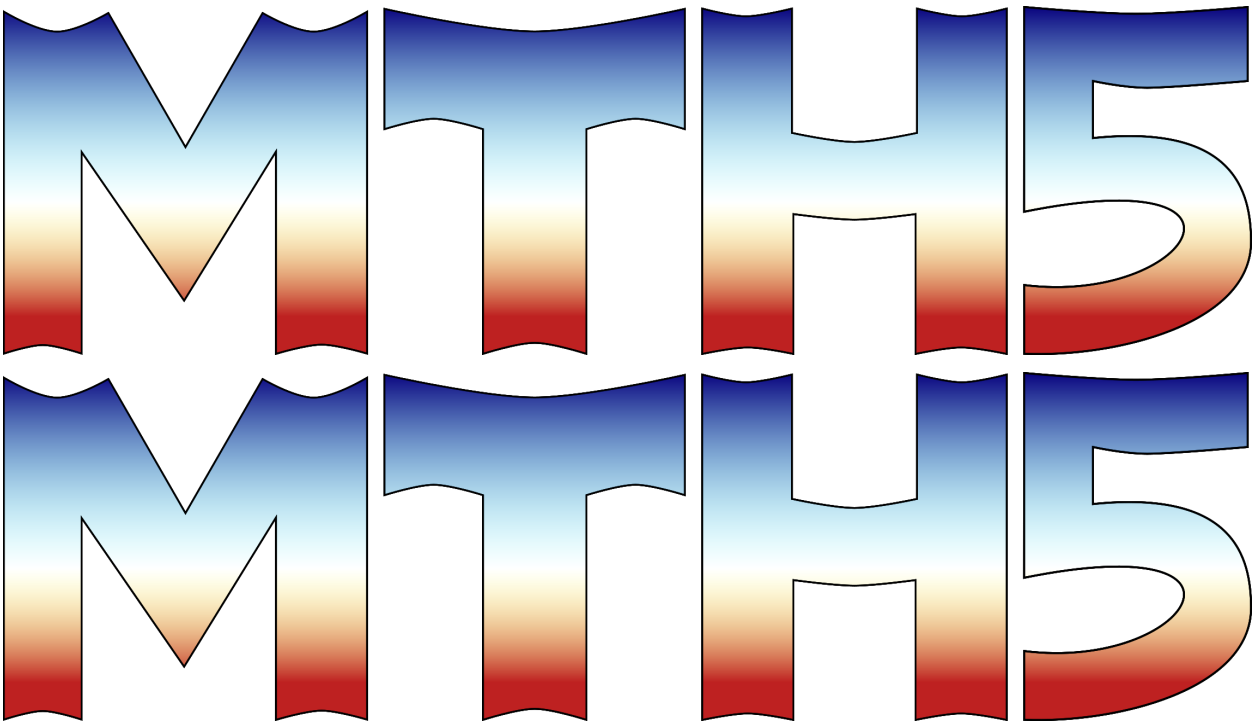
MTH5

Release 0.0.1

Jared Peacock

Jul 30, 2020

OVERVIEW



The goal of **MTH5** is to develop a standard format and tools for archiving magnetotelluric (MT) time series data.

The preferred format is HDF5 and has been adopted to conform to MT data, something that has been needed in the EM community for some time. The module mth5 contains reading/writing capabilities and will contain tools for retrieving data in useful ways to work with processing codes.

The metadata follows the standards proposed by the IRIS-PASSCAL MT Software working group and documented in MT Metadata Standards.

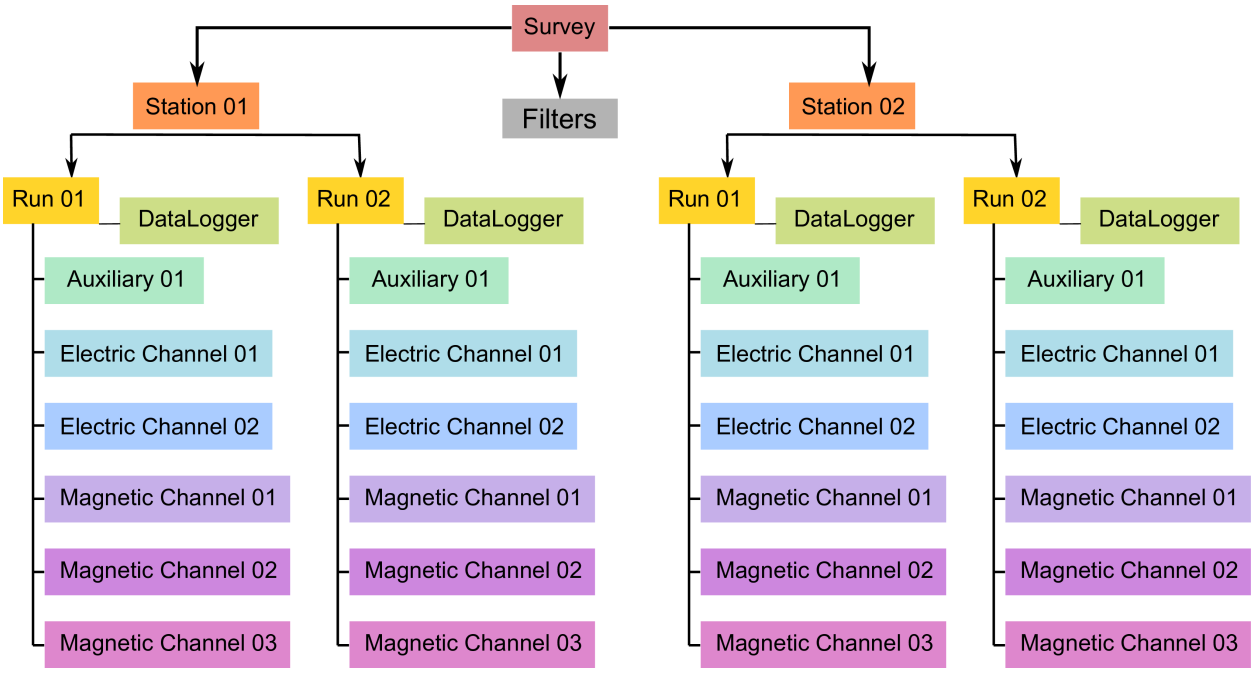
Note: This is a work in progress. Feel free to comment or send me a message at jpeacock@usgs.gov on the data format.

CHAPTER

ONE

MTH5 FORMAT

- The basic format of MTH5 is illustrated below, where metadata is attached at each level.



CHAPTER

TWO

BASICS

MTH5 is written to make read/writing an *.mth5* file easier.

Hint: MTH5 is comprehensively logged, therefore if any problems arise you can always check the mth5_debug.log and the mth5_error.log, which will be written to your current working directory.

Each MTH5 file has default groups. A 'group' is basically like a folder that can contain other groups or datasets. These are:

- **Survey** –> The master or root group of the HDF5 file
- **Filters** –> Holds all filters and filter information
- **Reports** –> Holds any reports relevant to the survey
- **Standards** –> A summary of metadata standards used
- **Stations** –> Holds all the stations an subsequent data

Each group also has a summary table to make it easier to search and access different parts of the file. Each entry in the table will have an HDF5 reference that can be directly used to get the appropriate group or dataset without using the path.

2.1 Opening and Closing Files

To open a new *.mth5* file:

```
>>> from mth5 import mth5
>>> mth5_obj = mth5.MTH5()
>>> mth5_obj.open(r"path/to/file.mth5", mode="w")
```

To open an exiting *.mth5* file:

```
>>> from mth5 import mth5
>>> mth5_obj = mth5.MTH5()
>>> mth5_obj.open(r"path/to/file.mth5", mode="a")
```

Note: If 'w' is used for the mode, it will overwrite any file of the same name, so be careful you don't overwrite any files. Using 'a' for the mode is safer as this will open an existing file of the same name and will give you write privilages.

To close a file:

```
>>> mth5_obj.close_mth5()
2020-06-26T15:01:05 - mth5.mth5.MTH5.close_mth5 - INFO - Flushed and
closed example_02.mth5
```

Note: Once a MTH5 file is closed any data contained within cannot be accessed. All groups are weakly referenced, therefore once the file closes the group can no longer access the HDF5 group and you will get a similar message as below. This is to remove any lingering references to the HDF5 file which will be important for parallel computing.

```
>>> 2020-06-26T15:21:47 - mth5.groups.Station.__str__ - WARNING - MTH5 file is closed␣
↪and cannot be accessed. MTH5 file is closed and cannot be accessed.
```

A MTH5 object is represented by the file structure and can be displayed at anytime from the command line.

```
>>> mth5_obj
/:
====================
        |- Group: Survey
        ----------------
                |- Group: Filters
                -----------------
                        --> Dataset: Summary
                        ......................
                |- Group: Reports
                -----------------
                        --> Dataset: Summary
                        ......................
                |- Group: Standards
                -------------------
                        --> Dataset: Summary
                        ......................
                |- Group: Stations
                ------------------
                        |- Group: MT001
```

(continues on next page)

(continued from previous page)

```
                        ---------------
                                --> Dataset: Summary
                                ....................
                        --> Dataset: Summary
                        ....................
```

This file does not contain a lot of stations, but this can get verbose if there are a lot of stations and filters. If you want to check what stations are in the current file.

```
>>> mth5_obj.station_list
['Summary', 'MT001']
```

Each group has a property attribute with an appropriate container including convenience methods. Each group has a property attribute called *group_list* that lists all groups the next level down.

See also:

`mth5.groups` and `mth5.metadata` for more information.

2.2 Metadata

Each group object has a container called *metadata* that holds the appropriate metadata (`mth5.metadata`) data according to the standards defined at MT Metadata Standards. The exceptions are the HDF5 file object which has metadata that describes the file type and is not part of the standards, and the stations_group, which is just a container to hold a collection of stations.

Input metadata will be validated against the standards and if it does not conform will throw an error.

The basic Python type used to store metadata is a dictionary, but there are three ways to input/output the metadata, dictionary, JSON, and XML. Many people have their own way of storing metadata so this should accommodate most everyone. If you store your metadata as JSON or XML you will need to read in the file first and input the appropriate element to the metadata.

2.2.1 Setting Attributes

Metadata can be input either manually by setting the appropriate attribute:

```
>>> existing_station = mth5_obj.get_station('MT001')
>>> existing_station.metadata.archive_id = 'MT010'
```

Hint: Currently, if you change any *metadata* attribute you will need to mannually update the attribute in the HDF5 group:

```
>>> existing_station.write_metadata()
```

2.2.2 Metadata Help

To get help with any metadata attribute you can use:

```
>>> existing_station.metadata.attribute_information('archive_id')
```

archive_id: alias: [] description: station name that is archived {a-z;A-Z;0-9} example: MT201 options: [] required: True style: alpha numeric type: string units: None

If no argument is given information for all metadata attributes will be printed.

2.2.3 Creating New Attributes

If you want to add new standard attributes to the metadata you can do this through **:function:`mth5.metadata.Base.add_base_attribute method`**

```
>>> extra = {'type': str,
...          'style': 'controlled vocabulary',
...          'required': False,
...          'units': 'celsius',
...          'description': 'local temperature',
...          'alias': ['temp'],
...          'options': [ 'ambient', 'air', 'other'],
...          'example': 'ambient'}
>>> existing_station.metadata.add_base_attribute('temperature', 'ambient', extra)
```

2.2.4 Dictionary Input/Output

You can input a dictionary of attributes

Note: The dictionary must be of the form {'level': {'key': 'value'}}, where 'level' is either ['survey' | 'station' | 'run' | 'channel' | 'filter']

```
>>> meta_dict = {'station': {'archive_id': 'MT010'}}
>>> existing_station.metadata.from_dict(meta_dict)
>>> exiting_station.metadata.to_dict()
{'station': OrderedDict([('acquired_by.author', None),
      ('acquired_by.comments', None),
      ('archive_id', 'MT010'),
      ('channel_layout', 'X'),
      ('channels_recorded', ['Hx', 'Hy', 'Hz', 'Ex', 'Ey']),
      ('comments', None),
      ('data_type', 'BB, LP'),
      ('geographic_name', 'Beachy Keen, FL, USA'),
      ('hdf5_reference', '<HDF5 object reference>'),
      ('id', 'FL001'),
      ('location.declination.comments',
       'Declination obtained from the instrument GNSS NMEA sequence'),
      ('location.declination.model', 'Unknown'),
      ('location.declination.value', -4.1),
      ('location.elevation', 0.0),
      ('location.latitude', 29.7203555),
      ('location.longitude', -83.4854715),
```

(continues on next page)

(continued from previous page)

```
      ('mth5_type', 'Station'),
      ('orientation.method', 'compass'),
      ('orientation.reference_frame', 'geographic'),
      ('provenance.comments', None),
      ('provenance.creation_time', '2020-05-29T21:08:40+00:00'),
      ('provenance.log', None),
      ('provenance.software.author', 'Anna Kelbert, USGS'),
      ('provenance.software.name', 'mth5_metadata.m'),
      ('provenance.software.version', '2020-05-29'),
      ('provenance.submitter.author', 'Anna Kelbert, USGS'),
      ('provenance.submitter.email', 'akelbert@usgs.gov'),
      ('provenance.submitter.organization',
       'USGS Geomagnetism Program'),
      ('time_period.end', '2015-01-29T16:18:14+00:00'),
      ('time_period.start', '2015-01-08T19:49:15+00:00')])}
```

2.2.5 JSON Input/Output

JSON input is as a string, therefore you will need to read the file first.

```
>>> json_string = '{"station": {"archive_id": "MT010"}}
>>> existing_station.metadata.from_json(json_string)
>>> print(existing_station.metadata.to_json(nested=True))
{
        "station": {
                "acquired_by": {
                        "author": null,
                        "comments": null
                },
                "archive_id": "FL001",
                "channel_layout": "X",
                "channels_recorded": [
                        "Hx",
                        "Hy",
                        "Hz",
                        "Ex",
                        "Ey"
                ],
                "comments": null,
                "data_type": "BB, LP",
                "geographic_name": "Beachy Keen, FL, USA",
                "hdf5_reference": "<HDF5 object reference>",
                "id": "MT010",
                "location": {
                        "latitude": 29.7203555,
                        "longitude": -83.4854715,
                        "elevation": 0.0,
                        "declination": {
                                "comments": "Declination obtained from the instrument␣
↪GNSS NMEA sequence",
                                "model": "Unknown",
                                "value": -4.1
                        }
                },
                "mth5_type": "Station",
```

(continues on next page)

(continued from previous page)

```
                "orientation": {
                        "method": "compass",
                        "reference_frame": "geographic"
                },
                "provenance": {
                        "creation_time": "2020-05-29T21:08:40+00:00",
                        "comments": null,
                        "log": null,
                        "software": {
                                "author": "Anna Kelbert, USGS",
                                "version": "2020-05-29",
                                "name": "mth5_metadata.m"
                        },
                        "submitter": {
                                "author": "Anna Kelbert, USGS",
                                "organization": "USGS Geomagnetism Program",
                                "email": "akelbert@usgs.gov"
                        }
                },
                "time_period": {
                        "end": "2015-01-29T16:18:14+00:00",
                        "start": "2015-01-08T19:49:15+00:00"
                }
        }
}
```

2.2.6 XML Input/Output

You can input as a XML element following the form previously mentioned. If you store your metadata in XML files you will need to read the and input the appropriate element into the metadata.

```
>>> from xml.etree import cElementTree as et
>>> root = et.Element('station')
>>> et.SubElement(root, 'archive_id', {'text': 'MT010'})
>>> existing_station.from_xml(root)
>>> print(existing_station.to_xml(string=True)
<?xml version="1.0" ?>
<station>
        <acquired_by>
                <author>None</author>
                <comments>None</comments>
        </acquired_by>
        <archive_id>MT010</archive_id>
        <channel_layout>X</channel_layout>
        <channels_recorded>
                <item>Hx</item>
                <item>Hy</item>
                <item>Hz</item>
                <item>Ex</item>
                <item>Ey</item>
        </channels_recorded>
        <comments>None</comments>
        <data_type>BB, LP</data_type>
        <geographic_name>Beachy Keen, FL, USA</geographic_name>
        <hdf5_reference type="h5py_reference">&lt;HDF5 object reference&gt;</hdf5_
↪reference>
```

(continues on next page)

(continued from previous page)

```
        <id>FL001</id>
        <location>
                <latitude type="float" units="degrees">29.7203555</latitude>
                <longitude type="float" units="degrees">-83.4854715</longitude>
                <elevation type="float" units="degrees">0.0</elevation>
                <declination>
                        <comments>Declination obtained from the instrument GNSS NMEA␣
↪sequence</comments>
                        <model>Unknown</model>
                        <value type="float" units="degrees">-4.1</value>
                </declination>
        </location>
        <mth5_type>Station</mth5_type>
        <orientation>
                <method>compass</method>
                <reference_frame>geographic</reference_frame>
        </orientation>
        <provenance>
                <creation_time>2020-05-29T21:08:40+00:00</creation_time>
                <comments>None</comments>
                <log>None</log>
                <software>
                        <author>Anna Kelbert, USGS</author>
                        <version>2020-05-29</version>
                        <name>mth5_metadata.m</name>
                </software>
                <submitter>
                        <author>Anna Kelbert, USGS</author>
                        <organization>USGS Geomagnetism Program</organization>
                        <email>akelbert@usgs.gov</email>
                </submitter>
        </provenance>
        <time_period>
                <end>2015-01-29T16:18:14+00:00</end>
                <start>2015-01-08T19:49:15+00:00</start>
        </time_period>
</station>
```

See also:

`mth5.metadata` for more information.

CHAPTER

THREE

STATIONS

Stations are the top level for an MT sounding. There are 2 station containers `mth5.groups.MasterStationsGroup` and `mth5.groups.StationGroup`.

3.1 Master Stations Group

`mth5.groups.MasterStationsGroup` is an umbrella container that holds a collection of `mth5.groups.StationGroup` objects and contains a summary table that summarizes all stations within the survey. Use `mth5.groups.MasterStationsGroup` to add/get/remove stations.

No metadata currently accompanies `mth5.groups.MasterStationsGroup`. There will soon be a list of `mth5.groups.StationGroup` objects for all stations.

There are 2 ways to add/remove/get stations. Add/get will return a `mth5.groups.StationGroup`. If adding a station that has the same name as an existing station the `mth5.groups.StationGroup` returned will be of the existing station and no station will be added. Change the name or update the existing staiton. If getting a station that does not exist a `mth5.utils.exceptions.MTH5Error` will be raised.

3.1.1 1) Using *stations_group*

The first way to add/get/remove stations is from the **:attribute:`mth5.MTH5.stations_group`** which is a `mth5.groups.MasterStationsGroup` object.

```
>>> stations = mth5_obj.stations_group
>>> type(stations)
mth5.groups.MasterStationsGroup
>>> stations
/Survey/Stations:
====================
        |- Group: MT001
        ---------------
                |- Group: MT001a
                ----------------
                        --> Dataset: Ex
                        ..................
                        --> Dataset: Ey
                        ..................
                        --> Dataset: Hx
                        ..................
                        --> Dataset: Hy
                        ..................
                        --> Dataset: Hz
                        ..................
                        --> Dataset: Summary
                        ......................
```

From the *stations_group* you can add/remove/get a station.

To add a station:

```
>>> new_station = stations.add_station('MT002')
>>> print(type(new_station))
mth5.groups.StationGroup
>>> new_station
/Survey/Stations/MT002:
====================
--> Dataset: Summary
......................
```

To get an existing station:

```
>>> existing_station = stations.get_station('MT001')
```

To remove an existing station:

```
>>> stations.remove_station('MT002')
>>> stations.group_list
['Summary', 'MT001']
```

3.1.2 2) Using Covnenience methods

The second way to add/remove/get stations is from the convenience functions in `mth5.MTH5`. These use the same methods as the `mth5.groups.MasterStationsGroup` but can be accessed directly.

To add a station:

```
>>> new_station = mth5_obj.add_station('MT002')
>>> mth5_obj
/:
====================
        |- Group: Survey
        ----------------
                |- Group: Filters
                -----------------
                        --> Dataset: Summary
                        ......................
                |- Group: Reports
                -----------------
                        --> Dataset: Summary
                        ......................
                |- Group: Standards
                -------------------
                        --> Dataset: Summary
                        ......................
                |- Group: Stations
                -------------------
                        |- Group: MT001
                        ---------------
                                --> Dataset: Summary
                                ......................
                        |- Group: MT002
                        ---------------
                                --> Dataset: Summary
                                ......................
                        --> Dataset: Summary
                        ......................
```

To get an existing station:

```
>>> existing_station = mth5_obj.get_station('MT002')
```

To remove an existing station:

```
>>> mth5_obj.remove_station('MT002')
```

3.1.3 Summary Table

Column	Description
archive_id	Station archive name
start	Start time of the station (ISO format)
end	End time of the station (ISO format)
components	All components measured by the station
measurement_type	All measurement types collected by the station
location.latitude	Station latitude (decimal degrees)
location.longitude	Station longitude (decimal degrees)
location.elevation	Station elevation (meters)
hdf5_reference	Internal HDF5 reference

3.2 Station Group

A single station is contained within a `mth5.groups.StationGroup` object, which has the appropriate metadata for a single station. `mth5.groups.StationGroup` contains all the runs for that station.

3.2.1 Summary Table

The summary table in `mth5.groups.StationGroup` summarizes all runs for that station.

Column	Description
id	Run ID
start	Start time of the run (ISO format)
end	End time of the run (ISO format)
components	All components measured for that run
measurement_type	Type of measurement for that run
sample_rate	Sample rate of the run (samples/second)
hdf5_reference	Internal HDF5 reference

3.2.2 Metadata

Metadata is accessed through the *metadata* property, which is a `mth5.metadata.Station` object.

```
>>> type(new_station.metadata)
mth5.metadata.Station
>>> new_station.metadata
{
        "station": {
                "acquired_by.author": null,
                "acquired_by.comments": null,
                "archive_id": "FL001",
                "channel_layout": "X",
                "channels_recorded": [
                        "Hx",
                        "Hy",
                        "Hz",
                        "Ex",
```

(continues on next page)

(continued from previous page)


```
                        "Ey"
                ],
                "comments": null,
                "data_type": "BB, LP",
                "geographic_name": "Beachy Keen, FL, USA",
                "hdf5_reference": "<HDF5 object reference>",
                "id": "FL001",
                "location.declination.comments": "Declination obtained from the␣
↪instrument GNSS NMEA sequence",
                "location.declination.model": "Unknown",
                "location.declination.value": -4.1,
                "location.elevation": 0.0,
                "location.latitude": 29.7203555,
                "location.longitude": -83.4854715,
                "mth5_type": "Station",
                "orientation.method": "compass",
                "orientation.reference_frame": "geographic",
                "provenance.comments": null,
                "provenance.creation_time": "2020-05-29T21:08:40+00:00",
                "provenance.log": null,
                "provenance.software.author": "Anna Kelbert, USGS",
                "provenance.software.name": "mth5_metadata.m",
                "provenance.software.version": "2020-05-29",
                "provenance.submitter.author": "Anna Kelbert, USGS",
                "provenance.submitter.email": "akelbert@usgs.gov",
                "provenance.submitter.organization": "USGS Geomagnetism Program",
                "time_period.end": "2015-01-29T16:18:14+00:00",
                "time_period.start": "2015-01-08T19:49:15+00:00"
        }
}
```


See also:

`mth5.groups.StationGroup`

CHAPTER

FOUR

RUNS

A run is a collection of channels that recorded at similar start and end times at the same sample rate for a given station. A run is contained within a `mth5.groups.RunGroup` object. A run is the next level down from a station.

The main way to add/remove/get a run object is through a `mth5.groups.StationGroup` object

4.1 Accessing through StationGroup

You can get a `mth5.groups.StationGroup` using either method in the previous section.

```
>>> new_station = mth5_obj.add_station('MT003')
```

or

```
>>> new_station = mth5_obj.stations_group.add_station('MT003')
```

4.1.1 Add Run

```
>>> # if you don't already have a run name one can be assigned based on existing runs
>>> new_run_name = new_station.make_run_name()
>>> new_run = new_station.add_run(new_run_name)
```

Or

```
>>> new_run = mth5_obj.add_run('MT003', 'MT003a')
```

4.1.2 Get Run

Similar methods for get/remove a run

```
>>> existing_run = new_station.get_run('MT003a')
```

or

```
>>> existing_run = mth5_obj.get_run('MT003', 'MT003a')
```

4.1.3 Remove Run

```
>>> new_station.remove_run('MT003a')
```

or

```
>>> mth5_obj.remove_run('MT003', 'MT003a')
```

4.2 Summary Table

The summary table summarizes all channels for that run.

Column	Description
component	Component name
start	Start time of the channel (ISO format)
end	End time of the channel (ISO format0
n_samples	Number of samples for the channel
measurement_type	Measuremnt type of the channel
units	Units of the channel data
hdf5_reference	HDF5 internal reference

4.3 Metadata

Metadata is accessed through the *metadata* property, which is a `mth5.metadata.Run` object.

```
>>> type(new_run)
mth5.metadata.Run
>>> new_run.metadata
{
        "run": {
                "acquired_by.author": "BB",
                "acquired_by.comments": "it's cold in florida",
                "channels_recorded_auxiliary": null,
                "channels_recorded_electric": null,
                "channels_recorded_magnetic": null,
                "comments": null,
                "data_logger.firmware.author": "Barry Narod",
                "data_logger.firmware.name": null,
                "data_logger.firmware.version": null,
```

(continues on next page)

(continued from previous page)


```
                "data_logger.id": "1305-1",
                "data_logger.manufacturer": "Barry Narod",
                "data_logger.model": "NIMS",
                "data_logger.power_source.comments": "voltage measurements not␣
↪recorded",
                "data_logger.power_source.id": null,
                "data_logger.power_source.type": "battery",
                "data_logger.power_source.voltage.end": null,
                "data_logger.power_source.voltage.start": null,
                "data_logger.timing_system.comments": null,
                "data_logger.timing_system.drift": 0.0,
                "data_logger.timing_system.type": "GPS",
                "data_logger.timing_system.uncertainty": 1.0,
                "data_logger.type": null,
                "data_type": "BB, LP",
                "hdf5_reference": "<HDF5 object reference>",
                "id": "MT003a",
                "metadata_by.author": "Anna Kelbert; Paul Bedrosian",
                "metadata_by.comments": "Paul Bedrosian: Ey, electrode dug up",
                "mth5_type": "Run",
                "provenance.comments": null,
                "provenance.log": null,
                "sample_rate": 8.0,
                "time_period.end": "2015-01-19T14:54:54+00:00",
                "time_period.start": "2015-01-08T19:49:15+00:00"
        }
}
```


See also:

`mth5.groups.RunGroup` and `mth5.metadata.Run` for more information.

CHAPTER

FIVE

A STANDARD FOR EXCHANGEABLE MAGNETOTELLURIC METADATA

Author Working Group for Data Handling and Software - PASSCAL

Magnetotelluric Program :Date: **Version 0.0.16 – July 2020**[1]

5.1 Introduction

Researchers using magnetotelluric (MT) methods lack a standardized format for storing time series data and metadata. Commercially available MT instruments produce data in formats that range from proprietary binary to ASCII, whereas recent datasets from the U.S. MT community have utilized institutional formats or heavily adapted formats like miniSEED. In many cases, the available metadata for MT time series are incomplete and loosely standardized; and overall, these datasets are not "user friendly". This lack of a standardized resource impedes the exchange and broader use of these data beyond a small community of specialists.

The IRIS PASSCAL MT facility maintains a pool of MT instruments that are freely available to U.S. Principal Investigators (PIs). Datasets collected with these instruments are subject to data sharing requirements, and an IRIS working group advises the development of sustainable data formats and workflows for this facility. Following in the spirit of the standard created for MT transfer function datasets, this document outlines a new metadata standard for level 0,1,and 2 MT time series data (Data Levels). Following community approval of these standards, MTH5 (an HDF5 MT specific format) will be developed later in 2020.

The Python 3 module written for these standards and MTH5 is being developed at https://github.com/kujaku11/MTarchive/tree/tables.

5.2 General Structure

The metadata for a full MT dataset are structured to cover details from single channel time series to a full survey. For simplicity, each of the different scales of an MT survey and measurements have been categorized starting from largest to smallest (Figure *1*). These categories are: `Survey`, `Station`, `Run`, `DataLogger`, `Electric Channel`, `Magnetic Channel`, and `Auxiliary Channel`. Each category is described in subsequent sections. Required keywords are labeled as and suggested keywords are labeled as . A user should use as much of the suggested metadata as possible for a full description of the data.

[1] **Corresponding Authors:**
Jared Peacock (jpeacock@usgs.gov)
Andy Frassetto (andy.frassetto@iris.edu)

5.2.1 Metadata Keyword Format

The metadata key names should be self-explanatory and are structured as follows:
`{category}.{name}`, or can be nested `{category1}.{categroy2}.{name}` where:

- `category` refers to a metadata category or level that has common parameters, such as `location`, which will have a latitude, longitude, and elevation ⟶ `location.latitude`, `location.longitude`, and `location.elevation`. These can be nested, for example, `station.location.latitude`
- `name` is a descriptive name, where words should be separated by an underscore. Note that only whole words should be used and abbreviations should be avoided, e.g. `data_quality`.

A '.' represents the separator between different categories. The metadata can be stored in many different forms. Common forms are XML or JSON formats. See examples below for various ways to represent the metadata.

5.2.2 Formatting Standards

Specific and required formatting standards for location, time and date, and angles are defined below and should be adhered to.

Time and Date Format

All time and dates are given as an ISO formatted date-time String in the UTC time zone. The ISO Date Time format is `YYYY-MM-DDThh:mm:ss.ms+00:00`, where the UTC time zone is represented by `+00:00`. UTC can also be denoted by `Z` at the end of the date-time string `YYYY-MM-DDThh:mm:ss.msZ`. Note that `Z` can also represent Greenwich Mean Time (GMT) but is an acceptable representation of UTC time. If the data requires a different time zone, this can be accommodated but it is recommended that UTC be used whenever possible to avoid confusion of local time and local daylight savings. Milliseconds can be accurate to 9 decimal places. ISO dates are formatted `YYYY-MM-DD`. Hours are given as a 24 hour number or military time, e.g. 4:00 PM is 16:00.

Location

All latitude and longitude locations are given in decimal degrees in the well known datum specified at the `Survey` level. **NOTE: The entire survey should use only one datum that is specified at the Survey level.**

- All latitude values must be $< |90|$ and all longitude values must be $< |180|$.
- Elevation and other distance values are given in meters.
- Datum should be one of the well known datums, WGS84 is preferred, but others are acceptable.

Angles

All angles of orientation are given in decimal degrees. Orientation of channels should be given in a geographic or a geomagnetic reference frame where the right-hand coordinates are assumed to be North = 0, East = 90, and vertical is positive downward (Figure *2*). The coordinate reference frame is given at the station level `station.orientation.reference_frame`. Two angles to describe the orientation of a sensor is given by `channel.measurement_azimuth` and `channel.measurement_tilt`. In a geographic or geomagnetic reference frame, the azimuth refers to the horizontal angle relative to north positive clockwise, and the tilt refers to the vertical angle with respect to the horizontal plane. In this reference frame, a tilt angle of 90 points downward, 0 is parallel with the surface, and -90 points upwards.

Archived data should remain in measurement coordinates. Any transformation of coordinates for derived products can store the transformation angles at the channel level in `channel.transformed_azimuth` and `channel.transformed_tilt`, the transformed reference frame can then be recorded in `station.orientation.transformed_reference_frame`.

5.2.3 Units

Acceptable units are only those from the International System of Units (SI). Only long names in all lower case are acceptable. Table *1* summarizes common acceptable units.

Table 1: Acceptable Units

Measurement Type	Unit Name
Angles	decimal degrees
Distance	meter
Electric Field	millivolt
Latitude/Longitude	decimal degrees
Magnetic Field	nanotesla
Resistance	ohms
Resistivity	ohm-meter
Temperature	celsius
Time	second
Voltage	volt

[tab:units]

5.2.4 String Formats

Each metadata keyword can have a specific string style, such as date and time or alpha-numeric. These are described in Table *2*. Note that any list should be comma separated.

Table 2: Acceptable String Formats

Style	Description	Example
Free Form	An unregulated string that can contain {a-z, A-Z, 0-9} and special characters	This is Free Form!
Alpha Numeric	A string that contains no spaces and only characters {a-z, A-Z, 0-9, -, /, _}	WGS84 or GEOMAG-USGS
Controlled Vocabulary	Only certain names or words are allowed. In this case, examples of acceptable values are provided in the documentation as [option01 | option02 | …]. The … indicates that other options are possible but have not been defined in the standards yet	reference_frame = geographic
List	List of entries using a comma separator	Ex, Ey, Hx, Hy, Hz, T
Number	A number according to the data type; number of decimal places has not been implemented yet	10.0 (float) or 10 (integer)
Date	ISO formatted date YYYY-MM-DD in UTC	2020-02-02
Date Time	ISO formatted date time YYYY-MM- DDThh:mm:ss.ms+00:00 in UTC	2020-02-02T12:20:45.123456+00:00
Email	A valid email address	person@mt.org
URL	A full URL that a user can view in a web browser	https://www.passcal.nmt.edu/

[tab:values]

5.3 Survey

A survey describes an entire data set that covers a specific time span and region. This may include multiple PIs in multiple data collection episodes but should be confined to a specific experiment or project. The `Survey` metadata category describes the general parameters of the survey.

Metadata Key | Description | Example |

+=====================+=====================+=====================+

acquired_by.author

None

String

Free Form

& Name of the person or persons who acquired the data. This can be different from the project lead if a contractor or different group collected the data. & person name

acquired_by.comments

None

String

Free Form

& Any comments about aspects of how the data were collected or any inconsistencies in the data. & Lightning strike caused a time skip at 8 am UTC.

archive_id

None

String

Alpha Numeric

& Alphanumeric name provided by the archive. For IRIS this will be the FDSN providing a code. & YKN20

archive_network

None

String

Alpha Numeric

& Network code given by PASSCAL/IRIS/FDSN. This will be a two character String that describes who and where the network operates. & EM

citation_dataset.doi

None

String

URL

& The full URL of the doi Number provided by the archive that describes the raw data & http://doi.10.adfabe

citation_journal.doi

None

String

URL

& The full URL of the doi Number for a journal article(s) that uses these data. If multiple journal articles use these data provide as a comma separated String of urls. & http://doi.10.xbsfs, or http://doi.10.xbsfs, http://doi.10.xbsfs2

[tab:survey]

Table 3: Attributes for Survey Continued

Metadata Key	Description	Example
comments None String Free Form	Any comments about the survey that are important for any user to know.	Solar activity low.
country None String Free Form	Country or countries that the survey is located in. If multiple input as comma separated names.	USA, Canada
datum None String Controlled Vocabulary	The reference datum for all geographic coordinates throughout the survey. It is up to the user to be sure that all coordinates are projected into this datum. Should be a well-known datum: [WGS84 | NAD83 | OSGB36 | GDA94 | ETRS89 | PZ-90.11 | …]	WGS84
geographic_name None String Free Form	Geographic names that encompass the survey. These should be broad geographic names. Further information can be found at https://www.usgs.gov/core-scien ce-systems/ngp/board -on-geographic-names	Eastern Mojave, Southwestern USA
name None String Free Form	Descriptive name of the survey, similar to the title of a journal article.	MT Characterization of Yukon Terrane
northwest_corner.latitude decimal degrees Float Number	Latitude of the northwest corner of the survey in the datum specified.	
northwest_corner.longitude decimal degrees Float Number	Longitude of the northwest corner of the survey in the datum specified.	

[tab:survey2]

Table 4: Attributes for Survey Continued

Metadata Key	Description	Example
project None String Free Form	Alphanumeric name for the project. This is different than the archive_id in that it describes a project as having a common project lead and source of funding. There may be multiple surveys within a project. For example if the project is to estimate geomagnetic hazards that project = GEOMAG but the archive_id = YKN20.	GEOMAG
project_lead.author None String Free Form	Name of the project lead. This should be a person who is responsible for the data.	Magneto
project_lead.email None String Email	Email of the project lead. This is in case there are any questions about data.	mt.guru@em.org
project_lead.organization None String Free Form	Organization name of the project lead.	MT Gurus
release_license None String Controlled Vocabulary	How the data can be used. The options are based on Creative Commons licenses. Options: [CC 0 | CC BY | CC BY-SA| CC BY-ND | CC BY-NC-SA | CC BY-NC-ND]. For details visit https://creati vecommons.org/licenses/	CC 0
southeast_corner.latitude decimal degrees Float Number	Latitude of the southeast corner of the survey in the datum specified.	
southeast_corner.longitude decimal degrees Float Number	Longitude of the southeast corner of the survey in the datum specified.	

[tab:survey3]

Table 5: Attributes for Survey Continued

Metadata Key	Description	Example
summary None String Free Form	Summary paragraph of the survey including the purpose; difficulties; data quality; summary of outcomes if the data have been processed and modeled.	Long project of characterizing mineral resources in Yukon
time_period.end_date None String Date	End date of the survey in UTC.	-02-01
time_period.start_date None String Date	Start date of the survey in UTC.	-06-21

[tab:survey4]

5.3.1 Example Survey XML Element

```
<?xml version="1.0" ?>
<survey>
    <acquired_by>
        <author>MT Graduate Students</author>
        <comments>Multiple over 5 years</comments>
    </acquired_by>
    <archive_id>SAM1990</archive_id>
    <archive_network>EM</archive_network>
    <citation_dataset>
        <doi>https://doi.###</doi>
    </citation_dataset>
    <citation_journal>
        <doi>https://doi.###</doi>
    </citation_journal>
    <comments>None</comments>
    <country>USA, Canada</country>
    <datum>WGS84</datum>
    <geographic_name>Yukon</geographic_name>
    <name>Imaging Gold Deposits of the Yukon Province</name>
    <northwest_corner>
        <latitude type="Float" units="decimal degrees">-130</latitude>
        <longitude type="Float" units="decimal degrees">75.9</longitude>
    </northwest_corner>
    <project>AURORA</project>
    <project_lead>
        <Email>m.tee@mt.org</Email>
        <organization>EM Ltd.</organization>
        <author>M. Tee</author>
    </project_lead>
    <release_license>CC0</release_license>
    <southeast_corner>
        <latitude type="Float" units="decimal degrees">-110.0</latitude>
```

(continues on next page)

(continued from previous page)

```
        <longitude type="Float" units="decimal degrees">65.12</longitude>
    </southeast_corner>
    <summary>This survey spanned multiple years with graduate students
             collecting the data.  Lots of curious bears and moose,
             some interesting signal from the aurora.  Modeled data
             image large scale crustal features like the
             "fingers of god" that suggest large mineral deposits.
    </summary>
    <time_period>
        <end_date>2020-01-01</end_date>
        <start_date>1995-01-01</start_date>
    </time_period>
</survey>
```

5.4 Station

A station encompasses a single site where data are collected. If the location changes during a run, then a new station should be created and subsequently a new run under the new station. If the sensors, cables, data logger, battery, etc. are replaced during a run but the station remains in the same location, then this can be recorded in the `Run` metadata but does not require a new station entry.

Table 6: Attributes for Station

Metadata Key	Description	Example
** acquired_by.author** None String Free Form	Name of person or group that collected the station data and will be the point of contact if any questions arise about the data.	person name
ac quired_by.comments None String Free Form	Any comments about who acquired the data.	Expert diggers.
archive_id None String Alpha Numeric	Station name that is archived a-z;A-Z;0-9. For IRIS this is a 5 character String.	MT201
channel_layout None String Controlled Vocabulary	How the dipoles and magnetic channels of the station were laid out. Options: [L | + | …]	•
• *channels_recorded** None String Controlled Vocabulary	List of components recorded by the station. Should be a summary of all channels recorded dropped channels will be recorded in Run. Options: [Ex | Ey | Hx | Hy | Hz | T | Battery | …]	Ex, Ey, Hx, Hy, Hz, T
comments None String Free Form	Any comments on the station that would be important for a user.	Pipeline near by.

[tab:station]

Table 7: Attributes for Station Continued

Metadata Key	Description	Example
data_type None String Controlled Vocabulary	All types of data recorded by the station. If multiple types input as a comma separated list. Options: [RMT | AMT | BBMT | LPMT | ULPMT | …]	BBMT
geographic_name None String Free Form	Closest geographic name to the station, should be rather general. For further details about geographic names see https://www .usgs.gov/core-scien ce-systems/ngp/board -on-geographic-names	"Whitehorse, YK"
id None String Free Form	Station name. This can be a longer name than the archive_id name and be a more explanatory name.	bear hallabaloo
location.de clination.comments None String Free Form	Any comments on declination that are important to an end user.	Different than recorded declination from data logger.
location .declination.model None String Controlled Vocabulary	Name of the geomagnetic reference model as {m odel_name}{-}{YYYY}. Model options:	WMM-2016
location .declination.value decimal degrees Float Number	Declination angle relative to geographic north positive clockwise estimated from location and geomagnetic model.	
** location.elevation** meters Float Number	Elevation of station location in datum specified at survey level.	

Table 8: Attributes for Station Continued

Metadata Key	Description	Example
• *location.latitude** decimal degrees Float Number	Latitude of station location in datum specified at survey level.	
** location.longitude** decimal degrees Float Number	Longitude of station location in datum specified at survey level.	
** orientation.method** None String Controlled Vocabulary	Method for orienting station channels. Options: [compass | GPS | theodolite | electric_compass | ...]	compass
orientati on.reference_frame None String Controlled Vocabulary	Reference frame for station layout. There are only 2 options geographic and geomagnetic. Both assume a right-handed coordinate system with North=0, E=90 and vertical positive downward. Options: [geographic | geomagnetic]	geomagnetic
o rientation.transform ed_reference_frame None Float Number	Reference frame rotation angel relative to orienta tion.reference_frame assuming positive clockwise. Should only be used if data are rotated.	
p rovenance.comments None String Free Form	Any comments on provenance of the data.	From a graduated graduate student.
proven ance.creation_time None String Date Time	Date and time the file was created.	-02-08 T12:23:40.324600 +00:00

Table 9: Attributes for Station Continued

Metadata Key	Description	Example
provenance.log None String Free Form	A history of any changes made to the data.	-02-10 T14:24:45+00:00 updated station metadata.
provenan ce.software.author None String Free Form	Author of the software used to create the data files.	programmer 01
proven ance.software.name None String Free Form	Name of the software used to create data files	mtrules
provenanc e.software.version None String Free Form	Version of the software used to create data files	12.01a
provenanc e.submitter.author None String Free Form	Name of the person submitting the data to the archive.	person name
provenan ce.submitter.email None String Email	Email of the person submitting the data to the archive.	mt.guru@em.org
provenance.subm itter.organization None String Free Form	Name of the organization that is submitting data to the archive.	MT Gurus

Table 10: Attributes for Station Continued

Metadata Key	Description	Example
time_period.end None String Date Time	End date and time of collection in UTC.	-02-04 T16:23:45.453670 +00:00
• *time_period.start** None String Date Time	Start date and time of collection in UTC.	-02-01 T09:23:45.453670 +00:00

5.4.1 Example Station JSON


```
{    "station": {
        "acquired_by": {
            "author": "mt",
            "comments": null},
        "archive_id": "MT012",
        "channel_layout": "L",
        "channels_recorded": "Ex, Ey, Hx, Hy",
        "comments": null,
        "data_type": "MT",
        "geographic_name": "Whitehorse, Yukon",
        "id": "Curious Bears Hallabaloo",
        "location": {
            "latitude": 10.0,
            "longitude": -112.98,
            "elevation": 1234.0,
            "declination": {
                "value": 12.3,
                "comments": null,
                "model": "WMM-2016"}},
        "orientation": {
            "method": "compass",
            "reference_frame": "geomagnetic"},
        "provenance": {
            "comments": null,
            "creation_time": "1980-01-01T00:00:00+00:00",
            "log": null,
            "software": {
                "author": "test",
                "version": "1.0a",
                "name": "name"},
            "submitter": {
                "author": "name",
                "organization": null,
                "email": "test@here.org"}},
        "time_period": {
            "end": "1980-01-01T00:00:00+00:00",
            "start": "1982-01-01T16:45:15+00:00"}
         }
}
```


5.5 Run

A run represents data collected at a single station with a single sampling rate. If the dipole length or other such station parameters are changed between runs, this would require adding a new run. If the station is relocated then a new station should be created. If a run has channels that drop out, the start and end period will be the minimum time and maximum time for all channels recorded.

Table 11: Attributes for Run

Metadata Key	Description	Example
** acquired_by.author** None String Free Form	Name of the person or persons who acquired the run data. This can be different from the station.acquired_by and survey.acquired_by.	M.T. Nubee
ac quired_by.comments None String Free Form	Any comments about who acquired the data.	Group of undergraduates.
channels_ recorded_auxiliary None String name list	List of auxiliary channels recorded.	T, battery
channels _recorded_electric None String name list	List of electric channels recorded.	Ex, Ey
channels _recorded_magnetic None String name list	List of magnetic channels recorded.	Hx, Hy, Hz
comments None String Free Form	Any comments on the run that would be important for a user.	Badger attacked Ex.

[tab:run]

Table 12: Attributes for Run Continued

Metadata Key	Description	Example
comments None String Free Form	Any comments on the run that would be important for a user.	cows chewed cables at 9am local time.
data_logg er.firmware.author None String Free Form	Author of the firmware that runs the data logger.	instrument engineer
data_lo gger.firmware.name None String Free Form	Name of the firmware the data logger runs.	mtrules
data_logge r.firmware.version None String Free Form	Version of the firmware that runs the data logger.	12.01a
data_logger.id None String Free Form	Instrument ID Number can be serial Number or a designated ID.	mt01
data_l ogger.manufacturer None String Free Form	Name of person or company that manufactured the data logger.	MT Gurus
• *data_logger.model** None String Free Form	Model version of the data logger.	falcon5

Table 13: Attributes for Run Continued

Metadata Key	Description	Example
data_logger.pow er_source.comments None String Name	Any comment about the power source.	Used a solar panel and it was cloudy.
data_logg er.power_source.id None String name	Battery ID or name	battery01
data_logger .power_source.type None String name	Battery type	pb-acid gel cell
data_logger.power_ source.voltage.end volts Float Number	End voltage	
** data_logger.power_so urce.voltage.start** volts Float Number	Starting voltage	
data_logger.timi ng_system.comments None String Free Form	Any comment on timing system that might be useful for the user.	GPS locked with internal quartz clock
data_logger.t iming_system.drift seconds Float Number	Estimated drift of the timing system.	

Table 14: Attributes for Run Continued

Metadata Key	Description	Example
data_logger. timing_system.type None String Free Form	Type of timing system used in the data logger.	GPS
• *data_logger.timing_ system.uncertainty** seconds Float Number	Estimated uncertainty of the timing system.	
data_logger.type None String Free Form	Type of data logger, this should specify the bit rate and any other parameters of the data logger.	broadband 32-bit
data_type None String Controlled Vocabulary	Type of data recorded for this run. Options: [RMT | AMT | BBMT | LPMT | ULPMT | . . .]	BBMT
id None String Alpha Numeric	Name of the run. Should be station name followed by an alphabet letter for the run.	MT302b
** metadata_by.author** None String Free Form	Person who input the metadata.	Metadata Zen
me tadata_by.comments None String Free Form	Any comments about the metadata that would be useful for the user.	Undergraduate did the input.

Table 15: Attributes for Run

Metadata Key	Description	Example
p rovenance.comments None String Free Form	Any comments on provenance of the data that would be useful to users.	all good
provenance.log None String Free Form	A history of changes made to the data.	-02-10 T14:24:45 +00:00 updated metadata
sampling_rate samples per second Float Number	Sampling rate for the recorded run.	
time_period.end None String Date Time	End date and time of collection in UTC.	-02-04 T16:23:45.453670 +00:00
• *time_period.start** None String Date Time	Start date and time of collection in UTC.	-02-01 T09:23:45.453670 +00:00

[tab:]

5.5.1 Example Run JSON


```
{
    "run": {
        "acquired_by.author": "Magneto",
        "acquired_by.comments": "No hands all telekinesis.",
        "channels_recorded_auxiliary": ["temperature", "battery"],
        "channels_recorded_electric": ["Ex", "Ey"],
        "channels_recorded_magnetic": ["Hx", "Hy", "Hz"],
        "comments": "Good solar activity",
        "data_logger.firmware.author": "Engineer 01",
        "data_logger.firmware.name": "MTDL",
        "data_logger.firmware.version": "12.23a",
        "data_logger.id": "DL01",
        "data_logger.manufacturer": "MT Gurus",
        "data_logger.model": "Falcon 7",
        "data_logger.power_source.comments": "Used solar panel but cloudy",
        "data_logger.power_source.id": "Battery_07",
        "data_logger.power_source.type": "Pb-acid gel cell 72 Amp-hr",
        "data_logger.power_source.voltage.end": 14.1,
        "data_logger.power_source.voltage.start": 13.7,
        "data_logger.timing_system.comments": null,
        "data_logger.timing_system.drift": 0.000001,
        "data_logger.timing_system.type": "GPS + internal clock",
        "data_logger.timing_system.uncertainty": 0.0000001,
        "data_logger.type": "Broadband 32-bit 5 channels",
```


(continues on next page)

(continued from previous page)

```
        "data_type": "BBMT",
        "id": "YKN201b",
        "metadata_by.author": "Graduate Student",
        "metadata_by.comments": "Lazy",
        "provenance.comments": "Data found on old hard drive",
        "provenance.log": "2020-01-02 Updated metadata from old records",
        "sampling_rate": 256,
        "time_period.end": "1999-06-01T15:30:00+00:00",
        "time_period.start": "1999-06-5T20:45:00+00:00"
    }
}
```

5.6 Electric Channel

Electric channel refers to a dipole measurement of the electric field for a single station for a single run.

Table 16: Attributes for Electric

Metadata Key	Description	Example
ac.end volts Float Number	Ending AC value; if more than one measurement input as a list of Number [1 2 . . .]	, 49.5
ac.start volts Float Number	Starting AC value; if more than one measurement input as a list of Number [1 2 . . .]	, 55.8
channel_number None Integer Number	Channel number on the data logger of the recorded channel.	
comments None String Free Form	Any comments about the channel that would be useful to a user.	Lightning storm at 6pm local time
component None String Controlled Vocabulary	Name of the component measured. Options:	Ex
cont act_resistance.end ohms Float Number list	Starting contact resistance; if more than one measurement input as a list [1, 2, . . .]	, 1.8

[tab:electric]

Table 17: Attributes for Electric Continued

Metadata Key	Description	Example
contact_resistance.start ohms Float Number list	Starting contact resistance; if more than one measurement input as a list [1, 2, . . .]	, 1.4
data_quality.rating.author None String Free Form	Name of person or organization who rated the data.	graduate student ace
data_quality.rating.method None String Free Form	The method used to rate the data. Should be a descriptive name and not just the name of a software package. If a rating is provided, the method should be recorded.	standard deviation
data_quality.rating.value None Integer Number	Rating from 1-5 where 1 is bad, 5 is good, and 0 is unrated. Options: [0 | 1 | 2 | 3 | 4 | 5]	
data_quality.warning None String Free Form	Any warnings about the data that should be noted for users.	periodic pipeline noise
dc.end volts Float Number	Ending DC value; if more than one measurement input as a list [1, 2, . . .]	
dc.start volts Float Number	Starting DC value; if more than one measurement input as a list [1, 2, . . .]	

Table 18: Attributes for Electric Continued

Metadata Key	Description	Example
dipole_length meters Float Number	Length of the dipole	
filter.applied None Boolean List	Boolean if filter has been applied or not. If more than one filter, input as a comma separated list. Needs to be the same length as filter.name. If only one entry is given, it is assumed to apply to all filters listed.	True, True
filter.comments None String Free Form	Any comments on filters that is important for users.	low pass is not calibrated
filter.name None String List	Name of filter applied or to be applied. If more than one filter, input as a comma separated list.	counts2mv, low-pass_electric
measurement_azimuth decimal degrees Float Number	Azimuth angle of the channel in the specified survey.orientation.reference_frame.	
measurement_tilt decimal degrees Float Number	Tilt angle of channel in survey.orientat ion.reference_frame.	
** negative.elevation** meters Float Number	Elevation of negative electrode in datum specified at survey level.	

Table 19: Attributes for Electric Continued

Metadata Key	Description	Example
negative.id None String Free Form	Negative electrode ID Number, can be serial number or a designated ID.	electrode01
negative.latitude decimal degrees Float Number	Latitude of negative electrode in datum specified at survey level.	
negative.longitude decimal degrees Float Number	Longitude of negative electrode in datum specified at survey level.	
** negative.manufacturer** None String Free Form	Person or organization that manufactured the electrode.	Electro-Dudes
negative.model None String Free Form	Model version of the electrode.	falcon5
negative.type None String Free Form	Type of electrode, should specify the chemistry.	Ag-AgCl
positive.elevation meters Float Number	Elevation of the positive electrode in datum specified at survey level.	

Table 20: Attributes for Electric Continued

Metadata Key	Description	Example
positive.id None String Free Form	Positive electrode ID Number, can be serial Number or a designated ID.	electrode02
positive.latitude decimal degrees Float Number	Latitude of positive electrode in datum specified at survey level.	
positive.longitude decimal degrees Float Number	Longitude of positive electrode in datum specified at survey level.	
** positive.manufacturer** None String Free Form	Name of group or person that manufactured the electrode.	Electro-Dudes
positive.model None String Free Form	Model version of the electrode.	falcon5
positive.type None String Free Form	Type of electrode, should include chemistry of the electrode.	Pb-PbCl
sample_rate samples per second Float Number	Sample rate of the channel.	

Table 21: Attributes for Electric Continued

Metadata Key	Description	Example
time_period.end None String Date Time	End date and time of collection in UTC	-02-04 T16:23:45.453670 +00:00
• *time_period.start** None String Date Time	Start date and time of collection in UTC.	-02-01T 09:23:45.453670 +00:00
t ransformed_azimuth decimal degrees Float Number	Azimuth angle of channel that has been transformed into a specified coordinate system. Note this value is only for derivative products from the archived data.	
transformed_tilt decimal degrees Float Number	Tilt angle of channel that has been transformed into a specified coordinate system. Note this value is only for derivative products from the archived data.	
type None String Free Form	Data type for the channel.	electric
units None String Controlled Vocabulary	Units of the data, if archived data should always be in counts. Options: [counts | millivolts]	counts

5.6.1 Example Electric Channel JSON


```
{
 "electric": {
    "ac.end": 10.2,
    "ac.start": 12.1,
    "channel_number": 2,
    "comments": null,
    "component": "EX",
    "contact_resistance.end": 1.2,
    "contact_resistance.start": 1.1,
    "data_quality.rating.author": "mt",
    "data_quality.rating.method": "ml",
    "data_quality.rating.value": 4,
    "data_quality.warning": null,
    "dc.end": 1.0,
    "dc.start": 2.0,
    "dipole_length": 100.0,
    "filter.applied": [false],
    "filter.comments": null,
    "filter.name": [ "counts2mv", "lowpass"],
    "measurement_azimuth": 90.0,
```


(continues on next page)

(continued from previous page)


```
    "measurement_tilt": 20.0,
    "negative.elevation": 100.0,
    "negative.id": "a",
    "negative.latitude": 12.12,
    "negative.longitude": -111.12,
    "negative.manufacturer": "test",
    "negative.model": "fats",
    "negative.type": "pb-pbcl",
    "positive.elevation": 101.0,
    "positive.id": "b",
    "positive.latitude": 12.123,
    "positive.longitude": -111.14,
    "positive.manufacturer": "test",
    "positive.model": "fats",
    "positive.type": "ag-agcl",
    "sample_rate": 256.0,
    "time_period.end": "1980-01-01T00:00:00+00:00",
    "time_period.start": "2020-01-01T00:00:00+00:00",
    "type": "electric",
    "units": "counts"
  }
}
```


5.7 Magnetic Channel

A magnetic channel is a recording of one component of the magnetic field at a single station for a single run.

Table 22: Attributes for Magnetic

Metadata Key	Description	Example
channel_number None Integer Number	Channel Number on the data logger.	
comments None String Free Form	Any comments about the channel that would be useful to a user.	Pc1 at 6pm local time.
component None String Controlled Vocabulary	Name of the component measured. Options:	Hx
data_quality.rating.author None String Free Form	Name of person or organization who rated the data.	graduate student ace
data_quality.rating.method None String Free Form	The method used to rate the data. Should be a descriptive name and not just the name of a software package. If a rating is provided, the method should be recorded.	standard deviation
data_quality.rating.value None Integer Number	Rating from 1-5 where 1 is bad, 5 is good, and 0 is unrated. Options: [0 | 1 | 2 | 3 | 4 | 5]	

[tab:magnetic]

Table 23: Attributes for Magnetic Continued

Metadata Key	Description	Example
da ta_quality.warning None String Free Form	Any warnings about the data that should be noted for users.	periodic pipeline noise
filter.applied None Boolean List	Boolean if filter has been applied or not. If more than one filter, input as a comma separated list. Needs to be the same length as filter.name. If only one entry is given, it is assumed to apply to all filters listed.	True, True
filter.comments None String Free Form	Any comments on filters that is important for users.	low pass is not calibrated
filter.name None String List	Name of filter applied or to be applied. If more than one filter, input as a comma separated list.	counts2mv, lowpass_electric
h_field_max.end nanotesla Float Number	Maximum magnetic field strength at end of measurement.	
• *h_field_max.start** nanotesla Float Number	Maximum magnetic field strength at beginning of measurement.	
h_field_min.end nanotesla Float Number	Minimum magnetic field strength at end of measurement.	

Table 24: Attributes for Magnetic Continued

Metadata Key	Description	Example
h_field_min.start nt Float Number	Minimum magnetic field strength at beginning of measurement.	
location.elevation meters Float Number	elevation of magnetometer in datum specified at survey level.	
location.latitude decimal degrees Float Number	Latitude of magnetometer in datum specified at survey level.	
location.longitude decimal degrees Float Number	Longitude of magnetometer in datum specified at survey level.	
measurement_azimuth decimal degrees Float Number	Azimuth of channel in the specified survey.orie ntation.reference_frame.	
measurement_tilt decimal degrees Float Number	Tilt of channel in survey.orie ntation.reference_frame.	
sample_rate samples per second Float Number	Sample rate of the channel.	

Table 25: Attributes for Magnetic Continued

Metadata Key	Description	Example
sensor.id None String Free Form	Sensor ID Number or serial Number.	mag01
s ensor.manufacturer None String Free Form	Person or organization that manufactured the magnetic sensor.	Magnets
sensor.model None String Free Form	Model version of the magnetic sensor.	falcon5
sensor.type None String Free Form	Type of magnetic sensor, should describe the type of magnetic field measurement.	induction coil
time_period.end None String Date Time	End date and time of collection in UTC.	-02-04 T16:23:45.453670 +00:00
• *time_period.start** None String Date Time	Start date and time of collection in UTC.	-02-01 T09:23:45.453670 +00:00
t ransformed_azimuth decimal degrees Float Number	Azimuth angle of channel that has been transformed into a specified coordinate system. Note this value is only for derivative products from the archived data.	

Table 26: Attributes for Magnetic Continued

Metadata Key	Description	Example
transformed_tilt decimal degrees Float Number	Tilt angle of channel that has been transformed into a specified coordinate system. Note this value is only for derivative products from the archived data.	
type None String Free Form	Data type for the channel	magnetic
units None String Controlled Vocabulary	Units of the data. if archiving should always be counts. Options: [counts | nanotesla]	counts

5.7.1 Example Magnetic Channel JSON


```
{    "magnetic": {
        "comments": null,
        "component": "Hz",
        "data_logger": {
            "channel_number": 2},
        "data_quality": {
            "warning": "periodic pipeline",
            "rating": {
                "author": "M. Tee",
                "method": "Machine Learning",
                "value": 3}},
        "filter": {
            "name": ["counts2nT", "lowpass_mag"],
            "applied": [true, false],
            "comments": null},
        "h_field_max": {
            "start": 40000.,
            "end": 420000.},
        "h_field_min": {
            "start": 38000.,
            "end": 39500.},
        "location": {
            "latitude": 25.89,
            "longitude": -110.98,
            "elevation": 1234.5},
        "measurement_azimuth": 0.0,
        "measurement_tilt": 180.0,
        "sample_rate": 64.0,
        "sensor": {
            "id": 'spud',
            "manufacturer": "F. McAraday",
            "type": "tri-axial fluxgate",
            "model": "top hat"},
        "time_period": {
            "end": "2010-01-01T00:00:00+00:00",
            "start": "2020-01-01T00:00:00+00:00"},
        "type": "magnetic",
        "units": "nT"
    }
}
```


5.8 Filters

`Filters` is a table that holds information on any filters that need to be applied to get physical units, and/or filters that were applied to the data to analyze the signal. This includes calibrations, notch filters, conversion of counts to units, etc. The actual filter will be an array of numbers contained within an array named `name` and formatted according to `type`. The preferred format for a filter is a look-up table which programatically can be converted to other formats.

It is important to note that filters will be identified by name and must be consistent throughout the file. Names should be descriptive and self evident. Examples:

- `coil_2284` ⟶ induction coil Number 2284
- `counts2mv` ⟶ conversion from counts to mV

- e_gain ⟶ electric field gain
- datalogger_response_024 ⟶ data logger Number 24 response
- notch_60hz ⟶ notch filter for 60 Hz and harmonics
- lowpass_10hz ⟶ low pass filter below 10 Hz

In each channel there are keys to identify filters that can or have been applied to the data to get an appropriate signal. This can be a list of filter names or a single filter name. An applied key also exists for the user to input whether that filter has been applied. A single Boolean can be provided True if all filters have been applied, or False if none of the filters have been applied. Or applied can be a list the same length as names identifying if the filter has been applied. For example name: "[counts2mv, notch_60hz, e_gain]" and applied: "[True, False, True] would indicate that counts2mv and e_gain have been applied but noth_60hz has not.

Table 27: Attributes for Filter

Metadata Key	Description	Example
type None String Controlled Vocabulary	Filter type. Options: [look up | poles zeros | converter | FIR | . . .]	lookup
name None String Alpha Numeric	Unique name for the filter such that it is easy to query. See above for some examples.	counts2mv
units_in None String Controlled Vocabulary	The input units for the filter. Should be SI units or counts.	counts
units_out None String Controlled Vocabulary	The output units for the filter. Should be SI units or counts.	millivolts
calibration_date None String Date Time	If the filter is a calibration, include the calibration date.	-01-01 T00:00:00 +00:00

[tab:filter]

5.8.1 Example Filter JSON

```
{
    "filter":{
        "type": "look up",
         "name": "counts2mv",
         "units_in": "counts",
         "units_out": "mV",
         "calibration_date": "2015-07-01",
        "comments": "Accurate to 0.001 mV"
    }
}
```

5.9 Auxiliary Channels

Auxiliary channels include state of health channels, temperature, etc.

Table 28: Attributes for Auxiliary

Metadata Key	Description	Example
channel_number None Integer Number	Channel Number on the data logger.	
comments None String Free Form	Any comments about the channel that would be useful to a user.	Pc1 at 6pm local time.
component None String Controlled Vocabulary	Name of the component measured. Options: [temperature | battery | ...]	temperature
data_quality.rating.author None String Free Form	Name of person or organization who rated the data.	graduate student ace
data_quality.rating.method None String Free Form	The method used to rate the data. Should be a descriptive name and not just the name of a software package. If a rating is provided, the method should be recorded.	standard deviation
data_quality.rating.value None Integer Number	Rating from 1-5 where 1 is bad, 5 is good, and 0 is unrated. Options: [0 | 1 | 2 | 3 | 4 | 5]	

[tab:auxiliary]

Table 29: Attributes for Auxiliary Continued

Metadata Key	Description	Example
da ta_quality.warning None String Free Form	Any warnings about the data that should be noted for users.	periodic pipeline noise
filter.applied None Boolean List	Boolean if filter has been applied or not. If more than one filter, input as a comma separated list. Needs to be the same length as filter.name. If only one entry is given, it is assumed to apply to all filters listed.	True, True
filter.comments None String Free Form	Any comments on filters that is important for users.	low pass is not calibrated
filter.name None String List	Name of filter applied or to be applied. If more than one filter, input as a comma separated list.	counts2mv, lowpass_auxiliary
** location.elevation** meters Float Number	Elevation of channel location in datum specified at survey level.	
• *location.latitude** decimal degrees Float Number	Latitude of channel location in datum specified at survey level.	
** location.longitude** decimal degrees Float Number	Longitude of channel location in datum specified at survey level.	

Table 30: Attributes for Auxiliary Continued

Metadata Key	Description	Example
m easurement_azimuth decimal degrees Float Number	Azimuth of channel in the specified survey.orientat ion.reference_frame.	
measurement_tilt decimal degrees Float Number	Tilt of channel in survey.orientat ion.reference_frame.	
sample_rate samples per second Float Number	Sample rate of the channel.	
time_period.end None String time	End date and time of collection in UTC.	-02-04 T16:23:45.453670 +00:00
• *time_period.start** None String time	Start date and time of collection in UTC.	-02-01 T09:23:45.453670 +00:00
t ransformed_azimuth decimal degrees Float Number	Azimuth angle of channel that has been transformed into a specified coordinate system. Note this value is only for derivative products from the archived data.	
transformed_tilt decimal degrees Float Number	Tilt angle of channel that has been transformed into a specified coordinate system. Note this value is only for derivative products from the archived data.	

Table 31: Attributes for Auxiliary Continued

Metadata Key	Description	Example
type None String Free Form	Data type for the channel.	temperature
units None String Controlled Vocabulary	Units of the data. Options: SI units or counts.	celsius

5.9.1 Example Auxiliary XML

```
<auxiliary>
    <comments>great</comments>
    <component>Temperature</component>
    <data_logger>
        <channel_number type="Integer">1</channel_number>
    </data_logger>
    <data_quality>
        <warning>None</warning>
        <rating>
            <author>mt</author>
            <method>ml</method>
            <value type="Integer">4</value>
        </rating>
    </data_quality>
    <filter>
        <name>
            <i>lowpass</i>
            <i>counts2mv</i>
        </name>
        <applied type="boolean">
            <i type="boolean">True</i>
        </applied>
        <comments>test</comments>
    </filter>
    <location>
        <latitude type="Float" units="degrees">12.324</latitude>
        <longitude type="Float" units="degrees">-112.03</longitude>
        <elevation type="Float" units="degrees">1234.0</elevation>
    </location>
    <measurement_azimuth type="Float" units="degrees">0.0</measurement_azimuth>
    <measurement_tilt type="Float" units="degrees">90.0</measurement_tilt>
    <sample_rate type="Float" units="samples per second">8.0</sample_rate>
    <time_period>
        <end>2020-01-01T00:00:00+00:00</end>
        <start>2020-01-04T00:00:00+00:00</start>
    </time_period>
    <type>auxiliary</type>
    <units>celsius</units>
</auxiliary>
```

5.10 Option Definitions

use the closest definition.

Data Type	Definition	Sample Rate [samples/s]
AMT	radio magnetotellurics	$> 10^3$
BBMT	broadband magnetotellurics	$10^3 - 10^0$
LPMT	long-period magnetotellurics	$< 10^0$

[tab:em]

added.

Channel Type	Definition
E	electric field measurement
H	magnetic field measurement
T	temperature
Battery	battery
SOH	state-of-health

[tab:channel_types]

right-hand-rule (Figure *2*) with X in the northern direction, Y in the eastern direction, and Z positive down. If the setup has multiple channels in the same direction, they can be labeled with a Number. For instance, if you measure multiple electric fields Ex01, Ey01, Ex02, Ey02.

Direction	Definition
x	north direction
y	east direction
z	vertical direction
# {0–9}	variable directions

[tab:diretions]

CHAPTER

SIX

MTH5 PACKAGE

6.1 Subpackages

6.1.1 mth5.standards package

Submodules

mth5.standards.schema module

Module contents

6.1.2 mth5.utils package

Submodules

mth5.utils.exceptions module

mth5.utils.helpers module

mth5.utils.mth5logger module

mth5.utils.mttime module

mth5.utils.stationxml_translator module

Module contents

6.2 Submodules

6.3 mth5.calibration module

6.4 mth5.groups module

6.5 mth5.helpers module

6.6 mth5.metadata module

6.7 mth5.mth5 module

6.8 mth5.mth5_tables module

6.9 mth5.timeseries module

6.10 mth5.to_stationxml module

6.11 Module contents

CHAPTER

SEVEN

INDICES AND TABLES

- genindex
- modindex
- search