on next page)

(continued from previous page)

```

97 dtype: float64
98 name: starting_time
99 quantity: '?'
100 - attributes:
101   - doc: Value is '1'
102     dtype: int32
103     name: interval
104     value: 1
105   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
106     dtype: text
107     name: unit
108     value: seconds
109 dims:
110   - num_times
111 doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
112     master-clock stored in NWBFile.timestamps_reference_time.
113 dtype: float64
114 name: timestamps
115 quantity: '?'
116 shape:
117   -
118 - dims:
119   - num_times
120 doc: Numerical labels that apply to each time point in data for the purpose of querying
121     and slicing data by these values. If present, the length of this array should
122     be the same size as the first dimension of data.
123 dtype: uint8
124 name: control
125 quantity: '?'
126 shape:
127   -
128 - dims:
129   - num_control_values
130 doc: Description of each control value. Must be present if control is present. If
131     present, control_description[0] should describe time points where control == 0.
132 dtype: text
133 name: control_description
134 quantity: '?'
135 shape:
136   -
137 doc: Voltage data from an intracellular current-clamp recording. A corresponding_
↪ CurrentClampStimulusSeries
138     (stored separately as a stimulus) is used to store the current injected.
139 groups:
140   - doc: Lab-specific time and sync information as provided directly from hardware devices
141     and that is necessary for aligning all acquired time information to a common_
↪ timebase.
142     The timestamp array stores time in the common timebase. This group will usually
143     only be populated in TimeSeries that are stored external to the NWB file, in files
144     storing raw data. Once timestamp data is calculated, the contents of 'sync' are
145     mostly for archival purposes.

```

(continues on next page)

(continued from previous page)

```
146   name: sync
147   quantity: '?'
148 links:
149 - doc: Link to IntracellularElectrode object that describes the electrode that was
150     used to apply or record this data.
151   name: electrode
152   target_type: IntracellularElectrode
153 neurodata_type_def: CurrentClampSeries
154 neurodata_type_inc: PatchClampSeries
```

5.10.3 IZeroClampSeries

Extends: *CurrentClampSeries*

Description: see [Section 4.9.3](#)

YAML Specification:

```

1 attributes:
2 - doc: An IZeroClampSeries has no stimulus, so this attribute is automatically set
3   to "N/A"
4   dtype: text
5   name: stimulus_description
6   value: N/A
7 - doc: Sweep number, allows to group different PatchClampSeries together.
8   dtype: uint32
9   name: sweep_number
10  required: false
11 - default_value: no description
12  doc: Description of the time series.
13  dtype: text
14  name: description
15  required: false
16 - default_value: no comments
17  doc: Human-readable comments about the TimeSeries. This second descriptive field
18    can be used to store additional information, or descriptive information if the
19    primary description field is populated with a computer-readable string.
20  dtype: text
21  name: comments
22  required: false
23 datasets:
24 - doc: Bias current, in amps, fixed to 0.0.
25   dtype: float32
26   name: bias_current
27 - doc: Bridge balance, in ohms, fixed to 0.0.
28   dtype: float32
29   name: bridge_balance
30 - doc: Capacitance compensation, in farads, fixed to 0.0.
31   dtype: float32
32   name: capacitance_compensation
33 - attributes:
34   - doc: Base unit of measurement for working with the data. which is fixed to 'volts'.
35     Actual stored values are not necessarily stored in these units. To access the
36     data in these units, multiply 'data' by 'conversion' and add 'offset'.
37     dtype: text
38     name: unit
39     value: volts
40   - default_value: 1.0
41     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
42       If the data are stored in acquisition system units or other units that require
43       a conversion to be interpretable, multiply the data by 'conversion' to convert
44       the data to the specified 'unit'. e.g. if the data acquisition system stores
45       values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
46       that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system

```

(continues on next page)

(continued from previous page)

```

47     gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
48     values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
49     dtype: float32
50     name: conversion
51     required: false
52 -   default_value: 0.0
53     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
54         to the specified 'unit'. Two common examples of this include (a) data stored
55         in an unsigned type that requires a shift after scaling to re-center the data,
56         and (b) specialized recording devices that naturally cause a scalar offset with
57         respect to the true units.
58     dtype: float32
59     name: offset
60     required: false
61 -   default_value: -1.0
62     doc: Smallest meaningful difference between values in data, stored in the specified
63         by unit, e.g., the change in value of the least significant bit, or a larger
64         number if signal noise is known to be present. If unknown, use -1.0.
65     dtype: float32
66     name: resolution
67     required: false
68 -   doc: Optionally describe the continuity of the data. Can be "continuous",
69     ↪ "instantaneous",
70     or "step". For example, a voltage trace would be "continuous", because samples
71     are recorded from a continuous process. An array of lick times would be
72     ↪ "instantaneous",
73     because the data represents distinct moments in time. Times of image presentations
74     would be "step" because the picture remains the same until the next timepoint.
75     This field is optional, but is useful in providing information about the underlying
76     data. It may inform the way this data is interpreted, the way it is visualized,
77     and what analysis methods are applicable.
78     dtype: text
79     name: continuity
80     required: false
81     doc: Recorded voltage.
82     name: data
83 -   doc: Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp).
84     dtype: float32
85     name: gain
86     quantity: '?'
87 -   attributes:
88     -   doc: Sampling rate, in Hz.
89         dtype: float32
90         name: rate
91     -   doc: Unit of measurement for time, which is fixed to 'seconds'.
92         dtype: text
93         name: unit
94         value: seconds
95     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
96         the timestamp of the first sample can be specified and all subsequent ones calculated
97         from the sampling rate attribute.
98     dtype: float64

```

(continues on next page)

(continued from previous page)

```

97  name: starting_time
98  quantity: '?'
99  - attributes:
100    - doc: Value is '1'
101      dtype: int32
102      name: interval
103      value: 1
104    - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
105      dtype: text
106      name: unit
107      value: seconds
108  dims:
109    - num_times
110  doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
111     master-clock stored in NWBFile.timestamps_reference_time.
112  dtype: float64
113  name: timestamps
114  quantity: '?'
115  shape:
116    -
117  - dims:
118    - num_times
119  doc: Numerical labels that apply to each time point in data for the purpose of querying
120     and slicing data by these values. If present, the length of this array should
121     be the same size as the first dimension of data.
122  dtype: uint8
123  name: control
124  quantity: '?'
125  shape:
126    -
127  - dims:
128    - num_control_values
129  doc: Description of each control value. Must be present if control is present. If
130     present, control_description[0] should describe time points where control == 0.
131  dtype: text
132  name: control_description
133  quantity: '?'
134  shape:
135    -
136  doc: Voltage data from an intracellular recording when all current and amplifier settings
137     are off (i.e., CurrentClampSeries fields will be zero). There is no_
↪ CurrentClampStimulusSeries
138     associated with an IZero series because the amplifier is disconnected and no stimulus
139     can reach the cell.
140  groups:
141  - doc: Lab-specific time and sync information as provided directly from hardware devices
142     and that is necessary for aligning all acquired time information to a common_
↪ timebase.
143     The timestamp array stores time in the common timebase. This group will usually
144     only be populated in TimeSeries that are stored external to the NWB file, in files
145     storing raw data. Once timestamp data is calculated, the contents of 'sync' are

```

(continues on next page)

(continued from previous page)

```
146     mostly for archival purposes.
147     name: sync
148     quantity: '?'
149 links:
150 - doc: Link to IntracellularElectrode object that describes the electrode that was
151     used to apply or record this data.
152     name: electrode
153     target_type: IntracellularElectrode
154 neurodata_type_def: IZeroClampSeries
155 neurodata_type_inc: CurrentClampSeries
```

5.10.4 CurrentClampStimulusSeries

Extends: *PatchClampSeries*

Description: see [Section 4.9.4](#)

YAML Specification:

```

1 attributes:
2 - doc: Protocol/stimulus name for this patch-clamp dataset.
3   dtype: text
4   name: stimulus_description
5 - doc: Sweep number, allows to group different PatchClampSeries together.
6   dtype: uint32
7   name: sweep_number
8   required: false
9 - default_value: no description
10  doc: Description of the time series.
11  dtype: text
12  name: description
13  required: false
14 - default_value: no comments
15  doc: Human-readable comments about the TimeSeries. This second descriptive field
16      can be used to store additional information, or descriptive information if the
17      primary description field is populated with a computer-readable string.
18  dtype: text
19  name: comments
20  required: false
21 datasets:
22 - attributes:
23   - doc: Base unit of measurement for working with the data. which is fixed to 'amperes'.
24       Actual stored values are not necessarily stored in these units. To access the
25       data in these units, multiply 'data' by 'conversion' and add 'offset'.
26     dtype: text
27     name: unit
28     value: amperes
29   - default_value: 1.0
30     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
31         If the data are stored in acquisition system units or other units that require
32         a conversion to be interpretable, multiply the data by 'conversion' to convert
33         the data to the specified 'unit'. e.g. if the data acquisition system stores
34         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
35         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
36         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
37         values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
38     dtype: float32
39     name: conversion
40     required: false
41   - default_value: 0.0
42     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
43         to the specified 'unit'. Two common examples of this include (a) data stored
44         in an unsigned type that requires a shift after scaling to re-center the data,
45         and (b) specialized recording devices that naturally cause a scalar offset with
46         respect to the true units.

```

(continues on next page)

(continued from previous page)

```

47     dtype: float32
48     name: offset
49     required: false
50 -   default_value: -1.0
51     doc: Smallest meaningful difference between values in data, stored in the specified
52         by unit, e.g., the change in value of the least significant bit, or a larger
53         number if signal noise is known to be present. If unknown, use -1.0.
54     dtype: float32
55     name: resolution
56     required: false
57 -   doc: Optionally describe the continuity of the data. Can be "continuous",
58     ↪ "instantaneous",
59     or "step". For example, a voltage trace would be "continuous", because samples
60     are recorded from a continuous process. An array of lick times would be
61     ↪ "instantaneous",
62     because the data represents distinct moments in time. Times of image presentations
63     would be "step" because the picture remains the same until the next timepoint.
64     This field is optional, but is useful in providing information about the underlying
65     data. It may inform the way this data is interpreted, the way it is visualized,
66     and what analysis methods are applicable.
67     dtype: text
68     name: continuity
69     required: false
70     doc: Stimulus current applied.
71     name: data
72 -   doc: Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp).
73     dtype: float32
74     name: gain
75     quantity: '?'
76 -   attributes:
77     -   doc: Sampling rate, in Hz.
78         dtype: float32
79         name: rate
80     -   doc: Unit of measurement for time, which is fixed to 'seconds'.
81         dtype: text
82         name: unit
83         value: seconds
84     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
85         the timestamp of the first sample can be specified and all subsequent ones calculated
86         from the sampling rate attribute.
87     dtype: float64
88     name: starting_time
89     quantity: '?'
90 -   attributes:
91     -   doc: Value is '1'
92         dtype: int32
93         name: interval
94         value: 1
95     -   doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
96         dtype: text
97         name: unit
98         value: seconds

```

(continues on next page)

(continued from previous page)

```

97  dims:
98  - num_times
99  doc: Timestamps for samples stored in data, in seconds, relative to the common_
    ↪ experiment
    master-clock stored in NWBFile.timestamps_reference_time.
100
101  dtype: float64
102  name: timestamps
103  quantity: '?'
104  shape:
105  -
106  - dims:
107  - num_times
108  doc: Numerical labels that apply to each time point in data for the purpose of querying
109  and slicing data by these values. If present, the length of this array should
110  be the same size as the first dimension of data.
111  dtype: uint8
112  name: control
113  quantity: '?'
114  shape:
115  -
116  - dims:
117  - num_control_values
118  doc: Description of each control value. Must be present if control is present. If
119  present, control_description[0] should describe time points where control == 0.
120  dtype: text
121  name: control_description
122  quantity: '?'
123  shape:
124  -
125  doc: Stimulus current applied during current clamp recording.
126  groups:
127  - doc: Lab-specific time and sync information as provided directly from hardware devices
128  and that is necessary for aligning all acquired time information to a common_
    ↪ timebase.
129  The timestamp array stores time in the common timebase. This group will usually
130  only be populated in TimeSeries that are stored external to the NWB file, in files
131  storing raw data. Once timestamp data is calculated, the contents of 'sync' are
132  mostly for archival purposes.
133  name: sync
134  quantity: '?'
135  links:
136  - doc: Link to IntracellularElectrode object that describes the electrode that was
137  used to apply or record this data.
138  name: electrode
139  target_type: IntracellularElectrode
140  neurodata_type_def: CurrentClampStimulusSeries
141  neurodata_type_inc: PatchClampSeries

```

5.10.5 VoltageClampSeries

Extends: *PatchClampSeries*

Description: see [Section 4.9.5](#)

YAML Specification:

```

1 attributes:
2 - doc: Protocol/stimulus name for this patch-clamp dataset.
3   dtype: text
4   name: stimulus_description
5 - doc: Sweep number, allows to group different PatchClampSeries together.
6   dtype: uint32
7   name: sweep_number
8   required: false
9 - default_value: no description
10  doc: Description of the time series.
11  dtype: text
12  name: description
13  required: false
14 - default_value: no comments
15  doc: Human-readable comments about the TimeSeries. This second descriptive field
16      can be used to store additional information, or descriptive information if the
17      primary description field is populated with a computer-readable string.
18  dtype: text
19  name: comments
20  required: false
21 datasets:
22 - attributes:
23   - doc: Base unit of measurement for working with the data. which is fixed to 'amperes'.
24       Actual stored values are not necessarily stored in these units. To access the
25       data in these units, multiply 'data' by 'conversion' and add 'offset'.
26     dtype: text
27     name: unit
28     value: amperes
29   - default_value: 1.0
30     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
31         If the data are stored in acquisition system units or other units that require
32         a conversion to be interpretable, multiply the data by 'conversion' to convert
33         the data to the specified 'unit'. e.g. if the data acquisition system stores
34         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
35         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
36         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
37         values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
38     dtype: float32
39     name: conversion
40     required: false
41   - default_value: 0.0
42     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
43         to the specified 'unit'. Two common examples of this include (a) data stored
44         in an unsigned type that requires a shift after scaling to re-center the data,
45         and (b) specialized recording devices that naturally cause a scalar offset with
46         respect to the true units.

```

(continues on next page)

(continued from previous page)

```

47     dtype: float32
48     name: offset
49     required: false
50 -   default_value: -1.0
51     doc: Smallest meaningful difference between values in data, stored in the specified
52         by unit, e.g., the change in value of the least significant bit, or a larger
53         number if signal noise is known to be present. If unknown, use -1.0.
54     dtype: float32
55     name: resolution
56     required: false
57 -   doc: Optionally describe the continuity of the data. Can be "continuous",
58     ↪ "instantaneous",
59     or "step". For example, a voltage trace would be "continuous", because samples
60     are recorded from a continuous process. An array of lick times would be
61     ↪ "instantaneous",
62     because the data represents distinct moments in time. Times of image presentations
63     would be "step" because the picture remains the same until the next timepoint.
64     This field is optional, but is useful in providing information about the underlying
65     data. It may inform the way this data is interpreted, the way it is visualized,
66     and what analysis methods are applicable.
67     dtype: text
68     name: continuity
69     required: false
70     doc: Recorded current.
71     name: data
72 -   attributes:
73     - doc: Unit of measurement for capacitance_fast, which is fixed to 'farads'.
74       dtype: text
75       name: unit
76       value: farads
77     doc: Fast capacitance, in farads.
78     dtype: float32
79     name: capacitance_fast
80     quantity: '?'
81 -   attributes:
82     - doc: Unit of measurement for capacitance_fast, which is fixed to 'farads'.
83       dtype: text
84       name: unit
85       value: farads
86     doc: Slow capacitance, in farads.
87     dtype: float32
88     name: capacitance_slow
89     quantity: '?'
90 -   attributes:
91     - doc: Unit of measurement for resistance_comp_bandwidth, which is fixed to 'hertz'.
92       dtype: text
93       name: unit
94       value: hertz
95     doc: Resistance compensation bandwidth, in hertz.
96     dtype: float32
97     name: resistance_comp_bandwidth
98     quantity: '?'

```

(continues on next page)

(continued from previous page)

```

97 - attributes:
98   - doc: Unit of measurement for resistance_comp_correction, which is fixed to 'percent'.
99     dtype: text
100    name: unit
101    value: percent
102  doc: Resistance compensation correction, in percent.
103  dtype: float32
104  name: resistance_comp_correction
105  quantity: '?'
106 - attributes:
107   - doc: Unit of measurement for resistance_comp_prediction, which is fixed to 'percent'.
108     dtype: text
109     name: unit
110     value: percent
111  doc: Resistance compensation prediction, in percent.
112  dtype: float32
113  name: resistance_comp_prediction
114  quantity: '?'
115 - attributes:
116   - doc: Unit of measurement for whole_cell_capacitance_comp, which is fixed to 'farads'.
117     dtype: text
118     name: unit
119     value: farads
120  doc: Whole cell capacitance compensation, in farads.
121  dtype: float32
122  name: whole_cell_capacitance_comp
123  quantity: '?'
124 - attributes:
125   - doc: Unit of measurement for whole_cell_series_resistance_comp, which is fixed
126     to 'ohms'.
127     dtype: text
128     name: unit
129     value: ohms
130  doc: Whole cell series resistance compensation, in ohms.
131  dtype: float32
132  name: whole_cell_series_resistance_comp
133  quantity: '?'
134 - doc: Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp).
135  dtype: float32
136  name: gain
137  quantity: '?'
138 - attributes:
139   - doc: Sampling rate, in Hz.
140     dtype: float32
141     name: rate
142   - doc: Unit of measurement for time, which is fixed to 'seconds'.
143     dtype: text
144     name: unit
145     value: seconds
146  doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
147    the timestamp of the first sample can be specified and all subsequent ones calculated
148    from the sampling rate attribute.

```

(continues on next page)

(continued from previous page)

```

149 dtype: float64
150 name: starting_time
151 quantity: '?'
152 - attributes:
153   - doc: Value is '1'
154     dtype: int32
155     name: interval
156     value: 1
157   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
158     dtype: text
159     name: unit
160     value: seconds
161 dims:
162   - num_times
163 doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
164   master-clock stored in NWBFile.timestamps_reference_time.
165 dtype: float64
166 name: timestamps
167 quantity: '?'
168 shape:
169   -
170 - dims:
171   - num_times
172 doc: Numerical labels that apply to each time point in data for the purpose of querying
173   and slicing data by these values. If present, the length of this array should
174   be the same size as the first dimension of data.
175 dtype: uint8
176 name: control
177 quantity: '?'
178 shape:
179   -
180 - dims:
181   - num_control_values
182 doc: Description of each control value. Must be present if control is present. If
183   present, control_description[0] should describe time points where control == 0.
184 dtype: text
185 name: control_description
186 quantity: '?'
187 shape:
188   -
189 doc: Current data from an intracellular voltage-clamp recording. A corresponding_
↪ VoltageClampStimulusSeries
190   (stored separately as a stimulus) is used to store the voltage injected.
191 groups:
192   - doc: Lab-specific time and sync information as provided directly from hardware devices
193     and that is necessary for aligning all acquired time information to a common_
↪ timebase.
194     The timestamp array stores time in the common timebase. This group will usually
195     only be populated in TimeSeries that are stored external to the NWB file, in files
196     storing raw data. Once timestamp data is calculated, the contents of 'sync' are
197     mostly for archival purposes.

```

(continues on next page)

(continued from previous page)

```
198   name: sync
199   quantity: '?'
200 links:
201 - doc: Link to IntracellularElectrode object that describes the electrode that was
202     used to apply or record this data.
203   name: electrode
204   target_type: IntracellularElectrode
205 neurodata_type_def: VoltageClampSeries
206 neurodata_type_inc: PatchClampSeries
```

5.10.6 VoltageClampStimulusSeries

Extends: *PatchClampSeries*

Description: see [Section 4.9.6](#)

YAML Specification:

```

1 attributes:
2 - doc: Protocol/stimulus name for this patch-clamp dataset.
3   dtype: text
4   name: stimulus_description
5 - doc: Sweep number, allows to group different PatchClampSeries together.
6   dtype: uint32
7   name: sweep_number
8   required: false
9 - default_value: no description
10  doc: Description of the time series.
11  dtype: text
12  name: description
13  required: false
14 - default_value: no comments
15  doc: Human-readable comments about the TimeSeries. This second descriptive field
16      can be used to store additional information, or descriptive information if the
17      primary description field is populated with a computer-readable string.
18  dtype: text
19  name: comments
20  required: false
21 datasets:
22 - attributes:
23   - doc: Base unit of measurement for working with the data. which is fixed to 'volts'.
24       Actual stored values are not necessarily stored in these units. To access the
25       data in these units, multiply 'data' by 'conversion' and add 'offset'.
26     dtype: text
27     name: unit
28     value: volts
29   - default_value: 1.0
30     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
31         If the data are stored in acquisition system units or other units that require
32         a conversion to be interpretable, multiply the data by 'conversion' to convert
33         the data to the specified 'unit'. e.g. if the data acquisition system stores
34         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
35         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
36         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
37         values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
38     dtype: float32
39     name: conversion
40     required: false
41   - default_value: 0.0
42     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
43         to the specified 'unit'. Two common examples of this include (a) data stored
44         in an unsigned type that requires a shift after scaling to re-center the data,
45         and (b) specialized recording devices that naturally cause a scalar offset with
46         respect to the true units.

```

(continues on next page)

(continued from previous page)

```

47     dtype: float32
48     name: offset
49     required: false
50 -   default_value: -1.0
51     doc: Smallest meaningful difference between values in data, stored in the specified
52         by unit, e.g., the change in value of the least significant bit, or a larger
53         number if signal noise is known to be present. If unknown, use -1.0.
54     dtype: float32
55     name: resolution
56     required: false
57 -   doc: Optionally describe the continuity of the data. Can be "continuous",
58     ↪ "instantaneous",
59     or "step". For example, a voltage trace would be "continuous", because samples
60     are recorded from a continuous process. An array of lick times would be
61     ↪ "instantaneous",
62     because the data represents distinct moments in time. Times of image presentations
63     would be "step" because the picture remains the same until the next timepoint.
64     This field is optional, but is useful in providing information about the underlying
65     data. It may inform the way this data is interpreted, the way it is visualized,
66     and what analysis methods are applicable.
67     dtype: text
68     name: continuity
69     required: false
70     doc: Stimulus voltage applied.
71     name: data
72 -   doc: Gain of the recording, in units Volt/Amp (v-clamp) or Volt/Volt (c-clamp).
73     dtype: float32
74     name: gain
75     quantity: '?'
76 -   attributes:
77     -   doc: Sampling rate, in Hz.
78         dtype: float32
79         name: rate
80     -   doc: Unit of measurement for time, which is fixed to 'seconds'.
81         dtype: text
82         name: unit
83         value: seconds
84     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
85         the timestamp of the first sample can be specified and all subsequent ones calculated
86         from the sampling rate attribute.
87     dtype: float64
88     name: starting_time
89     quantity: '?'
90 -   attributes:
91     -   doc: Value is '1'
92         dtype: int32
93         name: interval
94         value: 1
95     -   doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
96         dtype: text
97         name: unit
98         value: seconds

```

(continues on next page)

(continued from previous page)

```

97  dims:
98  - num_times
99  doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
100     master-clock stored in NWBFile.timestamps_reference_time.
101  dtype: float64
102  name: timestamps
103  quantity: '?'
104  shape:
105  -
106  - dims:
107    - num_times
108    doc: Numerical labels that apply to each time point in data for the purpose of querying
109        and slicing data by these values. If present, the length of this array should
110        be the same size as the first dimension of data.
111    dtype: uint8
112    name: control
113    quantity: '?'
114    shape:
115    -
116  - dims:
117    - num_control_values
118    doc: Description of each control value. Must be present if control is present. If
119        present, control_description[0] should describe time points where control == 0.
120    dtype: text
121    name: control_description
122    quantity: '?'
123    shape:
124    -
125  doc: Stimulus voltage applied during a voltage clamp recording.
126  groups:
127  - doc: Lab-specific time and sync information as provided directly from hardware devices
128    and that is necessary for aligning all acquired time information to a common_
↪ timebase.
129    The timestamp array stores time in the common timebase. This group will usually
130    only be populated in TimeSeries that are stored external to the NWB file, in files
131    storing raw data. Once timestamp data is calculated, the contents of 'sync' are
132    mostly for archival purposes.
133    name: sync
134    quantity: '?'
135  links:
136  - doc: Link to IntracellularElectrode object that describes the electrode that was
137    used to apply or record this data.
138    name: electrode
139    target_type: IntracellularElectrode
140  neurodata_type_def: VoltageClampStimulusSeries
141  neurodata_type_inc: PatchClampSeries

```

5.10.7 IntracellularElectrode

Extends: *NWBContainer*

Description: see [Section 4.9.7](#)

YAML Specification:

```

1 datasets:
2   - doc: unique ID of the cell
3     dtype: text
4     name: cell_id
5     quantity: '?'
6   - doc: Description of electrode (e.g., whole-cell, sharp, etc.).
7     dtype: text
8     name: description
9   - doc: Electrode specific filtering.
10    dtype: text
11    name: filtering
12    quantity: '?'
13   - doc: Initial access resistance.
14     dtype: text
15     name: initial_access_resistance
16     quantity: '?'
17   - doc: Location of the electrode. Specify the area, layer, comments on estimation
18       of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names
19       for anatomical regions when possible.
20     dtype: text
21     name: location
22     quantity: '?'
23   - doc: Electrode resistance, in ohms.
24     dtype: text
25     name: resistance
26     quantity: '?'
27   - doc: Information about seal used for recording.
28     dtype: text
29     name: seal
30     quantity: '?'
31   - doc: Information about slice used for recording.
32     dtype: text
33     name: slice
34     quantity: '?'
35 doc: An intracellular electrode and its metadata.
36 links:
37   - doc: Device that was used to record from this electrode.
38     name: device
39     target_type: Device
40 neurodata_type_def: IntracellularElectrode
41 neurodata_type_inc: NWBContainer

```

5.10.8 SweepTable

Extends: `DynamicTable`

Description: see [Section 4.9.8](#)

YAML Specification:

```

1 attributes:
2   - dims:
3     - num_columns
4     doc: The names of the columns in this table. This should be used to specify an order
5       to the columns.
6     dtype: text
7     name: colnames
8     shape:
9       -
10    - doc: Description of what is in this dynamic table.
11      dtype: text
12      name: description
13 datasets:
14   - doc: Sweep number of the PatchClampSeries in that row.
15     dtype: uint32
16     name: sweep_number
17     neurodata_type_inc: VectorData
18   - doc: The PatchClampSeries with the sweep number in that row.
19     dtype:
20       reftype: object
21       target_type: PatchClampSeries
22     name: series
23     neurodata_type_inc: VectorData
24   - doc: Index for series.
25     name: series_index
26     neurodata_type_inc: VectorIndex
27   - dims:
28     - num_rows
29     doc: Array of unique identifiers for the rows of this dynamic table.
30     dtype: int
31     name: id
32     neurodata_type_inc: ElementIdentifiers
33     shape:
34       -
35   - doc: Vector columns, including index columns, of this dynamic table.
36     neurodata_type_inc: VectorData
37     quantity: '*'
38 doc: '[DEPRECATED] Table used to group different PatchClampSeries. SweepTable is being
39 replaced by IntracellularRecordingsTable and SimultaneousRecordingsTable tables.
40 Additional SequentialRecordingsTable, RepetitionsTable, and ExperimentalConditions
41 tables provide enhanced support for experiment metadata.'
42 neurodata_type_def: SweepTable
43 neurodata_type_inc: DynamicTable

```

5.10.9 IntracellularElectrodesTable

Extends: [DynamicTable](#)

Description: see [Section 4.9.9](#)

YAML Specification:

```

1 attributes:
2   - doc: Description of what is in this dynamic table.
3     dtype: text
4     name: description
5     value: Table for storing intracellular electrode related metadata.
6   - dims:
7     - num_columns
8     doc: The names of the columns in this table. This should be used to specify an order
9       to the columns.
10    dtype: text
11    name: colnames
12    shape:
13      -
14 datasets:
15   - doc: Column for storing the reference to the intracellular electrode.
16     dtype:
17       reftype: object
18       target_type: IntracellularElectrode
19     name: electrode
20     neurodata_type_inc: VectorData
21   - dims:
22     - num_rows
23     doc: Array of unique identifiers for the rows of this dynamic table.
24     dtype: int
25     name: id
26     neurodata_type_inc: ElementIdentifiers
27     shape:
28       -
29   - doc: Vector columns, including index columns, of this dynamic table.
30     neurodata_type_inc: VectorData
31     quantity: '*'
32 doc: Table for storing intracellular electrode related metadata.
33 neurodata_type_def: IntracellularElectrodesTable
34 neurodata_type_inc: DynamicTable

```

5.10.10 IntracellularStimuliTable

Extends: `DynamicTable`

Description: see [Section 4.9.10](#)

YAML Specification:

```

1 attributes:
2 - doc: Description of what is in this dynamic table.
3   dtype: text
4   name: description
5   value: Table for storing intracellular stimulus related metadata.
6 - dims:
7   - num_columns
8   doc: The names of the columns in this table. This should be used to specify an order
9     to the columns.
10  dtype: text
11  name: colnames
12  shape:
13  -
14 datasets:
15 - doc: Column storing the reference to the recorded stimulus for the recording (rows).
16   name: stimulus
17   neurodata_type_inc: TimeSeriesReferenceVectorData
18 - dims:
19   - num_rows
20   doc: Array of unique identifiers for the rows of this dynamic table.
21   dtype: int
22   name: id
23   neurodata_type_inc: ElementIdentifiers
24   shape:
25   -
26 - doc: Vector columns, including index columns, of this dynamic table.
27   neurodata_type_inc: VectorData
28   quantity: '*'
29 doc: Table for storing intracellular stimulus related metadata.
30 neurodata_type_def: IntracellularStimuliTable
31 neurodata_type_inc: DynamicTable

```

5.10.11 IntracellularResponsesTable

Extends: [DynamicTable](#)

Description: see [Section 4.9.11](#)

YAML Specification:

```

1 attributes:
2 - doc: Description of what is in this dynamic table.
3   dtype: text
4   name: description
5   value: Table for storing intracellular response related metadata.
6 - dims:
7   - num_columns
8   doc: The names of the columns in this table. This should be used to specify an order
9     to the columns.
10  dtype: text
11  name: colnames
12  shape:
13    -
14 datasets:
15 - doc: Column storing the reference to the recorded response for the recording (rows)
16   name: response
17   neurodata_type_inc: TimeSeriesReferenceVectorData
18 - dims:
19   - num_rows
20   doc: Array of unique identifiers for the rows of this dynamic table.
21   dtype: int
22   name: id
23   neurodata_type_inc: ElementIdentifiers
24   shape:
25     -
26 - doc: Vector columns, including index columns, of this dynamic table.
27   neurodata_type_inc: VectorData
28   quantity: '*'
29 doc: Table for storing intracellular response related metadata.
30 neurodata_type_def: IntracellularResponsesTable
31 neurodata_type_inc: DynamicTable

```

5.10.12 IntracellularRecordingsTable

Extends: [AlignedDynamicTable](#)

Description: see [Section 4.9.12](#)

YAML Specification:

```

1 attributes:
2 - doc: Description of the contents of this table. Inherited from AlignedDynamicTable
3   and overwritten here to fix the value of the attribute.
4   dtype: text
5   name: description
6   value: A table to group together a stimulus and response from a single electrode
7     and a single simultaneous recording and for storing metadata about the intracellular
8     recording.
9 - dims:
10   - num_categories
11   doc: The names of the categories in this AlignedDynamicTable. Each category is
12     represented
13     by one DynamicTable stored in the parent group. This attribute should be used
14     to specify an order of categories and the category names must match the names
15     of the corresponding DynamicTable in the group.
16   dtype: text
17   name: categories
18   shape:
19     -
20 - dims:
21   - num_columns
22   doc: The names of the columns in this table. This should be used to specify an order
23     to the columns.
24   dtype: text
25   name: colnames
26   shape:
27     -
28 datasets:
29 - dims:
30   - num_rows
31   doc: Array of unique identifiers for the rows of this dynamic table.
32   dtype: int
33   name: id
34   neurodata_type_inc: ElementIdentifiers
35   shape:
36     -
37 - doc: Vector columns, including index columns, of this dynamic table.
38   neurodata_type_inc: VectorData
39   quantity: '*'
40 doc: A table to group together a stimulus and response from a single electrode and
41   a single simultaneous recording. Each row in the table represents a single recording
42   consisting typically of a stimulus and a corresponding response. In some cases,
43   however, only a stimulus or a response is recorded as part of an experiment. In
44   this case, both the stimulus and response will point to the same TimeSeries while
45   the idx_start and count of the invalid column will be set to -1, thus, indicating
   that no values have been recorded for the stimulus or response, respectively. Note,

```

(continues on next page)

(continued from previous page)

a recording MUST contain at least a stimulus or a response. Typically the stimulus and response are PatchClampSeries. However, the use of AD/DA channels that are not associated to an electrode is also common in intracellular electrophysiology, in which case other TimeSeries may be used.

groups:

- **doc:** Table for storing intracellular electrode related metadata.

name: electrodes

neurodata_type_inc: IntracellularElectrodesTable

- **doc:** Table for storing intracellular stimulus related metadata.

name: stimuli

neurodata_type_inc: IntracellularStimuliTable

- **doc:** Table for storing intracellular response related metadata.

name: responses

neurodata_type_inc: IntracellularResponsesTable

- **doc:** A DynamicTable representing a particular category for columns in the

↳ AlignedDynamicTable

parent container. The table MUST be aligned with (i.e., have the same number of

rows) as all other DynamicTables stored in the AlignedDynamicTable parent container.

The name of the category is given by the name of the DynamicTable and its description by the description attribute of the DynamicTable.

neurodata_type_inc: DynamicTable

quantity: '*'

name: intracellular_recordings

neurodata_type_def: IntracellularRecordingsTable

neurodata_type_inc: AlignedDynamicTable

5.10.13 SimultaneousRecordingsTable

Extends: [DynamicTable](#)

Description: see [Section 4.9.13](#)

YAML Specification:

```

1 attributes:
2 - dims:
3   - num_columns
4   doc: The names of the columns in this table. This should be used to specify an order
5     to the columns.
6   dtype: text
7   name: colnames
8   shape:
9     -
10  - doc: Description of what is in this dynamic table.
11    dtype: text
12    name: description
13 datasets:
14 - attributes:
15   - doc: Reference to the IntracellularRecordingsTable table that this table region
16     applies to. This specializes the attribute inherited from DynamicTableRegion
17     to fix the type of table that can be referenced here.
18     dtype:
19       reftype: object
20       target_type: IntracellularRecordingsTable
21       name: table
22     doc: A reference to one or more rows in the IntracellularRecordingsTable table.
23     name: recordings
24     neurodata_type_inc: DynamicTableRegion
25   - doc: Index dataset for the recordings column.
26     name: recordings_index
27     neurodata_type_inc: VectorIndex
28   - dims:
29     - num_rows
30     doc: Array of unique identifiers for the rows of this dynamic table.
31     dtype: int
32     name: id
33     neurodata_type_inc: ElementIdentifiers
34     shape:
35       -
36   - doc: Vector columns, including index columns, of this dynamic table.
37     neurodata_type_inc: VectorData
38     quantity: '*'
39 doc: A table for grouping different intracellular recordings from the
40 ↪ IntracellularRecordingsTable
41   table together that were recorded simultaneously from different electrodes.
42 name: simultaneous_recordings
43 neurodata_type_def: SimultaneousRecordingsTable
44 neurodata_type_inc: DynamicTable

```

5.10.14 SequentialRecordingsTable

Extends: [DynamicTable](#)

Description: see [Section 4.9.14](#)

YAML Specification:

```

1 attributes:
2   - dims:
3     - num_columns
4   doc: The names of the columns in this table. This should be used to specify an order
5     to the columns.
6   dtype: text
7   name: colnames
8   shape:
9     -
10  - doc: Description of what is in this dynamic table.
11    dtype: text
12    name: description
13 datasets:
14  - attributes:
15    - doc: Reference to the SimultaneousRecordingsTable table that this table region
16      applies to. This specializes the attribute inherited from DynamicTableRegion
17      to fix the type of table that can be referenced here.
18      dtype:
19        reftype: object
20        target_type: SimultaneousRecordingsTable
21        name: table
22      doc: A reference to one or more rows in the SimultaneousRecordingsTable table.
23      name: simultaneous_recordings
24      neurodata_type_inc: DynamicTableRegion
25    - doc: Index dataset for the simultaneous_recordings column.
26      name: simultaneous_recordings_index
27      neurodata_type_inc: VectorIndex
28    - doc: The type of stimulus used for the sequential recording.
29      dtype: text
30      name: stimulus_type
31      neurodata_type_inc: VectorData
32  - dims:
33    - num_rows
34    doc: Array of unique identifiers for the rows of this dynamic table.
35    dtype: int
36    name: id
37    neurodata_type_inc: ElementIdentifiers
38    shape:
39      -
40  - doc: Vector columns, including index columns, of this dynamic table.
41    neurodata_type_inc: VectorData
42    quantity: '*'
43 doc: A table for grouping different sequential recordings from the
44 ↪ SimultaneousRecordingsTable
45   table together. This is typically used to group together sequential recordings where
46   a sequence of stimuli of the same type with varying parameters have been presented

```

(continues on next page)

(continued from previous page)

```
46     in a sequence.  
47 name: sequential_recordings  
48 neurodata_type_def: SequentialRecordingsTable  
49 neurodata_type_inc: DynamicTable
```

5.10.15 RepetitionsTable

Extends: [DynamicTable](#)

Description: see [Section 4.9.15](#)

YAML Specification:

```

1 attributes:
2 - dims:
3   - num_columns
4   doc: The names of the columns in this table. This should be used to specify an order
5     to the columns.
6   dtype: text
7   name: colnames
8   shape:
9     -
10  - doc: Description of what is in this dynamic table.
11    dtype: text
12    name: description
13 datasets:
14 - attributes:
15   - doc: Reference to the SequentialRecordingsTable table that this table region applies
16     to. This specializes the attribute inherited from DynamicTableRegion to fix
17     the type of table that can be referenced here.
18     dtype:
19       reftype: object
20       target_type: SequentialRecordingsTable
21     name: table
22   doc: A reference to one or more rows in the SequentialRecordingsTable table.
23   name: sequential_recordings
24   neurodata_type_inc: DynamicTableRegion
25 - doc: Index dataset for the sequential_recordings column.
26   name: sequential_recordings_index
27   neurodata_type_inc: VectorIndex
28 - dims:
29   - num_rows
30   doc: Array of unique identifiers for the rows of this dynamic table.
31   dtype: int
32   name: id
33   neurodata_type_inc: ElementIdentifiers
34   shape:
35     -
36 - doc: Vector columns, including index columns, of this dynamic table.
37   neurodata_type_inc: VectorData
38   quantity: '*'
39 doc: A table for grouping different sequential intracellular recordings together.
40   With each SequentialRecording typically representing a particular type of stimulus,
41   the RepetitionsTable table is typically used to group sets of stimuli applied in
42   sequence.
43 name: repetitions
44 neurodata_type_def: RepetitionsTable
45 neurodata_type_inc: DynamicTable

```

5.10.16 ExperimentalConditionsTable

Extends: `DynamicTable`

Description: see [Section 4.9.16](#)

YAML Specification:

```

1 attributes:
2   - dims:
3     - num_columns
4     doc: The names of the columns in this table. This should be used to specify an order
5       to the columns.
6     dtype: text
7     name: colnames
8     shape:
9       -
10    - doc: Description of what is in this dynamic table.
11      dtype: text
12      name: description
13 datasets:
14   - attributes:
15     - doc: Reference to the RepetitionsTable table that this table region applies to.
16       This specializes the attribute inherited from DynamicTableRegion to fix the
17       type of table that can be referenced here.
18     dtype:
19       reftype: object
20       target_type: RepetitionsTable
21     name: table
22     doc: A reference to one or more rows in the RepetitionsTable table.
23     name: repetitions
24     neurodata_type_inc: DynamicTableRegion
25   - doc: Index dataset for the repetitions column.
26     name: repetitions_index
27     neurodata_type_inc: VectorIndex
28   - dims:
29     - num_rows
30     doc: Array of unique identifiers for the rows of this dynamic table.
31     dtype: int
32     name: id
33     neurodata_type_inc: ElementIdentifiers
34     shape:
35       -
36   - doc: Vector columns, including index columns, of this dynamic table.
37     neurodata_type_inc: VectorData
38     quantity: '*'
39   doc: A table for grouping different intracellular recording repetitions together that
40     belong to the same experimental condition.
41   name: experimental_conditions
42   neurodata_type_def: ExperimentalConditionsTable
43   neurodata_type_inc: DynamicTable

```

5.11 Optogenetics

This source module contains `neurodata_types` for opto-genetics data.

5.11.1 OptogeneticSeries

Extends: *TimeSeries*

Description: see [Section 4.10.1](#)

YAML Specification:

```

1  attributes:
2  - default_value: no description
3    doc: Description of the time series.
4    dtype: text
5    name: description
6    required: false
7  - default_value: no comments
8    doc: Human-readable comments about the TimeSeries. This second descriptive field
9        can be used to store additional information, or descriptive information if the
10       primary description field is populated with a computer-readable string.
11    dtype: text
12    name: comments
13    required: false
14 datasets:
15 - attributes:
16   - doc: Unit of measurement for data, which is fixed to 'watts'.
17     dtype: text
18     name: unit
19     value: watts
20   - default_value: 1.0
21     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
22         If the data are stored in acquisition system units or other units that require
23         a conversion to be interpretable, multiply the data by 'conversion' to convert
24         the data to the specified 'unit'. e.g. if the data acquisition system stores
25         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
26         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
27         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
28         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
29     dtype: float32
30     name: conversion
31     required: false
32   - default_value: 0.0
33     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
34         to the specified 'unit'. Two common examples of this include (a) data stored
35         in an unsigned type that requires a shift after scaling to re-center the data,
36         and (b) specialized recording devices that naturally cause a scalar offset with
37         respect to the true units.
38     dtype: float32
39     name: offset
40     required: false
41   - default_value: -1.0

```

(continues on next page)

(continued from previous page)

```

42     doc: Smallest meaningful difference between values in data, stored in the specified
43         by unit, e.g., the change in value of the least significant bit, or a larger
44         number if signal noise is known to be present. If unknown, use -1.0.
45     dtype: float32
46     name: resolution
47     required: false
48     - doc: Optionally describe the continuity of the data. Can be "continuous",
49       ↪ "instantaneous",
50         or "step". For example, a voltage trace would be "continuous", because samples
51         are recorded from a continuous process. An array of lick times would be
52         ↪ "instantaneous",
53         because the data represents distinct moments in time. Times of image presentations
54         would be "step" because the picture remains the same until the next timepoint.
55         This field is optional, but is useful in providing information about the underlying
56         data. It may inform the way this data is interpreted, the way it is visualized,
57         and what analysis methods are applicable.
58     dtype: text
59     name: continuity
60     required: false
61     dims:
62     - num_times
63     doc: Applied power for optogenetic stimulus, in watts.
64     dtype: numeric
65     name: data
66     shape:
67     -
68     - attributes:
69       - doc: Sampling rate, in Hz.
70         dtype: float32
71         name: rate
72       - doc: Unit of measurement for time, which is fixed to 'seconds'.
73         dtype: text
74         name: unit
75         value: seconds
76     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
77         the timestamp of the first sample can be specified and all subsequent ones calculated
78         from the sampling rate attribute.
79     dtype: float64
80     name: starting_time
81     quantity: '?'
82     - attributes:
83       - doc: Value is '1'
84         dtype: int32
85         name: interval
86         value: 1
87       - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
88         dtype: text
89         name: unit
90         value: seconds
91     dims:
92     - num_times
93     doc: Timestamps for samples stored in data, in seconds, relative to the common_

```

(continues on next page)

(continued from previous page)

```

↪ experiment
    master-clock stored in NWBFile.timestamps_reference_time.
    dtype: float64
    name: timestamps
    quantity: '?'
    shape:
    -
- dims:
    - num_times
    doc: Numerical labels that apply to each time point in data for the purpose of querying
        and slicing data by these values. If present, the length of this array should
        be the same size as the first dimension of data.
    dtype: uint8
    name: control
    quantity: '?'
    shape:
    -
- dims:
    - num_control_values
    doc: Description of each control value. Must be present if control is present. If
        present, control_description[0] should describe time points where control == 0.
    dtype: text
    name: control_description
    quantity: '?'
    shape:
    -
doc: An optogenetic stimulus.
groups:
- doc: Lab-specific time and sync information as provided directly from hardware devices
    and that is necessary for aligning all acquired time information to a common_
↪ timebase.
    The timestamp array stores time in the common timebase. This group will usually
    only be populated in TimeSeries that are stored external to the NWB file, in files
    storing raw data. Once timestamp data is calculated, the contents of 'sync' are
    mostly for archival purposes.
    name: sync
    quantity: '?'
links:
- doc: Link to OptogeneticStimulusSite object that describes the site to which this
    stimulus was applied.
    name: site
    target_type: OptogeneticStimulusSite
neurodata_type_def: OptogeneticSeries
neurodata_type_inc: TimeSeries

```

5.11.2 OptogeneticStimulusSite

Extends: *NWBContainer*

Description: see [Section 4.10.2](#)

YAML Specification:

```

1 datasets:
2   - doc: Description of stimulation site.
3     dtype: text
4     name: description
5   - doc: Excitation wavelength, in nm.
6     dtype: float32
7     name: excitation_lambda
8   - doc: Location of the stimulation site. Specify the area, layer, comments on estimation
9     of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names
10    for anatomical regions when possible.
11    dtype: text
12    name: location
13 doc: A site of optogenetic stimulation.
14 links:
15   - doc: Device that generated the stimulus.
16     name: device
17     target_type: Device
18 neurodata_type_def: OptogeneticStimulusSite
19 neurodata_type_inc: NWBContainer

```

5.12 Optical physiology

This source module contains neurodata_types for optical physiology data.

5.12.1 OnePhotonSeries

Extends: *ImageSeries*

Description: see [Section 4.11.1](#)

YAML Specification:

```

1 attributes:
2 - doc: Photomultiplier gain.
3   dtype: float32
4   name: pmt_gain
5   required: false
6 - doc: Lines imaged per second. This is also stored in /general/optophysiology but
7   is kept here as it is useful information for analysis, and so good to be stored
8   w/ the actual data.
9   dtype: float32
10  name: scan_line_rate
11  required: false
12 - doc: Exposure time of the sample; often the inverse of the frequency.
13  dtype: float32
14  name: exposure_time
15  required: false
16 - doc: Amount of pixels combined into 'bins'; could be 1, 2, 4, 8, etc.
17  dtype: uint8
18  name: binning
19  required: false
20 - doc: Power of the excitation in mW, if known.
21  dtype: float32
22  name: power
23  required: false
24 - doc: Intensity of the excitation in mW/mm^2, if known.
25  dtype: float32
26  name: intensity
27  required: false
28 - default_value: no description
29  doc: Description of the time series.
30  dtype: text
31  name: description
32  required: false
33 - default_value: no comments
34  doc: Human-readable comments about the TimeSeries. This second descriptive field
35  can be used to store additional information, or descriptive information if the
36  primary description field is populated with a computer-readable string.
37  dtype: text
38  name: comments
39  required: false
40 datasets:
41 - attributes:

```

(continues on next page)

(continued from previous page)

```

42 - default_value: 1.0
43 doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
44     If the data are stored in acquisition system units or other units that require
45     a conversion to be interpretable, multiply the data by 'conversion' to convert
46     the data to the specified 'unit'. e.g. if the data acquisition system stores
47     values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
48     that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
49     gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
50     values to recorded volts is  $2.5/32768/8000 = 9.5367e-9$ .
51 dtype: float32
52 name: conversion
53 required: false
54 - default_value: 0.0
55 doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
56     to the specified 'unit'. Two common examples of this include (a) data stored
57     in an unsigned type that requires a shift after scaling to re-center the data,
58     and (b) specialized recording devices that naturally cause a scalar offset with
59     respect to the true units.
60 dtype: float32
61 name: offset
62 required: false
63 - default_value: -1.0
64 doc: Smallest meaningful difference between values in data, stored in the specified
65     by unit, e.g., the change in value of the least significant bit, or a larger
66     number if signal noise is known to be present. If unknown, use -1.0.
67 dtype: float32
68 name: resolution
69 required: false
70 - doc: Base unit of measurement for working with the data. Actual stored values
71     are not necessarily stored in these units. To access the data in these units,
72     multiply 'data' by 'conversion' and add 'offset'.
73 dtype: text
74 name: unit
75 - doc: Optionally describe the continuity of the data. Can be "continuous",
76     ↪ "instantaneous",
77     or "step". For example, a voltage trace would be "continuous", because samples
78     are recorded from a continuous process. An array of lick times would be
79     ↪ "instantaneous",
80     because the data represents distinct moments in time. Times of image presentations
81     would be "step" because the picture remains the same until the next timepoint.
82     This field is optional, but is useful in providing information about the underlying
83     data. It may inform the way this data is interpreted, the way it is visualized,
84     and what analysis methods are applicable.
85 dtype: text
86 name: continuity
87 required: false
88 dims:
89 - - frame
90 - - x
91 - - y
92 - - frame
93 - - x

```

(continues on next page)

(continued from previous page)

```

92     - y
93     - z
94     doc: Binary data representing images across frames. If data are stored in an external
95         file, this should be an empty 3D array.
96     dtype: numeric
97     name: data
98     shape:
99     - -
100     -
101     -
102     - -
103     -
104     -
105     -
106 - dims:
107     - rank
108     doc: Number of pixels on x, y, (and z) axes.
109     dtype: int32
110     name: dimension
111     quantity: '?'
112     shape:
113     -
114 - attributes:
115     - dims:
116         - num_files
117         doc: Each external image may contain one or more consecutive frames of the full
118             ImageSeries. This attribute serves as an index to indicate which frames each
119             file contains, to facilitate random access. The 'starting_frame' attribute, hence,
120             contains a list of frame numbers within the full ImageSeries of the first frame
121             of each file listed in the parent 'external_file' dataset. Zero-based indexing
122             is used (hence, the first element will always be zero). For example, if the
123             'external_file' dataset has three paths to files and the first file has 5 frames,
124             the second file has 10 frames, and the third file has 20 frames, then this
125             attribute
126             will have values [0, 5, 15]. If there is a single external file that holds all
127             of the frames of the ImageSeries (and so there is a single element in the
128             'external_file'
129             dataset), then this attribute should have value [0].
130         dtype: int32
131         name: starting_frame
132         shape:
133         -
134     dims:
135     - num_files
136     doc: Paths to one or more external file(s). The field is only present if format=
137     'external'.
138     This is only relevant if the image series is stored in the file system as one
139     or more image file(s). This field should NOT be used if the image is stored in
140     another NWB file and that file is linked to this file.
141     dtype: text
142     name: external_file
143     quantity: '?'

```

(continues on next page)

(continued from previous page)

```

141 shape:
142 -
143 - default_value: raw
144 doc: Format of image. If this is 'external', then the attribute 'external_file'
145     contains the path information to the image files. If this is 'raw', then the raw
146     (single-channel) binary data is stored in the 'data' dataset. If this attribute
147     is not present, then the default format='raw' case is assumed.
148 dtype: text
149 name: format
150 quantity: '?'
151 - attributes:
152   - doc: Sampling rate, in Hz.
153     dtype: float32
154     name: rate
155   - doc: Unit of measurement for time, which is fixed to 'seconds'.
156     dtype: text
157     name: unit
158     value: seconds
159 doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
160     the timestamp of the first sample can be specified and all subsequent ones calculated
161     from the sampling rate attribute.
162 dtype: float64
163 name: starting_time
164 quantity: '?'
165 - attributes:
166   - doc: Value is '1'
167     dtype: int32
168     name: interval
169     value: 1
170   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
171     dtype: text
172     name: unit
173     value: seconds
174 dims:
175   - num_times
176 doc: Timestamps for samples stored in data, in seconds, relative to the common_
177     ↪ experiment
178     master-clock stored in NWBFile.timestamps_reference_time.
179 dtype: float64
180 name: timestamps
181 quantity: '?'
182 shape:
183 -
184 - dims:
185   - num_times
186 doc: Numerical labels that apply to each time point in data for the purpose of querying
187     and slicing data by these values. If present, the length of this array should
188     be the same size as the first dimension of data.
189 dtype: uint8
190 name: control
191 quantity: '?'
192 shape:

```

(continues on next page)

(continued from previous page)

```

192 -
193 - dims:
194   - num_control_values
195   doc: Description of each control value. Must be present if control is present. If
196     present, control_description[0] should describe time points where control == 0.
197   dtype: text
198   name: control_description
199   quantity: '?'
200   shape:
201   -
202 doc: Image stack recorded over time from 1-photon microscope.
203 groups:
204 - doc: Lab-specific time and sync information as provided directly from hardware devices
205   and that is necessary for aligning all acquired time information to a common_
206   ↳timebase.
207   The timestamp array stores time in the common timebase. This group will usually
208   only be populated in TimeSeries that are stored external to the NWB file, in files
209   storing raw data. Once timestamp data is calculated, the contents of 'sync' are
210   mostly for archival purposes.
211   name: sync
212   quantity: '?'
213 links:
214 - doc: Link to ImagingPlane object from which this TimeSeries data was generated.
215   name: imaging_plane
216   target_type: ImagingPlane
217 - doc: Link to the Device object that was used to capture these images.
218   name: device
219   quantity: '?'
220   target_type: Device
221 neurodata_type_def: OnePhotonSeries
neurodata_type_inc: ImageSeries

```

5.12.2 TwoPhotonSeries

Extends: *ImageSeries*

Description: see [Section 4.11.2](#)

YAML Specification:

```

1 attributes:
2 - doc: Photomultiplier gain.
3   dtype: float32
4   name: pmt_gain
5   required: false
6 - doc: Lines imaged per second. This is also stored in /general/optophysiology but
7   is kept here as it is useful information for analysis, and so good to be stored
8   w/ the actual data.
9   dtype: float32
10  name: scan_line_rate
11  required: false
12 - default_value: no description
13  doc: Description of the time series.
14  dtype: text
15  name: description
16  required: false
17 - default_value: no comments
18  doc: Human-readable comments about the TimeSeries. This second descriptive field
19  can be used to store additional information, or descriptive information if the
20  primary description field is populated with a computer-readable string.
21  dtype: text
22  name: comments
23  required: false
24 datasets:
25 - dims:
26   - - width|height
27   - - width|height|depth
28   doc: Width, height and depth of image, or imaged area, in meters.
29   dtype: float32
30   name: field_of_view
31   quantity: '?'
32   shape:
33   - - 2
34   - - 3
35 - attributes:
36   - default_value: 1.0
37   doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
38     If the data are stored in acquisition system units or other units that require
39     a conversion to be interpretable, multiply the data by 'conversion' to convert
40     the data to the specified 'unit'. e.g. if the data acquisition system stores
41     values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
42     that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
43     gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
44     values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
45   dtype: float32
46   name: conversion

```

(continues on next page)

(continued from previous page)

```

47     required: false
48 -   default_value: 0.0
49     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
50         to the specified 'unit'. Two common examples of this include (a) data stored
51         in an unsigned type that requires a shift after scaling to re-center the data,
52         and (b) specialized recording devices that naturally cause a scalar offset with
53         respect to the true units.
54     dtype: float32
55     name: offset
56     required: false
57 -   default_value: -1.0
58     doc: Smallest meaningful difference between values in data, stored in the specified
59         by unit, e.g., the change in value of the least significant bit, or a larger
60         number if signal noise is known to be present. If unknown, use -1.0.
61     dtype: float32
62     name: resolution
63     required: false
64 -   doc: Base unit of measurement for working with the data. Actual stored values
65         are not necessarily stored in these units. To access the data in these units,
66         multiply 'data' by 'conversion' and add 'offset'.
67     dtype: text
68     name: unit
69 -   doc: Optionally describe the continuity of the data. Can be "continuous",
70     ↪ "instantaneous",
71         or "step". For example, a voltage trace would be "continuous", because samples
72         are recorded from a continuous process. An array of lick times would be
73     ↪ "instantaneous",
74         because the data represents distinct moments in time. Times of image presentations
75         would be "step" because the picture remains the same until the next timepoint.
76         This field is optional, but is useful in providing information about the underlying
77         data. It may inform the way this data is interpreted, the way it is visualized,
78         and what analysis methods are applicable.
79     dtype: text
80     name: continuity
81     required: false
82     dims:
83     - - frame
84       - x
85     - - y
86     - - frame
87       - x
88       - y
89       - z
90     doc: Binary data representing images across frames. If data are stored in an external
91         file, this should be an empty 3D array.
92     dtype: numeric
93     name: data
94     shape:
95     - -
96     - -

```

(continues on next page)

(continued from previous page)

```

97     -
98     -
99     -
100 - dims:
101   - rank
102   doc: Number of pixels on x, y, (and z) axes.
103   dtype: int32
104   name: dimension
105   quantity: '?'
106   shape:
107   -
108 - attributes:
109   - dims:
110     - num_files
111     doc: Each external image may contain one or more consecutive frames of the full
112       ImageSeries. This attribute serves as an index to indicate which frames each
113       file contains, to facilitate random access. The 'starting_frame' attribute, hence,
114       contains a list of frame numbers within the full ImageSeries of the first frame
115       of each file listed in the parent 'external_file' dataset. Zero-based indexing
116       is used (hence, the first element will always be zero). For example, if the
117       'external_file' dataset has three paths to files and the first file has 5 frames,
118       the second file has 10 frames, and the third file has 20 frames, then this
119   ↪ attribute
120       will have values [0, 5, 15]. If there is a single external file that holds all
121       of the frames of the ImageSeries (and so there is a single element in the
122   ↪ 'external_file'
123       dataset), then this attribute should have value [0].
124     dtype: int32
125     name: starting_frame
126     shape:
127     -
128     dims:
129     - num_files
130     doc: Paths to one or more external file(s). The field is only present if format=
131   ↪ 'external'.
132     This is only relevant if the image series is stored in the file system as one
133     or more image file(s). This field should NOT be used if the image is stored in
134     another NWB file and that file is linked to this file.
135     dtype: text
136     name: external_file
137     quantity: '?'
138     shape:
139     -
140   - default_value: raw
141     doc: Format of image. If this is 'external', then the attribute 'external_file'
142       contains the path information to the image files. If this is 'raw', then the raw
143       (single-channel) binary data is stored in the 'data' dataset. If this attribute
144       is not present, then the default format='raw' case is assumed.
145     dtype: text
146     name: format
147     quantity: '?'
148   - attributes:

```

(continues on next page)

(continued from previous page)

```

146 - doc: Sampling rate, in Hz.
147   dtype: float32
148   name: rate
149 - doc: Unit of measurement for time, which is fixed to 'seconds'.
150   dtype: text
151   name: unit
152   value: seconds
153 doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
154     the timestamp of the first sample can be specified and all subsequent ones calculated
155     from the sampling rate attribute.
156 dtype: float64
157 name: starting_time
158 quantity: '?'
159 - attributes:
160   - doc: Value is '1'
161     dtype: int32
162     name: interval
163     value: 1
164   - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
165     dtype: text
166     name: unit
167     value: seconds
168 dims:
169   - num_times
170 doc: Timestamps for samples stored in data, in seconds, relative to the common_
171     ↪ experiment
172     master-clock stored in NWBFile.timestamps_reference_time.
173 dtype: float64
174 name: timestamps
175 quantity: '?'
176 shape:
177 -
178 - dims:
179   - num_times
180 doc: Numerical labels that apply to each time point in data for the purpose of querying
181     and slicing data by these values. If present, the length of this array should
182     be the same size as the first dimension of data.
183 dtype: uint8
184 name: control
185 quantity: '?'
186 shape:
187 -
188 - dims:
189   - num_control_values
190 doc: Description of each control value. Must be present if control is present. If
191     present, control_description[0] should describe time points where control == 0.
192 dtype: text
193 name: control_description
194 quantity: '?'
195 shape:
196 -
197 doc: Image stack recorded over time from 2-photon microscope.

```

(continues on next page)

(continued from previous page)

groups:

- **doc:** Lab-specific time and sync information as provided directly from hardware devices and that is necessary for aligning all acquired time information to a common ↵
↵timebase.

The timestamp array stores time in the common timebase. This group will usually only be populated in TimeSeries that are stored external to the NWB file, in files storing raw data. Once timestamp data is calculated, the contents of 'sync' are mostly for archival purposes.

name: sync

quantity: '?'

links:

- **doc:** Link to ImagingPlane object from which this TimeSeries data was generated.

name: imaging_plane

target_type: ImagingPlane

- **doc:** Link to the Device object that was used to capture these images.

name: device

quantity: '?'

target_type: Device

neurodata_type_def: TwoPhotonSeries

neurodata_type_inc: ImageSeries

5.12.3 RoiResponseSeries

Extends: *TimeSeries*

Description: see [Section 4.11.3](#)

YAML Specification:

```

1 attributes:
2 - default_value: no description
3   doc: Description of the time series.
4   dtype: text
5   name: description
6   required: false
7 - default_value: no comments
8   doc: Human-readable comments about the TimeSeries. This second descriptive field
9       can be used to store additional information, or descriptive information if the
10      primary description field is populated with a computer-readable string.
11   dtype: text
12   name: comments
13   required: false
14 datasets:
15 - attributes:
16   - default_value: 1.0
17     doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
18         If the data are stored in acquisition system units or other units that require
19         a conversion to be interpretable, multiply the data by 'conversion' to convert
20         the data to the specified 'unit'. e.g. if the data acquisition system stores
21         values in this object as signed 16-bit integers (int16 range -32,768 to 32,767)
22         that correspond to a 5V range (-2.5V to 2.5V), and the data acquisition system
23         gain is 8000X, then the 'conversion' multiplier to get from raw data acquisition
24         values to recorded volts is 2.5/32768/8000 = 9.5367e-9.
25     dtype: float32
26     name: conversion
27     required: false
28   - default_value: 0.0
29     doc: Scalar to add to the data after scaling by 'conversion' to finalize its coercion
30         to the specified 'unit'. Two common examples of this include (a) data stored
31         in an unsigned type that requires a shift after scaling to re-center the data,
32         and (b) specialized recording devices that naturally cause a scalar offset with
33         respect to the true units.
34     dtype: float32
35     name: offset
36     required: false
37   - default_value: -1.0
38     doc: Smallest meaningful difference between values in data, stored in the specified
39         by unit, e.g., the change in value of the least significant bit, or a larger
40         number if signal noise is known to be present. If unknown, use -1.0.
41     dtype: float32
42     name: resolution
43     required: false
44   - doc: Base unit of measurement for working with the data. Actual stored values
45       are not necessarily stored in these units. To access the data in these units,
46       multiply 'data' by 'conversion' and add 'offset'.

```

(continues on next page)

(continued from previous page)

```

47     dtype: text
48     name: unit
49     - doc: Optionally describe the continuity of the data. Can be "continuous",
↳ "instantaneous",
50       or "step". For example, a voltage trace would be "continuous", because samples
51       are recorded from a continuous process. An array of lick times would be
↳ "instantaneous",
52       because the data represents distinct moments in time. Times of image presentations
53       would be "step" because the picture remains the same until the next timepoint.
54       This field is optional, but is useful in providing information about the underlying
55       data. It may inform the way this data is interpreted, the way it is visualized,
56       and what analysis methods are applicable.
57     dtype: text
58     name: continuity
59     required: false
60     dims:
61     - - num_times
62     - - num_times
63     - num_ROIs
64     doc: Signals from ROIs.
65     dtype: numeric
66     name: data
67     shape:
68     - -
69     - -
70     -
71     - doc: DynamicTableRegion referencing into an ROITable containing information on the
72       ROIs stored in this timeseries.
73     name: rois
74     neurodata_type_inc: DynamicTableRegion
75     - attributes:
76     - doc: Sampling rate, in Hz.
77       dtype: float32
78       name: rate
79     - doc: Unit of measurement for time, which is fixed to 'seconds'.
80       dtype: text
81       name: unit
82       value: seconds
83     doc: Timestamp of the first sample in seconds. When timestamps are uniformly spaced,
84       the timestamp of the first sample can be specified and all subsequent ones calculated
85       from the sampling rate attribute.
86     dtype: float64
87     name: starting_time
88     quantity: '?'
89     - attributes:
90     - doc: Value is '1'
91       dtype: int32
92       name: interval
93       value: 1
94     - doc: Unit of measurement for timestamps, which is fixed to 'seconds'.
95       dtype: text
96       name: unit

```

(continues on next page)

(continued from previous page)

```

97     value: seconds
98     dims:
99     - num_times
100     doc: Timestamps for samples stored in data, in seconds, relative to the common_
↪ experiment
101         master-clock stored in NWBFile.timestamps_reference_time.
102     dtype: float64
103     name: timestamps
104     quantity: '?'
105     shape:
106     -
107 - dims:
108     - num_times
109     doc: Numerical labels that apply to each time point in data for the purpose of querying
110         and slicing data by these values. If present, the length of this array should
111         be the same size as the first dimension of data.
112     dtype: uint8
113     name: control
114     quantity: '?'
115     shape:
116     -
117 - dims:
118     - num_control_values
119     doc: Description of each control value. Must be present if control is present. If
120         present, control_description[0] should describe time points where control == 0.
121     dtype: text
122     name: control_description
123     quantity: '?'
124     shape:
125     -
126     doc: ROI responses over an imaging plane. The first dimension represents time. The
127         second dimension, if present, represents ROIs.
128     groups:
129     - doc: Lab-specific time and sync information as provided directly from hardware devices
130         and that is necessary for aligning all acquired time information to a common_
↪ timebase.
131         The timestamp array stores time in the common timebase. This group will usually
132         only be populated in TimeSeries that are stored external to the NWB file, in files
133         storing raw data. Once timestamp data is calculated, the contents of 'sync' are
134         mostly for archival purposes.
135     name: sync
136     quantity: '?'
137 neurodata_type_def: RoiResponseSeries
138 neurodata_type_inc: TimeSeries

```

5.12.4 DfOverF

Extends: *NWBDataInterface*

Description: see [Section 4.11.4](#)

YAML Specification:

```

1 default_name: DfOverF
2 doc: dF/F information about a region of interest (ROI). Storage hierarchy of dF/F
3   should be the same as for segmentation (i.e., same names for ROIs and for image
4   planes).
5 groups:
6 - doc: RoiResponseSeries object(s) containing dF/F for a ROI.
7   neurodata_type_inc: RoiResponseSeries
8   quantity: +
9 neurodata_type_def: DfOverF
10 neurodata_type_inc: NWBDataInterface

```

5.12.5 Fluorescence

Extends: *NWBDataInterface*

Description: see [Section 4.11.5](#)

YAML Specification:

```

1 default_name: Fluorescence
2 doc: Fluorescence information about a region of interest (ROI). Storage hierarchy
3   of fluorescence should be the same as for segmentation (ie, same names for ROIs
4   and for image planes).
5 groups:
6 - doc: RoiResponseSeries object(s) containing fluorescence data for a ROI.
7   neurodata_type_inc: RoiResponseSeries
8   quantity: +
9 neurodata_type_def: Fluorescence
10 neurodata_type_inc: NWBDataInterface

```

5.12.6 ImageSegmentation

Extends: *NWBDataInterface*

Description: see Section 4.11.6

YAML Specification:

```

1 default_name: ImageSegmentation
2 doc: Stores pixels in an image that represent different regions of interest (ROIs)
3   or masks. All segmentation for a given imaging plane is stored together, with storage
4   for multiple imaging planes (masks) supported. Each ROI is stored in its own subgroup,
5   with the ROI group containing both a 2D mask and a list of pixels that make up this
6   mask. Segments can also be used for masking neuropil. If segmentation is allowed
7   to change with time, a new imaging plane (or module) is required and ROI names should
8   remain consistent between them.
9 groups:
10 - doc: Results from image segmentation of a specific imaging plane.
11   neurodata_type_inc: PlaneSegmentation
12   quantity: +
13 neurodata_type_def: ImageSegmentation
14 neurodata_type_inc: NWBDataInterface

```

5.12.7 PlaneSegmentation

Extends: `DynamicTable`

Description: see [Section 4.11.7](#)

YAML Specification:

```

1 attributes:
2   - dims:
3     - num_columns
4     doc: The names of the columns in this table. This should be used to specify an order
5       to the columns.
6     dtype: text
7     name: colnames
8     shape:
9       -
10  - doc: Description of what is in this dynamic table.
11    dtype: text
12    name: description
13 datasets:
14   - dims:
15     - - num_roi
16       - num_x
17       - num_y
18     - - num_roi
19       - num_x
20       - num_y
21     - num_z
22    doc: ROI masks for each ROI. Each image mask is the size of the original imaging
23      plane (or volume) and members of the ROI are finite non-zero.
24    name: image_mask
25    neurodata_type_inc: VectorData
26    quantity: '?'
27    shape:
28      - -
29        -
30        -
31      - -
32        -
33        -
34      -
35   - doc: Index into pixel_mask.
36     name: pixel_mask_index
37     neurodata_type_inc: VectorIndex
38     quantity: '?'
39   - doc: 'Pixel masks for each ROI: a list of indices and weights for the ROI. Pixel
40     masks are concatenated and parsing of this dataset is maintained by the
41     ↪PlaneSegmentation'
42     dtype:
43       - doc: Pixel x-coordinate.
44         dtype: uint32
45         name: x
46       - doc: Pixel y-coordinate.
```

(continues on next page)

(continued from previous page)

```

46     dtype: uint32
47     name: y
48     - doc: Weight of the pixel.
49     dtype: float32
50     name: weight
51     name: pixel_mask
52     neurodata_type_inc: VectorData
53     quantity: '?'
54     - doc: Index into voxel_mask.
55     name: voxel_mask_index
56     neurodata_type_inc: VectorIndex
57     quantity: '?'
58     - doc: 'Voxel masks for each ROI: a list of indices and weights for the ROI. Voxel
59       masks are concatenated and parsing of this dataset is maintained by the
60       ↪PlaneSegmentation'
61     dtype:
62     - doc: Voxel x-coordinate.
63       dtype: uint32
64       name: x
65     - doc: Voxel y-coordinate.
66       dtype: uint32
67       name: y
68     - doc: Voxel z-coordinate.
69       dtype: uint32
70       name: z
71     - doc: Weight of the voxel.
72       dtype: float32
73       name: weight
74     name: voxel_mask
75     neurodata_type_inc: VectorData
76     quantity: '?'
77     - dims:
78       - num_rows
79       doc: Array of unique identifiers for the rows of this dynamic table.
80       dtype: int
81       name: id
82       neurodata_type_inc: ElementIdentifiers
83       shape:
84     - doc: Vector columns, including index columns, of this dynamic table.
85       neurodata_type_inc: VectorData
86       quantity: '*'
87     doc: Results from image segmentation of a specific imaging plane.
88     groups:
89     - doc: Image stacks that the segmentation masks apply to.
90       groups:
91     - doc: One or more image stacks that the masks apply to (can be one-element stack).
92       neurodata_type_inc: ImageSeries
93       quantity: '*'
94       name: reference_images
95     links:
96     - doc: Link to ImagingPlane object from which this data was generated.

```

(continues on next page)

(continued from previous page)

```
97  name: imaging_plane
98  target_type: ImagingPlane
99  neurodata_type_def: PlaneSegmentation
100 neurodata_type_inc: DynamicTable
```

5.12.8 ImagingPlane

Extends: *NWBContainer*

Description: see [Section 4.11.8](#)

YAML Specification:

```

1 datasets:
2   - doc: Description of the imaging plane.
3     dtype: text
4     name: description
5     quantity: '?'
6   - doc: Excitation wavelength, in nm.
7     dtype: float32
8     name: excitation_lambda
9   - doc: Rate that images are acquired, in Hz. If the corresponding TimeSeries is present,
10     the rate should be stored there instead.
11     dtype: float32
12     name: imaging_rate
13     quantity: '?'
14   - doc: Calcium indicator.
15     dtype: text
16     name: indicator
17   - doc: Location of the imaging plane. Specify the area, layer, comments on estimation
18     of area/layer, stereotaxic coordinates if in vivo, etc. Use standard atlas names
19     for anatomical regions when possible.
20     dtype: text
21     name: location
22   - attributes:
23     - default_value: 1.0
24       doc: Scalar to multiply each element in data to convert it to the specified 'unit'.
25         If the data are stored in acquisition system units or other units that require
26         a conversion to be interpretable, multiply the data by 'conversion' to convert
27         the data to the specified 'unit'. e.g. if the data acquisition system stores
28         values in this object as pixels from x = -500 to 499, y = -500 to 499 that
29         correspond
30         to a 2 m x 2 m range, then the 'conversion' multiplier to get from raw data
31         acquisition pixel units to meters is 2/1000.
32       dtype: float32
33       name: conversion
34       required: false
35     - default_value: meters
36       doc: Base unit of measurement for working with the data. The default value is
37         'meters'.
38       dtype: text
39       name: unit
40       required: false
41   dims:
42     - height
43     - width
44     - x, y, z
45     - height
46     - width

```

(continues on next page)

(continued from previous page)

```

46     - depth
47     - x, y, z
48     doc: DEPRECATED Physical position of each pixel. 'xyz' represents the position of
49         the pixel relative to the defined coordinate space. Deprecated in favor of origin_
↳ coords
50         and grid_spacing.
51     dtype: float32
52     name: manifold
53     quantity: '?'
54     shape:
55         - -
56         -
57         - 3
58         - -
59         -
60         -
61         - 3
62 - attributes:
63     - default_value: meters
64       doc: Measurement units for origin_coords. The default value is 'meters'.
65       dtype: text
66       name: unit
67       required: false
68     dims:
69         - - x, y
70         - - x, y, z
71     doc: Physical location of the first element of the imaging plane (0, 0) for 2-D
72         data or (0, 0, 0) for 3-D data. See also reference_frame for what the physical
73         location is relative to (e.g., bregma).
74     dtype: float32
75     name: origin_coords
76     quantity: '?'
77     shape:
78         - - 2
79         - - 3
80 - attributes:
81     - default_value: meters
82       doc: Measurement units for grid_spacing. The default value is 'meters'.
83       dtype: text
84       name: unit
85       required: false
86     dims:
87         - - x, y
88         - - x, y, z
89     doc: Space between pixels in (x, y) or voxels in (x, y, z) directions, in the specified
90         unit. Assumes imaging plane is a regular grid. See also reference_frame to interpret
91         the grid.
92     dtype: float32
93     name: grid_spacing
94     quantity: '?'
95     shape:
96         - - 2

```

(continues on next page)

(continued from previous page)

```

97 - - 3
98 - doc: Describes reference frame of origin_coords and grid_spacing. For example, this
99   can be a text description of the anatomical location and orientation of the grid
100   defined by origin_coords and grid_spacing or the vectors needed to transform or
101   rotate the grid to a common anatomical axis (e.g., AP/DV/ML). This field is necessary
102   to interpret origin_coords and grid_spacing. If origin_coords and grid_spacing
103   are not present, then this field is not required. For example, if the microscope
104   takes 10 x 10 x 2 images, where the first value of the data matrix (index (0,
105   0, 0)) corresponds to (-1.2, -0.6, -2) mm relative to bregma, the spacing between
106   pixels is 0.2 mm in x, 0.2 mm in y and 0.5 mm in z, and larger numbers in x means
107   more anterior, larger numbers in y means more rightward, and larger numbers in
108   z means more ventral, then enter the following -- origin_coords = (-1.2, -0.6,
109   -2) grid_spacing = (0.2, 0.2, 0.5) reference_frame = "Origin coordinates are relative
110   to bregma. First dimension corresponds to anterior-posterior axis (larger index
111   = more anterior). Second dimension corresponds to medial-lateral axis (larger
112   index = more rightward). Third dimension corresponds to dorsal-ventral axis (larger
113   index = more ventral)."
```

dtype: text

name: reference_frame

quantity: '?'

doc: An imaging plane and its metadata.

groups:

- **doc**: An optical channel used to record from an imaging plane.

neurodata_type_inc: OpticalChannel

quantity: +

links:

- **doc**: Link to the Device object that was used to record from this electrode.

name: device

target_type: Device

neurodata_type_def: ImagingPlane

neurodata_type_inc: NWBContainer

5.12.9 OpticalChannel

Extends: *NWBContainer*

Description: see [Section 4.11.9](#)

YAML Specification:

```

1 datasets:
2 - doc: Description or other notes about the channel.
3   dtype: text
4   name: description
5 - doc: Emission wavelength for channel, in nm.
6   dtype: float32
7   name: emission_lambda
8 doc: An optical channel used to record from an imaging plane.
9 neurodata_type_def: OpticalChannel
10 neurodata_type_inc: NWBContainer

```

5.12.10 MotionCorrection

Extends: *NWBDataInterface*

Description: see Section 4.11.10

YAML Specification:

```

1 default_name: MotionCorrection
2 doc: 'An image stack where all frames are shifted (registered) to a common coordinate
3   system, to account for movement and drift between frames. Note: each frame at each
4   point in time is assumed to be 2-D (has only x & y dimensions).'
5 groups:
6 - doc: Results from motion correction of an image stack.
7   neurodata_type_inc: CorrectedImageStack
8   quantity: +
9 neurodata_type_def: MotionCorrection
10 neurodata_type_inc: NWBDataInterface

```

5.12.11 CorrectedImageStack

Extends: *NWBDataInterface*

Description: see [Section 4.11.11](#)

YAML Specification:

```
1 doc: Results from motion correction of an image stack.
2 groups:
3 - doc: Image stack with frames shifted to the common coordinates.
4   name: corrected
5   neurodata_type_inc: ImageSeries
6 - doc: Stores the x,y delta necessary to align each frame to the common coordinates,
7   for example, to align each frame to a reference image.
8   name: xy_translation
9   neurodata_type_inc: TimeSeries
10 links:
11 - doc: Link to ImageSeries object that is being registered.
12   name: original
13   target_type: ImageSeries
14 neurodata_type_def: CorrectedImageStack
15 neurodata_type_inc: NWBDataInterface
```

5.13 Retinotopy

This source module contains neurodata_type for retinotopy data.

5.13.1 ImagingRetinotopy

Extends: *NWBDataInterface*

Description: see [Section 4.12.1](#)

YAML Specification:

```

1 datasets:
2   - attributes:
3     - dims:
4       - num_rows, num_cols
5       doc: 'Number of rows and columns in the image. NOTE: row, column representation
6         is equivalent to height, width.'
7       dtype: int32
8       name: dimension
9       shape:
10        - 2
11     - dims:
12       - height, width
13       doc: Size of viewing area, in meters.
14       dtype: float32
15       name: field_of_view
16       shape:
17        - 2
18     - doc: Unit that axis data is stored in (e.g., degrees).
19       dtype: text
20       name: unit
21   dims:
22     - num_rows
23     - num_cols
24   doc: Phase response to stimulus on the first measured axis.
25   dtype: float32
26   name: axis_1_phase_map
27   shape:
28     -
29     -
30   - attributes:
31     - dims:
32       - num_rows, num_cols
33       doc: 'Number of rows and columns in the image. NOTE: row, column representation
34         is equivalent to height, width.'
35       dtype: int32
36       name: dimension
37       shape:
38        - 2
39     - dims:
40       - height, width
41       doc: Size of viewing area, in meters.

```

(continues on next page)

(continued from previous page)

```

42     dtype: float32
43     name: field_of_view
44     shape:
45     - 2
46 - doc: Unit that axis data is stored in (e.g., degrees).
47     dtype: text
48     name: unit
49 dims:
50 - num_rows
51 - num_cols
52 doc: Power response on the first measured axis. Response is scaled so 0.0 is no
53     power in the response and 1.0 is maximum relative power.
54 dtype: float32
55 name: axis_1_power_map
56 quantity: '?'
57 shape:
58 -
59 -
60 - attributes:
61 - dims:
62     - num_rows, num_cols
63     doc: 'Number of rows and columns in the image. NOTE: row, column representation
64         is equivalent to height, width.'
65     dtype: int32
66     name: dimension
67     shape:
68     - 2
69 - dims:
70     - height, width
71     doc: Size of viewing area, in meters.
72     dtype: float32
73     name: field_of_view
74     shape:
75     - 2
76 - doc: Unit that axis data is stored in (e.g., degrees).
77     dtype: text
78     name: unit
79 dims:
80 - num_rows
81 - num_cols
82 doc: Phase response to stimulus on the second measured axis.
83 dtype: float32
84 name: axis_2_phase_map
85 shape:
86 -
87 -
88 - attributes:
89 - dims:
90     - num_rows, num_cols
91     doc: 'Number of rows and columns in the image. NOTE: row, column representation
92         is equivalent to height, width.'
93     dtype: int32

```

(continues on next page)

(continued from previous page)

```

94     name: dimension
95     shape:
96     - 2
97 -   dims:
98     - height, width
99     doc: Size of viewing area, in meters.
100    dtype: float32
101    name: field_of_view
102    shape:
103    - 2
104 -   doc: Unit that axis data is stored in (e.g., degrees).
105    dtype: text
106    name: unit
107  dims:
108  - num_rows
109  - num_cols
110  doc: Power response on the second measured axis. Response is scaled so 0.0 is no
111      power in the response and 1.0 is maximum relative power.
112  dtype: float32
113  name: axis_2_power_map
114  quantity: '?'
115  shape:
116  -
117  -
118 -   dims:
119     - axis_1, axis_2
120     doc: Two-element array describing the contents of the two response axis fields.
121         Description should be something like ['altitude', 'azimuth'] or ['radius', 'theta'].
122     dtype: text
123     name: axis_descriptions
124     shape:
125     - 2
126 -   attributes:
127     - doc: Number of bits used to represent each value. This is necessary to determine
128         maximum (white) pixel value.
129       dtype: int32
130       name: bits_per_pixel
131     - dims:
132       - num_rows, num_cols
133       doc: 'Number of rows and columns in the image. NOTE: row, column representation
134           is equivalent to height, width.'
135       dtype: int32
136       name: dimension
137       shape:
138       - 2
139     - dims:
140       - height, width
141       doc: Size of viewing area, in meters.
142       dtype: float32
143       name: field_of_view
144       shape:
145       - 2

```

(continues on next page)

(continued from previous page)

```

146 - doc: Focal depth offset, in meters.
147   dtype: float32
148   name: focal_depth
149 - doc: Format of image. Right now only 'raw' is supported.
150   dtype: text
151   name: format
152 dims:
153 - num_rows
154 - num_cols
155 doc: 'Gray-scale image taken with same settings/parameters (e.g., focal depth,
↪wavelength)
156   as data collection. Array format: [rows][columns].'
157 dtype: uint16
158 name: focal_depth_image
159 quantity: '?'
160 shape:
161 -
162 -
163 - attributes:
164   - dims:
165     - num_rows, num_cols
166     doc: 'Number of rows and columns in the image. NOTE: row, column representation
167       is equivalent to height, width.'
168     dtype: int32
169     name: dimension
170     shape:
171     - 2
172   - dims:
173     - height, width
174     doc: Size of viewing area, in meters.
175     dtype: float32
176     name: field_of_view
177     shape:
178     - 2
179   dims:
180   - num_rows
181   - num_cols
182   doc: Sine of the angle between the direction of the gradient in axis_1 and axis_2.
183   dtype: float32
184   name: sign_map
185   quantity: '?'
186   shape:
187   -
188   -
189 - attributes:
190   - doc: Number of bits used to represent each value. This is necessary to determine
191     maximum (white) pixel value
192     dtype: int32
193     name: bits_per_pixel
194   - dims:
195     - num_rows, num_cols
196     doc: 'Number of rows and columns in the image. NOTE: row, column representation

```

(continues on next page)

(continued from previous page)

```

197     is equivalent to height, width.'
198     dtype: int32
199     name: dimension
200     shape:
201     - 2
202     - dims:
203     - height, width
204     doc: Size of viewing area, in meters.
205     dtype: float32
206     name: field_of_view
207     shape:
208     - 2
209     - doc: Format of image. Right now only 'raw' is supported.
210     dtype: text
211     name: format
212     dims:
213     - num_rows
214     - num_cols
215     doc: 'Gray-scale anatomical image of cortical surface. Array structure: [rows][columns]
216     dtype: uint16
217     name: vasculature_image
218     shape:
219     -
220     -
221     default_name: ImagingRetinotopy
222     doc: 'Intrinsic signal optical imaging or widefield imaging for measuring retinotopy.
223     Stores orthogonal maps (e.g., altitude/azimuth; radius/theta) of responses to specific
224     stimuli and a combined polarity map from which to identify visual areas. This group
225     does not store the raw responses imaged during retinotopic mapping or the stimuli
226     presented, but rather the resulting phase and power maps after applying a Fourier
227     transform on the averaged responses. Note: for data consistency, all images and
228     arrays are stored in the format [row][column] and [row, col], which equates to [y][x].
229     Field of view and dimension arrays may appear backward (i.e., y before x).'
230     neurodata_type_def: ImagingRetinotopy
231     neurodata_type_inc: NWBDataInterface

```


Part III

History and Legal

MAKING A PULL REQUEST

Actions to take on each PR that modifies the schema and does not prepare the schema for a public release (this is also in the [GitHub PR template](#)):

If the current schema version on “dev” is a public release, then:

1. Update the version string in `docs/format/source/conf.py` and `common/namespace.yaml` to the next version with the suffix “-alpha”
2. Add a new section in the release notes for the new version with the date “Upcoming”

Always:

1. Add release notes for the PR to `docs/format/source/format_release_notes.rst`

Documentation or internal changes to the repo (i.e., changes that do not affect the schema files) do not need to be accompanied with a version bump or addition to the release notes.

MERGING PRS AND MAKING RELEASES

Public release: a tagged release of the schema. The version string **MUST NOT** have a suffix indicating a pre-release, such as “-alpha”. The current “dev” branch of PyNWB and all PyNWB releases **MUST** point to a public release of nwb-schema. All schema that use nwb-schema as a submodule **MUST** also point only to public releases.

Internal release: a state of the schema “dev” branch where the version string ends with “-alpha”.

The default branch of nwb-schema is “dev”. **The “dev” branch holds the bleeding edge version of the nwb-schema specification.**

PRs should be made to “dev”. Every PR should include an update to docs/format/source/format_release_notes.rst. If the current version is a public release, then the PR should also update the version of the schema in two places: docs/format/source/conf.py and core/nwb.namespace.yaml. The new version should be the next bugfix/minor/major version of the schema with the suffix “-alpha”. For example, if the current schema on “dev” has version “2.2.0”, then a PR implementing a bug fix should update the schema version from “2.2.0” to “2.2.1-alpha”. Appending the “-alpha” suffix ensures that any person or API accessing the default “dev” branch of the repo containing an internal release of the schema receives the schema with a version string that is distinct from public releases of the schema. If the current schema on “dev” is already an internal release, then the version string does not need to be updated unless the PR requires an upgrade in the version (e.g., from bugfix to minor).

PyNWB should contain a branch and PR that tracks the “dev” branch of nwb-schema. Before a public release of nwb-schema is made, this PyNWB branch should be checked to ensure that when the new release is made, the branch can be merged without issue.

Immediately prior to making a new public release, the version of the schema should be updated to remove the “-alpha” suffix and the documentation and release notes should be updated as needed (see next section).

The current “dev” branch of PyNWB and all PyNWB releases **MUST** always point to a public release of nwb-schema. If a public release contains an internally released version of nwb-schema, e.g., from an untagged commit on the “dev” branch, then it will be difficult to find the version (commit) of nwb-schema that was used to create an NWB file when the schema is not cached.

MAKING A RELEASE CHECKLIST

Before merging:

1. Update requirements versions as needed
2. Update legal file dates and information in `Legal.txt`, `license.txt`, `README.md`, `docs/format/source/conf.py`, and any other locations as needed
3. Update `README.rst` as needed
4. Update the version string in `docs/format/source/conf.py`, `core/nwb.namespace.yaml`, and `/core/nwb.file.yaml` (remove “-alpha” suffix)
5. Update `docs/format/source/conf.py` as needed
6. Update release notes (set release date) in `docs/format/source/format_release_notes.rst` and any other docs as needed
7. Test docs locally (`cd docs/format; make fulldoc`) where the `nwb-schema` submodule in the local version of `PyNWB` is fully up-to-date with the head of the dev branch.
8. Push changes to a new PR and make sure all PRs to be included in this release have been merged. Add `?template=release.md` to the PR URL to auto-populate the PR with this checklist.
9. Check that the `readthedocs` build for this PR succeeds (build latest to pull the new branch, then activate and build docs for new branch): <https://readthedocs.org/projects/nwb-schema/builds/>

After merging:

1. Create a new git tag. Pull the latest dev branch locally, run `git tag [version] --sign`, copy and paste the release notes into the tag message, and run `git push --tags`.
2. On the [GitHub tags](#) page, click “...” -> “Create release” for the new tag on the right side of the page. Copy and paste the release notes into the release message, update the formatting if needed (reST to Markdown), and set the title to the version string.
3. Check that the `readthedocs` “latest” and “stable” builds run and succeed. Delete the `readthedocs` build for the merged PR. <https://readthedocs.org/projects/nwb-schema/builds/>
4. Update the `nwb-schema` submodule in the `PyNWB` branch corresponding to this schema version to point to the tagged commit.

This checklist can also be found in the [GitHub release PR template](#).

The time between merging this PR and creating a new public release should be minimized.

THE “DEV” BRANCH AND RELEASES

The default branch is “dev”. The “dev” branch will hold the bleeding edge version of the NWB format specification, language specification, and storage specification. PRs should be made to “dev”. Every PR should include an update to the corresponding format/language/storage release notes. If the PR involves a change to the NWB format, the PR should also update the version of the format in three places: `/docs/format/conf.py`, `/core/nwb.namespace.yaml` and `/core/nwb.file.yaml`. The new version string should be the next bugfix/minor/major version of the format with the suffix “a” (for “alpha”). For example, if the format is currently in version “2.2.0” and the format is then updated / released internally with a bug fix, then the PR for that bug fix should update the format version from “2.2.0” to “2.2.1a”. Appending the “a” suffix ensures that any person or API accessing the default dev branch of the repo containing an internally released version of the schema receives the schema with a version string that is distinct from the full public released versions of the schema.

Before merging a PR on nwb-schema, developers should take care to test their changes locally with the latest version of PyNWB and MatNWB to ensure compatibility. If the APIs require changes to work with the PR, those changes should be implemented and tested locally before merging the PR to ensure that the API changes can be implemented and no further changes to the schema are required.

When a new public release is ready, the branches of the APIs, PyNWB and MatNWB, that track nwb-schema should be checked to ensure that when the new release is made, the branches in the APIs can be merged without issue.

Immediately prior to a new public release, the version of the format should be updated to remove the any alphabetic, suffixes, e.g., “a”, “b”, and “rc”, and the documentation and release notes should be checked and updated (see release process documents).

It is important that all public releases of nwb-schema contain a publicly released version of hdmf-common-schema. If a public release contains an internally released version of hdmf-common-schema, e.g., from an untagged commit on the “dev” branch, then tracking the identity of the included hdmf-common-schema becomes difficult and the same version string could point to two different versions of the schema.

For the same reason, it is important that all public releases of the APIs, PyNWB and MatNWB, contain a publicly released version of nwb-schema. Starting with nwb-schema version 2.2.0, the dev branch and all public releases of PyNWB and MatNWB include only publicly released versions of nwb-schema. For more details, see the [PyNWB software process documentation](https://pynwb.readthedocs.io/en/stable/software_process.html).

The [NWB Extensions Versioning Guidelines](https://nwb-extensions.github.io/versioning_guidelines) are used to guide versioning of the NWB core schema, as well as extensions to NWB.

RELEASE NOTES

10.1 2.7.0 (February 7, 2024)

10.1.1 Minor changes

- Fixed typos in docstrings. (#560)
- Deprecated `ImagingRetinotopy` neurodata type. (#565)
- Modified `OptogeneticSeries` to allow 2D data, primarily in extensions of `OptogeneticSeries`. (#564)
- Added optional `stimulus_template` column to `IntracellularStimuliTable` as part of the `IntracellularRecordingsTable`. (#545)
- Added support for `NWBDataInterface` and `DynamicTable` in `NWBFile.stimulus`. (#559)

10.2 2.6.0 (January 17, 2023)

10.2.1 Minor changes

- Added `OnePhotonSeries`. (#523)
- `Subject.age` has a new optional attribute, `reference`, which can take a value of “birth” (default) or “gestational”. (#525)
- Added “in seconds” to the doc of `Units.spike_times`. (#530)

10.3 2.5.0 (June 14, 2022)

10.3.1 Major changes

- Shape of `SpatialSeries.data` is more restrictive to prevent > 3 columns. (#510)

10.3.2 Minor changes

- The elements *x*, *y*, *z*, *imp* and *filtering* are now optional instead of required. (#506)
- Added an offset attribute to all TimeSeries objects to allow enhanced translation to scientific units. (#494)
- Allowed NWBFile/stimulus/templates to contain Images objects. (#459)
- Added new optional “order_of_images” dataset to Images that contains an ordered list of object references to Image objects stored in the same Images object. This dataset must be used if the images are referred to by index, e.g., from an IndexSeries object. Created new neurodata type ImageReferences which should be used for this dataset. (#459, #518, #519, #520)
- Overhauled IndexSeries type (#459): - Fixed dtype of data dataset of IndexSeries (int32 -> uint32). - Updated unit attribute of data to have fixed value “N/A”. - Updated docstrings for the conversion, resolution, and offset attributes of data to indicate that these fields are not used. - Added link to an Images object, which contains an ordered collection of images. - Discouraged use of the indexed_timeseries link to an ImageSeries.
- Updated TimeIntervals to use the new TimeSeriesReferenceVectorData type. This does not alter the overall structure of TimeIntervals in a major way aside from changing the value of the neurodata_type attribute in the file from VectorData to TimeSeriesReferenceVectorData. This change replaces the existing TimeIntervals.timeseries column with a TimeSeriesReferenceVectorData type column of the same name and overall schema. This change facilitates creating common functionality around TimeSeriesReferenceVectorData. This change affects all existing TimeIntervals tables as part of the intervals/group, i.e., intervals/epochs, intervals/trials, and intervals/invalid_times. (#486)
- Clarified the doc string for the reference column of the electrodes table. (#498)
- Added cell_id field to IntracellularElectrode. (#512)

10.4 2.4.0 (Aug. 11, 2021)

10.4.1 Major changes

- Added new TimeSeriesReferenceVectorData type for referencing ranges of TimeSeries from a VectorData column (#470)
- Integrated the intracellular electrophysiology experiment metadata table structure developed as part of the [ndx-icephys-meta](#) extension project with NWB (#470). This includes the following new types:
 - IntracellularRecordingsTable is an AlignedDynamicTable for managing individual intracellular recordings and to group together a stimulus and response from a single electrode recording. The table contains the following category tables:
 - * IntracellularElectrodesTable; a DynamicTable for storing metadata about the IntracellularElectrode used
 - * IntracellularStimuliTable; a DynamicTable for storing metadata about the recorded stimulus TimeSeries using the new TimeSeriesReferenceVectorData type to reference TimeSeries
 - * IntracellularResponsesTable; a DynamicTable for storing metadata about the recorded response TimeSeries using the new TimeSeriesReferenceVectorData type to reference TimeSeries
 - SimultaneousRecordingsTable is a DynamicTable for grouping different intracellular recordings from the IntracellularRecordingsTable together that were recorded simultaneously from different electrodes and for storing metadata about simultaneous recordings

- `SequentialRecordingsTable` is a `DynamicTable` for grouping different sequential recordings from the `SimultaneousRecordingsTable` together and storing metadata about sequential recordings
- `RepetitionsTable` a `DynamicTable` for grouping different sequential intracellular recordings from the `SequentialRecordingsTable` together and storing metadata about repetitions
- `ExperimentalConditionsTable` is a `DynamicTable` for grouping different intracellular recording repetitions from the `RepetitionsTable` together and storing metadata about experimental conditions
- Added the new intracellular electrophysiology metadata tables to `/general/intracellular_ephys` as part of NWBFile (#470)

10.4.2 Deprecations

- `SweepTable` has been deprecated in favor of the new intracellular electrophysiology metadata tables. Use of `SweepTable` is still possible but no longer recommended. (#470)
- `/general/intracellular_ephys/filtering` has been deprecated in favor of `IntracellularElectrode.filtering` (#470)

10.4.3 Bug Fixes

- Fixed incorrect dtype for electrodes table column “filtering” (float -> text) (#478)
- Removed `quantity:` * from the type definitions of `OptogeneticStimulusSite` and `ImagingPlane`. This change improves clarity of the schema to follow best practices. It has no functional effect on the schema. (#472)
- Updated `ImageSeries` to have its data dataset be required. Since `ImageSeries` is a `TimeSeries` and `TimeSeries.data` is required, `ImageSeries.data` should also be a required dataset. Otherwise this creates problems for inheritance and validation. If `ImageSeries` data are stored in an external file, then `ImageSeries.data` should be set to an empty 3D array. (#481)

10.5 2.3.0 (May 12, 2021)

- Add optional `waveforms` column to the `Units` table.
- Add optional `strain` field to `Subject`.
- Add to `DecompositionSeries` an optional `DynamicTableRegion` called `source_channels`.
- Add to `ImageSeries` an optional link to `Device`.
- Add optional `continuity` field to `TimeSeries`.
- Add optional `filtering` attribute to `ElectricalSeries`.
- Clarify documentation for electrode impedance and filtering.
- Add description of extra fields.
- Set the `stimulus_description` for `IZeroCurrentClamp` to have the fixed value `N/A`.
- Update `hdmf-common-schema` from 1.1.3 to version 1.5.0.
 - The `HDMF-experimental` namespace was added, which includes the `ExternalResources` and `EnumData` data types. Schema in the `HDMF-experimental` namespace are experimental and subject to breaking changes at any time.
 - Added experimental data type `ExternalResources` for storing ontology information / external resource references.
 - Added experimental data type `EnumData` to store data from a set of fixed values.
 - Changed dtype for datasets within `CSRMatrix` from ‘int’ to ‘uint’ and added missing `data_type_inc:` Container to the `CSRMatrix` type.
 - Added data

type `SimpleMultiContainer`, a `Container` for storing other `Container` and `Data` objects together. - Added data type `AlignedDynamicTable`, a `DynamicTable` type with support for categories (or sub-headings) each described by a separate `DynamicTable`. - Fixed missing dtype for `VectorIndex`. - `VectorIndex` now extends `VectorData` instead of `Index`. - Removed unused and non-functional `Index` data type. - See https://hdmf-common-schema.readthedocs.io/en/latest/format_release_notes.html for full release notes.

10.6 2.2.5 (May 29, 2020)

- Add schema validation CI.
- Fix incorrect dims and shape for `ImagingPlane.origin_coords` and `ImagingPlane.grid_spacing`, and fix incorrect dims for `TwoPhotonSeries.field_of_view`.

10.7 2.2.4 (April 14, 2020)

- Fix typo in `nwb.phys.yaml` that prevents proper parsing of the schema.

10.8 2.2.3 (April 13, 2020)

- Move nested type definitions to root of YAML files. This does not functionally change the schema but simplifies parsing of the schema and extensions by APIs.
- Make `ImagingPlane.imaging_rate` optional to handle cases where an imaging plane is associated with multiple time series with different rates.
- Add release process documentation.

10.9 2.2.2 (March 2, 2020)

- Fix shape and dims of `OpticalSeries.data` for color images
- Allow more than one `OpticalChannel` object in `ImagingPlane`
- Update `hdmf-common-schema` to 1.1.3. This fixes missing 'shape' and 'dims' key for types `VectorData`, `VectorIndex`, and `DynamicTableRegion`.
- Revert changes to `nwb.retinotopy.yaml` in 2.1.0 which break backward compatibility and were not supported by the APIs in any case. Changes will be revisited in a future version.

10.10 2.2.1 (January 14, 2020)

- Fixed incorrect version numbers in `nwb.file.yaml` and `hdmf-common-schema/common/namespace.yaml`.

10.11 2.2.0 (January 6, 2020)

- Moved common data structures such as `Container` and `DynamicTable` to `hdmf-common-schema`.
 - The `hdmf-common-schema` repo is now included as a submodule
 - See <https://github.com/NeurodataWithoutBorders/nwb-schema/pull/307> for details
- Added “channel_conversion” dataset to `ElectricalSeries` to represent per-channel conversion factors.
- Added “sampling_rate” and “unit” attributes to “waveform_mean” and “waveform_sd” datasets/columns in `Units` table.
- Added “description” and “manufacturer” attributes to `Device`.
- Deprecated `ImagingPlane` “manifold” in favor of “origin_coords” and “grid_spacing”
- Use “text” data type for all `DynamicTable` “colnames”. Previously, only ASCII was allowed.
- Use “text” data type for electrode table columns “location” and “group_name”. Previously, only ASCII was allowed.
- Added to description to make electrode x,y,z consistent with CCF reference. https://allensdk.readthedocs.io/en/latest/reference_space.html
- Added “position” dataset with compound data type x,y,z in `ElectrodeGroup`.
- Avoid enforcing “uint64” for sweep numbers for better compatibility. Use `uint` instead which is 32bit.
- Set `dtype` for `Image` and its subtypes to `numeric`. (note: technically this breaks backwards compatibility, in the schema, but the `pynwb` API has always enforced that Images have a numeric type, and realistically we do not think users are storing strings in an `Image` dataset.)
- Added “resolution” attribute to “spike_times” column of `Units`.
- Changed the “quantity” key of attribute `Units.resolution` to “required” for schema language compliance.
- Removed “required” key from dataset `ImageSeries.field_of_view` for schema language compliance.
- Replaced “required” keys with “quantity” keys for `ImagingPlane.origin_coords` and `ImagingPlane.grid_spacing` for schema language compliance.
- Refactored `ImagingRetinotopy` type to reduce redundancy.
- Added “doc” key to `ImagingRetinotopy.axis_2_power_map` for schema language compliance.
- Fixed makefiles for generating documentation on Windows.
- Added optional “reference” column in “electrodes” table.
- Changed dims of `ImageSeries` from (frame, y, x) to (frame, x, y) and (frame, z, y, x) to (frame, x, y, z) to be consistent with the dimension ordering in `PlaneSegmentation`.
- Changed dims of `Image` from (y, x) to (x, y). (note: as far as we know, users of NWB 2.0 that use the `Image` type encode their data as (x, y)) to be consistent with the dimension ordering in `ImageSeries`.
- Updated `hdmf-common-schema` to version 1.1.0 which includes:
 - The ‘colnames’ attribute of `DynamicTable` changed from data type ‘ascii’ to ‘text’.
 - Improved documentation and type docstrings.

10.12 2.1.0 (September 2019)

- Improved documentation in “doc” attribute of many types
- Removed “help” attribute
 - Now that the schema is cached in an NWB file by default, this attribute is redundant, confusing, used inconsistently, clutters the file and documentation, and adds substantial boilerplate to writing extensions
 - See <https://github.com/NeurodataWithoutBorders/nwb-schema/issues/270> for details
- Removed static “description” attribute from some types
 - These were intended to be a “help” attribute, which has now been removed
 - For example, TimeIntervals dataset “start_time” attribute “description” had a fixed value that is now removed
- Reordered keys
 - This standardizes the order of keys across types and makes the schema more readable
 - See <https://github.com/NeurodataWithoutBorders/nwb-schema/issues/274> for details
- Added “dims” attribute for datasets where “shape” was specified without “dims”
 - The “dims” attribute describes the data along each dimension of the dataset and is helpful to provide alongside “shape”
 - For example, NWBFile dataset “keywords” has attribute “shape” has one entry: “null”. The attribute “dims” was added with one entry: “num_keywords”
- Removed redundant specifications that are inherited from a parent type
- ElectrodeGroup link “device”: optional -> required
 - This was previously required by PyNWB
 - See <https://github.com/NeurodataWithoutBorders/pynwb/issues/1025> for details
- Matched default and fixed values of datasets and attributes with the documentation and intended use
 - IZeroClampSeries dataset “bias_current” unspecified value -> fixed value 0.0
 - IZeroClampSeries dataset “bridge_balance” unspecified value -> fixed value 0.0
 - IZeroClampSeries dataset “capacitance_compensation” unspecified value -> fixed value 0.0
 - TimeSeries dataset “resolution” default value: 0.0 -> -1.0
 - ImagingRetinotopy dataset “axis_descriptions” attribute “shape”: null -> 2
 - DecompositionSeries dataset “data” attribute “unit” default value unspecified -> default value “no unit”
 - VoltageClampStimulusSeries, CurrentClampSeries, IZeroClampSeries attribute “unit” has fixed value “volts”
 - CurrentClampStimulusSeries, VoltageClampSeries, attribute “unit” has fixed value “amperes”
- NWBFile dataset “experimenter” and “related_publications” change from scalar to 1-D, unlimited arrays
 - This allows the “experimenter” and “related_publications” dataset to encode multiple values
 - See <https://github.com/NeurodataWithoutBorders/pynwb/issues/985> and <https://github.com/NeurodataWithoutBorders/nwb-schema/issues/299> for details
- Standardized units to be plural, lower-case, SI units

- TimeSeries dataset “starting_time” attribute “unit” fixed value: “Seconds” -> “seconds”
- TimeSeries dataset “timestamps” attribute “unit” fixed value: “Seconds” -> “seconds”
- ElectricalSeries dataset “data” attribute “unit” default value (previously optional): “volt” -> fixed value “volts”
- SpikeEventSeries dataset “data” attribute “unit” default value (previously optional): “volt” -> fixed value “volts”
- SpikeEventSeries dataset “timestamps” attribute “unit” fixed value (previously optional): “Seconds” -> fixed value “seconds”
- EventDetection dataset “times” attribute “unit” default value: “Seconds” -> “seconds”
- VoltageClampSeries dataset “capacitance_fast” attribute “unit” default value “Farad” -> fixed value “farads”
- VoltageClampSeries dataset “capacitance_slow” attribute “unit” default value “Farad” -> fixed value “farads”
- VoltageClampSeries dataset “resistance_comp_bandwidth” attribute “unit” default value “Hz” -> fixed value “hertz”
- VoltageClampSeries dataset “resistance_comp_correction” attribute “unit” default value “percent” -> fixed value “percent”
- VoltageClampSeries dataset “resistance_comp_prediction” attribute “unit” default value “percent” -> fixed value “percent”
- VoltageClampSeries dataset “whole_cell_capacitance_comp” attribute “unit” default value “Farad” -> fixed value “farads”
- VoltageClampSeries dataset “whole_cell_series_resistance_comp” attribute “unit” default value “Ohm” -> fixed value “ohms”
- OptogeneticSeries dataset “data” attribute “unit” default value “watt” -> fixed value “watts”
- ImagingPlane dataset “manifold” attribute “unit” default value “Meter” -> default value “meters”
- see <https://github.com/NeurodataWithoutBorders/nwb-schema/issues/277> for details
- Made Units table column “waveform_mean” and “waveform_sd” have shape num_units x num_samples x num_electrodes
 - See <https://github.com/NeurodataWithoutBorders/pynwb/pull/1008> for details
- Made CorrectedImageStack and ImagingRetinotopy inherit from the more specific NWBDataInterface instead of NWBContainer
- Added a scratch space for saving arbitrary datasets to an NWBFile
 - NWB is cumbersome as a format for day-to-day work. There is a lot of overhead to save one-off analysis results to an NWB file. To save new datasets, a user has write an extension. This is a lot of work for a result that may just be tossed out.
 - “scratch” is now an optional top-level group under NWBFile that can hold NWBContainer groups and ScratchData datasets
 - The scratch space is explicitly for non-standardized data that is not intended for reuse by others. Standard NWB types, and extensions if required, should always be used for any data that you intend to share. As such, published data should not include scratch data and a user should be able to ignore any data stored in scratch to use a file.
 - See <https://github.com/NeurodataWithoutBorders/nwb-schema/issues/286> for details

- Set the default value for the dataset “format” to “raw” and clarified the documentation for ImageSeries
 - See <https://github.com/NeurodataWithoutBorders/nwb-schema/pull/308/files> for details
- Add required attribute `object_id` to all NWB Groups and Datasets with an assigned `neurodata_type`
 - See https://nwb-schema.readthedocs.io/en/latest/format_description.html#common-attributes for details

Backwards compatibility: The PyNWB and MatNWB APIs can read 2.0 files with the 2.1 schema.

10.13 2.0.2 (June 2019)

- Updated authors
- Removed NWBFile subgroup “specifications” because schema is now cached
 - See <https://github.com/NeurodataWithoutBorders/pynwb/pull/953> for details
- Made DecompositionSeries link “source_timeseries” optional
 - See <https://github.com/NeurodataWithoutBorders/pynwb/pull/955> for details

10.14 2.0.1 (March 2019)

Change: Added `doc` and `title` descriptions for the individual source files included in the main namespace.

Reason: Enhance documentation of the namespace and facilitate presentation of types in autogenerated docs by making it easier to sort `neurodata_types` into meaningful categories (i.e., sections) with appropriate tiles and descriptions.

Backwards compatibility: No changes to the actual specification of the format are made. 2.0.1 is fully compatible with 2.0.0.

10.15 2.0.0 (January 2019)

Main release: November 2017 (Beta), November 2018 (RC), January 2019 (final)

10.15.1 Added new base data types: *NWBContainer*, *NWBData*, *NWBDataInterface*

Change: Added common base types for Groups, Datasets, and for Groups storing primary experiment data

Reason Collect common functionality and ease future evolution of the standard

Specific Changes

- *NWBContainer* defines a common base type for all Groups with a `neurodata_type` and is now the base type of all main data group types in the NWB format, including *TimeSeries*. This also means that all group types now inherit the required `help` and `source` attribute from *NWBContainer*. A number of `neurodata_types` have been updated to add the missing `help` (see <https://github.com/NeurodataWithoutBorders/nwb-schema/pull/37/files> for details)
- *NWBDataInterface* extends *NWBContainer* and replaces *Interface* from NWB 1.x. It has been renamed to ease intuition. *NWBDataInterface* serves as base type for primary data (e.g., experimental or analysis data) and is used to distinguish in the schema between non-metadata data containers and metadata containers. (see <https://github.com/NeurodataWithoutBorders/nwb-schema/pull/116/files> for details)

- *NWBData* defines a common base type for all Datasets with a `neurodata_type` and serves a similar function to *NWBContainer* only for Datasets instead of Groups.

10.15.2 Support general data structures for data tables and vector data

10.15.2.1 Support row-based and column-based tables

Change: Add support for storing tabular data via row-based and column-based table structures.

Reason: Simplify storage of complex metadata. Simplify storage of dynamic and variable-length metadata.

Format Changes:

- **Row-based tables:** are implemented via a change in the specification language through support for compound data types. The advantage of row-based tables is that they 1) allow referencing of sets of rows via region-references to a single dataset (e.g., a set of electrodes), 2) make it easy to add rows by appending to a single dataset, 3) make it easy to read individual rows of a table (but require reading the full table to extract the data of a single column). Row-based tables are used to simplify, e.g., the organization of electrode-metadata in NWB 2 (see above). (See the [specification language release notes](#) for details about the addition of compound data types in the schema).
 - *Referencing rows in a row-based tables:* Subsets of rows can be referenced directly via a region-reference to the row-based table. Subsets
 - *Referencing columns in a row-based table:* This is currently not directly supported, but could be implemented via a combination of an object-reference to the table and a list of the labels of columns.
- **Column-based tables:** are implemented via the new `neurodata_type` *DynamicTable*. A *DynamicTable* is simplified-speaking just a collection of an arbitrary number of *VectorData* table column datasets (all with equal length) and a dataset storing row ids and a dataset storing column names. The advantage of the column-based store is that it 1) makes it easy to add new columns to the table without the need for extensions and 2) the column-based storage makes it easy to read individual columns efficiently (while reading full rows requires reading from multiple datasets). *DynamicTable* is used, e.g., to enhance storage of trial data. (See <https://github.com/NeurodataWithoutBorders/pynwb/pull/536/files>)
 - *Referencing rows in column-based tables:* As *DynamicTable* consists of multiple datasets (compared to row-based tables which consists of a single 1D dataset with a compound datatype) it is not possible to reference a set of rows with a single region reference. To address this issue, NWB defines *DynamicTableRegion* (added later in [PR634 \(PyNWB\)](#)) dataset type, which stores a list of integer indices (row index) and also has an attribute `table` with the object reference to the corresponding *DynamicTable*.
 - *Referencing columns in a column-based table:* As each column is a separate dataset, columns of a column-based *DynamicTable* can be directly referenced via links, object-references and region-references.

10.15.2.2 Enable efficient storage of large numbers of vector data elements

Change Introduce `neurodata_types` *VectorData*, *VectorIndex*, *ElementIdentifiers*

Reason To efficiently store spike data as part of *UnitTimes* a new, more efficient data structure was required. This builds the general, reusable types to define efficient data storage for large numbers of data vectors in efficient, consolidated arrays, which enable more efficient read, write, and search (see *Improved storage of unit-based data*).

Format Changes

- *VectorData* : Data values from a series of data elements are concatenated into a single array. This allows all elements to be stored efficiently in a single data array.
- *VectorIndex* : 1D dataset of exclusive stop-indices selecting subranges in *VectorData*. In addition, the `target` attribute stores an object reference to the corresponding *VectorData* dataset. With this we can efficiently access

single sub-vectors associated with single elements from the [VectorData](#) collection. An alternative approach would be store region-references as part of the [VectorIndex](#). We opted for stop-indices mainly because they are more space-efficient and are easier to use for introspection of index values than region references.

- [ElementIdentifiers](#) : 1D array for storing unique identifiers for the elements in a [VectorIndex](#).

See [Improved storage of unit-based data](#) for an illustration and specific example use in practice. See also [I116 \(nwb-schema\)](#) and [PR382 \(PyNWB\)](#) for further details.

10.15.3 Use new table and vector data structures to improve data organization

10.15.3.1 Improved organization of electrode metadata in `/general/extracellular_ephys`

Change: Consolidate metadata from related electrodes (e.g., from a single device) in a single location.

Example: Previous versions of the format specified in `/general/extracellular_ephys` for each electrode a group `<electrode_group_X>` that stores 3 text datasets with a description, device name, and location, respectively. The main `/general/extracellular_ephys` group then contained in addition the following datasets:

- `electrode_group` text array describing for each `electrode_group` (implicitly referenced by index) which device (shank, probe, tetrode, etc.) was used,
- `electrode_map` array with the x,y,z locations of each electrode
- `filtering`, i.e., a single string describing the filtering for all electrodes (even though each electrode might be from different devices), and `iv`),
- `impedance`, i.e., a single text array for impedance (i.e., the user has to know which format the string has, e.g., a float or a tuple of floats for impedance ranges).

Reason:

- Avoid explosion of the number of groups and datasets. For example, in the case of an ECoG grid with 128 channels one had to create 128 groups and corresponding datasets to store the required metadata about the electrodes using the original layout.
- Simplify access to related metadata. E.g., access to metadata from all electrodes of a single device requires resolution of a potentially large number of implicit links and access to a large number of groups (one per electrode) and datasets.
- Improve performance of metadata access operations. E.g., to access the `location` of all electrodes corresponding to a single recording in an `<ElectricalSeries>` in the original layout required iterating over a potentially large number of groups and datasets (one per electrode), hence, leading to a large number of small, independent read/write/seek operations, causing slow performance on common data accesses. Using the new layout, these kind of common data accesses can often be resolved via a single read/write
- Ease maintenance, use, and development through consolidation of related metadata

Format Changes

- Added specification of a new neurodata type `<ElectrodeGroup>` group. Each `<ElectrodeGroup>` contains the following datasets to describe the metadata of a set of related electrodes (e.g., all electrodes from a single device):
 - `description` : text dataset (for the group)
 - `device`: Soft link to the device in `/general/devices/`
 - `location`: Text description of the location of the device
- Added table-like dataset `electrodes` that consolidates all electrode-specific metadata. This is a [DynamicTable](#) describing for each electrode:

- `id` : a user-specified unique identifier
- `x, y, z` : The floating point coordinate for the electrode
- `imp` : the impedance of the channel
- `location` : The location of channel within the subject e.g. brain region
- `filtering` : Description of hardware filtering
- `group` : Object reference to the `ElectrodeGroup` object
- `group_name` : The name of the `ElectrodeGroup`
- Updated `/general/extracellular_ephys` as follows:
 - Replaced `/general/extracellular_ephys/<electrode_group_X>` group (and all its contents) with the new `<ElectrodeGroup>`
 - Removed `/general/extracellular_ephys/electrode_map` dataset. This information is now stored in the `ElectrodeTable`.
 - Removed `/general/extracellular_ephys/electrode_group` dataset. This information is now stored in `<ElectrodeGroup>/device`.
 - Removed `/general/extracellular_ephys/impedance` This information is now stored in the `ElectrodeTable`.
 - Removed `/general/extracellular_ephys/filtering` This information is now stored in the `ElectrodeTable`.

Note: In NWB 2.0Beta the refactor originally used a row-based table for the `ElectrodeTable` based on a compound data type as described in [#I6 \(new-schema\)](#), i.e., `electrodes` was a 1D compound dataset. This was later changed to a column-based `DynamicTable` (see [Support row-based and column-based tables](#)). The main reason for this later change was mainly to avoid the need for large numbers of user-extensions to add electrode metadata (see [#I623 \(PyNWB\)](#) and [PR634 \(PyNWB\)](#) for details.) This change also removed the optional `description` column as it can be added easily by the user to the `DynamicTable` if required.

10.15.3.2 Improved storage of lab-specific meta-data

Reason: Labs commonly have specific meta-data associated with sessions, and we need a good way to organize this within NWB.

Changes: The datatype `LabMetaData` has been added to the schema within `/general` so that an extension can be added to `/general` by inheriting from `LabMetaData`.

For further details see [I19 \(nwb-schema\)](#) and [PR751 \(PyNWB\)](#).

10.15.3.3 Improved storage of Spectral Analyses (Signal Decomposition)

Reason: Labs commonly use analyses that involve frequency decomposition or bandpass filtering of neural or behavioral data, and it is difficult to standardize this data and meta-data across labs.

Changes: A new datatype, `DecompositionSeries` has been introduced to offer a common interface for labs to exchange the result of time-frequency analysis. The new type offers a `DynamicTable` to allow users to flexibly add features of bands, and a place to directly link to the `TimeSeries` that was used.

For further details see [#I46 \(nwb-schema\)](#) and [#PR764 \(PyNWB\)](#)

10.15.3.4 Improved storage of Images

Reason:

- **Improve consistency of schema:** Previously there was a reference to Image from ImageSeries, however Image was not defined in the schema
- **Support different static image types**

Changes: *Image* was added as a base type, and subtypes were defined: *GrayscaleImage*, *RGBImage*, and *RGBAImage* (The “A” in “RGBA” is for alpha, i.e., opacity).

10.15.3.5 Improved storage of ROIs

Reason:

- **Improve efficiency:** Similar to epochs, in NWB 1.x ROIs were stored as a single group per ROI. This structure is inefficient for storing large numbers of ROIs.
- **Make links explicit:** The relationship of *RoiResponseSeries* to ROI objects was implicit (i.e. ROI was specified by a string), so one had to know a priori which *ImageSegmentation* and *ImagingPlane* was used to produce the ROIs.
- **Support 3D ROIs:** Allow users to add 3D ROIs collected from a multi-plane image.

Changes: The main types for storing ROIs in NWB 2 are *ImageSegmentation* which stores 0 or more *PlaneSegmentation*. *PlaneSegmentation* is a *DynamicTable* for managing image segmentation results of a specific imaging plane. The ROIs are referenced by *RoiResponseSeries* which stores the ROI responses over an imaging plane. During the development of NWB 2 the management of ROIs has been improved several times. Here we outline the main changes (several of which were ultimately merged together in the *PlaneSegmentation* type).

1. Added neurodata_type *ImageMasks* replacing *ROI.img_mask* (from NWB 1.x) with (a) a 3D dataset with shape [num_rois, num_x_pixels, num_y_pixels] (i.e. an array of planar image masks) or (b) a 4D dataset with shape [num_rois, num_x_pixels, num_y_pixels, num_z_pixels] (i.e. an array of volumetric image masks) *ImageMasks* was subsequently merged with *PlaneSegmentation* and is represented by the *VectorData* table column *image_mask* in the table.
2. Added neurodata_type *PixelMasks* which replaces *ROI.pix_mask/ROI.pix_mask_weight* (from NWB 1.x) with a table that has columns “x”, “y”, and “weight” (i.e. combining *ROI.pix_mask* and *ROI.pix_mask_weight* into a single table). *PixelMasks* was subsequently merged with *PlaneSegmentation* and is represented by the *VectorData* dataset *pixel__mask* that is referenced from the table via the *VectorIndex* column *pixel_mask_index*.
3. Added analogous neurodata_type *VoxelMasks* with a table that has columns “x”, “y”, “z”, and “weight” for 3D ROIs. *VoxelMasks* was subsequently merged with *PlaneSegmentation* and is represented by the *VectorData* dataset *voxel_mask* that is referenced from the table via the *VectorIndex* column *voxel_mask_index*.
4. Added neurodata_type *ROItable* which defines a table for storing references to the image mask and pixel mask for each ROI (see item 1,2). The *ROItable* type was subsequently merged with the *PlaneSegmentation* type and as such does no longer appear as a separate type in the NWB 2 schema but *PlaneSegmentation* takes the function of *ROItable*.
5. Added neurodata_type *ROItableRegion* for referencing a subset of elements in an *ROItable*. Subsequently *ROItableRegion* has been replaced by *DynamicTableRegion* as the *ROItable* changed to a *DynamicTable* and was merged with *PlaneSegmentation* (see 8.)
6. Replaced *RoiResponseSeries.roi_names* with *RoiResponseSeries.rois*, which is a *DynamicTableRegion* into the *PlaneSegmentation* table of ROIs (see items 3,4). (Before *ROItable* was converted from a row-based to a column-based table, *RoiResponseSeries.rois* had been changed to a *ROItableRegion* which was then subsequently changed to a corresponding *DynamicTableRegion*)

7. Removed `RoiResponseSeries.segmentation_interface`. This information is available through `RoiResponseSeries.rois` (described above in 5.)
8. Assigned neurodata_type *PlaneSegmentation* to the image_plan group in *ImageSegmentation* and updated it to use the `ROITable`, `ImageMasks`, `PixelMasks`, and `VoxelMasks` (see items 1-4 above). Specifically, *PlaneSegmentation* has been changed to be a *DynamicTable* and `ROITable`, `ImageMasks`, `PixelMasks`, and `VoxelMasks` have been merged into the *PlaneSegmentation* table, resulting in the removal of the `ROITable`, `ROITableRegion`, `ImageMasks`, `PixelMasks`, and `VoxelMasks` types.

For additional details see also:

- [PR391 \(PyNWB\)](#) and [I118 \(nwb-schema\)](#) for details on the main refactoring of ROI storage,
- [PR665 \(PyNWB\)](#) and [I663 \(PyNWB\)](#) (and previous issue [I643 \(PyNWB\)](#)) for details on the subsequent refactor using *DynamicTable*, and
- [PR688 \(PyNWB\)](#) and [I554 \(nwb-schema\)](#) for details on 3D ROIs,

10.15.3.6 Improved storage of unit-based data

In NWB 1.0.x data about spike units was stored across a number of different neurodata_types, specifically `UnitTimes`, `ClusterWaveforms`, and `Clustering`. This structure had several critical shortcomings, which were addressed in three main phases during the development of NWB 2.

Problem 1: Efficiency: In NWB 1.x each unit was stored as a separate group `unit_n` containing the times and `unit_description` for unit with index `n`. In cases where users have a very large number of units, this was problematic with regard to performance. To address this challenge `UnitTimes` has been restructured in NWB 2 to use the new *VectorData*, *VectorIndex*, *ElementIdentifiers* data structures (see *Enable efficient storage of large numbers of vector data elements*). Specifically, NWB 2 replaced `unit_n` (from NWB 1.x, also referred to by neurodata_type *SpikeUnit* in NWB 2beta) groups in `UnitTimes` with the following data:

- `unit_ids`: *ElementIdentifiers* dataset for storing unique ids for each element
- `spike_times_index`: *VectorIndex* dataset with region references into the spike times dataset
- `spike_times`: *VectorData* dataset storing the actual spike times data of all units in a single data array (for efficiency).

See also [I116 \(nwb-schema\)](#) and [PR382 \(PyNWB\)](#) for further details.

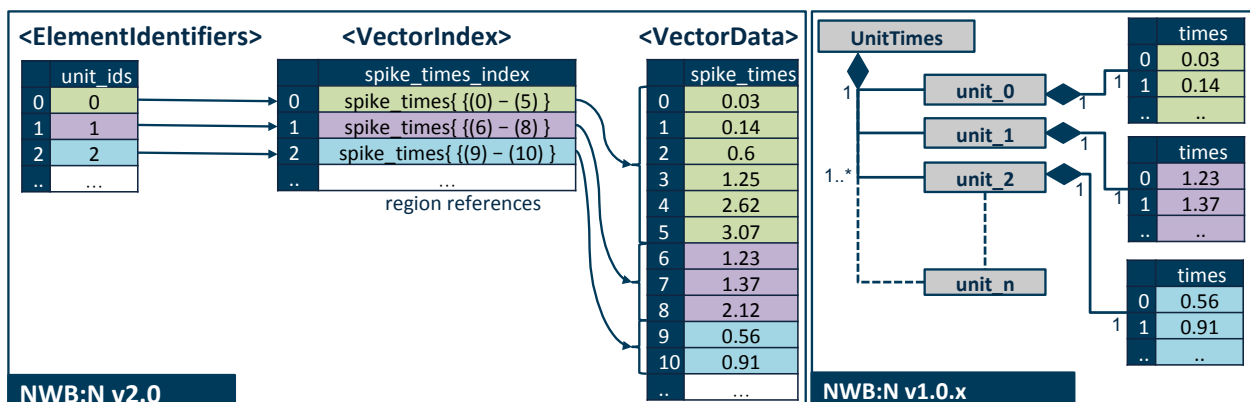


Fig. 10.1: Overview of the basic data structure for storing `UnitTimes` using the *VectorData* (`spike_times`), *VectorIndex* (`spike_times_index`), and *ElementIdentifiers* (`unit_ids`) data structures.

Problem 2: Dynamic Metadata: Users indicated that it was not easy to store user-defined metadata about units. To address this challenge, NWB 2 added an optional top-level group `units/` (which was subsequently moved to `/`

intervals/units) which is a [DynamicTable](#) with `id` and `description` columns and optional additional user-defined table columns. See [PR597 on PyNWB](#) for detailed code changes. See the [PyNWB docs](#) for a short tutorial on how to use unit metadata. See [NWBFile Groups: /units](#) for an overview of the unit schema.

Problem 3: Usability: Finally, users found that storing unit data was challenging due to the fact that the information was distributed across a number of different types. To address this challenge, NWB 2.0 integrates `UnitTimes`, `ClusterWaveforms`, and `Clustering` (deprecated) into the new column-based table `units/` (i.e., `intervals/units`) (which still uses the optimized vector data storage to efficiently store spike times). See for discussions and [I674 on PyNWB](#) (and related [I675 on PyNWB](#)) and the pull request [PR684 on PyNWB](#) for detailed changes.

Together these changes have resulted in the following improved structure for storing unit data and metadata in NWB 2.0.

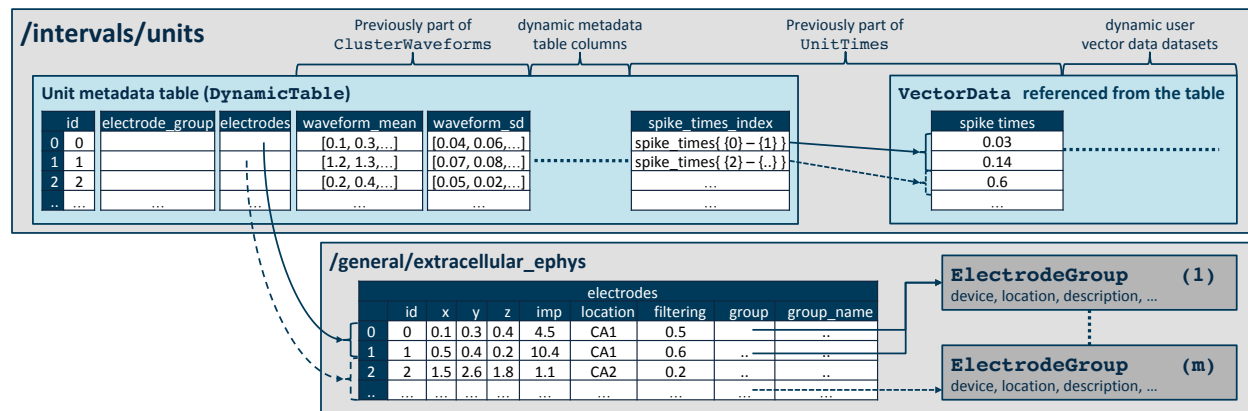


Fig. 10.2: Overview of the data structure for storing spiking unit data and metadata in NWB 2.0.

In addition to `spike_times`, the `units` table includes the following optional columns:

- `obs_intervals`: intervals indicating the time intervals over which this unit was recorded.
- `electrodes`: range references to the electrodes table indicating which electrodes from which this unit was recorded.
- `electrode_group`: may be used instead of `electrodes` if mutually exclusive electrode groups are sufficient.
- `waveform_mean`: mean waveform across all recorded spikes.
- `waveform_sd`: standard deviation from the mean across all recorded spikes.

10.15.3.7 Improved support for sweep-based information

Changes Added [SweepTable](#) type stored in `/general/intracellular_ephys`

Reason: In Icephys it is common to have sweeps (i.e., a group of `PatchClampSeries` belonging together, were up to two `TimeSeries` are from one electrode, including other `TimeSeries` not related to an electrode (aka TTL channels)). NWB 1.0.x did not support the concept of sweeps, so it was not possible to link different `TimeSeries` for sweeps. The goal of this change is to allow users to find the `TimeSeries` which are from one sweep without having to iterate over all present `TimeSeries`.

Format Changes Added `neurodata_type SweepTable` to `/general/intracellular_ephys`. `SweepTable` is a `DynamicTable` <sec-DynamicTable> storing for each sweep a the `sweep_number` and the `series_index`. The later is a `VectorIndex` pointing to a `VectorData` dataset describing belonging `PatchClampSeries` to the sweeps. See [I499 \(PyNWB\)](#) and [PR701 \(PyNWB\)](#) for further details.

10.15.4 Improved specification of reference time stamp(s)

To improve the specification of reference time, NWB adopts ISO8061 for storing datetimes and adds `timestamps_reference_time` as explicit zero for all timestamps in addition to the `session_start_time`.

10.15.4.1 Improve standardization of reference time specification using ISO8061

Changes: Modify `session_start_time` and `file_create_date` to enforce use of ISO 8601 datetime strings

Reason: Standardize the specification of timestamps to ensure consistent programmatic and human interpretation

Format Changes: Updated `session_start_time` and `file_create_date` to use dtype: `isodatetime` that was added as dedicated dtype to the specification language. For details see [PR641 \(PyNWB\)](#) and [I50 \(nwb-schema\)](#).

10.15.4.2 Improved specification of reference time

Change: Add field `timestamps_reference_time`, allowing users to explicitly specify a date and time corresponding to time zero for all timestamps in the nwb file.

Reason: Previously `session_start_time` served both as the indicator for the start time of a session as well as the global reference time for a file. Decoupling the two makes the global reference time explicit and enables users to use times relative to the session start as well as other reference time frames, e.g., using POSIX time. This also makes the specification easier to develop against, since this will explicitly specify the offset to obtain relative timestamps, eliminating the need for APIs to guess based on range.

Format Changes: Added top-level field `timestamps_reference_time`. See [PR709 \(PyNWB\)](#) and [I49 \(nwb-schema\)](#) for further details.

10.15.5 Improved storage of time intervals

10.15.5.1 Improved storage of epoch data

Change: Store epoch data as a table to improve efficiency, usability and extensibility.

Reason: In NWB 1.x Epochs are stored as a single group per Epoch. Within each Epoch, the index into each `TimeSeries` that the Epoch applies to was stored as a single group. This structure is inefficient for storing large numbers of Epochs.

Format Changes: In NWB 2 epochs are stored via a *TimeIntervals* table (i.e., a *DynamicTable* for storing time intervals) that is stored in the group `/intervals/epochs`. Over the course of the development of NWB 2 the epoch storage has been refined in several phases:

- First, we create a new neurodata_type Epochs which was included in *NWBFile* as the group epochs. This simplified the extension of the epochs structure. `/epochs` at that point contained a compound (row-based) table with neurodata_type EpochTable that described the start/stop times, tags, and a region reference into the `TimeSeriesIndex` to identify the timeseries parts the epoch applies to. Note, the types Epochs, EpochTable and `TimeSeriesIndex` have been removed/superseded in subsequent changes. (See [PR396 \(PyNWB\)](#) and [I119 \(nwb-schema\)](#)).
- Later, an additional *DynamicTable* for storing dynamic metadata about epochs was then added to the Epochs neurodata_type to support storage of dynamic metadata about epochs without requiring users to create custom extensions (see [PR536 \(PyNWB\)](#)).
- Subsequently the epoch table was then fully converted to a *DynamicTable* (see [PR682 \(PyNWB\)](#) and [I664 \(PyNWB\)](#)).

- Finally, the EpochTable was then moved to `/intervals/epochs` and the EpochTable type was replaced by the more general type *TimeIntervals*. This also led to removal of the Epochs type (see [PR690 \(PyNWB\)](#) and [I683 \(PyNWB\)](#))

10.15.5.2 Improved support for trial-based data

Change: Add dedicated concept for storing trial data.

Reason: Users indicated that it was not easy to store trial data in NWB 1.x.

Format Changes: Added optional group `/intervals/trials/` which is a *DynamicTable* with `id`, `start_time`, and `stop_time` columns and optional additional user-defined table columns. See [PR536 on PyNWB](#) for detailed code changes. See the [PyNWB docs](#) for a short tutorial on how to use trials. See *NWBFile Groups: /trials* for an overview of the trial schema. **Note:** Originally trials was added a top-level group `trials` which was then later moved to `/intervals/trials` as part of the generalization of time interval storage as part of [PR690 \(PyNWB\)](#).

10.15.5.3 Generalized storage of time interval

Change: Create general type *TimeIntervals* (which is a generalization of the previous EpochTable type) and create top-level group `/intervals` for organizing time interval data.

Reason: Previously all time interval data was stored in either `epochs` or `trials`. To facilitate reuse and extensibility this has been generalized to enable users to create arbitrary types of intervals in addition to the predefined types, i.e., `epochs` or `trials`.

Format Changes: See [PR690 \(PyNWB\)](#) and [I683 \(PyNWB\)](#) for details:

- Renamed EpochTable type to the more general type *TimeIntervals* to facilitate reuse.
- Created top-level group `/intervals` for organizing time interval data.
 - Moved `/epochs` to `/intervals/epochs` and reused the *TimeIntervals* type
 - Moved `/trials` to `/intervals/trials` and reused the *TimeIntervals* type
 - Allow users to add arbitrary *TimeIntervals* tables to `/intervals`
 - Add optional *TimeIntervals* object named `invalid_times` in `"/intervals"`, which specifies time intervals that contain artifacts. See [I224 \(nwb-schema\)](#) and [PR731 \(PyNWB\)](#) for details.

10.15.6 Replaced Implicit Links/Data-Structures with Explicit Links

Change Replace implicit links with explicit soft-links to the corresponding HDF5 objects where possible, i.e., use explicit HDF5 mechanisms for expressing basic links between data rather than implicit ones that require users/developers to know how to use the specific data. In addition to links, NWB 2 adds support for object- and region references, enabling the creation of datasets (i.e., arrays) that store links to other data objects (groups or datasets) or regions (i.e., subsets) of datasets.

Reason: In several places datasets containing arrays of either 1) strings with object names, 2) strings with paths, or 3) integer indexes are used that implicitly point to other locations in the file. These forms of implicit links are not self-describing (e.g., the kind of linking, target location, implicit size and numbering assumptions are not easily identified). This hinders human interpretation of the data as well as programmatic resolution of these kind of links.

Format Changes:

- Text dataset `image_plane` of `<TwoPhotonSeries>` is now a link to the corresponding `<ImagingPlane>` (which is stored in `/general/optophysiology`)

- Text dataset `image_plane_name` of `<ImageSegmentation>` is now a link to the corresponding `<ImagingPlane>` (which is stored in `/general/optophysiology`). The dataset is also renamed to `image_plane` for consistency with `<TwoPhotonSeries>`
- Text dataset `electrode_name` of `<PatchClampSeries>` is now a link to the corresponding `<IntracellularElectrode>` (which is stored in `/general/intracellular_ephys`). The dataset is also renamed to `electrode` for consistency.
- Text dataset `site` in `<OptogeneticSeries>` is now a link to the corresponding `<StimulusSite>` (which is stored in `/general/optogenetics`).
- Integer dataset `electrode_idx` of `FeatureExtraction` is now a dataset `electrodes` of type `DynamicTableRegion` pointing to a region of the `ElectrodeTable` stored in `/general/extracellular_ephys/electrodes`.
- Integer array dataset `electrode_idx` of `<ElectricalSeries>` is now a dataset `electrodes` of type `DynamicTableRegion` pointing to a region of the `ElectrodeTable` stored in `/general/extracellular_ephys/electrodes`.
- Text dataset `/general/extracellular_ephys/<electrode_group_X>/device` is now a link `<ElectrodeGroup>/device`
- The Epochs, Unit, Trial and other dynamic tables in NWB 2 also support (and use) region and object references to explicitly reference other data (e.g., vector data as part of the unit tables).

10.15.7 Improved consistency, identifiability, and readability

10.15.7.1 Improved identifiability of objects

Change: All groups and datasets are now required to either have a unique name or a unique `neurodata_type` defined.

Reason: This greatly simplifies the unique identification of objects with variable names.

Format Changes: Defined missing `neurodata_types` for a number of objects, e.g.,:

- Group `/general/optophysiology/<imaging_plane_X>` now has the `neurodata_type` `ImagingPlane`
- Group `/general/intracellular_ephys/<electrode_X>` now has the `neurodata_type` `IntracellularElectrode`
- Group `/general/optogenetics/<site_X>` now has the `neurodata_type` `StimulusSite`
- ...

To enable identification of the type of objects, the `neurodata_type` is stored in HDF5 files as an attribute on the corresponding object (i.e., group or dataset). Also information about the namespace (e.g., the name and version) are stored as attributed to allow unique identification of the specification for storage objects.

10.15.7.2 Simplified extension of subject metadata

Specific Change: Assigned `neurodata_type` to `/general/subject` to enable extension of the subject container directly without having to extend `NWBFile` itself. (see <https://github.com/NeurodataWithoutBorders/nwb-schema/issues/120> and <https://github.com/NeurodataWithoutBorders/nwb-schema/pull/121> for details)

10.15.7.3 Reduce requirement for potentially empty groups

Change: Make several previously required fields optional

Reason: Reduce need for empty groups.

Format Changes: The following groups/datasets have been made optional:

- /epochs : not all experiments may require epochs.
- /general/optogenetics : not all experiments may use optogenetic data
- device in *IntracellularElectrode*
-

10.15.7.4 Added missing metadata

Change: Add a few missing metadata attributes/datasets.

Reason: Ease data interpretation, improve format consistency, and enable storage of additional metadata

Format Changes:

- /general/devices text dataset becomes group with neurodata type Device to enable storage of more complex and structured metadata about devices (rather than just a single string)
- Added attribute unit=Seconds to <EventDetection>/times dataset to explicitly describe time units and improve human and programmatic data interpretation
- Added filtering dataset to type <IntracellularElectrode> (i.e., /general/intracellular_ephys/<electrode_X>) to enable specification of per-electrode filtering data
- Added default values for <TimeSeries>/description and <TimeSeries>/comments

10.15.7.5 Improved Consistency

Change: Rename objects, add missing objects, and refine types

Reason: Improve consistency in the naming of data objects that store similar types of information in different places and ensure that the same kind of information is available.

Format Changes:

- Added missing help attribute for <BehavioralTimeSeries> to improve consistency with other types as well as human data interpretation
- Renamed dataset image_plan_name in <ImageSegmentation> to image_plane to ensure consistency in naming with <TwoPhotonSeries>
- Renamed dataset electrode_name in <PatchClampSeries> to electrode for consistency (and since the dataset is now a link, rather than a text name).
- Renamed dataset electrode_idx in <FeatureExtraction> to electrode_group for consistency (and since the dataset is now a link to the <ElectrodeGroup>)
- Renamed dataset electrode_idx in <ElectricalSeries> to electrode_group for consistency (and since the dataset is now a link to the <ElectrodeGroup>)
- Changed imaging_rate field in *ImagingPlane* from text to float. See [PR697 \(PyNWB\)](#) and [I136 \(nwb-schema\)](#) for details

10.15.7.6 Added keywords field

Change: Added keywords fields to `/general`

Reason: Data archive and search tools often rely on user-defined keywords to facilitate discovery. This enables users to specify keywords for a file. (see [PR620 \(PyNWB\)](#))

10.15.7.7 Removed source field

Change: Remove required attribute `source` from all `neurodata_types`

Reason: In NWB 1.0.x the attribute `source` was defined as a free text entry intended for storage of provenance information. In practice, however, this attribute was often either ignored, contained no useful information, and/or was misused to encode custom metadata (that should have been defined via extensions).

Specific Change: Removed attribute `source` from the core base `neurodata_types` which effects a large number of the types throughout the NWB schema. For further details see [PR695 \(PyNWB\)](#)

10.15.7.8 Removed ancestry field

Change: Removed the explicit specification of ancestry as an attribute as part of the format specification

Reason: 1) avoid redundant information as the ancestry is encoded in the inheritance of types, 2) ease maintenance, and 3) avoid possible inconsistencies between the ancestry attribute and the true ancestry (i.e., inheritance hierarchy) as defined by the spec.

Note The new specification API as part of PyNWB/HDMF makes the ancestry still easily accessible to users. As the ancestry can be easily extracted from the spec, we currently do not write a separate ancestry attribute but this could be easily added if needed. (see also [PR707 \(PyNWB\)](#), [I24 \(nwb-schema\)](#))

10.15.8 Improved organization of processed and acquisition data

10.15.8.1 Improved organization of processed data

Change: Relaxed requirements and renamed and refined core types used for storage of processed data.

Reason: Ease user intuition and provide greater flexibility for users.

Specific Changes: The following changes have been made to the organization of processed data:

- **Module has been renamed to *ProcessingModule* to avoid possible confusion** and to clarify its purpose. Also *ProcessingModule* may now contain any *NWBDataInterface*.
- With *NWBDataInterface* now being a general base class of *TimeSeries*, this means that it is now possible to define data processing types that directly inherit from *TimeSeries*, which was not possible in NWB 1.x.
- *Interface* has been renamed to *NWBDataInterface* to avoid confusion and ease intuition (see above)
- All *Interface* types in the original format had fixed names. The fixed names have been replaced by specification of corresponding default names. This change enables storage of multiple instances of the same analysis type in the same *ProcessingModule* by allowing users to customize the name of the data processing types, whereas in version 1.0.x only a single instance of each analysis could be stored in a *ProcessingModule* due to the requirement for fixed names.

10.15.8.2 Simplified organization of acquisition data

Specific Changes:

- /acquisition may now store any primary data defined via an *NWBDataInterface* type (not just TimeSeries).
- /acquisition/timeseries and /acquisition/images have been removed
- Created a new neurodata_type *Images* for storing a collection of images to replace acquisition/images and provide a more general container for use elsewhere in NWB (i.e., this is not meant to replace *ImageSeries*)

10.15.9 Other changes:

- **PR765** made the timestamps in *SpikeEventSeries* required

10.15.10 Improved governance and accessibility

Change: Updated release and documentation mechanisms for the NWB format specification

Reason: Improve governance, ease-of-use, extensibility, and accessibility of the NWB format specification

Specific Changes

- The NWB format specification is now released in separate Git repository
- Format specifications are released as YAML files (rather than via Python .py file included in the API)
- Organized core types into a set of smaller YAML files to ease overview and maintenance
- Converted all documentation documents to Sphinx reStructuredText to improve portability, maintainability, deployment, and public access
- Sphinx documentation for the format are auto-generated from the YAML sources to ensure consistency between the specification and documentation
- The PyNWB API now provides dedicated data structured to interact with NWB specifications, enabling users to programmatically access and generate format specifications

10.15.11 Specification language changes

Change: Numerous changes have been made to the specification language itself in NWB 2.0. Most changes to the specification language effect mainly how the format is specified, rather than the actual structure of the format. The changes that have implications on the format itself are described next. For an overview and discussion of the changes to the specification language see [specification language release notes](#).

10.15.11.1 Specification of dataset dimensions

Change: Updated the specification of the dimensions of dataset

Reason: To simplify the specification of dimension of datasets and attribute

Format Changes:

- The shape of various dataset is now specified explicitly for several datasets via the new `shape` key
- The `unit` for values in a dataset are specified via an attribute on the dataset itself rather than via `unit` definitions in structs that are available only in the specification itself but not the format.

- In some cases the length of a dimension was implicitly described by the length of structs describing the components of a dimension. This information is now explicitly described in the `shape` of a dataset.

10.15.11.2 Added Link type

Change Added new type for links

Reason:

- Links are usually a different type than datasets on the storage backend (e.g., HDF5)
- Make links more readily identifiable
- Avoid special type specification in datasets

Format Changes: The format itself is not affected by this change aside from the fact that datasets that were links are now explicitly declared as links.

10.15.11.3 Removed datasets defined via autogen

Change Support for `autogen` has been removed from the specification language. After review of all datasets that were produced via `autogen` it was decided that all `autogen` datasets should be removed from the format.

Reason The main reasons for removal of `autogen` dataset is to ease use and maintenance of NWB files by 1) avoiding redundant storage of information (i.e., improve normalization of data) and 2) avoiding dependencies between data (i.e., datasets having to be updated due to changes in other locations in a file).

Format Changes

- Datasets/Attributes that have been removed due to redundant storage of the path of links stored in the same group:
 - `IndexSeries/indexed_timeseries_path`
 - `RoiResponseSeries/segmentation_interface_path`
 - `ImageMaskSeries/masked_imageseries_path`
 - `ClusterWaveforms/clustering_interface_path`
 - `EventDetection/source_electricalseries_path`
 - `MotionCorrection/image_stack_name/original_path`
 - `NWBFile/epochs/epoch_X.links`
- Datasets//Attributes that have been removed because they stored only a list of groups/datasets (of a given type or property) in the current group.
 - `Module.interfaces` (now `ProcessingModule`)
 - `ImageSegmentation/image_plane/roi_list`
 - `UnitTimes/unit_list`
 - `TimeSeries.extern_fields`
 - `TimeSeries.data_link`
 - `TimeSeries.timestamp_link`
 - `TimeSeries.missing_fields`
- Other datasets/attributes that have been removed to ease use and maintenance because the data stored is redundant and can be easily extracted from the file:

- NWBFile/epochs.tags
- TimeSeries/num_samples
- Clustering/cluster_nums

10.15.11.4 Removed 'neurodata_type=Custom'

Change The 'neurodata_type=Custom' has been removed.

Reason All additions of data should be governed by extensions. Custom datasets can be identified based on the specification, i.e., any objects that are not part of the specification are custom.

10.16 1.0.x (09/2015 - 04/2017)

NWB 1.0.x has been deprecated. For documents relating to the 1.0.x schema please see https://github.com/NeurodataWithoutBorders/specification_nwb_1_0_x.

10.16.1 1.0.6, April 8, 2017

Minor fixes:

- Modify <IntervalSeries>/ documentation to use html entities for < and >.
- Fix indentation of unit attribute data_type, and conversion attribute description in /general/optophysiology/<imaging_plane_X>/manifold.
- Fix typos in <AnnotationSeries>/ conversion, resolution and unit attributes.
- Update documentation for IndexSeries to reflect more general usage.
- Change to all numerical version number to remove warning message when installing using setuptools.

10.16.2 1.0.5i_beta, Dec 6, 2016

Removed some comments. Modify author string in info section.

10.16.3 1.0.5h_beta, Nov 30, 2016

Add dimensions to /acquisition/images/<image_X>

10.16.4 1.0.5g_beta, Oct 7, 2016

- Replace group options: autogen: {"type": "create"} and "closed": True with "_properties": {"create": True} and "_properties": {"closed": True}. This was done to make the specification language more consistent by having these group properties specified in one place ("_properties" dictionary).

10.16.5 1.0.5f_beta, Oct 3, 2016

- Minor fixes to allow validation of schema using json-schema specification in file `meta-schema.py` using utility `check\_schema.py`.

10.16.6 1.0.5e_beta, Sept 22, 2016

- Moved definition of `<Module>` out of `/processing` group to allow creating subclasses of `Module`. This is useful for making custom `Module` types that specified required interfaces. Example of this is in `python-api/examples/create\_scripts/module-e.py` and the extension it uses (`extensions/e-module.py`).
- Fixed malformed html in `nwb\_core.py` documentation.
- Changed html generated by `doc\_tools.py` to html5 and fixed so passes validation at <https://validator.w3.org>.

10.16.7 1.0.5d_beta, Sept 6, 2016

- Changed `ImageSeries` `img_mask` dimensions to: `"dimensions": ["num\_y", "num\_x"]` to match description.

10.16.8 1.0.5c_beta, Aug 17, 2016

- Change `IndexSeries` to allow linking to any form of `TimeSeries`, not just an `ImageSeries`

10.16.9 1.0.5b_beta, Aug 16, 2016

- Make `'manifold'` and `'reference\_frame'` (under `/general/optophysiology`) recommended rather than required.
- In all cases, allow subclasses of a `TimeSeries` to fulfill validation requirements when an instance of `TimeSeries` is required.
- Change unit attributes in `VoltageClampSeries` series datasets from required to recommended.
- Remove `'const'=True` from `TimeSeries` attributes in `AnnotationSeries` and `IntervalSeries`.
- Allow the base `TimeSeries` class to store multi-dimensional arrays in `'data'`. A user is expected to describe the contents of `'data'` in the comments and/or description fields.

10.16.10 1.0.5a_beta, Aug 10, 2016

Expand class of `Ids` allowed in `TimeSeries` `missing\_fields` attribute to allow custom uses.

10.16.11 1.0.5_beta Aug 2016

- Allow subclasses to be used for merges instead of base class (specified by 'merge+' in format specification file).
- Use 'neurodata_type=Custom' to flag additions that are not describe by a schema.
- Exclude TimeSeries timestamps and starting time from under /stimulus/templates

10.16.12 1.0.4_beta June 2016

- Generate documentation directly from format specification file.”
- Change ImageSeries external_file to an array.
- Made TimeSeries description and comments recommended.

10.16.13 1.0.3 April, 2016

- Renamed "ISI_Retinotopy" to "ISIRetinotopy"
- Change ImageSeries external_file to an array. Added attribute starting_frame.
- Added IZeroClampSeries.

10.16.14 1.0.2 February, 2016

- Fixed documentation error, updating 'neurodata_version' to 'nwb_version'
- Created ISI_Retinotopy interface
- In ImageSegmentation module, moved pix_mask::weight attribute to be its own dataset, named pix_mask_weight. Attribute proved inadequate for storing sufficiently large array data for some segments
- Moved 'gain' field from Current/VoltageClampSeries to parent PatchClampSeries, due need of stimuli to sometimes store gain
- Added Ken Harris to the Acknowledgements section

10.16.15 1.0.1 October 7th, 2015

- Added 'required' field to tables in the documentation, to indicate if group/dataset/attribute is required, standard or optional
- Obsoleted 'file_create_date' attribute 'modification_time' and made file_create_date a text array
- Removed 'resistance_compensation' from CurrentClampSeries due being duplicate of another field
- Upgraded TwoPhotonSeries::imaging_plane to be a required value
- Removed 'tags' attribute to group 'epochs' as it was fully redundant with the 'epoch/tags' dataset
- Added text to the documentation stating that specified sizes for integer values are recommended sizes, while sizes for floats are minimum sizes
- Added text to the documentation stating that, if the TimeSeries::data::resolution attribute value is unknown then store a NaN
- Declaring the following groups as required (this was implicit before)

```
acquisition/  
  
  \_ images/  
  \_ timeseries/  
  
analysis/  
  
epochs/  
  
general/  
  
processing/  
  
stimulus/  
  
  \_ presentation/  
  
  \_ templates/
```

This is to ensure consistency between .nwb files, to provide a minimum expected structure, and to avoid confusion by having someone expect time series to be in places they're not. I.e., if 'acquisition/timeseries' is not present, someone might reasonably expect that acquisition time series might reside in 'acquisition/'. It is also a subtle reminder about what the file is designed to store, a sort of built-in documentation. Subfolders in 'general/' are only to be included as needed. Scanning 'general/' should provide the user a quick idea what the experiment is about, so only domain-relevant subfolders should be present (e.g., 'optogenetics' and 'optophysiology'). There should always be a 'general/devices', but it doesn't seem worth making it mandatory without making all subfolders mandatory here.

10.16.16 1.0.0 September 28th, 2015

- Convert document to .html
- TwoPhotonSeries::imaging_plane was upgraded to mandatory to help enforce inclusion of important metadata in the file.

11.1 Acknowledgments

For details on the partners, members, and supporters of NWB please see the <http://www.nwb.org/> project website. For specific contributions to the format specification and this document see the change logs of the Git repository at <https://github.com/NeurodataWithoutBorders/nwb-schema>.

11.2 Authors

11.2.1 NWB: Version 2.0.0 and later

Documentation for Version 2 of the NWB format and later have been created by Oliver Ruebel and Andrew Tritt et al. in collaboration with the NWB community.

11.2.2 NWB: Version 1.0.x and earlier

Version 1.0.5g (and earlier) of the NWB file format were created by Jeff Teeters et al. as part of the first NWB pilot project. The documents for NWB 2 have been adopted from the final version of format docs released by the original NWB pilot project.

12.1 Copyright

“nwb-schema” Copyright (c) 2017-2024, The Regents of the University of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from the U.S. Dept. of Energy). All rights reserved.

If you have questions about your rights to use or distribute this software, please contact Berkeley Lab’s Innovation & Partnerships Office at IPO@lbl.gov.

NOTICE. This Software was developed under funding from the U.S. Department of Energy and the U.S. Government consequently retains certain rights. As such, the U.S. Government has been granted for itself and others acting on its behalf a paid-up, nonexclusive, irrevocable, worldwide license in the Software to reproduce, distribute copies to the public, prepare derivative works, and perform publicly and display publicly, and to permit other to do so.

12.2 Licence

“nwb-schema” Copyright (c) 2017-2024, The Regents of the University of California, through Lawrence Berkeley National Laboratory (subject to receipt of any required approvals from the U.S. Dept. of Energy). All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- (1) Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- (2) Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- (3) Neither the name of the University of California, Lawrence Berkeley National Laboratory, U.S. Dept. of Energy nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS “AS IS” AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

You are under no obligation whatsoever to provide any bug fixes, patches, or upgrades to the features, functionality or performance of the source code (“Enhancements”) to anyone; however, if you choose to make your Enhancements available either publicly, or directly to Lawrence Berkeley National Laboratory, without imposing a separate written license agreement for such Enhancements, then you hereby grant the following license: a non-exclusive, royalty-free perpetual license to install, use, modify, prepare derivative works, incorporate into other computer software, distribute, and sublicense such enhancements or derivative works thereof, in binary and source code form.