ASDF

Adaptable Seismic Data Format

ASDF Definition

Release 1.0.2

Lion Krischer, James Smith, Jeroen Tromp

Mar 26, 2018

Contents

ASDF

Adaptable Seismic Data Format

This is the formal definition of the *Adaptable Seismic Data Format* (**ASDF**).

Note: For more details please see our paper:

Lion Krischer, James Smith, Wenjie Lei, Matthieu Lefebvre, Youyi Ruan, Elliott Sales de Andrade, Norbert Podhorszki, Ebru Bozdağ and Jeroen Tromp (2016), **An Adaptable Seismic Data Format**, Geophysical Journal International, 207(2). http://dx.doi.org/10.1093/gji/ggw319

Note: This document is for version **1.0.2** of the format.

This is the **A**daptable **S**eismic **D**ata **F**ormat - if you are looking for the **A**dvanced **S**cientific **D**ata **F**ormat, go here: https://asdf.readthedocs.io/en/latest/

For further information and contact information please see these two web sites:

- Landing page of the ASDF data format: http://seismic-data.org
- Github repository of this document http://www.github.com/SeismicData/ASDF_definition

Additionally these pages and software projects are of further interest:

- C/Fortran implementation intended to be integrated into numerical solvers: https://github.com/SeismicData/asdf-library
- Python implementation: http://seismicdata.github.io/pyasdf/
- ASDF validation tool: https://github.com/SeismicData/asdf_validate
- Graphical user interface for ASDF: https://github.com/SeismicData/asdf_sextant

- SEIS-PROV: http://seismicdata.github.io/SEIS-PROV/

CHAPTER 1

ASDF Format Changelog

Version 1.0.2 (March 1st, 2018)

- Allow adding very short waveforms that might start and end within the same second.

Version 1.0.1 (Oktober 19th, 2017)

- Allow little and big endian 16 bit integer waveform data.

Version 1.0.0 (March 22nd, 2016)

- Initial ASDF version.

1.1 Introduction

We submitted a paper containing a large section about the motivation for this new format and why we think we really need this. Once published we will put a link to it here.

1.1.1 Why introduce a new seismic data format?

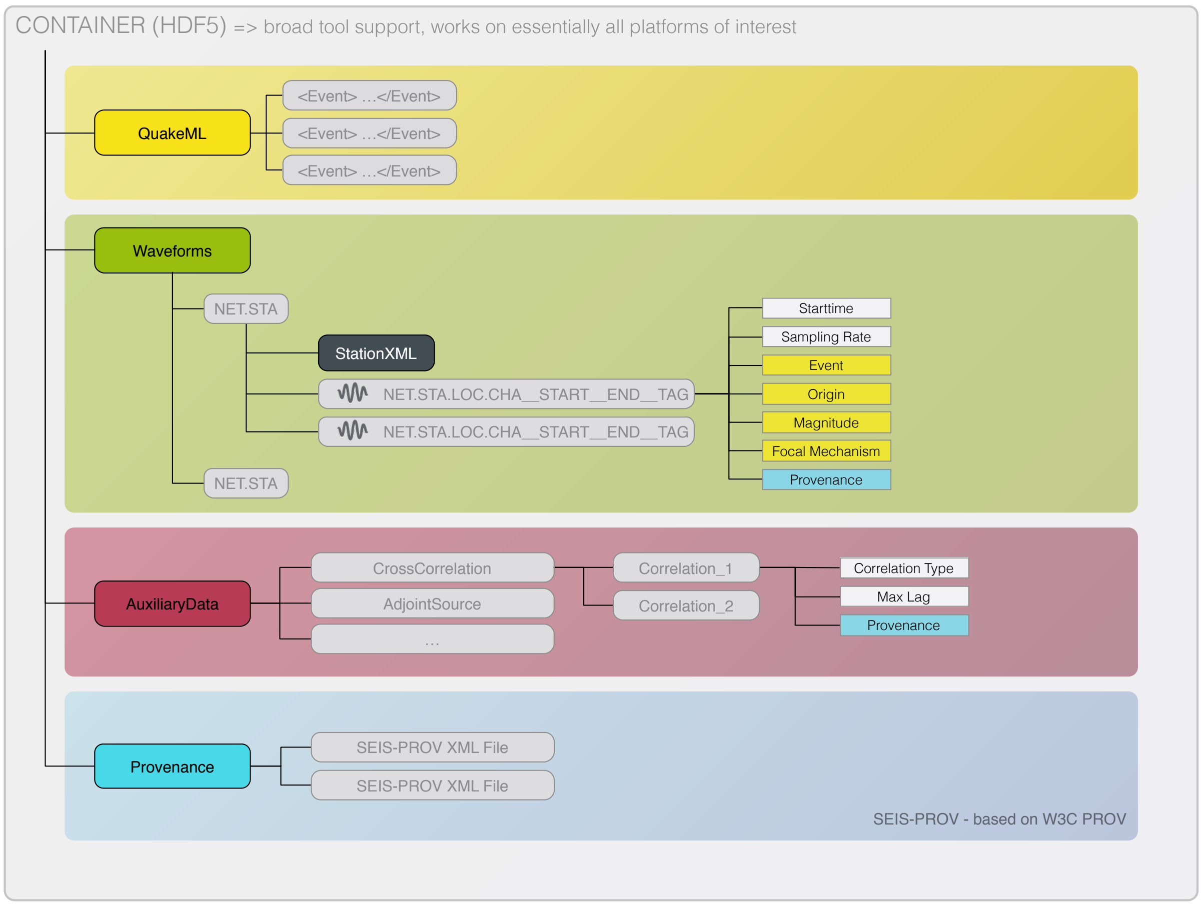
1. The amount of seismic data available for analysis worldwide is rapidly growing. Seismic arrays, such as USArray and ChinaArray, give access to datasets on the terabyte scale that are not suited for existing seismic data formats.
2. Disk space is rapidly growing and data organization should improve such that the different types of seismic data (waveforms, receivers, earthquakes, adjoint sources, cross correlations, etc.) can be easily exchanged among the community under one container.
3. Modern workflows in seismology use supercomputers and the number of files is an I/O bottleneck. The performance of these workflows would be increased if the data was stored by combining all time series into one file and taking advantage of parallel processing capabilities.

4. New methods, such as ambient-noise seismology, should not be limited by data formats that were developed for other applications in seismology. In addition, seismologists often ignore standards because adherence increases development time. An adaptable seismic data format with an open, modular design will be able to evolve and handle future advances in seismology.
5. Reproducibility is a goal in science and seismology has yet to develop a standardized way of storing provenance in the current seismic data formats. We introduce a format that contains flexible provenance that lets the user know where the data comes from and what has been done to it.

1.2 Big Picture

ASDF stores everything in an HDF5 file with a layout that is illustrated by the following picture. It contains four parts, all of which are elaborated upon in another section:

1. Earthquake information (yellow) is stored in the form of a single QuakeML file. It can store an arbitrary number of seismic events of various types.
2. Waveform information is stored in the green part at a per station granularity. This section also contains the station meta information in form of StationXML files. Please also note the possible associations of waveforms with parts of a seismic event and the possible provenance links.
3. Anything not a waveform is stored in the red section. Arbitrary nesting is possible alongside the capability to store provenance information.
4. The blue section finally contains a collection of provenance documents that can be referred to from other parts of the ASDF file.



1.3 The Container

The chosen container format for ASDF is HDF5. The Hierarchical Data Format version 5 is a quasi standard format for binary data with a large amount of tools and support available. This format has also shown promise for dealing with large datasets in other applications.

ASDF delegates the allowed compression, chunking, checksumming, etc. settings to the HDF5 definition. Best to not use use custom filters that are not shipping with the HDF5 library as it will severely hurt exchangeability and possibly future safety. The HDF5 group puts a lot of emphasis on backwards compatibility.

1.4 Header

This section defines all elements that the root group (`/`) of a proper ASDF file should contain.

1.4.1 Attributes

The root group contains two attributes that serve to mark the format and the version of the format for a particular file.

`file_format` Attribute	
Type	Attribute
Name	`file_format`
Description	The file format name
Required	True
Value	`"ASDF"`
Details	

```
ATTRIBUTE "file_format" {
    DATATYPE  H5T_STRING {
        STRPAD H5T_STR_NULLPAD;
        CSET H5T_CSET_ASCII;
        CTYPE H5T_C_S1;
    }
    DATASPACE  SCALAR;
}
```

`file_format_version` Attribute	
Type	Attribute
Name	`file_format_version`
Description	The file format version.
Required	True
Value	`"1.0.2"`
Details	

```
ATTRIBUTE "file_format_version" {
    DATATYPE  H5T_STRING {
        STRPAD H5T_STR_NULLPAD;
        CSET H5T_CSET_ASCII;
        CTYPE H5T_C_S1;
    }
    DATASPACE  SCALAR;
}
```

1.4.2 Data Sets

It can optionally contain a data set representing a QuakeML file.

Data Set - `/QuakeML`

See *Events* for further information

1.4.3 Groups

Waveform, auxiliary, and provenance data is stored into three sub groups.

Group - `/Waveforms`

See *Waveform Data* for further information.

Group - `/AuxiliaryData`

See *Auxiliary Data* for further information.

Group - `/Provenance`

See *Provenance* for further information.

1.5 Events

Event information is stored as a QuakeML file in the `/QuakeML` data set. It contains a binary dump (determining the encoding is left to the XML header which is quite good at it) of a QuakeML file.

It is recommended to always use the most wide-spread QuakeML version but ASDF will work fine as long as the QuakeML version has the concept of resource identifiers and public IDs.

`/QuakeML` Data Set	
Type	Data Set
Path	`/QuakeML`
Description	Event information as a single QuakeML file.
Data Type	`H5T_STD_I8LE`
Required	False
Details	`DATASET "QuakeML" { DATATYPE H5T_STD_I8LE DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) } }`

```
DATASET "QuakeML" {
    DATATYPE  H5T_STD_I8LE
    DATASPACE  SIMPLE {
      ( * ) / ( H5S_UNLIMITED )
    }
}
```

1.6 Waveform Data

Waveform data and station meta information is organized by utilizing the network, station, location, and channel codes described in the SEED Manual. They are widely employed in the seismological community and serve to uniquely identify a recording instrument.

ASDF organizes all data that can be considered a seismic waveform (an actual seismic recording, barometric or temperature data, ...) under the /Waveforms group. Data within that group is organized per seismic station, i.e. a unique combination of network and station code.

/Waveforms Group	
Type	Group
Path	/Waveforms
Required	False
Description	Contains waveforms and station meta information.

1.6.1 Station Groups

Data under the /Waveforms group is organized per station in a separate group, named {NET}.{STA} where {NET} and {STA} are placeholders for the network and station codes as defined by the SEED standard. The network code can have 1 or 2 letter, the station code between 1 and 5.

/Waveforms/{NET}.{STA} Group	
Type	Group
Full Path	/Waveforms/{NET}.{STA}
RegEx	^[A-Z0-9]{1,2}\\.[A-Z0-9]{1,5}$
Required	False
Description	Contains waveforms and station meta information for a single station.

StationXML

Each station can have a single FDSN StationXML file associated with it. ASDF does not care about the version of the StationXML file but the file must not contain information about other channels. The file are stored as a binary dump in a data set called StationXML. Thus dealing with the encoding is delegated to whatever XML library reads the binary dump.

/Waveforms/{NET}.{STA}.StationXML Data Set	
Type	Data Set
Path	/Waveforms/{NET}.{STA}.StationXML
Description	Station information as a single StationXML file for a particular station.
Data Type	H5T_STD_I8LE
Required	False
Details	`DATASET "StationXML" { DATATYPE H5T_STD_I8LE DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) } }`

```
DATASET "StationXML" {
    DATATYPE  H5T_STD_I8LE
    DATASPACE  SIMPLE {
         ( * ) / ( H5S_UNLIMITED )
    }
}
```

Waveform Traces

The actual waveform data is stored on a per trace basis. A trace is a continuous nice behaving time series from a single channel with a regular sampling interval. It has a defined start time and a sampling rate and no gaps and overlaps. Gaps and overlaps are then represented by having multiple traces. Each station group can have an arbitrary number of traces and each trace can be arbitrarily long.

Allowed data types are half, single, and double precision two's complement signed integers and single and double precision IEEE 768 floating point numbers which happen to be the native data types on most platforms. HDF5 transparently deals with byte order issues so choose whichever endianness is most suitable for your platform.

Data sets are named according to the scheme

`{NET}.{STA}/{NET}.{STA}.{LOC}.{CHA}__{ST}__{ET}__{TAG}`

with the following placeholders:

- `{NET}`: The network code.
- `{STA}`: The station code.
- `{LOC}`: The location code.
- `{CHA}`: The channel code.
- `{ST}`: The approximate start time in UTC as an ISO 8601 date time string. Serves only to generate a suitable data set name. The actual start time is specified as an attribute.
- `{ET}`: The approximate end time in UTC as an ISO 8601 date time string. Serves only to generate a suitable data set name.
- `{TAG}`: The hierarchical tag. Please read *Of Tags and Labels* for further explanations.

A full example is

`CN.FRB..BHZ__1998-09-01T10:24:49__1998-09-01T12:09:49__tag`

and the exact pattern is specificed as a regular expression in the following table.

`/Waveforms/{NET}.{STA}/{NET}.{STA}.{LOC}.{CHA}__{ST}__{ET}__{TAG}` Data Set	
Type	Data Set
Full Path	`/Waveforms/{NET}.{STA}/{NET}.{STA}.{LOC}.{CHA}__{ST}__{ET}__{TAG}`
RegEx	`^[A-Z0-9]{1,2}\\.[A-Z0-9]{1,5}\\.[A-Z0-9]{0,2}\\.[A-Z0-9]{3}__(18|19|20|21)\\d{2}-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[01])T([0-1][0-9]|2[0-4]):([0-5]\\d|60):[0-5]\\d(\\.\\d{9})?__(18|19|20|21)\\d{2}-(0[1-9]|1[012])-(0[1-9]|[12][0-9]|3[01])T([0-1][0-9]|2[0-4]):([0-5]\\d|60):[0-5]\\d(\\.\\d{9})?__[A-Za-z_0-9]+$`
Description	Waveform data for a single trace.
Data Type	`H5T_IEEE_F32LE`, `H5T_IEEE_F64LE`, `H5T_IEEE_F32BE`, `H5T_IEEE_F64BE`, `H5T_STD_I32LE`, `H5T_STD_I64LE`, `H5T_STD_I32BE`, `H5T_STD_I64BE`, `H5T_STD_I16LE`, `H5T_STD_I16BE`
Required	False

Details	`DATASET`
	`    "NET.STA.LOC.CHA__ST__ET__TAG" {`
	`      DATATYPE  H5T_IEEE_F32LE`
	`      DATASPACE  SIMPLE {`
	`          ( * ) / ( H5S_UNLIMITED )`
	`      }`
	`}`

Waveform Trace Attributes

The attributes of each trace are described here. Only two are mandatory, the time of the first sample and the sampling interval.

`sampling_rate` Attribute	
Type	Attribute
Name	`sampling_rate`
Description	The sampling rate of the the waveform trace in `Hz`. Must naturally be positive and non-zero.
Data Type	`H5T_IEEE_F64LE`, `H5T_IEEE_F64BE`
Required	True

Details	`ATTRIBUTE "sampling_rate" {`
	`   DATATYPE  H5T_IEEE_F64LE`
	`   DATASPACE  SCALAR`
	`}`

`starttime` Attribute	
Type	Attribute
Name	`starttime`
Description	The time of the first sample as a UNIX epoch time in nanoseconds in UTC. It provides an approximate temporal range from the year 1680 to 2260 which is plenty for all envisioned applications.
Data Type	`H5T_STD_I64LE`, `H5T_STD_I64BE`
Required	True

Details	`ATTRIBUTE "starttime" {`
	`   DATATYPE  H5T_STD_I64LE`
	`   DATASPACE  SCALAR`
	`}`

Provenance for that trace can be stored as an identifier to a certain provenance record which represents that particular trace. It is possible (and recommended) but not necessary that a provenance document in the *Provenance* contains a record with that id.

`provenance_id` Attribute	
Type	Attribute
Name	`provenance_id`
Description	The id of a provenance record representing the current state of the waveform trace.
Required	False

Details	`ATTRIBUTE "provenance_id" {`
	`    DATATYPE  H5T_STRING {`
	`        STRPAD H5T_STR_NULLPAD;`
	`        CSET H5T_CSET_ASCII;`
	`        CTYPE H5T_C_S1;`
	`    }`
	`    DATASPACE  SCALAR;`
	`}`

Next are four optional identifiers that refer to different elements within a QuakeML file and enable the association of a waveform trace with an event or a specific origin, magnitude, or focal mechanism. The later three are mainly of interest for synthetic data where these three are exactly known.

`event_id` Attribute	
Type	Attribute
Name	`event_id`
Description	The id of the event associated with that waveform. Can contain several comma-separated ids.
Required	False
Details	`ATTRIBUTE "event_id" { DATATYPE H5T_STRING { STRPAD H5T_STR_NULLPAD; CSET H5T_CSET_ASCII; CTYPE H5T_C_S1; } DATASPACE SCALAR; }`

```
ATTRIBUTE "event_id" {
    DATATYPE  H5T_STRING {
        STRPAD H5T_STR_NULLPAD;
        CSET H5T_CSET_ASCII;
        CTYPE H5T_C_S1;
    }
    DATASPACE  SCALAR;
}
```

`origin_id` Attribute	
Type	Attribute
Name	`origin_id`
Description	The id of the orgin associated with that waveform. Can contain several comma-separated ids.
Required	False
Details	`ATTRIBUTE "origin_id" { DATATYPE H5T_STRING { STRPAD H5T_STR_NULLPAD; CSET H5T_CSET_ASCII; CTYPE H5T_C_S1; } DATASPACE SCALAR; }`

```
ATTRIBUTE "origin_id" {
    DATATYPE  H5T_STRING {
        STRPAD H5T_STR_NULLPAD;
        CSET H5T_CSET_ASCII;
        CTYPE H5T_C_S1;
    }
    DATASPACE  SCALAR;
}
```

`magnitude_id` Attribute	
Type	Attribute
Name	`magnitude_id`
Description	The id of the magnitude associated with that waveform. Can contain several comma-separated ids.
Required	False
Details	`ATTRIBUTE "magnitude_id" { DATATYPE H5T_STRING { STRPAD H5T_STR_NULLPAD; CSET H5T_CSET_ASCII; CTYPE H5T_C_S1; } DATASPACE SCALAR; }`

```
ATTRIBUTE "magnitude_id" {
    DATATYPE  H5T_STRING {
        STRPAD H5T_STR_NULLPAD;
        CSET H5T_CSET_ASCII;
        CTYPE H5T_C_S1;
    }
    DATASPACE  SCALAR;
}
```

`focal_mechanism_id` Attribute	
Type	Attribute
Name	`focal_mechanism_id`
Description	The id of the focal mechanism associated with that waveform. Can contain several comma-separated ids.
Required	False

Details	
	```
	ATTRIBUTE "focal_mechanism_id" {
	    DATATYPE  H5T_STRING {
	        STRPAD H5T_STR_NULLPAD;
	        CSET H5T_CSET_ASCII;
	        CTYPE H5T_C_S1;
	    }
	    DATASPACE  SCALAR;
	}
	```

Last but not least each waveform trace can also have any number of labels associated with it. Please note that these are different from tags, see *Of Tags and Labels* for details. The labels are stored as comma separated UTF-8 strings so the two labels `label 1`, and `äöü` would be stored as `"label 1, äöü"`.

`labels` Attribute	
Type	Attribute
Name	`labels`
Description	The labels of this waveform as a comma-separated UTF-8 string.
Required	False

Details	
	```
	ATTRIBUTE "labels" {
	   DATATYPE  H5T_STRING {
	      STRSIZE H5T_VARIABLE;
	      STRPAD H5T_STR_NULLTERM;
	      CSET H5T_CSET_UTF8;
	      CTYPE H5T_C_S1;
	   }
	   DATASPACE  SCALAR
	}
	```

1.7 Auxiliary Data

Anything that is not a seismic waveform will be stored here. Conceptually this group stores any data array (arbitrary data type and number of dimensions) with associated meta information in an arbitrarily nested path.

`/AuxiliaryData` Group	
Type	Group
Full Path	`/AuxiliaryData`
Required	False
Description	Any data that cannot be considered a waveform will be stored here.

The path is intended to group the data by data type and whatever grouping makes sense for the data at hand. For cross correlations this might be `/CrossCorrelations/Station_A/Station_B` and for adjoint sources `/AdjointSources/Event_A`. This is, by design, not fixed within the `ASDF` format as we are not experts in every field and it would take a long time to come up with the definitions. Additionally many areas are still actively researched so flexibility to store any data and meta data in whatever grouping must be retained. The minimum nesting is 1 layer, e.g. one cannot directly attach data to the `/AuxiliaryData` group.

Each path segment is just an HDF5 group whose name has to match a certain pattern. Keep in mind that you can nest any number of these.

`/AuxiliaryData/{...}/{...}` Group	
Type	Group
Full Path	`/AuxiliaryData/{...}/{...}`
RegEx	`^[A-Z][A-Za-z0-9_]*[a-zA-Z0-9]$`
Required	False
Description	A group storing further auxiliary data groups and/or actual auxiliary data.

Data itsself is stored in a data set within one of these groups. The name of data set can be freely chosen but has to match the regular expression in the following table.

`/AuxiliaryData/{...}/{...}/{TAG}` Data Set	
Type	Data Set
Full Path	`AuxiliaryData/{...}/{...}/{TAG}`
RegEx	`^[a-zA-Z0-9][a-zA-Z0-9_]*[a-zA-Z0-9]$`
Description	Auxiliary Data.
Data Type	Any data type and dimension. For interoperability best keep to data types supported by HDF5.
Required	False

Meta information is stored as attributes on the data sets and they are once again completely free form and highly dependent on the application. The only reserved attribute is the `provenance_id` attribute which works exactly as in the waveform traces.

Provenance for a particular piece of auxiliary data can be stored as an identifier to a certain provenance record which represents that piece of data. It is possible (and recommended) but not necessary that a provenance document in the *Provenance* contains a record with that id.

`provenance_id` Attribute	
Type	Attribute
Name	`provenance_id`
Description	The id of a provenance record representing the current state of the auxiliary data piece.
Required	False
Details	`ATTRIBUTE "provenance_id" { DATATYPE H5T_STRING { STRPAD H5T_STR_NULLPAD; CSET H5T_CSET_ASCII; CTYPE H5T_C_S1; } DATASPACE SCALAR; }`

```
ATTRIBUTE "provenance_id" {
    DATATYPE  H5T_STRING {
        STRPAD H5T_STR_NULLPAD;
        CSET H5T_CSET_ASCII;
        CTYPE H5T_C_S1;
    }
    DATASPACE  SCALAR;
}
```

1.8 Provenance

The `/Provenance` group contains any number of data sets, each containing a SEIS-PROV document.

`/Provenance` Group	
Type	Group
Full Path	`/Provenance`
Required	False
Description	Contains a number of `SEIS-PROV` documents.

Each SEIS-PROV document is serialized with PROV-XML and stored as a binary dump to a data set. Thus dealing with the encoding is once again delegated to the XML libraries. The name of the data set is arbitrary but has to match the pattern in the following table.

/Provenance/{NAME} Data Set	
Type	Data Set
Full Path	/Provenance/{NAME}
RegEx	^[0-9a-z][0-9a-z_]*[0-9a-z]$
Description	A SEIS-PROV document with any number of records.
Data Type	H5T_STD_I8LE
Required	False
Details	`DATASET "373fe_9bca_43ed10b" { DATATYPE H5T_STD_I8LE DATASPACE SIMPLE { ( * ) / ( H5S_UNLIMITED ) } }`

```
DATASET "373fe_9bca_43ed10b" {
    DATATYPE  H5T_STD_I8LE
    DATASPACE  SIMPLE {
        ( * ) / ( H5S_UNLIMITED )
    }
}
```

1.9 Potential Shortcomings

This section aims to point out several shortcomings of the ASDF format and potential ways to deal with them where applicable.

1.9.1 Irregularly sampled data

The ASDF format in the initial definition can not deal with this and neither can most signal processing tools in use in seismology. If this ever become a serious issue, the format definition will have to be extended. One possibility would be to use 2D arrays for irregularly sampled components; one dimension denoting time, the other the data.

1.9.2 Finite Sources

This is mainly a limitation of the QuakeML format and thus should be dealt with therein. Currently this could be worked around by either specifying a finite source as a large number of point sources in a QuakeML file or by storing a more appropriate representation of finite sources in the auxiliary data section of ASDF.

1.9.3 Source Time Functions

This is mainly a limitation of the QuakeML format and thus should be dealt with therein. As of now this can be worked around by storing the source time functions in the auxiliary data section.

1.10 Of Tags and Labels

The ASDF data format has tags and labels which, while similar to a certain extent, serve a different purpose. This is sometimes a bit confusing to people new to the format - this page explains and clarifies everything.

1.10.1 Tags

Tags are used as an additional hierarchical layer. They are for example used to distinguish observed and synthetic data or two synthetic waveforms calculated with slightly different earth models. Each waveform trace must have a tag - it is used as part of the arrays' names in the HDF5 file. There are little rules to them but they should be pretty short.

The `raw_recording` tag is by convention reserved for raw data counts straight from a digitizer.

Other names depend on the use case - common choices are `synthetic_prem` or `processed_1_10_s`.

1.10.2 Labels

Labels on the other hand are an optional list of words potentially assigned to a waveform. They can be used to describe and label similar waveforms without influencing how they are stored on disc. **They are a piece of meta information** useful to organize the data a bit better. Its always a list of simple UTF-8 encoded words.