



DEWETRON



---

## **MATLAB IMPORT OF DMD FILES**

**Version:** 1.2

**Date:** 02/10/2023

**Author:** Verena Niederkofler



DEWETRON

The information contained in this document is subject to change without notice.

DEWETRON GmbH (DEWETRON) shall not be liable for any errors contained in this document. DEWETRON MAKES NO WARRANTIES OF ANY KIND WITH REGARD TO THIS DOCUMENT, WHETHER EXPRESS OR IMPLIED. DEWETRON SPECIFICALLY DISCLAIMS THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

DEWETRON shall not be liable for any direct, indirect, special, incidental, or consequential damages, whether based on contract, tort, or any other legal theory, in connection with the furnishing of this document or the use of the information in this document.

## Technical Support

Please contact your local authorized DEWETRON representative first for any support and service questions.

For Asia and Europe, please contact:

### **DEWETRON GmbH**

Parking 4  
8074 Grambach  
AUSTRIA

Tel.: +43 316 3070  
Fax: +43 316 307090  
Email: [support@dewetron.com](mailto:support@dewetron.com)  
Web: <http://www.dewetron.com>

For America, please contact:

### **DEWETRON, Inc.**

PO Box 1460  
Charlestown, RI 02813  
U.S.A.

Tel.: +1 401 364 9464  
Toll-free: +1 877 431 5166  
Fax: +1 401 364 8565  
Email: [support@dewamerica.com](mailto:support@dewamerica.com)  
Web: <http://www.dewamerica.com>

The telephone hotline is available Monday to Friday between 08:00 and 17:00 GST (GMT -5:00)

## Restricted Rights Legend:

Use Austrian law for duplication or disclosure.

### **DEWETRON GmbH**

Parking 4  
8074 Grambach  
AUSTRIA

## Printing History:

Please refer to the page bottom for printing version. Copyright © DEWETRON GmbH



DEWETRON

This document contains information which is protected by copyright. All rights are reserved. Reproduction, adaptation, or translation without prior written permission is prohibited, except as allowed under the copyright laws.

All trademarks and registered trademarks are acknowledged to be the property of their owners.

Before updating your software please contact DEWETRON. Use only original software from DEWETRON.

Please find further information at [www.dewetron.com](http://www.dewetron.com).



DEWETRON

# Table of Contents

1	INTRODUCTION .....	6
1.1	SUPPORTED FEATURES .....	6
1.2	NEEDED FILES .....	6
1.3	IMPORTANT REMARKS .....	7
1.4	ADJUSTING THE SCRIPT BEFORE USAGE .....	7
1.4.1	REDUCED DATA .....	8
1.5	DATA STORING.....	8
2	DATA STRUCTURE.....	9
2.1	ERROR CODES .....	9
2.2	CHANNEL INFORMATION.....	10
2.3	TIMESTAMP .....	10
2.4	MARKERS AND HEADER FIELDS .....	11
2.5	SWEEPS .....	11
2.5.1	REDUCED DATA.....	11
3	FUNCTIONS.....	12
3.1	DMDREADER_INITIALIZE.....	12
3.2	DMDREADER_OPENFILE.....	12
3.3	DMDREADER_GETMEASUREMENTSTARTTIME.....	12
3.4	DMDREADER_GETNUMCHANNELS .....	12
3.5	DMDREADER_GETCHANNELS.....	13
3.6	DMDREADER_GETCHANNELINFORMATION .....	13
3.7	DMDREADER_GETSAMPLETYPE .....	13
3.8	DMDREADER_GETVECTORSAMPLETYPE .....	14
3.9	DMDREADER_GETNUMDATASWEEPS .....	14
3.10	DMDREADER_GETDATASWEEPS.....	14
3.11	DMDREADER_GETNUMREDUCEDSWEEPS.....	15
3.12	DMDREADER_GETREDUCEDSWEEPS .....	15
3.13	DMDREADER_GETSAMPLESANDTS_SCALEDVALUE_SECONDS .....	15
3.14	DMDREADER_GETSAMPLESWITHTS_REDUCEDVALUE_SECONDS.....	16
3.15	DMDREADER_GETSAMPLESANDTS_DIGITALVALUE_SECONDS.....	17
3.16	DMDREADER_GETSAMPLESANDTS_SCALARVECTOR_SECONDS .....	17
3.17	DMDREADER_GETSAMPLESANDTS_COMPLEXVECTOR_SECONDS .....	18



DEWETRON

3.18	DMDREADER_GETNUMMARKERS .....	18
3.19	DMDREADER_GETMARKERS.....	19
3.20	DMDREADER_GETNUMHEADERFIELDS.....	19
3.21	DMDREADER_GETHEADERFIELDS .....	19
3.22	DMDREADER_CLOSEFILE.....	20
4	EXAMPLE USAGE OF DMD-READER .....	21
4.1	EXAMPLE SCREEN IN MATLAB AFTER IMPORT AND PLOTS.....	23
4.1.1	PLOT EXAMPLE 1.....	24
4.1.2	PLOT EXAMPLE 2.....	25
4.1.3	PLOT EXAMPLE 3.....	26
5	TROUBLESHOOTING .....	28

# 1 INTRODUCTION

This documentation gives an overview of the provided MATLAB code to import *dmd* files. It gives an overview of the needed features, explains the data structures and functions used in the script. At the end of the document, some plot examples are shown, also including on how to set the UTC time on the x-axis for plots. This code is available in the script but is commented out and must be uncommented before using it.

## 1.1 SUPPORTED FEATURES

- Import raw data: analog, digital, counter, FFT channel data (scalar and complex vector data)
- Import reduced data: statistical data that is optionally stored during slow monitoring
- Import meta data: file headers, basic channel parameters, markers

## 1.2 NEEDED FILES

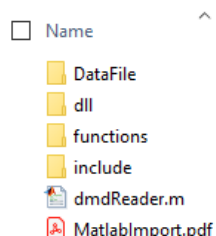
To be able to import *dmd* files into MATLAB a C/C++ compiler compatible with MATLAB is required. Additionally, the following C header files and *dll* files are needed.

- `dmd_reader_api_interface.h`
- `dmd_reader_api_load.h`
- `dmd_reader_api_special.h`
- `dmd_reader_api.dll`
- `dmd_reader_api_x64.dll` (for 64-bit systems)

The current version of the \*.h header files and *dll* files, can be downloaded from the DEWETRON CCC portal: <https://ccc.dewetron.com/>. In order to download files, a registration on the portal is necessary. The needed download file can be found under *Downloads > Software & Tools > OXYGEN > DMD Import > DEWETRON DMDReader Rx.x*.

Please download the latest DEWETRON *DMDReader Rx.x* package. This package contains two folders named *dll* and *include*. Copy those two folders into the same folder where the `dmdReader.m` file is at.

The folder should now contain the following items:





DEWETRON

### 1.3 IMPORTANT REMARKS

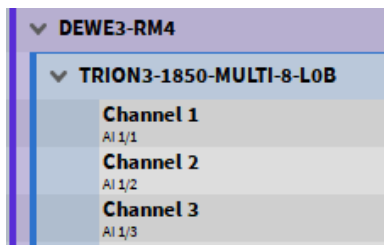
- ➔ C/C++ Compiler is needed for the import to work. Refer to <https://de.mathworks.com/support/requirements/previous-releases.html> to install the right compiler depending on your MATLAB version.
- ➔ For high sample rates (>1/2MS) it might take some seconds to read in each channel. This may cause long waiting times for a high number of channels. Consider the direct export to \*.mat files from OXYGEN.
- ➔ The RAM usage by MATLAB can be adapted in MATLAB: Home – Preferences (Environment) - MATLAB – Workspace - MATLAB array size limit

### 1.4 ADJUSTING THE SCRIPT BEFORE USAGE

This section will explain what lines of code need to be adjusted for an individual usage of the script.

#### DO NOT CHANGE

- All functions are stored in the folder named *'functions'* – do not rename!



- If this folder is renamed, the respective line of code in the script must be adjusted also:

```
dataPath = [path 'DataFile\'];
```

#### ADJUST BEFORE USAGE

1. The files to import must be stored in a folder named *'DataFile'*, in the folder with the .m file.
- a. If this folder has another name, the respective line of code in the script must be adjusted also:

```
funcPath = [path 'functions'];
```
2. Set the filename with the parameter `filename`.
3. Define if all or only specific channels should be read in.
  - a. To read in all channels the parameter `readChannels` must be set to *'all'*.
  - b. To read in only specific channels, define each channel name in the script according to the example. The channels must have the same name as in OXYGEN. By default, all the parameter is set to *'all'* and all channels are read in.

```
readChannels = [string('Channel 1'), ...  
               string('Channel 2'), ...  
               string('Channel 3')];
```



DEWETRON

4. Define if the reduced data should be read in. To read it in, set the parameter `readReducedData` to 1, otherwise to 0. For more details about the reduced data see section 1.4.1. By default, the parameter is set to 0.
5. Define if markers and header data should be read in. To read in either data, set the according parameters `readMarkers` and `readHeader` to 1, otherwise to 0. By default, both parameters are set to 1.
6. Eventually set the path or stay in the same folder as where you have opened the dmd-reader to run the code.

#### 1.4.1 REDUCED DATA

OXYGEN stores two different types of data, the waveform (red box in Figure 1-1) and the statistics or reduced data (blue box in Figure 1-1). The waveform corresponds to the raw data, which is sampled with the set sample rate in OXYGEN. This data is read in using the script.

The reduced data corresponds to some statistical values (min, max, avg, rms), which are also calculated with a specific time window and stored as background information for each channel in OXYGEN by default. This statistical data storing can be deactivated in OXYGEN. For more details, please refer to the OXYGEN Technical Reference Manual, which can be downloaded on <https://ccc.dewetron.com/>.

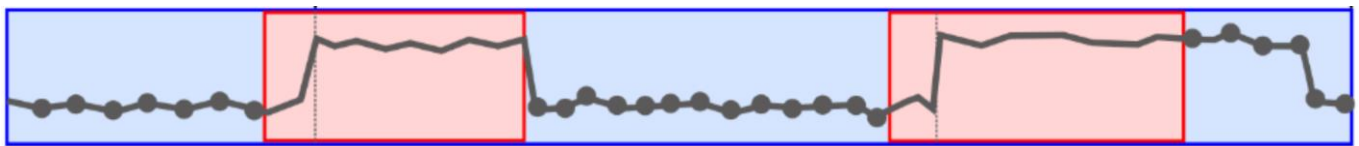


Figure 1-1: Illustration of the waveform and statistical data recording in OXYGEN.

#### 1.5 DATA STORING

A struct is created for each channel with the name specified in OXYGEN and contains all the data with different fields for each sweep. Further, each sweep contains two fields for time and data. A sweep is defined as the timespan from the beginning of the recording until the recording paused or stopped. Therefore, one recording can have multiple sweeps if the recording was paused within the measurement.

Note: Optionally, reduced data gets stored in an additional struct with time, min, max, avg and rms values. For more information about the reduced data see section 1.4.1.

Markers and header fields will be available as structs with all the relevant information.

The Timestamp information will also be available as struct.

The struct `ChannelInfo` contains all the relevant information for each channel in an individual field. E.g. the sample rate can be obtained here for the first Channel by `ChannelInfo(1).channels.sampleRate`

All other parameters/information will be cleared at the end of the script. If it is desired to keep more parameters, it can be changed at the end of the script in the **Dispose and clear** section by deleting/commenting out these parameters.





DEWETRON

## 2 DATA STRUCTURE

The **data structure** for MATLAB is described in the C header file *dmd\_reader\_api\_interface.h*

### 2.1 ERROR CODES

Description of the possible error codes:

Specification of all the possible error codes returned by the dmd reader api interface. DMDReader\_NoError indicates the ok case, whereas all other codes denote the error that occurred.

```
typedef enum
{
    DMDReader_NoError                = 0,
    DMDReader_FileDoesNotExist       = -5001,
    DMDReader_FileInvalid            = -5002,
    DMDReader_InternalError          = -5003,
    DMDReader_InvalidArgument        = -5004,
    DMDReader_InvalidFileHandle      = -5005,
    DMDReader_InvalidChannelHandle   = -5006,
    DMDReader_InvalidMemorySize      = -5007,
    DMDReader_MaximumNumberOfDataValuesExceeded = -5008,
    DMDReader_ChannelDataTypeMismatch = -5009,
    DMDReader_InvalidMarkerHandle    = -5010,
    DMDReader_OutOfMemory             = -5011,
    DMDReader_IncompatibleVersion    = -5012,
    DMDReader_APINotInitialized       = -5013,
    DMDReader_InputBufferTooSmall     = -5014,

    DMDReader_ErrorEnd               = -8000
} DMDReader_ErrorCode_Enum;
```

Another Error Code appears **'Error: Not enough RAM available.'** if the available RAM memory drops below 20% of the total RAM memory. To remove this delete the following code

```
[~,sys] = memory;
if (sys.PhysicalMemory.Available < sys.PhysicalMemory.Total * 0.2)
    error('Error: Not enough RAM available.')
end
```

in the following functions:

```
readSweep.m
readScaledValues.m
readReducedValues.m
readDIValues.m
readVectorValues.m
```



DEWETRON

## 2.2 CHANNEL INFORMATION

The Channel Information is stored as struct named `ChannelInfo` with the following information:

```
typedef struct
{
    double sampleRate;                /**< Sample rate [Hz] */
    DMDReader_ChannelType channelType; /**< Channel Type */
#ifdef BUILD_X64
    DMDReader_uint32 _padding0;
#endif
    DMDReader_STRING name;            /**< Name of the channel */
    DMDReader_STRING unit;
    /**< Physical unit of the channel sample data */
    DMDReader_STRING description;
    /**< Description of the channel */
    DMDReader_TIMESTAMP_SECONDS measurementDuration;
    /**< Measurement duration of the channel's sample data [seconds] */
    DMDReader_SAMPLEVALUE_SCALED rangeMin;
    /**< Minimum range of the channel's sample data. Specified in the physical unit of the
    channel */
    DMDReader_SAMPLEVALUE_SCALED rangeMax;
    /**< Maximum range of the channel's sample data. Specified in the physical unit of the
    channel */
} DMDReader_CHANNELINFORMATION, *DMDReader_P_CHANNELINFORMATION;
```

**Note:** if the sample rate is 0 the channel contains asynchronous data.

## 2.3 TIMESTAMP

The Timestamp is stored in the following format:

Specifies the Timestamp Format.

```
typedef struct
{
    DMDReader_sint32 year;            /**< The year */
    DMDReader_sint32 yearDay;         /**< The day of the year*/

    double timeOfDay;
    /**< Representing the number of seconds after midnight (local time; subtract timeOffset
    to get UTC time)*/

    DMDReader_sint32 timeOffset;
    /**< Representing the offset (in minutes) between UTC and the local time when the
    measurement started (for recordings in CET: +60) */
    DMDReader_bool valid;             /**< True, if time is available */
} DMDReader_TIMESTAMP_UTC;
```

If the parameter `isUTC = 1` the function returns the UTC time, if `isUTC = 0` the function returns the local time and the UTC offset.



DEWETRON

## 2.4 MARKERS AND HEADER FIELDS

Markers are stored in one struct containing the following information:

Specifies the marker event data.

```
typedef struct
{
    DMDReader_MarkerEvent_Source source;
    DMDReader_MarkerEvent_Type type;
    DMDReader_TIMESTAMP_SECONDS time;
    DMDReader_STRING text;
} DMDReader_MARKEREVENT, *DMDReader_P_MARKEREVENT;
```

Header fields are also stored in one struct containing the Header information.

## 2.5 SWEEPS

One sweep is defined as the time between the start and the stop trigger. For synchronous data the `sampleFrequency` is set to samples per second and a single sweep only contains equidistant samples. The timestamp of the first sample is `startTime` and each index from `firstSample` to `lastSample` references an individual sample in this case. For asynchronous data the `sampleFrequency` is set 0.0 and the `*Sample` & `*Time` fields reference the range of the sweep but not necessarily individual samples. Respecting the `numValidSamples` and `nextSample` parameters of `GetSamples` functions is essential to read async data correctly.

```
typedef struct
{
    DMDReader_SampleIndex firstSample;
    DMDReader_SampleIndex lastSample;
    DMDReader_TIMESTAMP_SECONDS startTime;
    DMDReader_TIMESTAMP_SECONDS endTime;
    double sampleFrequency;
} DMDReader_SWEEP, *DMDReader_P_SWEEP;
```

### 2.5.1 REDUCED DATA

If reduced data will be imported into MATLAB, it contains the following information:

```
typedef struct
{
    double min;
    double max;
    double avg;
    double rms;
} DMDReader_SAMPLEVALUE_REDUCED, *DMDReader_P_SAMPLEVALUE_REDUCED;
```



DEWETRON

## 3 FUNCTIONS

### 3.1 DMDREADER\_INITIALIZE

Parameters: interface Version

Return: DMDReader\_ErrorCode      Error Code, see error description

Description: Initializing interface version; This function must be made prior to any other calls

```
DMDReader_ErrorCode = calllib('dmdlib', 'DMDReader_Initialize', interfaceVersion(1),  
interfaceVersion(2));
```

### 3.2 DMDREADER\_OPENFILE

Parameters: fileFullName      name of data file

file\_handle      a pointer on a data file

Return: DMDReader\_ErrorCode      Error Code, see error description

file\_handle      returns the pointer on a data file

Description: Opens data file specified by fileFullName

```
[DMDReader_ErrorCode, ~ , file_handle] = calllib('dmdlib', 'DMDReader_OpenFile',  
fileFullName, file_handle);
```

### 3.3 DMDREADER\_GETMEASUREMENTSTARTTIME

Parameters: file\_handle      a pointer on a data file

p\_TimestampUTC.Value      a pointer on UTC timestamp

isUTC      defines whether to get UTC or local time

Return: DMDReader\_ErrorCode      Error Code, see error description

file\_handle      returns the pointer on a data file

TimestampUTC      Timestamp information

Description: Gets the timestamp information; format described above

```
[DMDReader_ErrorCode, file_handle, TimestampUTC] = calllib('dmdlib',  
'DMDReader_GetMeasurementStartTime', file_handle, p_TimestampUTC.Value, isUTC);
```

### 3.4 DMDREADER\_GETNUMCHANNELS

Parameters: file\_handle      a pointer on a data file

channelType      0 – to get all available channels

numChannels      number of channels (uint64)

Return: DMDReader\_ErrorCode      Error Code, see error description



DEWETRON

file_handle	returns the pointer on a data file
numChannels	number of channels returned

**Description:** Gets number of all available channels within a dmd file

```
[DMDReader_ErrorCode, file_handle, numChannels] = calllib('dmdlib',  
'DMDReader_GetNumChannels', file_handle, 0, numChannels);
```

### 3.5 DMDREADER\_GETCHANNELS

<b>Parameters:</b> file_handle	returns the pointer on a data file
channelType	specific channel type 0 – to get all available channels
idx	channel index
maxChannels	maximum number of channels that will be returned; 1 – to get one channel per iteration
channel_handle	returns the pointer on a channel
outNumChRet	number of channels returned (unused)
<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
file_handle	a pointer on a data file
channel_handle	a pointer on a channel

**Description:** Gets a channel

```
[DMDReader_ErrorCode, file_handle, channel_handle, ~] = calllib('dmdlib',  
'DMDReader_GetChannels', file_handle, 0, idx, 1, channel_handle, outNumChRet);
```

### 3.6 DMDREADER\_GETCHANNELINFORMATION

<b>Parameters:</b> channel_handle	a pointer on a channel
p_channelInfo.Value	a pointer on a channel information
<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel
channelInfo	contains the channel information

**Description:** Gets channel information

```
[DMDReader_ErrorCode, channel_handle, channelInfo] = calllib('dmdlib',  
'DMDReader_GetChannelInformation', channel_handle, p_channelInfo.Value);
```

### 3.7 DMDREADER\_GETSAMPLETYPE

<b>Parameters:</b> channel_handle	a pointer on a channel
sampleTypeData	sample type of the data of one specific channel
sampleTypeReduced	sample type of reduced data of one specific channel
<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel



DEWETRON

sampleTypeData	returns the sample type of the data
sampleTypeReduced	returns the sample type of the reduced data

**Description:** Gets the data type of one channel

```
[DMDReader_ErrorCode, channel_handle, sampleTypeData, sampleTypeReduced] =  
calllib('dmdlib', 'DMDReader_GetSampleType', channel_handle, sampleTypeData,  
sampleTypeReduced);
```

### 3.8 DMDREADER\_GETVECTORSAMPLETYPE

<b>Parameters:</b> channel_handle	a pointer on a channel
sampleTypeData	sample type of the data of one specific channel
sampleTypeReduced	sample type of reduced data of one specific channel
maxDim	maximum dimension for vector channel data

<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel (unused)
sampleTypeData	returns the sample type of the data
sampleTypeReduced	returns the sample type of the reduced data

**Description:** Gets the vector data type of one channel

```
[DMDReader_ErrorCode, ~, sampleTypeData, sampleTypeReduced, maxDim] =  
calllib('dmdlib', 'DMDReader_GetVectorSampleType', channel_handle, sampleTypeData,  
sampleTypeReduced, maxDim);
```

### 3.9 DMDREADER\_GETNUMDATASWEEPS

<b>Parameters:</b> channel_handle	a pointer on a channel
numDataSweeps	number of sweeps
<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel
numDataSweeps	returned number of sweeps

**Description:** Gets number of sweeps for one channel

```
[DMDReader_ErrorCode, channel_handle, numDataSweeps] = calllib('dmdlib',  
'DMDReader_GetNumDataSweeps', channel_handle, numDataSweeps);
```

### 3.10 DMDREADER\_GETDATASWEEPS

<b>Parameters:</b> channel_handle	a pointer on a channel
firstSweep	index of the first sweep (k of loop)
maxSweeps	maximum number of sweeps
	1 – one sweep per iteration
p_sweep.Value	a pointer on the sweep
numSweeps	number of sweeps



DEWETRON

**Return:** DMDReader\_ErrorCode      Error Code, see error description  
channel\_handle      returns the pointer on a channel  
sweep      returned array with sweep information  
numSweeps      number of sweeps (unused)

**Description:** Gets sweeps for one channel

```
[DMDReader_ErrorCode, channel_handle, sweep, ~] = calllib('dmdlib',  
'DMDReader_GetDataSweeps', channel_handle, k, 1, p_sweep.Value, numSweeps);
```

### 3.11 DMDREADER\_GETNUMREDUCEDSWEEPS

**Parameters:** channel\_handle      a pointer on a channel  
numReducedSweeps      number of sweeps  
**Return:** DMDReader\_ErrorCode      Error Code, see error description  
channel\_handle      returns the pointer on a channel  
numReducedSweeps      returned number of sweeps

**Description:** Gets number of sweeps for one channel

```
[DMDReader_ErrorCode, channel_handle, numReducedSweeps] = calllib('dmdlib',  
'DMDReader_GetNumReducedSweeps', channel_handle, numReducedSweeps);
```

### 3.12 DMDREADER\_GETREDUCEDSWEEPS

**Parameters:** channel\_handle      a pointer on a channel  
firstSweep      index of the first sweep (k of loop)  
maxSweeps      maximum number of sweeps  
1 – one sweep per iteration  
p\_sweep.Value      a pointer on the sweep  
numRSweeps      number of sweeps  
**Return:** DMDReader\_ErrorCode      Error Code, see error description  
channel\_handle      returns the pointer on a channel  
sweep      returned array with sweep information  
numRSweeps      number of reduced sweeps (unused)

**Description:** Gets reduced sweeps for one channel

```
[DMDReader_ErrorCode, channel_handle, sweep, ~] = calllib('dmdlib',  
'DMDReader_GetReducedSweeps', channel_handle, j, 1, p_sweep.Value, numRSweeps);
```

### 3.13 DMDREADER\_GETSAMPLESANDTS\_SCALEDVALUE\_SECONDS

**Parameters:** channel\_handle      a pointer on a channel  
firstSample      index of the first sample index (including)  
block      block to be read out



DEWETRON

blockData	block of data
blockTS	block of timestamps
numValidSamples	number of valid samples, written to the output array
nextSample	index of the sample just beyond the last sample that was returned by the function. Used as firstSample parameter to get the next block of samples

<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel (unused)
blockData	block of data of the size of block
blockTS	block of timestamps of the size of block
numValidSamples	number of valid samples, written to the output array
nextSample	index of the sample just beyond the last sample that was returned by the function

**Description:** Reads in data of type analog and counter for one channel with timestamp; block-wise reading with blocks of size 1e5

```
[DMDReader_ErrorCode, ~, blockData, blockTS, numValidSamples, nextSample] =  
calllib('dmdlib', 'DMDReader_GetSamplesAndTS_ScaledValue_Seconds', channel_handle,  
firstSample, block, blockData, blockTS, numValidSamples, nextSample);
```

### 3.14 DMDREADER\_GETSAMPLESWITHTS\_REDUCEDVALUE\_SECONDS

<b>Parameters:</b> channel_handle	a pointer on a channel
firstSampleIndex	index of the first sample index (including)
maxSamples	maximum samples to read out
reducedSampleTS_Ptr.Value	a pointer to the reduced sample timestamp value
numValidSamples	number of valid samples, written to the output array
nextSample	index of the sample just beyond the last sample that was returned by the function. Used as firstSample parameter to get the next block of samples

<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel
data_temp	temporary array with data information
numValidSamples	number of valid samples, written to the output array
nextSample	index of the sample just beyond the last sample that was returned by the function

**Description:** Reads in reduced data for one channel with timestamp

```
[DMDReader_ErrorCode, channel_handle, data_temp, numValidSamples, nextSample] =  
calllib('dmdlib', 'DMDReader_GetSamplesWithTS_ReducedValue_Seconds', channel_handle,  
k + firstSampleIndex, 1, reducedSampleTS_Ptr.Value, numValidSamples, nextSample);
```

**Note:** k: 0 to numReducedSamples





DEWETRON

### 3.15 DMDREADER\_GETSAMPLESANDTS\_DIGITALVALUE\_SECONDS

<u>Parameters:</u>	
channel_handle	a pointer on a channel
firstSample	index of the first sample index (including)
block	block to be read out
blockData	block of data
blockTS	block of timestamps
numValidSamples	number of valid samples, written to the output array
nextSample	index of the sample just beyond the last sample that was returned by the function. Used as firstSample parameter to get the next block of samples
<u>Return:</u>	Error Code, see error description
DMDReader_ErrorCode	
channel_handle	returns the pointer on a channel (unused)
blockData	block of data of the size of block
blockTS	block of timestamps of the size of block
numValidSamples	number of valid samples, written to the output array
nextSample	index of the sample just beyond the last sample that was returned by the function

Description: Reads in digital data for one channel with timestamp; block-wise reading with blocks of size 1e5

```
[DMDReader_ErrorCode, ~, blockData, blockTS, numValidSamples, nextSample] =  
calllib('dmdlib', 'DMDReader_GetSamplesAndTS_DigitalValue_Seconds', channel_handle,  
firstSample, block, blockData, blockTS, numValidSamples, nextSample);
```

### 3.16 DMDREADER\_GETSAMPLESANDTS\_SCALARVECTOR\_SECONDS

<u>Parameters:</u>	
channel_handle	a pointer on a channel
firstSample	index of the first sample index (including)
block	block to be read out
maxDim	maximum dimension of data array per timestamp
blockData	block of data
blockTS	block of timestamps
sampleDimensions_out	number of dimensions for each sample
numValidTS	number of valid timestamps
nextSample	index of the sample just beyond the last sample that was returned by the function. Used as firstSample parameter to get the next block of samples
<u>Return:</u>	Error Code, see error description
DMDReader_ErrorCode	
channel_handle	returns the pointer on a channel (unused)
blockData	block of data of the size of block
blockTS	block of timestamps of the size of block
sampleDimensions_out	number of dimensions for each sample (unused)
numValidTS	number of valid timestamps
nextSample	index of the sample just beyond the last sample that was returned by the function



DEWETRON

**Description:** Reads in scalar vector data for one channel with timestamp; block-wise reading with blocks of size 1e5

```
[DMDReader_ErrorCode, channel_handle, blockData, blockTS, ~, numValidTS, nextSample]
= calllib('dmdlib', 'DMDReader_GetSamplesAndTS_ScalarVector_Seconds', channel_handle,
firstSample, block, maxDim, blockData, blockTS, sampleDimensions_out, numValidTS,
nextSample);
```

### 3.17 DMDREADER\_GETSAMPLESANDTS\_COMPLEXVECTOR\_SECONDS

<b>Parameters:</b> channel_handle	a pointer on a channel
firstSample	index of the first sample index (including)
block	block to be read out
maxDim	maximum dimension of data array per timestamp
blockData	block of data
blockTS	block of timestamps
sampleDimensions_out	number of dimensions for each sample
numValidTS	number of valid timestamps
nextSample	index of the sample just beyond the last sample that was returned by the function. Used as firstSample parameter to get the next block of samples
<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
channel_handle	returns the pointer on a channel (unused)
blockData	block of data of the size of block
blockTS	block of timestamps of the size of block
sampleDimensions_out	number of dimensions for each sample (unused)
numValidTS	number of valid timestamps
nextSample	index of the sample just beyond the last sample that was returned by the function

**Description:** Reads in complex vector data for one channel with timestamp; block-wise reading with blocks of size 1e5

```
[DMDReader_ErrorCode, channel_handle, blockData, blockTS, ~, numValidTS, nextSample]
= calllib('dmdlib', 'DMDReader_GetSamplesAndTS_ComplexVector_Seconds',
channel_handle, firstSample, block, maxDim, blockData, blockTS, sampleDimensions_out,
numValidTS, nextSample);
```

### 3.18 DMDREADER\_GETNUMMARKERS

<b>Parameters:</b> file_handle	a pointer on a file
numMarkers	number of markers
<b>Return:</b> DMDReader_ErrorCode	Error Code, see error description
file_handle	returns the pointer on a file
numMarkers	returned number of markers

**Description:** Gets number of markers for one channel



DEWETRON

```
[DMDReader_ErrorCode, file_handle, numMarkers] = calllib('dmdlib',  
'DMDReader_GetNumMarkers', file_handle, numMarkers);
```

### 3.19 DMDREADER\_GETMARKERS

**Parameters:** file\_handle                      a pointer on a file  
                 firstSweep                    index of the first marker (j (numMarkers) of loop)  
                 maxMarkers                   maximum number of markers that can be stored  
                                                      1 – one marker per iteration  
                 markerEventPtr.Value           a pointer on the marker  
                 numMarkers2                   number of markers

**Return:** DMDReader\_ErrorCode              Error Code, see error description  
                 file\_handle                    returns the pointer on a file  
                 markerEvent                   returned marker event  
                 numMarker2                    returned number of markers

**Description:** Gets marker events for one channel

```
[DMDReader_ErrorCode, file_handle, markerEvent, numMarkers2] = calllib('dmdlib',  
'DMDReader_GetMarkers', file_handle, j, 1, markerEventPtr.Value, numMarkers2);
```

### 3.20 DMDREADER\_GETNUMHEADERFIELDS

**Parameters:** file\_handle                      a pointer on a file  
                 numHeaderFields                number of header fields

**Return:** DMDReader\_ErrorCode              Error Code, see error description  
                 file\_handle                    returns the pointer on a file  
                 numHeaderFields                returned number of header fields

**Description:** Gets number of header fields for one channel

```
[DMDReader_ErrorCode, file_handle, numHeaderFields] = calllib('dmdlib',  
'DMDReader_GetNumHeaderFields', file_handle, numHeaderFields);
```

### 3.21 DMDREADER\_GETHEADERFIELDS

**Parameters:** file\_handle                      a pointer on a file  
                 firstSweep                    index of the first marker (i (numHeaderFields) of loop)  
                 maxFields                     maximum number of fields that can be stored  
                                                      1 – one field per iteration  
                 headerfieldPtr.Value           a pointer on the header field  
                 numFields                     number of header fields

**Return:** DMDReader\_ErrorCode              Error Code, see error description  
                 file\_handle                    returns the pointer on a file  
                 headerfield                    returned header field



DEWETRON

numFields

returned number of header fields

**Description:** Gets header fields events for one channel

```
[DMDReader_ErrorCode, file_handle, headerfield, numFields] = calllib('dmdlib',  
'DMDReader_GetHeaderFields', file_handle, i, 1, headerfieldPtr.Value, numFields);
```

### 3.22 DMDREADER\_CLOSEFILE

**Parameters:** file\_handle

a pointer on a data file

**Return:** DMDReader\_ErrorCode

Error Code, see error description

file\_handle

returns the pointer on a data file (unused)

**Description:** Closes the data file and frees all library-managed resources (handles for channels...) for this dmd file

```
[DMDReader_ErrorCode, ~] = calllib('dmdlib', 'DMDReader_CloseFile', file_handle);
```



DEWETRON

## 4 EXAMPLE USAGE OF DMD-READER

```
dmdReader.m
1 % DMD (DEWETRON Measurement Data) MATLAB import prototype
2 % Copyright (c) Dewetron GmbH 2021
3 % R4.0 for dmd_reader_api 4.0 (Interface V1.1)
4 %*****
5 % Reduce Matlab usage of RAM:
6 % Home-Preferences(Environment)-Matlab-Workspace-Matlab array size limit
7 %*****
8 % Plot examples
9 % At the end of the script you can find plot examples, including an example
10 % on plotting the UTC time on your x-axis
11 %*****
12 % Please refer to the manual to adjust the script accordingly before usage
13
14 clear;
15 clc;
16
17 %% Set filename and path
18 % Data files have to be stored in a folder named 'DataFile'
19
20 filename = 'DMD20_Marker&Header_Sow_Fast_Sweeps.dmd';
21
22 path = [pwd '\'];
23 dataPath = [path 'DataFile\'];
24 funcPath = [path 'functions'];
25 dllPath = [path 'dll\'];
26 hPath = [path 'include\'];
27 addpath(funcPath);
28
```

Dll: (*dmd\_reader\_api\_x64.dll*)

- Set the filename and path if you are not in the same folder, otherwise *pwd* identifies the current folder.
- Define if all or only specific channels should be read in.
- Define if markers and header data should be read in.
- Load dll and initialize to dynamic link library.

```
loadlibrary([dllPath, DMDReaderFile, '.dll'], [hPath, 'dmd_reader_api_special.h'],  
'alias', 'dmdlib');
```

```
% initialize with interface version
```

```
interfaceVersion = [1,1];  
DMDReader_ErrorCode = calllib('dmdlib', 'DMDReader_Initialize', interfaceVersion(1),  
interfaceVersion(2));
```

- Open Dewetron data file (\*.dmd) specified in the beginning.

```
[DMDReader_ErrorCode, ~, file_handle] = calllib('dmdlib', 'DMDReader_OpenFile',  
fileFullName, file_handle);
```

- Get number of channels.



DEWETRON

```
[DMDReader_ErrorCode, file_handle, numChannels] = calllib('dmdlib',  
'DMDReader_GetNumChannels', file_handle, 0, numChannels); %0 - get all available  
channels
```

- Get channel information.

```
[DMDReader_ErrorCode, channel_handle, channelInfo] = calllib('dmdlib',  
'DMDReader_GetChannelInformation', channel_handle, p_channelInfo.Value);
```

- Get sweeps.

```
[infoSweep, numDataSweeps] = readSweep(1, channel_handle, numChannels, idx);  
[DMDReader_ErrorCode, channel_handle, numDataSweeps] = calllib('dmdlib',  
'DMDReader_GetNumDataSweeps', channel_handle, numDataSweeps);  
  
[DMDReader_ErrorCode, channel_handle, sweep, ~] = calllib('dmdlib',  
'DMDReader_GetDataSweeps', channel_handle, k, 1, p_sweep.Value, numSweeps);
```

- Read scaled values and timestamps.

```
data_temp = readScaledValues(firstSampleIndex, numSamples, channel_handle);  
[DMDReader_ErrorCode, ~, blockData, blockTS, numValidSamples, nextSample] =  
calllib('dmdlib', 'DMDReader_GetSamplesAndTS_ScaledValue_Seconds', channel_handle,  
firstSample, block, blockData, blockTS, numValidSamples, nextSample);
```

- Close the open dmd file.

```
[DMDReader_ErrorCode, ~] = calllib('dmdlib', 'DMDReader_CloseFile', file_handle);
```

- Disposes the dmd reader library and all associated global variables.

```
DMDReader_ErrorCode = calllib('dmdlib', 'DMDReader_Dispose');
```

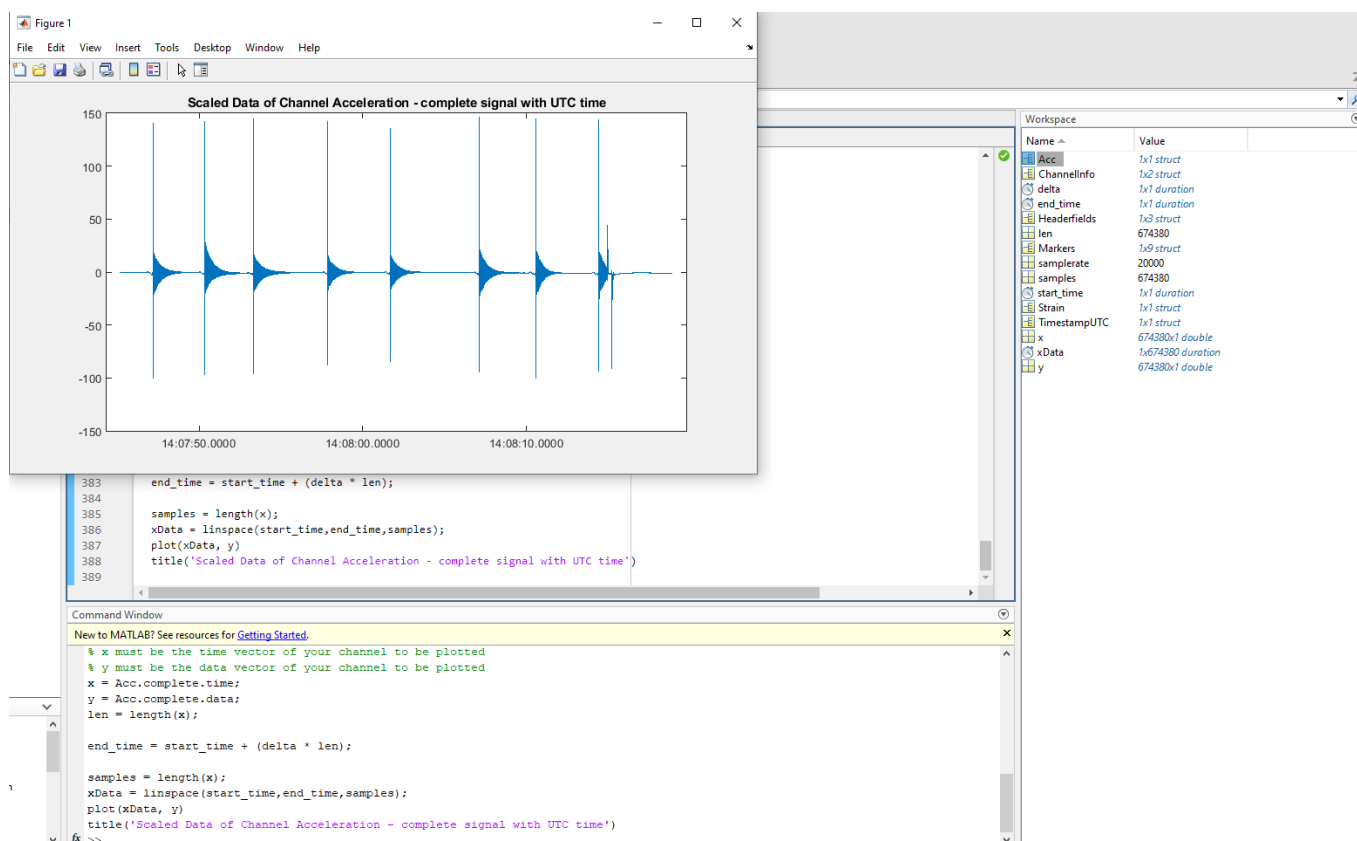
- Unload dll.

```
unloadlibrary('dmdlib');
```



DEWETRON

## 4.1 EXAMPLE SCREEN IN MATLAB AFTER IMPORT AND PLOTS



Each channel consists of a struct, containing fields with its sweeps and one field with the composed data:

Acc	
1x1 struct with 4 fields	
Field	Value
complete	1x1 struct
sweep1	1x1 struct
sweep2	1x1 struct
sweep3	1x1 struct

Each sweep contains an array with time and data information:

Acc	
Acc.sweep1	
Field	Value
time	216321x1 double
data	216321x1 double



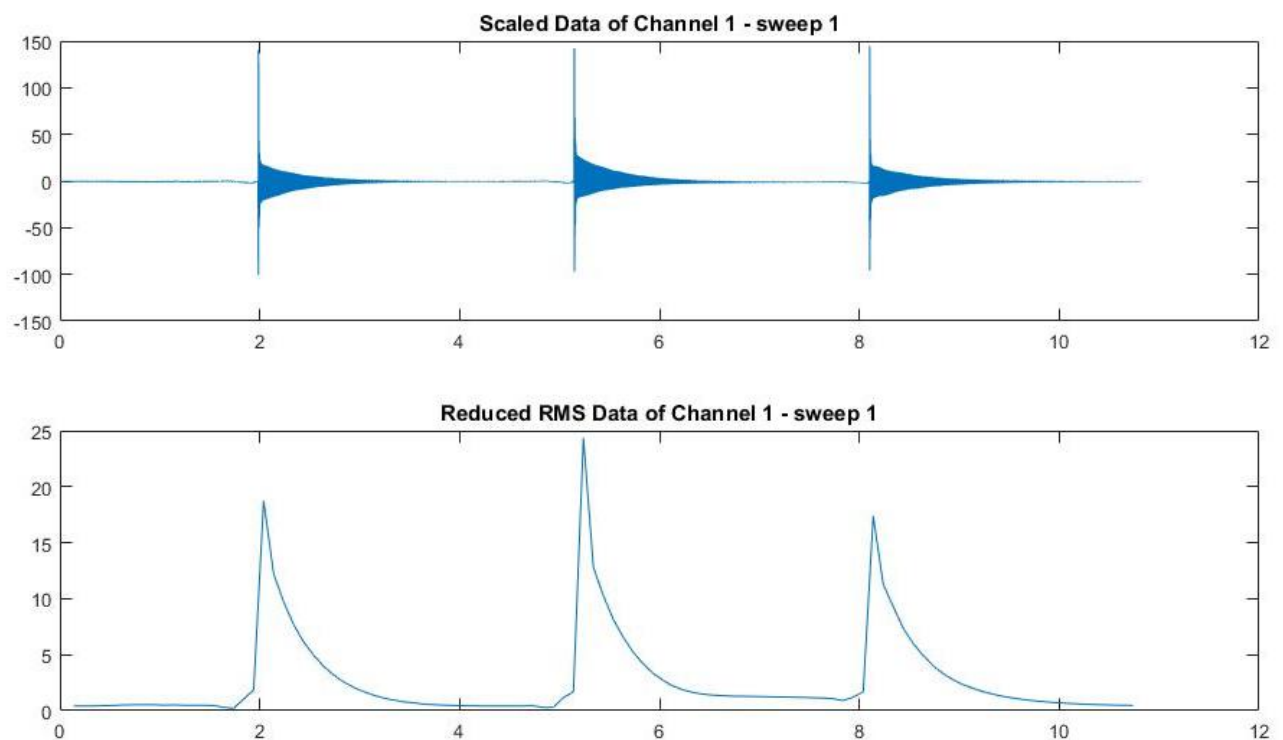
DEWETRON

#### 4.1.1 PLOT EXAMPLE 1

```
subplot(2,1,1)
plot(Channel1.sweep1.time, Channel1.sweep1.data)
title('Scaled Data of Channel 1')

subplot(2,1,2)
plot(Channel1_Reduced.sweep1.time, Channel1_Reduced.sweep1.rms)
title('Reduced RMS Data of Channel 1')
```

Note: to plot the reduced data, the parameter `readReducedData` must be set to 1.







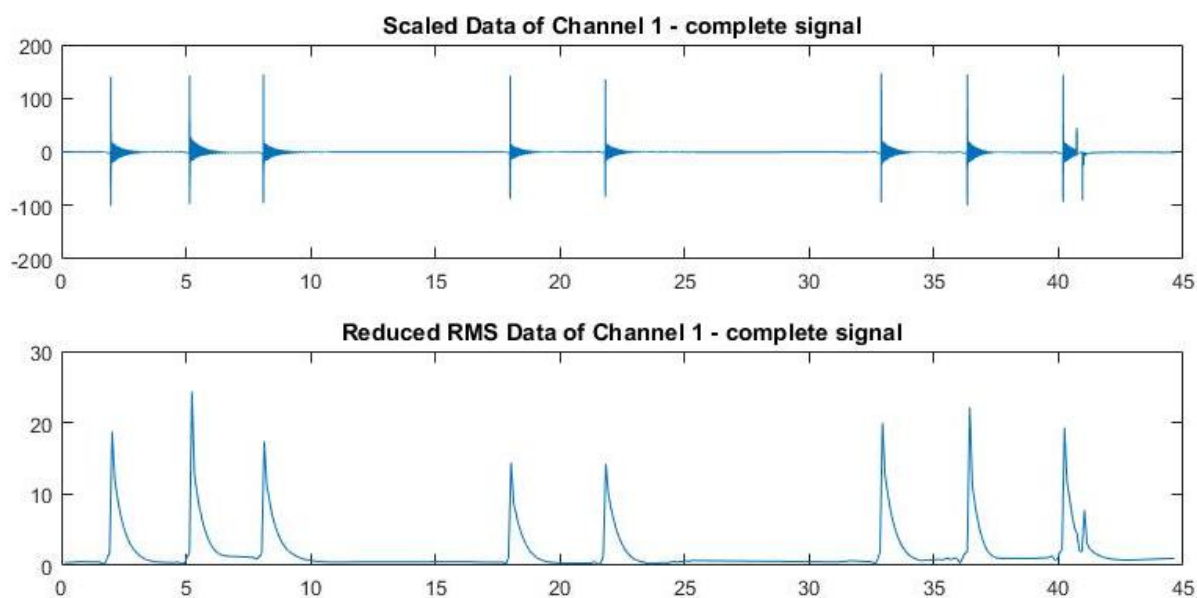
DEWETRON

#### 4.1.2 PLOT EXAMPLE 2

```
subplot(2,1,1)
plot(Channel1.complete.time, Channel1.complete.data)
title('Scaled Data of Channel 1 - complete signal')

subplot(2,1,2)
plot(Channel1_Reduced.complete.time, Channel1_Reduced.complete.rms)
title('Reduced RMS Data of Channel 1 - complete signal')
```

Note: to plot the reduced data, the parameter `readReducedData` must be set to 1.





DEWETRON

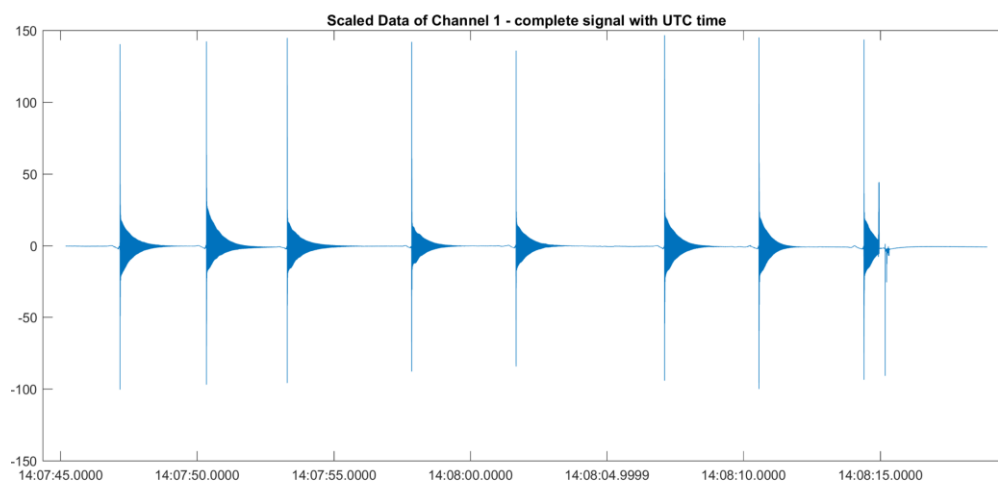
### 4.1.3 PLOT EXAMPLE 3

This example shows how to plot the data with the UTC time on the x-axis.

```
% Plotting data with UTC time on the x axis
start_time = seconds(TimestampUTC.timeOfDay);
start_time.Format = 'hh:mm:ss.SSSS';
samplerate = ChannelInfo(1).channels.sampleRate;
delta = seconds(1 / samplerate);

% x must be the time vector of your channel to be plotted
% y must be the data vector of your channel to be plotted
x = Acc.complete.time;
y = Acc.complete.data;
len = length(x);

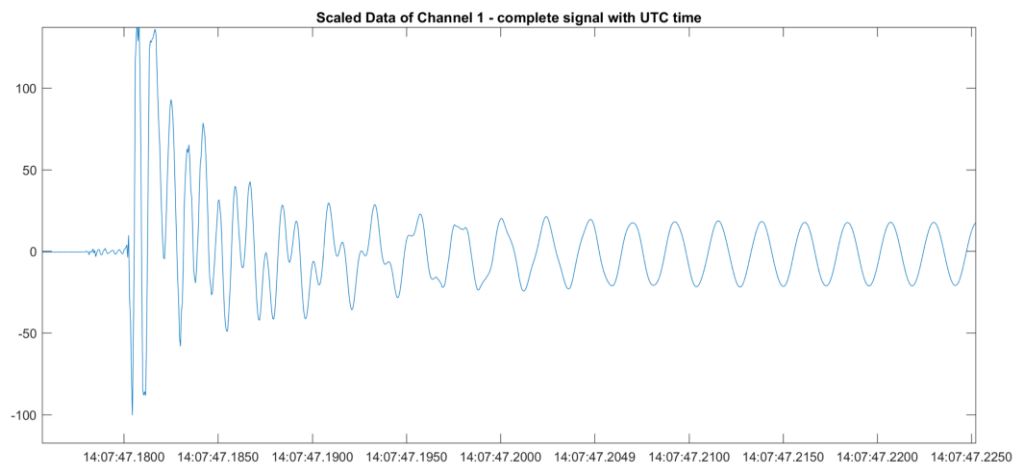
end_time = start_time + (delta * len);
samples = length(x);
xData = linspace(start_time,end_time,samples);
plot(xData, y)
```





DEWETRON

When zooming in, the resolution of the time axis also increases.





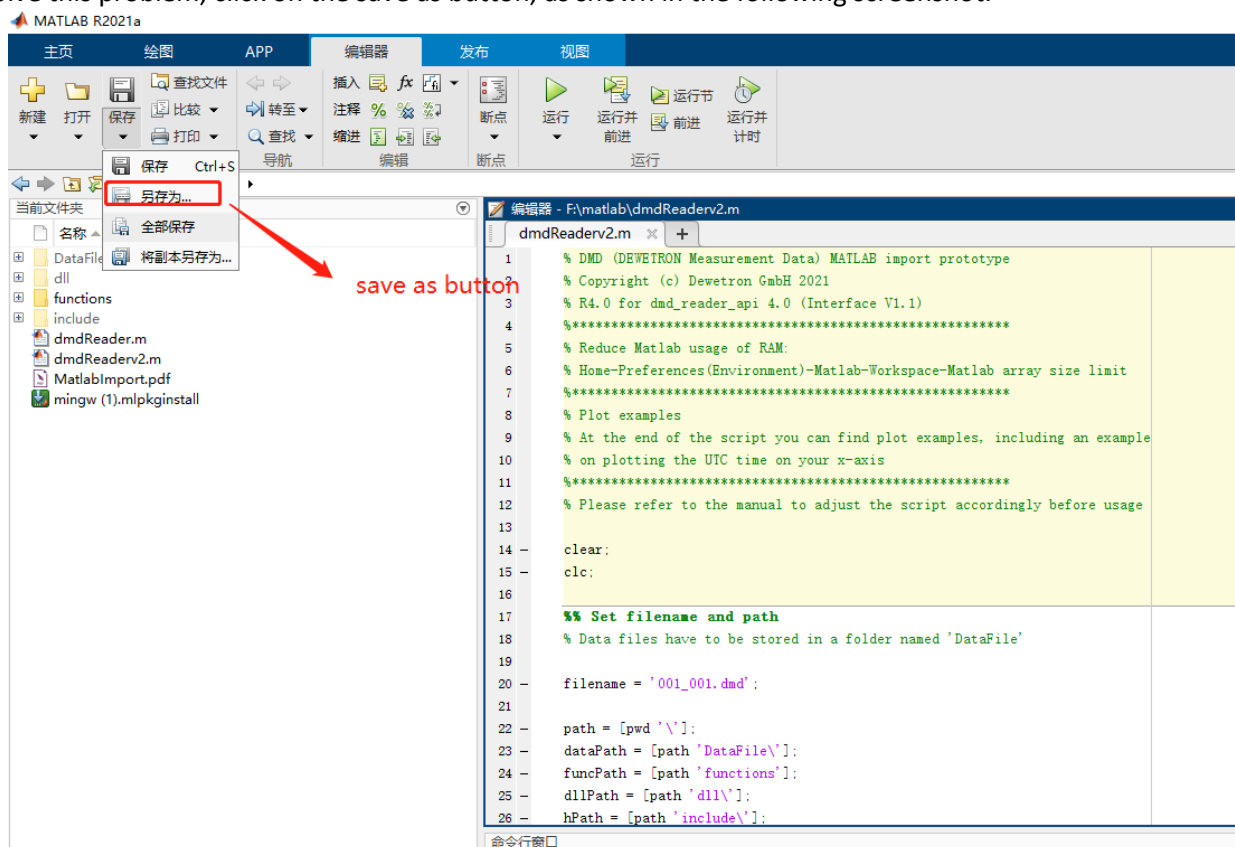
DEWETRON

## 5 TROUBLESHOOTING

When using a Chinese Window version the following error message could show up when first using the script:



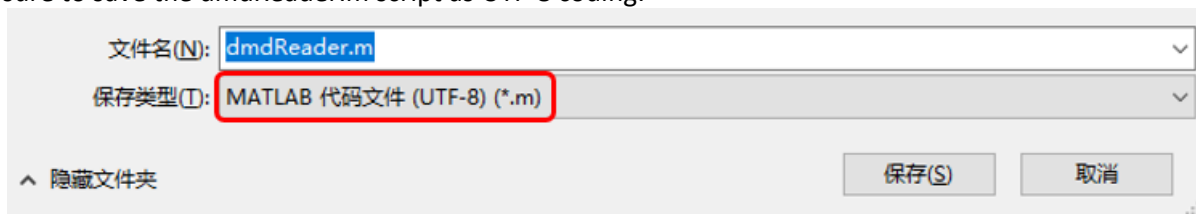
To solve this problem, click on the save as button, as shown in the following screenshot:





DEWETRON

Make sure to save the dmdReader.m script as UTF-8 coding:



These steps should solve this problem. If you still need help, please contact our support team.