General Time Series Data Format

The General Time Series Data Format is a binary hdf5 data format for storing time series data.

Mads M Pedersen, mmpe@dtu.dk, DTU-Wind Energy (AED)

Features:

- Single file
- Compact data representation (default: 2 byte pr. data value)
- Optional data type
- Precise time representation (including absolute times)
- Additional data blocks can be appended continuously
- Optional specification of name and description of dataset
- Optional specification of name, unit and description of attributes
- NaN support

File contents

Name in file	Hdf5 type	Description
type	Attribute	"General Time Series Data Format"
[name]	Attribute	Dataset name
[description]	Attribute	Dataset description
[attribute_names]	Dataset (no_attributes x 1)	Attribute names
[attribute_units]	Dataset (no_attributes x 1)	Attribute units
[attribute_descriptions]	Dataset(no_attributes x 1)	Attribute descriptions
no_blocks	Attribute	Number of blocks in file
block0000	Group	Data block group
[Block0001]	Group	Data block group
…		
[blockxxxx]	Group	Data block group

Block contents

blockxxxx/data	Dataset (no_observations x no_attributes)	Data values may be compressed using gain and offset
[blockxxxx/time]	Dataset (no_observations x 1)	Absolute or relative time (may be compressed by time_step and time_start)
[blockxxxx/time_step]	Attribute	Real time (e.g. seconds) of one time unit
[blockxxxx/time_start]	Attribute	Absolute or relative time start
[blockxxxx/gains]	Dataset (no_observations x 1)	Data scale factors
[blockxxxx/offsets]	Dataset (no_observations x 1)	Data offsets

How to save

Required parameters
Filename
Data (no_obsercations x no_attributes)
Data type (default uint16)

Optional parameters	**Check**
Dataset name	
Dataset description	
Attribute names	Len==no_attributes
Attribute units	Len==no_attributes
Attribute descriptions	Len==no_attributes
Absolute or relative time	Len==no_observations
Time step i.e. real time of one time unit	
Time start	

Procedure

```
Create hdf5 file
save type, value="General Time Series Data Format"
create group "block0000"
if Data type is integer type then
            offsets = ColumnSums(Data) #Ignore NaN
            data = Data – offsets
            gains = ColumnMax(data) / (MaxInt(Data type)-1) #Ignore NaN
            data = data / gains # where gains > 0, Ignore NaN
            convert data to dtype
            set data to MaxInt(Data Type) where data==NaN
            save data, gains and offsets in block0000-group
else
            convert data to dtype
            save data in block0000-group
end if
check and save present optional parameters
```

How to append blocks

Required parameters
Filename
Data (no_obsercations x no_attributes)

Optional parameters	Check
Absolute or relative time	Len==no_observations
Time step i.e. real time of one time unit	
Time start	

Procedure

```
Open hdf5 file for append
Check lcase(file.type)="general time series data format"
blocknr = file.no_blocks
file.no_blocks = blocknr+1
create group "blockxxxx", e.g. "block0004"
dtype = file.block0000.data.dtype
if dtype is integer type then
            offsets = ColumnSums(Data) #Ignore NaN
            data = Data – offsets
            gains = ColumnMax(data) / (MaxInt(Data type)-1) #Ignore NaN
            data = data / gains # where gains > 0, Ignore NaN
            convert data to dtype
            set data to MaxInt(Data Type) where data==NaN
            save data, gains and offsets in blockxxxx-group
else
            convert data to dtype
            save data in blockxxxx-group
end if
check and save present optional parameters in blockxxxx
```

How to load

Default values

Name in file	Default value if not present
type	Required!!!
data	Required!!!
name	<filename>
description	""
attribute_names	None
attribute_units	None
attribute_descriptions	None
time	0..no_observations-1
time_step	1
time_start	0
gains	1
offsets	0

```
Read values from file or defaults values
Check lcase(type) == "general time series data format"
data = []
time = []
for i = 0 to file.no_blocks
            block = file.blockxxxx
            if block.dtype is integer then
                        set block = NaN where block==MaxInt(block.dtype)
            end if
            block_data = block.data * block.gains + block.offset
            data.append(block_data)
            block.time = block.time * block.time_step + block.time_start
            time.append(block_time)
return time, data, <optional values>
```