How to read h5m

The content of an H5M file can be accessed by any HDF5 compatible tool.

The following viewer is available from the HDF Group:

HDFView : https://www.hdfgroup.org/downloads/hdfview/

Please visit the general support page of the HDF Group for more information.

With Matlab

Read and Plot

h5m_read_and_plot.m 1.0

```
% author: Y. Klinkenberg

% file used in this example, replace this by your file:
fileName = '<path to your file.h5m>';

% this contains all metadata information and can be used to extract data:
fileInfo = h5info(fileName);

% these groups (blocks of data with the same shape exist in the file):
allGroups = {fileInfo.Groups.Name};

% loop over the groups to extract data:
for iGroup = 1:length(allGroups)

        % identify signals per group:
        signalsPerGroup = {fileInfo.Groups(iGroup).Datasets.Name};

        % used to plot the data
        figure
        hold on

        % loop over each signal, load data and plot
        for iSignal = 1:length(signalsPerGroup)

                % retrieve data per signal
                dataSetName = ['/' allGroups{iGroup} '/' signalsPerGroup{iSignal}];
                if ismember('bases', {fileInfo.Groups(iGroup).Datasets(iSignal).Attributes.Name})
                        signalData = h5read(fileName, dataSetName);
                        baseDataC = h5readatt(fileName, dataSetName, 'bases');

                        % show in plot
                        plot(baseDataC{1}, signalData)
                end
        end
end
```

Read to DataStruct

Code below shows how to read an h5m file and yield it's content as a data struct.

h5m_read_struct.m 2.0

```
function [DataStruct] = read_h5m2struct(fileName,nameOption)
```

```
% This file reads H5M files produced by Shark.
% Input: fileName, (optional: nameOption)
% fileName: complete path and filename to H5M-file
% nameOption (optional): true or false [DEFAULT] (or 0 / 1). If true: signals in the output datastruture are
created as fields with the name of the signal as fieldname; Otherwise, accessible by index.
% The data is permuted such that the order of dimensions of the data match the sequence of bases listed for
each signal
% <PLEASE CHECK THIS CAREFULLY; IT HAS BEEN TESTED BUT MAY NOT BE 100% FOOLPROOF>
% Note: The bases themselves (DataStruct.Bases) are not ordered based on the order used for the signals!
They are ordered based on the sequence in the H5M-file.
% Different signals can have a different sequence of bases. The correct order of each signal's bases is in
the "Bases" field of the respective signal (DataStruct.Signals(#).Bases)

% Modified by S van Essen (Jul 2020)
% Modified by FH Lafeber (Aug 2020)


if nargin < 2
    nameOption = false;
end

% this contains all metadata information and can be used to extract data
fileInfo = h5info(fileName);

% these groups (blocks of data with the same shape exist in the file)
allGroups = {fileInfo.Groups.Name};

% Initialise
DataStruct = struct('Signals',[],'Bases',[],'GroupName',[],'StepSize',[]);

% loop over the groups to extract data
for iGroup = 1:length(allGroups)

    signalsPerGroup = {fileInfo.Groups(iGroup).Datasets.Name};          % Check how many signals there are
in total (including the base signals)
    signalCounter = 0;
    baseCounter = 0;
    % loop over each signal, load data and plot
    for iSignal = 1:length(signalsPerGroup)

        % retrieve data per signal
        dataSetName = ['/' allGroups{iGroup} '/' signalsPerGroup{iSignal}];

        if ismember('bases', {fileInfo.Groups(iGroup).Datasets(iSignal).Attributes.Name})   % Put all the
Signals (i.e. signals that have a base) in an array
            signalCounter = signalCounter + 1;
            signalName = {fileInfo.Groups(iGroup).Datasets(iSignal).Name};  % And the signal name
            signalName = matlab.lang.makeValidName(strrep(signalName{1},' ','_'));

            for iattrib = 1:length(fileInfo.Groups(iGroup).Datasets(iSignal).Attributes)
                if strcmp(fileInfo.Groups(iGroup).Datasets(iSignal).Attributes(iattrib).Name,'unit')
                    signalUnit = fileInfo.Groups(iGroup).Datasets(iSignal).Attributes(iattrib).Value;
                    break
                end
            end

            for iattrib = 1:length(fileInfo.Groups(iGroup).Datasets(iSignal).Attributes)
                if strcmp(fileInfo.Groups(iGroup).Datasets(iSignal).Attributes(iattrib).Name,'baseNames')
                    signalBases = fileInfo.Groups(iGroup).Datasets(iSignal).Attributes(iattrib).Value;
                    break
                end
            end

            signalValues = h5read(fileName, dataSetName);
            signalSize = fileInfo.Groups(iGroup).Datasets(iSignal).Dataspace.Size;

            dimensionOrder = length(signalSize):-1:1;   % I am assuming it is always the "wrong way around"

            % Check with the dimensions of the bases? Then I need to find the sequence of the bases for each
```

```
signal and get the dimensions of each of the bases (but that's only once)

            if isstruct(signalValues)
                signalValueNames = fieldnames(signalValues);

                for ival = 1:length(signalValues)
                    for iField = signalValueNames'
                        signalValues(ival).(iField{1}) = permute(signalValues(ival).(iField{1}),
dimensionOrder);
                    end
                end
            elseif min(size(signalValues)) > 1
                signalValues = permute(signalValues,dimensionOrder);
            end

            if ~nameOption
                DataStruct(iGroup).Signals(signalCounter).Bases = signalBases;
                DataStruct(iGroup).Signals(signalCounter).Value = signalValues;               % Get all the
data
                DataStruct(iGroup).Signals(signalCounter).Unit = signalUnit{1};
                DataStruct(iGroup).Signals(signalCounter).Name = signalName;
            else
                DataStruct(iGroup).Signals.(signalName).Unit = signalUnit{1};
                DataStruct(iGroup).Signals.(signalName).Value = signalValues;               % Get all the
data
                DataStruct(iGroup).Signals.(signalName).Base = signalBases;
            end

        else % If it doesn't have a base, it IS a base; put in a separate array

            baseCounter = baseCounter + 1;
            baseName = {fileInfo.Groups(iGroup).Datasets(iSignal).Name};           % Get the base name
            baseName = matlab.lang.makeValidName(strrep(baseName{1},' ','_'));


            for iattrib = 1:length(fileInfo.Groups(iGroup).Datasets(iSignal).Attributes)
                if strcmp(fileInfo.Groups(iGroup).Datasets(iSignal).Attributes(iattrib).Name,'unit')
                    signalUnit = fileInfo.Groups(iGroup).Datasets(iSignal).Attributes(iattrib).Value;
                    break
                end
            end

            if ~nameOption
                DataStruct(iGroup).Bases(baseCounter).Value = h5read(fileName, dataSetName);
                DataStruct(iGroup).Bases(baseCounter).Unit = signalUnit{1};
                DataStruct(iGroup).Bases(baseCounter).Name = baseName;
            else
                DataStruct(iGroup).Bases.(baseName).Value = h5read(fileName, dataSetName);
                DataStruct(iGroup).Bases.(baseName).Unit = signalUnit{1};
            end
        end
    end

    DataStruct(iGroup).GroupName = allGroups{iGroup}(2:end);               % Add the name of the group to
the structure for identification
    DataStruct(iGroup).StepSize = fileInfo.Groups(iGroup).Attributes(find(strcmp({fileInfo.Groups(iGroup).
Attributes.Name},'stepSize'),1,'first')).Value;       % Add the stepsize

end
end
```

Code above can be tested using the code below and using the download test.h5m

example_read_h5m2struct.m 1.0

```
% test H5M reader
clear variables

% For this file the reader works:
path1 = cd;

file0 = '\test.h5m';
fileName0 = [path1,file0];

TestSet = read_h5m2struct(fileName0);
TestSet2 = read_h5m2struct(fileName0,'true');  % Now all signals have their own name as field in the structur

% Explore and do as you please with either TestSet or TestSet2... :-)
```

With Python

With h5py

read_h5m.py

```
import h5py
import numpy

fid = h5py.File('myfile.h5m', 'r')

# Get the name of the measurements/groups in the file:
print(fid.keys)

# Get the name of the signals in the measurement:
print(fid['measurement'].keys())

# Get the data as numpy array:
numpy.array(fid['name of the test']['name of the signal'])

# Get the units of the signal:
fid['name of the test']['name of the signal'].attrs['unit']
```

For more information regarding h5py, see https://docs.h5py.org/

With pymarin

read_h5m.py

```
import pymarin

# Get the groups as list of pymarin.SignalSet:
sets_list = pymarin.io.h5m.read('afile.h5m')
# Or rather get the groups as a dictionary:
sets_dict = pymarin.io.h5m.read('afile.h5m', returnAsDict=True)

# Pick some measurement in the list/dict:
aset = sets_list[0]
another_set = sets_dict['100.00 Hz']

# Get a signal using its name:
asignal = aset['signal_name']
# Or using its index:
another_signal = aset[1]

# Accessing some attributes of the signal:
data = asignal.data
name = asignal.key.name
unit = asignal.key.unit
```

For more help using pymarin, see PYMARIN.