# 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](mailto: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>