OPT-TN1287 OptoDAS HDF5 File Format Description

1 Introduction

The purpose of this document is to help gettig started with reading and interpreting OptoDAS HDF5 files containing the raw DAS recordings. This document complements the Python file *simpleDASreader4.py*, which we recommend reading through.

Only meta data fields needed for basic DAS analysis are explained.

1.1 Definitions/ Abbreviations

DAS	Distributed Acoustic Sensing
ROI	Region of Interest
HDF5	Hierarchical Data Format 5
UTC	Coordinated Universal Time
FWHM	Full Width at Half Maximum

1.2 HDF5 File Format

Hierarchical Data Format 5 (HDF5) is a file format developed by the non-profit organization The HDF Group. Their website has guides and documentation for an in depth understanding of the format: www.hdfgroup.org/solutions/hdf5

2 OptoDAS HDF5 Format

2.1.1 File Naming

Continuous DAS recordings are saved to file every 10 seconds with file names hhmmss.hdf5, where hhmmss is the start time of recording. Usually the files are stored in a folder hierarchy with folder names describing the experiment name, date, and data type.

Example:

A recording started at 2020-04-22T07:50:11Z (UTC), with the experiment name "Vibration_monitoring" will be stored under:

'*... /Vibration_monitoring/20200422/dphi/075011.hdf5*'

Here 'dphi' is short for demodulated. Differential phase data is stored under 'dphi'. Note that the filename has a leading zero. The filename will always have six digits.

2.2 Signal Data and Metadata

Figure 1 shows the most important fields and groups of fields contained in the raw HDF5 file. There are also several other fields and groups in addition to those shown here, which will usually not be relevant for the end user.

The metadata is organized into several groups. Two of the groups should be relevant to the end users: 'header' and 'demodSpec'. In the simpleDASreader4.py these are packed in a common dictionary called meta.

The raw DAS data is stored under the root group, '/', with field name '*data'*

```
/
data(int32)[nSamples][nChannels]
```

```
header
channels(int32)[nChannels]
dataScale(float64)
dataType(int32)
dt(float64)
dx(float64)
exp(string)
gaugeLength(float64)
missingSamples(int32)[]
nChannels(int64)
nSamples(int64)
phiOffs(float64)[nChannels]
phiOfsStartTime(float64)
sensitivity(float64)
sensitivityUnit(string)
spatialUnwrRange(float32)
unit(string)
time(float64)
```

```
timing
ppses(int32)[]
sampleSkew(float64)
triggers(int32)[]
```

```
demodSpec
dTau(float64)
nAvgTau(uint8)
nDiffTau(uint8)
roiDec(uint32)[]
roiEnd(uint32)[]
roiStart(uint32)[]
```

```
acqSpec
nSkipLead(int32)
```

```
cableSpec
fiberOverLength(float64)
refractiveIndex(float64)
zeta(float64)
distances(float64)[nChannels]
```

```
monitoring/gps
gpsPosE(float64)
gpsPosH(float64)
gpsPosN(float64)
gpsStatus(uint32)
```

Figure 1 The three most relevant OptoDAS HDF5 groups and their datasets.

A short description of each metadata field is given below.

Group: /		
data	(int32)[nSamples][nChannels]	Two-dimensional array containing differential phase data. Time index in first dimension and channel index in second dimension.

Group: *header*		
channels	int32[nChannels]	Array containing the absolute channel number for each recorded channel.
dataScale	float64	Scaling factor to multiply in order to express data in *unit*.
dataType	int32	Integer defining the data type stored in the dataset *data*. 0 = decimated ADC 1 = complex reflectivity 2 = reflected power 3 = time differentiated phase 4 = post processed
dt	float64	Time between samples [s]. 1/dt = sampling frequency.

dx	float64	Spatial sampling interval [m]
exp	string	Name of recording/experiment.
gaugeLength	float64	Gauge length [m]
missingSamples	int32[]	Array of sample indexes where data has been lost due to sampling errors.
nChannels	int64	Number of sensor channels in data field
nSamples	int64	Number of time samples in data field
phiOffs	float64[nChannels]	Phase offset of first sample (Accumulated phase vs time)
phiOffsStartTime	float64	Time since 1970.01.01 (UTC) at start time of integration to phiOffs [s].
dataScale	float64	Scaling factor to multiply in order to express data in unit.
unit	string	Unit of recorded data after scaling with dataScale. ‘rad/m/s’ for datatype=3
sensitivity	float64	Nominal sensitivity of sensor channels.
sensitivityUnit	string	Unit for sensitivity value. rad/m/ε
spatialUnwrRange	float32	The range that phase data is wrapped into in the spatial domain.
time	float64	Time since 1970.01.01 (UTC) at start of file [s].

Group: *timing*		
ppses	int32[]	List of sample indexes where a pulse on the PPS input is received.
triggers	int32[]	List of sample indexes where a pulse on the Trig input is received.
sampleSkew	float64	The sampling offset in fraction of dt for first sample relative to the time field in the header. This is relevant when then sampling interval is set such that length of the file (10s) is not divisible by the sampling interval.

Group: *demodSpec*		
dTau	float64	Optical return propagation delay between spatial samples [s].
nAvgTau	uint8	Number of spatial samples averaged to calculate the reflector phase at one point.
nDiffTau	uint8	Number of spatial samples over which the phase is differentiated. The recorded phase thus represents the phase change rate over a fiber length with return delay nDiffTau·dTau.
roiDec	uint32[8]	Array containing channel decimation factors of the ROIs.
roiEnd	uint32[8]	Array containing the last absolute channel of each ROI.
roiStart	uint32[8]	Array containing the first absolute channel of each ROI.

Group: acqSpec

nSkipLead	int32	Offset of the absolute channel numbering from the instrument input/output connector. nSkipLead is usually set such that channel zero will be right after the optical circulator.

Group: *cableSpec*		
fiberOverLength	float64	Ratio between fiber length and distance along the cable route. Larger than 1 when the fiber is longer than the cable. (Default: 1.0).
refractiveIndex	float64	Refractive index (Group Index) of fiber (Default: 1.4677).
zeta	float64	Strain-optic coefficient of fiber (Default: 0.78).
sensorDistances	float64[nChannels]	The distance to each sensing point, given by channels*dx

Group / *Subgroup*: *monitoring / Gps*		
gpsPosE	float64	Longitude [deg]
gpsPosH	float64	Height [m]
gpsPosN	float64	Latitude [deg]
gpsStatus	int32	Status of GPS, 0: position (and time) is locked, 1: communication error, 2: GPS clock not synced, 4: position not determined

2.3 DataType = 3: Time differentiated Phase

This section describes how the differential phase data is calculated and the impact of the parameters *nAvgTau* and *nDiffTau* (in the demodSpec group).

Interrogation is performed by sending frequency swept light pulses into the sensing fiber which are backscattered from the whole fiber length. From the detected backscattered response from one sweep the instrument calculates the phase change since the last time sample for each sampled fiber position. The spatial sample separation is given by *dx* (in the header group) in meters.

A spatial moving average is applied with an averaging length of *nAvgTau* samples. Thereafter a spatial differentiation is performed where each output sample is the difference between the phase at two locations separated by *nDiffTau* spatial samples. Subsections of the data may later be extracted for recording, as defined by the region of interest (ROI) parameters (see section 2.5).

Let Φ_N [rad] be the phase of the backscattered light to the instrument from a spatial sampling location N (distance $N{\cdot}dx$ from absolute channel 0). The phase change between consecutive time samples, $\dot{\Phi}_N$ [rad/*dt*] is extracted for every spatial sample location. A spatial moving average is applied to suppress the effect of Rayleigh fading such that $\dot{\Phi}_N$ is averaged over *nAvgTau* sample locations. The result may be labeled $\dot{\Phi}_{avgN}$.

Finally, the change of $\dot{\Phi}_{avgN}$ over *nDiffTau* spatial samples is calculated as:

$$\dot{\varphi}_N = \dot{\Phi}_{avg(N+nDiffTau)} - \dot{\Phi}_{avgN} \qquad \textbf{Eq. 1}$$

$\dot{\varphi}$ is what we refer to as the differential phase data that is stored in the HDF5 file. Since the data is stored as scaled integer values it needs processing before it can be used, see section 2.6.

2.4 Gauge length GL and spatial sampling interval

The instrument has a fixed optical return propagation delay sampling interval of *dTau* = 10^{-9} s, which corresponds to a spatial sampling interval *dx* ≈ 1.02 m for standard telecom fiber. The formula for calculating *dx* is given in appendix 3.1. If the optical fiber length differs from the cable length (i.e. an overlength of fiber in the cable), the *fiberOverLength* (cableSpec group) parameter can be set > 1.0 to give the spatial sampling interval along the cable.

In default settings, *nAvgTau* = *nDiffTau.* In this case the full width half maximum (FWHM) spatial resolution of the measurement equals the *gaugeLength* [m] parameter in the header group, which is defined by Eq. 5 in the appendix.

Figure 2 illustrates the spatial distribution of the response to a Dirac "impulse" strain, which is essentially an elongation of the fiber at a single point with a unit length. We see that the impulse response is a triangular function with FWHM resolution = *gaugeLength* = GL, while the full width of the response is 2·GL.

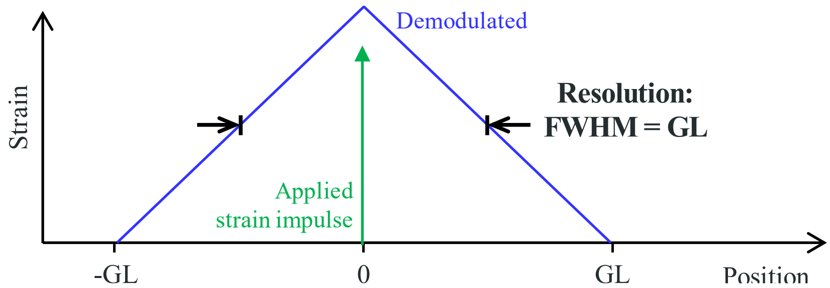

Figure 2 Spatial impulse response.

The red curve in Figure 3 shows the theoretical instrument response to a uniform strain applied to a 20 m long section of the fiber when GL = 10 m. This is a convolution of the applied strain (orange curve) with the blue curve in Figure 2. The measured response will vary somewhat due to the random distribution of the Rayleigh reflectivity in the fiber, as illustrated by the measurement standard deviation shown in the plot.

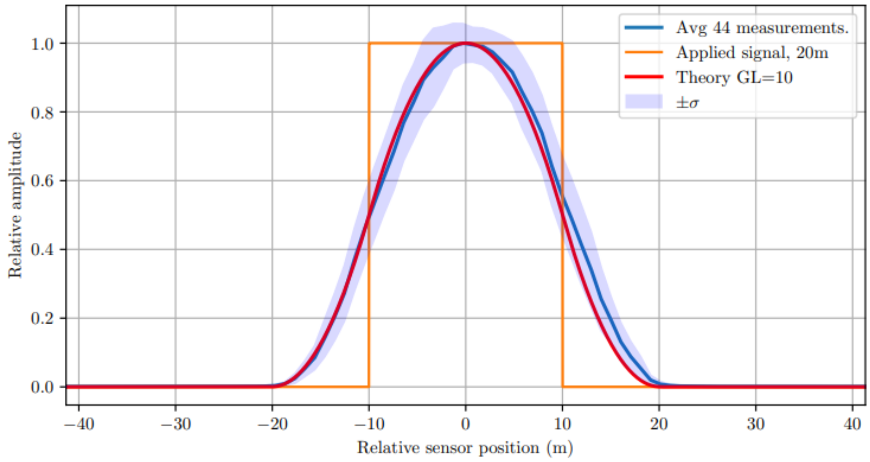

Figure 3 Measured spatial strain response from 20 m of modulated fiber using a gauge length of 10 m.

2.5 Channels and Regions of Interest

We refer to spatial sample locations as channels. Each channel corresponds to a point on the sensing fiber. Channel 0 is usually set right after the optical circulator where light starts reflecting to the interrogator.

For a given sampling rate the maximum number of channels that can be recorded is given by *dt/dTau.* However, the resulting data rate if all these channels are to be transferred is 3.2 Gb/s, which is more than the ethernet transmission capacity connection to the PC (max 2x 1 Gb/s). To reduce the bitrate and save disk space it is therefore possible to reduce the number of transferred channels by defining a set of regions of interest (ROIs). Each ROI defines a region of the cable where channels should be recorded, with an optional channel step, or decimation factor.

The differential phase data is stored as a 2D-array where the channels of the different ROIs are chained (concatenated) after each other. Therefore, the channel index of the data array will not be linearly dependent on the channel position. All absolute channels present in a file are listed in *channels* in the *header* group.

The ROIs are defined by three arrays: *roiStart*, *roiEnd*, and *roiDec*, where each array can have up to 8 elements. Thus, up to 8 ROIs can be defined. The n^{th} ROI will have absolute start channel *roiStart*[*n*], absolute end channel *roiEnd*[*n*], and a spatial decimation factor *roiDec*[*n*]. Note that the channels given by *roiEnd* are included in the ROIs. Also, remember that first absolute channel index is 0.

Example:

roiStart = [0, 4000]

roiEnd = [199, 5999]

roiDec = [1, 5]

This gives two ROIs. The first ROI contains all channels from 0 to 199, 200 channels in total. The second ROI contains every fifth channel from 4000 to 5999, 400 channels in total.

The *data* array will contain 600 channels. Relative channel 100 refers to absolute channel 100 on the fiber. Relative channel 201 refers to absolute channel 4005.

Signal Conditioning

Since the raw data is stored as time differentiated scaled integer values, some conditioning of the data is usually required. This section describes how scaling of the raw data is defined in the file, and a scale factor is provided that can be used to convert to a physical measurand unit.

2.6 Scaling, Phase Unwrapping and Integration

The differential phase data is stored in the two-dimensional int32 array called *data*. The data is phase wrapped along the spatial dimension. In most cases, three processing steps are needed before performing DAS analysis: scaling, phase unwrapping and time integration.

1. The integer values are converted to rad/m/s by multiplying with the header entry *dataScale*. This unit can be found in the string variable *unit*. If *proc*-data is recorded (data that is preproceesed in DasControl/DasDisplay), dataScale is equal to 1.0.
2. The resulting phase will be wrapped in a range $[-spatialUnwrRange/2, spatialUnwrRange/2\rangle$. Unwrapping should be performed along the spatial dimension by adding multiples of *spatialUnwrRange* such that absolute phase steps are < *spatialUnwrRange* in absolute value.
3. Integration of the data along the time axis, resulting in data with unit rad/m.

All three steps above are handled in the Python code bundled with this document *simpleDASreader4.py*. In some cases, the data may be affected by phase unwrapping errors. It is then possible to activate spike detection and removal along the time axis. See the code documentation for more details.

In the DASdisplay software, steps 1 and 2 are handled automatically when loading a file, while step 3 is performed by the Temporal Integration module. Again, a Spike Removal module is available.

In order to evaluate the phase change over multiple files, the field *phiOffs* can be used. *phiOffs* give the time integrated phase at the beginning of the file for all channels since *phiOffsStartTime*. The value of *phiOffs* can thus be added to the result from step 3.

2.7 Conversion from Radians to a Physical Measurand

The phase change φ in a gauge length has a linear relationship to the fiber strain ε. When the instrument is used for ordinary Rayleigh measurements the conversion factor from phase to strain will be found in the header group as *sensitivity*. The unit of the *sensitivity* is given as *sensitivityUnit* = 'rad/m/ε'. The measured strain can thus be calculated as

$$\varepsilon = \varphi / sensitivity \qquad \textbf{Eq. 2}$$

The formula for calculating *sensitivity* is given in appendix 3.1.

The *sensitivity* and *sensitivityUnit* may be different in cases where the fiber measures other measurands than strain.

3 Appendix

3.1 Expressions for scale factors and gauge length

The scale parameters needed to convert absolute channel number to fiber length, the gauge length, as well as the conversion factor from phase to strain are all is given in the *header* group. These parameters are derived from the parameters in the *cableSpec* group, which are set during acquisition.

These parameters are computed from the following formulas:

$$\boldsymbol{dx = \frac{c \cdot dTau}{2 \cdot refractiveIndex \cdot fiberOverLength}} \qquad \textbf{Eq. 3}$$

$$\boldsymbol{sensitivity = \frac{4\pi \cdot zeta \cdot refractiveIndex}{wavelength}} \qquad \textbf{Eq. 4}$$

$$\boldsymbol{gaugeLength = \frac{c \cdot dTau}{2 \cdot refractiveIndex} \cdot nDiffTau} \qquad \textbf{Eq. 5}$$

Here, $c = 299792458\ m/s$ is the speed of light in vacuum.

Document Change History

Date	Rev.	Change by	Check by	Approve by	Change
27.04.2020	01	AH	ER	JKB	First Issue.
14.12.2020	02	OHW	AH, ER	JKB	Update to HDF5 file format rev 5.
04.03.2021	03	AH	OHW	JKB	Update to HDF5 file format rev 7.