A Guide to TITANIC Data Products

Andrew Clifton

March 18, 2016

Contents

1 Introduction **3**

1.1 Met tower data . . . 3

1.2 Version . . . 3

1.3 How to use this document . . . 3

2 High-frequency time-series data **5**

2.1 What data are measured by the towers? . . . 5

2.2 MATLAB 20-Hz files . . . 9

2.2.1 Loading the data file . . . 10

2.2.2 Data structure . . . 10

2.2.3 Metadata structure . . . 11

2.3 ASCII (text) time series file . . . 13

2.3.1 Loading the data file . . . 13

2.3.2 Data structure . . . 13

3 10-minute summary MATLAB data files **14**

3.1 Variable Structure . . . 14

3.2 Metadata . . . 15

3.2.1 Data file records . . . 15

3.3 Instrument data . . . 15

3.3.1 Channel means and standard deviations . . . 15

3.4 Cup wind speed and turbulence intensity . . . 15

3.5 Sonic Anemometer data . . . 15

3.5.1 Horizontal wind speed (sonic anemometer) . . . 15

3.5.2 Cup-equivalent wind speed (sonic anemometer) . . . 16

3.5.3 Cup-equivalent turbulence intensity (sonic anemometer) . . . 16

3.5.4 Total wind speed (sonic anemometer) . . . 16

3.5.5 Flow angle (sonic anemometer) . . . 16

3.5.6 Advection wind speed (sonic anemometer) . . . 16

3.5.7 Standard deviations of velocities and temperatures (sonic anemometer) . . . 16

3.5.8 Friction velocity (sonic anemometer) . . . 17
3.5.9 Convective temperature scale (sonic anemometer) . . . 17
3.5.10 Turbulent kinetic energy (sonic anemometer) . . . 17
3.5.11 Coherent TKE (sonic anemometers) . . . 17
3.5.12 integral length scales (sonic anemometers) . . . 17
3.5.13 Structure functions of velocity and temperature (sonic anemometers) . . . 18
3.5.14 Dissipation rate (sonic anemometer) . . . 18
3.6 Derived Data . . . 18
3.6.1 Wind direction (cups and vanes) . . . 18
3.6.2 Power law velocity profile exponent (cups) . . . 18
3.6.3 Log law friction velocity and roughness length . . . 18
3.6.4 Wind veer (cups) . . . 19
3.6.5 Rain . . . 19
3.6.6 Air temperature . . . 19
3.6.7 Relative humidity . . . 19
3.6.8 Air pressure . . . 19
3.6.9 Potential temperature . . . 19
3.6.10 Virtual potential temperature . . . 19
3.6.11 Gradient Richardson number . . . 20
3.6.12 Speed Gradient Richardson number . . . 20
3.6.13 Brunt-Vaisala Frequency . . . 20
3.6.14 Heat flux (sonic anemometers) . . . 20
3.6.15 Monin-Obukhov length . . . 20

4 Low-frequency (10-minute) time-series data files 22
4.1 MATLAB files . . . 22
4.1.1 Example: checking the version of the data . . . 22
4.1.2 Example: plotting the time series of wind speed . . . 23
4.1.3 Example: plotting only good data . . . 23
4.1.4 Example: finding a subset of data . . . 23
4.2 ASCII files . . . 25

List of Tables

2.1 Measurement devices 5
2.2 M4 instrumentation 6
2.3 M5 instrumentation 6
2.4 M4 channels 7
2.5 M5 channels 8

1 Introduction

This document is an unofficial companion to the NREL Turbine Inflow and Turbulence Analysis Code (TITANIC), which can be found at https://github.com/NREL/TITANIC.

The purpose of this document is to help users of data created by TITANIC navigate that data and perform useful operations on the data. This document is not supported, and is not guaranteed to be current.

This document uses the example of the NREL M4 and M4 towers from 2011 to 2016. These towers are described in **?**.

1.1 Met tower data

Meteorological towers generate 2 main types of data that are differentiated by the measurement frequency. These are:

- 20 Hz data, which is the frequency with which measurements are made by the tower instruments.
- 10-minute data, which is the period over which the 20-Hz data are averaged to quantify atmospheric conditions.

The data flows can be visualized using the data flow diagrams in Figure 1.1.

1.2 Version

This document was last updated on March 18, 2016. It documents data that are created using the tower software version 1.21 and higher, released on January 19, 2013.

1.3 How to use this document

Assume that you have some data from the National Wind technology Center's (NWTC) 135-m tall meteorological towers, and want to know how to interpret it:

- if it is high-frequency MATLAB[1] data (file extension *.mat, containing lots of variables in a single file), see Section 2.2.
- if it is high-frequency ASCII text data (file extension *.txt, text headers followed by 3 header rows, 12,000 data rows, around 100 columns), see Section 2.3.
- if it is a MATLAB file giving averaged data for a single, ten-minute interval, see Chapter 3.
- if it is a MATLAB file giving averaged data for lots of ten-minute intervals, see Section 4.1.

[1] MATLAB is a registered trademark of The MathWorks, Inc. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Figure 1.1: Data Flow Diagrams for the meteorological tower data.

- if it is an ASCII text file giving data for lots of ten-minute intervals (2 header rows, 1 data row per 10 minutes, 400 or more columns), see Section 4.2.

This PDF document is also searchable.

2 High-frequency time-series data

High-frequency (20 Hz) time-series data are made available in two forms:

1. MATLAB structures, and
2. ASCII text files

2.1 What data are measured by the towers?

Data files for the two towers at the National Wind Technology Center (M4 and M5) have different contents. As described in **?**, the two towers have slightly different instrumentation, which is reflected in the variable names. The devices used to measure different parameters are listed in Table 2.1. Instrumentation on the different towers is listed in Tables 2.2 (M4) and 2.3 (M5). Channel names are listed in Tables 2.4 and 2.5.

Table 2.1: Devices used to measure atmospheric properties and mast behavior on the M4 and M5 masts. Accuracy data are from the manufacturer's specification sheets unless otherwise stated.

Parameter	Description	Device	Range	Accuracy
WS	Wind speed	Met One SS-201 Cup anem.	0 to 90 m s^{-1}	0.5 m s^{-1} or 2%
$WS(1)$	Wind speed (class one)	Thies 4.3351.10.0000	0 to 75 m s^{-1}	
WD	Wind Direction	Met One SD-201 Vane	0 to 360°	3.6°
T	Air temperature	Met One T-200A platinum RTD	± 50°	
T_{dp}	Dew point temperature	Therm-x 9400ASTD	± 50 °C	
ΔT	Differential temperature	Met One T-200A	-4.44 °C to +6.66 °C	
u_x, u_y, u_z	Wind components	ATI 'K' Type sonic anem.	± 30 m s^{-1}	0.01 m s^{-1}
accn	Boom triaxial acceleration	Summit 34201A	± 2.4g (all axes)	
T_s	Sonic temperature	ATI 'K' Type sonic anem.	-50 °C to +60 °C	0.1°C
P	Barometric pressure	AIR AB-2AX	740 to 1000 mBar	
Precip	Precipitation	Vaisala DRD11A	0 (heavy) to 3 (dry)	None given

Table 2.2: Heights of instrumentation on the M4 tower. Parameters are described in more detail in Table 2.1.

Height	Parameter	Boom
134	WD, WS, T_{dp}, ΔT	Short
131	u_x, u_y, u_z, T_s, accn	Long
100	u_x, u_y, u_z, T_s, accn	Long
88	WD, WS, T, T_{dp}, ΔT	Short
80	$WS(1)$	Short
76	u_x, u_y, u_z, T_s, accn	Long
50	u_x, u_y, u_z, T_s, accn	Long
30	u_x, u_y, u_z, T_s, accn	Long
26	WD, WS, T, T_{dp}, ΔT	Short
15	u_x, u_y, u_z, T_s, accn	Long
10	WD, WS	Short
3	WD, WS, T, T_{dp}	Short
3	P, precip	N/A

Table 2.3: Heights of instrumentation on the M5 tower. Parameters are described in more detail in Table 2.1.

Height	Parameter	Boom
130	$WS(1)$	Short
122	WD, WS, T_{dp}, ΔT	Short
119	Sonic u_x, u_y, u_z, T_s, accn	Long
105	$WS(1)$	Short
100	Sonic u_x, u_y, u_z, T_s, accn	Long
92	–	Short
90	–	Short
87	WD, WS, T,T_{dp}, ΔT	Short
80	$WS(1)$	Short
74	Sonic u_x, u_y, u_z, T_s, accn	Long
61	Sonic u_x, u_y, u_z, T_s, accn	Long
55	$WS(1)$	Short
41	Sonic u_x, u_y, u_z, T_s, accn	Long
38	WD, WS, T, T_{dp}, ΔT	Short
30	$WS(1)$	Short
15	Sonic u_x, u_y, u_z, T_s, accn	Long
10	WD, WS	Short
3	WD, WS, T, T_{dp}	Short
3	P, precip	N/A

Table 2.4: M4 data channels. See also Table 2.2.

Channel	Description	Variable	height
6	Sonic x velocity	Raw_Sonic_x_131	131
7	Sonic y velocity	Raw_Sonic_y_131	131
8	Sonic z velocity	Raw_Sonic_z_131	131
9	Sonic temperature	Raw_Sonic_Temp_131	131
10	Sonic x velocity	Raw_Sonic_x_100	100
11	Sonic y velocity	Raw_Sonic_y_100	100
12	Sonic z velocity	Raw_Sonic_z_100	100
13	Sonic temperature	Raw_Sonic_Temp_100	100
14	Sonic x velocity	Raw_Sonic_x_76	76
15	Sonic y velocity	Raw_Sonic_y_76	76
16	Sonic z velocity	Raw_Sonic_z_76	76
17	Sonic temperature	Raw_Sonic_Temp_76	76
18	Sonic x velocity	Raw_Sonic_x_50	50
19	Sonic y velocity	Raw_Sonic_y_50	50
20	Sonic z velocity	Raw_Sonic_z_50	50
21	Sonic Temperature	Raw_Sonic_Temp_50	50
22	Sonic x velocity	Raw_Sonic_x_30	30
23	Sonic y velocity	Raw_Sonic_y_30	30
24	Sonic z velocity	Raw_Sonic_z_30	30
25	Sonic temperature	Raw_Sonic_temp_30	30
26	Sonic x velocity	Raw_Sonic_x_15	15
27	Sonic y velocity	Raw_Sonic_y_15	15
28	Sonic z velocity	Raw_Sonic_z_15	15
29	Sonic temperature	Raw_Sonic_Temp_15	15
30	Air temperature	Raw_Air_Temp_88m	88
31	Air temperature	Raw_Air_Temp_26m	26
32	Air temperature	Raw_Air_Temp_3m	3
33	Dewpoint temperature	Raw_Dewpt_Temp_134m	134
34	Dewpoint temperature	Raw_Dewpt_Temp_88m	88
35	Dewpoint temperature	Raw_Dewpt_Temp_26m	26
36	Dewpoint temperature	Raw_Dewpt_Temp_3m	3
37	Δ T	Raw_DeltaT_134_88m	134
38	Δ T	Raw_DeltaT_88_26m	26
39	Δ T	Raw_DeltaT_26_3m	3
40	Vane wind direction	Raw_Vane_WD_134m	134
41	Vane wind direction	Raw_Vane_WD_88m	88
42	Vane wind direction	Raw_Vane_WD_26m	26
43	Vane wind direction	Raw_Vane_WD_10m	10
44	Vane wind direction	Raw_Vane_WD_3m	3
45	Acceleration in x	Raw_Accel_x_131	131
46	Acceleration in y	Raw_Accel_y_131	131
47	Acceleration in z	Raw_Accel_z_131	131
48	Acceleration in x	Raw_Accel_x_100	100
49	Acceleration in y	Raw_Accel_y_100	100
50	Acceleration in z	Raw_Accel_z_100	100
51	Acceleration in x	Raw_Accel_x_76	76
52	Acceleration in y	Raw_Accel_y_76	76
53	Acceleration in z	Raw_Accel_z_76	76

Channel	Description	Variable	height
54	Acceleration in x	Raw_Accel_x_50	50
55	Acceleration in y	Raw_Accel_y_50	50
56	Acceleration in z	Raw_Accel_z_50	50
57	Acceleration in x	Raw_Accel_x_30	30
58	Acceleration in y	Raw_Accel_y_30	30
59	Acceleration in z	Raw_Accel_z_30	30
60	Acceleration in x	Raw_Accel_x_15	15
61	Acceleration in y	Raw_Accel_y_15	15
62	Acceleration in z	Raw_Accel_z_15	15
63	Station Pressure	Raw_Baro_Presr_3m	3
64	Precipitation intensity	Raw_PRECIP_INTEN	0
65	Cup wind speed	Raw_Cup_WS_134m	134
66	Cup wind speed	Raw_Cup_WS_88m	88
67	Cup wind speed	Raw_Cup_WS_80m	80
68	Cup wind speed	Raw_Cup_WS_26m	26
69	Cup wind speed	Raw_Cup_WS_10m	10
70	Cup wind speed	Raw_Cup_WS_3m	3

Table 2.5: M5 data channels. See also Table 2.3.

Channel	Description	Variable	height
6	Sonic x velocity	Raw_Sonic_x_119	119
7	Sonic y velocity	Raw_Sonic_y_119	119
8	Sonic z velocity	Raw_Sonic_z_119	119
9	Sonic temperature	Raw_Sonic_Temp_119	119
10	Sonic x velocity	Raw_Sonic_x_100	100
11	Sonic y velocity	Raw_Sonic_y_100	100
12	Sonic z velocity	Raw_Sonic_z_100	100
13	Sonic temperature	Raw_Sonic_Temp_100	100
14	Sonic x velocity	Raw_Sonic_x_74	74
15	Sonic y velocity	Raw_Sonic_y_74	74
16	Sonic z velocity	Raw_Sonic_z_74	74
17	Sonic temperature	Raw_Sonic_Temp_74	74
18	Sonic x velocity	Raw_Sonic_x_61	61
19	Sonic y velocity	Raw_Sonic_y_61	61
20	Sonic z velocity	Raw_Sonic_z_61	61
21	Sonic Temperature	Raw_Sonic_Temp_61	61
22	Sonic x velocity	Raw_Sonic_x_41	41
23	Sonic y velocity	Raw_Sonic_y_41	41
24	Sonic z velocity	Raw_Sonic_z_41	41
25	Sonic temperature	Raw_Sonic_temp_41	41
26	Sonic x velocity	Raw_Sonic_x_15	15
27	Sonic y velocity	Raw_Sonic_y_15	15
28	Sonic z velocity	Raw_Sonic_z_15	15
29	Sonic temperature	Raw_Sonic_Temp_15	15
30	Air temperature	Raw_Air_Temp_87m	87
31	Air temperature	Raw_Air_Temp_38m	38
32	Air temperature	Raw_Air_Temp_3m	3

Channel	Description	Variable	height
33	Dewpoint temperature	Raw_Dewpt_Temp_122m	122
34	Dewpoint temperature	Raw_Dewpt_Temp_87m	87
35	Dewpoint temperature	Raw_Dewpt_Temp_38m	38
36	Dewpoint temperature	Raw_Dewpt_Temp_3m	3
37	Δ T	Raw_DeltaT_122_87m	122
38	Δ T	Raw_DeltaT_87_38m	38
39	Δ T	Raw_DeltaT_38_3m	3
40	Vane wind direction	Raw_Vane_WD_122m	122
41	Vane wind direction	Raw_Vane_WD_87m	87
42	Vane wind direction	Raw_Vane_WD_38m	38
43	Vane wind direction	Raw_Vane_WD_10m	10
44	Vane wind direction	Raw_Vane_WD_3m	3
45	Acceleration in x	Raw_Accel_x_119	119
46	Acceleration in y	Raw_Accel_y_119	119
47	Acceleration in z	Raw_Accel_z_119	119
48	Acceleration in x	Raw_Accel_x_100	100
49	Acceleration in y	Raw_Accel_y_100	100
50	Acceleration in z	Raw_Accel_z_100	100
51	Acceleration in x	Raw_Accel_x_74	74
52	Acceleration in y	Raw_Accel_y_74	74
53	Acceleration in z	Raw_Accel_z_74	74
54	Acceleration in x	Raw_Accel_x_61	61
55	Acceleration in y	Raw_Accel_y_61	61
56	Acceleration in z	Raw_Accel_z_61	61
57	Acceleration in x	Raw_Accel_x_41	41
58	Acceleration in y	Raw_Accel_y_41	41
59	Acceleration in z	Raw_Accel_z_41	41
60	Acceleration in x	Raw_Accel_x_15	15
61	Acceleration in y	Raw_Accel_y_15	15
62	Acceleration in z	Raw_Accel_z_15	15
63	Station Pressure	Raw_Baro_Presr_3m	3
64	Precipitation intensity	Raw_PRECIP_INTEN	0
65	Cup wind speed	Raw_Cup_WS_C1_130m	130
66	Cup wind speed	Raw_Cup_WS_122m	122
67	Cup wind speed	Raw_Cup_WS_C1_105m	105
68	Cup wind speed	Raw_Cup_WS_87m	87
69	Cup wind speed	Raw_Cup_WS_C1_80m	80
70	Cup wind speed	Raw_Cup_WS_C1_55m	55
71	Cup wind speed	Raw_Cup_WS_38m	38
72	Cup wind speed	Raw_Cup_WS_C1_30m	30
73	Cup wind speed	Raw_Cup_WS_10m	10
74	Cup wind speed	Raw_Cup_WS_3m	3

2.2 MATLAB 20-Hz files

The MATLAB 20-Hz file includes all of the time-series (20 Hz) data that were written out from each of the instruments listed in Tables 2.2 (M4) and 2.3 (M5). The file includes both measurement data and metadata.

2.2.1 Loading the data file

Load the file using MATLAB's `load` function. If `myfile.mat` is the name of the file, use

```
load{myfile}
```

2.2.2 Data structure

Each channel is written out into a separate variable. To get a listing of all of the variables in the file, load the file and then use the `who` command. The following variables will be listed in the workspace:

- ***time_UTC***: the UTC time stamp of the data. This is a MATLAB serial date number, and can be converted to seconds from the start of the file using the following MATLAB code:

```
time_elapsed = (time_UTC.val-time_UTC.val(1))*60*60*24;
```

- ***<Channel_name>***: Time series data from all channels with all extreme values (i..e at full scale) removed.
- ***Sonic_ <x,y,z,Temp>_ clean_ <z>m***: Time series data from the sonics with all extreme values and spikes removed. Missing data have been replaced by linearly interpolating. These data should be used together with the time series in ***Sonic_ cleaned_ timestamp***.
- ***Sonic_ dt_ clean_ <z>m***: time elapsed since the start of the data acquisition [seconds].
- ***Sonic_ <u,v,w>_ <z>m***: data from the sonic anemometers that have been processed so that they are oriented into the prevailing wind direction during that 10-minute interval. These time series have the property that the mean lateral and vertical velocities ($\overline{v}$ and $\overline{w}$, respectively) are zero. These data should be used together with the time series in ***Sonic_ rotated_ timestamp***.
- ***Sonic_ Temp_ rotated_ <z>m***: the temperature from the sonic anemometers, at the same temporal resolution as the rotated data.

Each variable is a structure[1] containing information that help to identify the data. The structure includes:

- ***.val*** All data obtained during the 10-minute interval. For a 10-minute file from instruments that were sampled at 20 Hz, this should include approximately 12,000 samples. There may be less samples if a measurement was skipped.
- ***.label***. Text string to use as a label for charts, etc.
- ***.units***. Text string containing a LaTeX-formatted string describing the units, e.g. `'m s^{-1}'`.
- ***.height***. The height z above ground [m].

To plot the raw, cleaned, and rotated sonic anemometer temperature at 100 m a.g.l., versus the elapsed time since the start of the file, try the following Matlab code:

```
% get the time in seconds that is elapsed since the start of the file
time_elapsed = (time_UTC.val-time_UTC.val(1))*60*60*24;
```

[1] See documentation from Mathworks, e.g. http://www.mathworks.com/help/techdoc/matlab_prog/br04bw6-38.html

```
figure
plot(time_elapsed,Sonic_Temp_100.val,'ko')
hold on
plot(Sonic_dt_clean_100m.val, Sonic_Temp_clean_100m.val,'b+')
plot(Sonic_dt_clean_100m.val,Sonic_Temp_rotated_100m.val,'rx')

legend('Raw data','Cleaned data','Rotated data')
set(legend, 'location', 'best')
```

2.2.3 Metadata structure

The data file also includes a variable called ***tower***. This is a structure containing information (metadata) about the meteorological tower and the data processing. Not all fields are relevant for the end user. For completeness, the following fields are found in the ***tower*** structure:

- ***tower.config.date***: A date array of the last update to the configuration file.
- ***tower.name***: A name to use for the tower, e.g. 'M4'.
- ***tower.id*** A unique identifier for the tower, e.g. '4.4'.
- ***tower.baseheight***: The height of the tower base above sea level [m], e.g. 1845.
- ***tower.UTCoffset***: The number of hours offset between the local time zone and UTC, where local = UTC + offset, e.g. -7.
- ***tower.timezone***: a string identifying the timezone that the tower is operating in, e.g. 'MST'.
- ***tower.daqfreq***: the frequency at which the DAQ is operating [Hz], e.g. 20.
- ***tower.windowsize***: the number of expected samples per data file, e.g. 12000.
- ***tower.veldirpairs***: An $m \times 2$ array of channel numbers of wind speed and direction to use to calculate a velocity profile. Each row includes the channel number of a cup and a vane[2].
- ***tower.veldetrendingorder***: The order of detrending to use on velocity data to estimate turbulence, e.g. 0 (no detrending).
- ***tower.thermodynamics***: A structure containing information about the mean thermodynamic properties during the 10 minutes.
- ***tower.sonicrotationmethod***: A description of the technique used to rotate the sonic anemometer data into the prevailing wind, e.g. 'pitchnyaw'.
- ***tower.sonicdetrendingorder***: The order of detrending to use on sonic anemometer velocity data to estimate turbulence, e.g. 0 (no detrending).
- ***tower.sonicinterpmethod***: The method to use to fill gaps in the sonic anemometer time series, e.g. 'linear' for linear interpolation.
- ***tower.sonicpassrate***: The minimum frequency of good data that is required to calculate statistics from the sonic anemometer data. For example, 0.92 indicates that 92% of data must be good.
- ***tower.sonicrotaterate***: The minimum frequency of good data that is required for the sonic anemometer data to be rotated into the prevailing wind. For example, 0.95 indicates that 95% of data must be good.

[2] Channel numbers are listed in Table 2.4 and 2.5.

- ***tower.sonictype***: A cell array of strings that describe the sonic anemometers. These can be used to switch between different processing routines for different types of instrument.
- ***tower.sonicdespike***: A logical array that determines if a sonic anemometer time series should be despiked (true) or not (false), e.g. [1 1 1 1 1 1] means that all sonic anemometers should be despiked.
- ***tower.sonicpairs***: Channel numbers for the sonic anemometers and accelerometer. Each row includes the x, y, z velocities and temperature, and the x, y, and z acceleration.
- ***tower.shearpairs***: A cell array of instruments that should be used to calculate a shear exponent from.
- ***tower.precipsensor***: The channel number of the device that detects precipitation, e.g. 64.
- ***tower.velprofile***: A list of channels to use to display a velocity profile over.
- ***tower.tempprofile***: A list of channels to use to display a temperature profile over.
- ***tower.processing***: A structure containing information about the processing that this file has had.
- ***tower.outage***: A cell array of structures containing data about known outages on this tower.

2.3 ASCII (text) time series file

2.3.1 Loading the data file

The ASCII time series file can be opened by any program that can read ASCII text. Before opening the file, the reader is recommended to move the file to their own local file space or hard drive.

For simplicity, the following example assumes the reader has access to Microsoft Excel.

- Open Excel
- From the 'File' menu, select 'open' and find the file using the browser
- Follow the prompts to import the data as comma-delimited text

Note that alternative approaches such as right-click and 'open with', loading the file directly from the internet, and covering text to data columns, will not work.

2.3.2 Data structure

The ASCII time series file has 3 header lines and then data lines.

The header files are:

1. Variable names
2. Units
3. Measurement height [m]

The data lines are comma-separated, fixed width fields. All of the time series data from the individual channels are included, as well as the cleaned and rotated data from the sonic anemometers. The following variables are included:

- ***time (elapsed)***: time since the start of the data acquisition [seconds].
- *<**Channel_name**>*: Time series data from all channels with all extreme values (i..e at full scale) removed.
- ***Sonic_*** *<**x,y,z,Temp**>* ***_clean_*** *<**z**>**m***: Time series data from the sonics with all extreme values and spikes removed. Missing data have been replaced by linearly interpolating. These data should be used together with the time series in ***Sonic_cleaned_timestamp***.
- ***Sonic_cleaned_timestamp***: time elapsed since the start of the data acquisition [seconds].
- ***Sonic_*** *<**u,v,w**>* ***_*** *<**z**>**m***: data from the sonic anemometers that have been processed so that they are oriented into the prevailing wind direction during that 10-minute interval. These time series have the property that the mean lateral and vertical velocities ($\overline{v}$ and $\overline{w}$, respectively) are zero. These data should be used together with the time series in ***Sonic_rotated_timestamp***.
- ***Sonic_Temp_rotated_*** *<**z**>**m***: the temperature from the sonic anemometers, at the same temporal resolution as the rotated data.
- ***Sonic_rotated_timestamp***: time elapsed since the start of the data acquisition [seconds].

3 10-minute summary MATLAB data files

Summary MATLAB data files are generated automatically for every 10-minute period. These file are stored separately from the raw data. The files also have the naming convention `mmddy_HH_MM_SS_FFF.mat`. The files contain multiple structures for each of the channels, the sonic anemometer data, and the derived data.

3.1 Variable Structure

The structure of *<**variable name**>* is as follows:

- ***.val*** Mean value during the 10-minute interval, after removing data outside of manufacturer's limits.
- ***.date*** Serial date number of the start of the 10-minute interval.
- ***.label*** Text string to use as a label for charts, etc.
- ***.units*** Text string containing a LaTeX-formatted string describing the units, e.g. `'m s^{-1}'`.
- ***.height*** The height z above ground [m].
- ***.npoints*** The number of points in the 10-minute interval.
- ***.flags*** Quality-control (QC) codes:
  - QC codes indicating that data are 'flagged' (possibly bad) are in the range 1000 to 4999. Reasons for flagging channels include:
    - **1001** irregular timing. The period between measurements should be 0.05 seconds at a data acquisition rate of 20 Hz. If more than 1% of data are more than 5% from the ideal period, this QC code is set.
    - **1002, 1003** If the number of points within the manufacturer's limits or users' limits is below a threshold set in the configuration file. These threshold values are the ***range*** rate (QC code 1002) and the ***accept*** rate (QC code 1003).
    - 1004 indicates that the sonic anemometer time series contain less than 95% of the expected number of records, and thus the data have not been rotated into the mean flow for that interval.
    - **1006** if the standard deviation drops below 0.01% of the mean and so a channel is assumed to have a constant value during the measurement interval.
    - **20***nn* if a channel is flagged because it is linked with another channel that has been flagged, where *nn* is the number of the channel that was flagged.
  - QC codes indicating that channels or data have failed are greater than 5000. Reasons for marking channels as failed include:
    - **5001** if a channel is empty.
    - **5002** if all data in a channel have known 'bad' values, e.g. -999.
    - **5003** if all data in a channel are not-a-number (NaN).

* **5004** if the boom speed exceeds 0.1 m/s at any time during the 10 minute interval.
* **5005** if the channel is affected by a known outage.
* **60***nn* if a channel fails because it is linked with another channel that has failed, where *nn* is the number of the channel that failed.

For brevity, usually only the output value in ***<variable_ name>.val*** and the quality control output to ***<variable_ name>.flags*** will be described in the following sections.

3.2 Metadata

3.2.1 Data file records

- ***Data_ File_ Records*** The number of records in the data file (rows).
- ***Data_ File_ Valid_ Records*** The number of the first record at which data import fails.

3.3 Instrument data

3.3.1 Channel means and standard deviations

- ***Raw_ <Channel_ name>_ <z>m_ mean*** Mean value of all channel data within the manufacturer's limits from the 10-minute interval.
- ***Raw_ <Channel_ name>_ <z>m_ sdev*** Standard deviation of all channel data within the manufacturer's limits from the 10-minute interval.

3.4 Cup wind speed and turbulence intensity

- ***Wind_ Speed_ Cup_ <z>m.val*** Mean value of all cup wind speeds within the manufacturer's limits from the 10-minute interval.
- ***Wind_ Speed_ Cup_ <z>m.flags*** Quality flags for the wind speed are inherited from the channel data.
- ***Ti_ Cup_ <z>m.val*** Ratio of standard deviation to mean (percentage) of all cup wind speeds within the manufacturer's limits from the 10-minute interval.
- ***Ti_ Cup_ <z>m.flags*** Quality flags for the turbulence intensity are inherited from the mean wind speed at the heights at which the turbulence intensity is calculated.

3.5 Sonic Anemometer data

If any of the individual sonic velocity component channels or temperature channel are coded with the codes **1006** (low standard deviation), **5001** (empty data channel), **5002** (all bad values) or **5003** (all NaNs), these quality codes are inherited. Other quality codes are ignored.

3.5.1 Horizontal wind speed (sonic anemometer)

- ***Wind_ Speed_ Horizontal_ Sonic_ <z>m.val*** Horizontal wind speed at height z [m s^{-1}].
- ***Wind_ Speed_ Horizontal_ Sonic_ <z>m.flags*** QC flags are defined depending on the amount of data, the boom motion and individual channel status.

3.5.2 Cup-equivalent wind speed (sonic anemometer)

- ***Wind_Speed_CupEq_Sonic_<z>m.val*** Cup-equivalent wind speed at height z [m s^{-1}].
- ***Wind_Speed_CupEq_Sonic_<z>m.flags*** Quality control codes used for the cup-equivalent wind speed are inherited from the horizontal wind speed (Section 3.5.1).

3.5.3 Cup-equivalent turbulence intensity (sonic anemometer)

- ***Ti_CupEq_Sonic_<z>m.val*** The cup-equivalent turbulence intensity of the streamwise velocity component at height z [%].
- ***Ti_CupEq_Sonic_<z>m.flags*** Turbulence statistics are only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded. Other quality control codes are inherited from the horizontal wind speed (Section 3.5.1).

3.5.4 Total wind speed (sonic anemometer)

- ***Wind_Speed_Total_Sonic_<z>m.val*** Total wind speed at height z [m s^{-1}].
- ***Wind_Speed_Total_Sonic_<z>m.flags*** Quality control codes used for the total wind speed are inherited from the horizontal wind speed (Section 3.5.1).

3.5.5 Flow angle (sonic anemometer)

- ***Wind_Inflow_Angle_Sonic_<z>m.val*** Inflow angle at height z. A positive angle implies a mean vertical velocity above zero (rising flow).
- ***Wind_Inflow_Angle_Sonic_<z>m.flags*** Quality control codes used for the inflow angle are inherited from the horizontal wind speed (Section 3.5.1).

3.5.6 Advection wind speed (sonic anemometer)

- ***Wind_Speed_Advection_Sonic_<z>m.val*** Advection wind speed at height z [m s^{-1}].
- ***Wind_Speed_Advection_Sonic_<z>m.flags*** Quality control codes used for the advection wind speed are inherited from the horizontal wind speed (Section 3.5.1).

3.5.7 Standard deviations of velocities and temperatures (sonic anemometer)

- ***Sigma_u_Sonic_<z>m.val*** The standard deviation of the streamwise velocity component at height z [m s^{-1}].
- ***Sigma_v_Sonic_<z>m.val*** The standard deviation of the lateral velocity component at height z [m s^{-1}].
- ***Sigma_w_Sonic_<z>m.val*** The standard deviation of the vertical velocity component at height z [m s^{-1}].
- ***Sigma_T_Sonic_<z>m.val*** The standard deviation of the temperature at height z [K].
- ***Sigma_<u,v,w,T>_Sonic_<z>m.flags*** Turbulence statistics are only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code

1004 is recorded. Other quality control codes used are inherited from the horizontal wind speed (Section 3.5.1).

3.5.8 Friction velocity (sonic anemometer)

- ***ustar_ Sonic_ <z>m.val*** The friction velocity at height z m [m s^{-1}].
- ***ustar_ Sonic_ <z>m.flags*** Friction velocity is only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded. Other quality control codes are inherited from the horizontal wind speed (Section 3.5.1).

3.5.9 Convective temperature scale (sonic anemometer)

- ***Tstar_ Sonic_ <z>m_ mean.val*** The convective temperature scale at height z [k].

3.5.10 Turbulent kinetic energy (sonic anemometer)

- ***TKE_ Sonic_ <z>m_ mean.val*** The mean turbulent kinetic energy ($\overline{\text{TKE}}$) at height z [m^2 s^{-2}].
- ***TKE_ Sonic_ <z>m_ peak.val*** The maximum turbulent kinetic energy (TKE(t)) at height z during the 10-minute interval [m^2 s^{-2}].
- ***TKE_ Sonic_ <z>m_ <mean,peak>.flags*** Turbulence statistics are only calculated if the number of data points exceeds the target percentage given by the parameter ***tower.sonicrotate*** (Section 2.2.3). If there is not enough data, the quality control code **1004** is recorded. Other quality control codes used are inherited from the horizontal wind speed (Section 3.5.1).

3.5.11 Coherent TKE (sonic anemometers)

- ***CTKE_ Sonic_ <z>m_ mean.val*** The RMS coherent turbulent kinetic energy at height z [m^2 s^{-2}].
- ***CTKE_ Sonic_ <z>m_ peak.val*** The maximum coherent turbulent kinetic energy at height z during the 10-minute interval [m^2 s^{-2}].
- ***CTKE_ Sonic_ <z>m_ <mean,peak>.flags*** CTKE is only calculated if the number of data points exceeds the target percentage given by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded. Other quality control codes are inherited from the horizontal wind speed (Section 3.5.1).

3.5.12 integral length scales (sonic anemometers)

Several different integral length scales are defined, after **?**.

- ***L_ integral_ <u,v,w>_ Sonic_ <z>m.val*** The integral of the autocorrelation function from 0 to infiinty at height z [m].
- ***L_ zc_ <u,v,w>_ Sonic_ <z>m.val*** The integral of the autocorrelation function from 0 to the zero crossing at height z [m].
- ***L_ sz_ <u,v,w>_ Sonic_ <z>m.val*** The integral length scale from $t = 0$ to the zero-crossing time at height z [m].
- ***L_ Peak_ <u,v,w>_ Sonic_ <z>m.val*** The length scale associated with the peak of the spectrum $fS(f)$ at height z [m].

- ***L_Kaim_<u,v,w>_Sonic_<z>m.val*** The length scale associated with a fit to the spectrum assuming the spectrum has a Kaimal spectral, at height z [m].
- ***L_<integral, zc, sz, Peak, Kaim>_<u,v,w>Sonic_<z>m.flags*** Turbulence time and length scales are only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded. Other quality control codes used are inherited from the horizontal wind speed (Section 3.5.1).

3.5.13 Structure functions of velocity and temperature (sonic anemometers)

- ***CV2m_Sonic_<z>m.val*** The median value of C_{V^2} at height z [m^2 s^{-3}].
- ***CT2m_Sonic_<z>m.val*** The median value of C_{T^2} at height z [k^2 $m^{2/3}$].

The structure function parameters C_{V^2} and C_{T^2} are not quality controlled.

3.5.14 Dissipation rate (sonic anemometer)

- ***Drate_SF_Sonic_<z>m.val*** The dissipation rate ϵ at height z .
- ***Drate_SF_Sonic_<z>m.flags*** The dissipation rate is only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded. Other quality control codes used are inherited from the horizontal wind speed (Section 3.5.1).

3.6 Derived Data

3.6.1 Wind direction (cups and vanes)

- ***Wind_Direction_Vane_<z>m.val*** Mean wind direction measured by the vane during the 10 minute interval [°].
- ***Wind_Direction_Vane_<z>m_sdev.val*** Standard deviation of the wind direction measured by the vane during the 10 minute interval [°].
- ***....flags*** Quality flags for the wind speed and wind direction are inherited from the time-resolved wind speed and direction measurement channels. For example, if there are insufficient data in the wind speed time series, the mean wind speed will also be flagged as having too few data points with the QC code **1002** or **1003**.

3.6.2 Power law velocity profile exponent (cups)

- ***Wind_Shear_z2_z2m.val*** Power law exponent calculated from the mean velocities at heights $z1$ and $z2$ [-].
- ***Wind_Shear_z2_z2m.flags*** Quality flags for the wind shear are inherited from the mean wind speed at the heights over which the shear is calculated. Data area also flagged if the best fit does not converge.

3.6.3 Log law friction velocity and roughness length

- ***Friction_velocity_cup_<z1>_<z2>m.val*** Friction velocity [m s^{-1}].

- ***Friction_ velocity_ cup_ <z1>_ <z2>m.flags*** Quality codes for the friction velocity are inherited from codes for the wind speed and wind direction at the heights over which the friction velocity is calculated.
- ***Roughness_ Length_ cup_ <z1>_ <z2>m.val*** Roughness length [m].
- ***Roughness_ Length_ cup_ <z1>_ <z2>m.flags*** Quality codes for the roughness length are inherited from codes for the wind speed and wind direction at the heights over which the friction velocity is calculated.

3.6.4 Wind veer (cups)

- ***Wind_ Veer_ <z1>_ <z2>m.val*** Wind veer between heights $z1$ and $z2$. If multiple heights are defined, for example, across the rotor disk of the turbine, the largest difference is used [°].
- ***Wind_ Veer_ <z1>_ <z2>m.flags*** Quality codes for the wind veer are inherited from the mean wind direction at the heights over which the veer is calculated.

3.6.5 Rain

The channel number of the rain sensor is given in the configuration file (***tower.precipsensor*** = ***c***, see Section 2.2.3).

- ***Raw_ PRECIP_ INTEN_ mean.val*** Mean during the measurement interval.

3.6.6 Air temperature

- ***Air_ Temperature_ <z>m.val*** Air temperature at height z [°C].
- ***Air_ Temperature_ <z>m.flags*** The air temperature inherits the quality codes of the tower-base temperature measurement and the differential temperature measurements.

3.6.7 Relative humidity

- ***Relative_ Humidity_ <z>m.val*** The relative humidity at height z [%].
- ***Relative_ Humidity_ <z>m.flags*** The relative humidity inherits the quality codes of the relative and saturation vapor pressure.

3.6.8 Air pressure

- ***Air_ Pressure_ <z>m.val*** The air pressure at height z [mBar].
- ***Air_ Pressure_ <z>m.flags*** The pressure at each height inherits the quality codes of the ground pressure and pressure gradient.

3.6.9 Potential temperature

- ***Potential_ Temperature_ <z>m.val*** The potential temperature at height z.
- ***Potential_ Temperature_ <z>m.flags*** The potential temperature inherits the quality codes from the pressure profile and calculated air temperature profile.

3.6.10 Virtual potential temperature

- ***Virtual_ Potential_ Temperature_ <z>m.val*** The 10-minute mean virtual potential temperature at height z [°].
- ***Virtual_ Potential_ Temperature_ <z>m.flags*** The virtual potential temperature inherits the quality codes from the local specific humidity, air temperature and pressure.

3.6.11 Gradient Richardson number

- ***Ri_ grad_ <z1>_ <z2>m.val*** Gradient Richardson number between heights z_1 and z_2 [-].

  Where the height interval scans multiple measurement heights, all of those heights will be used in the name, e.g ***Ri_ grad_ <z1>_ <z2>_ <z3>_ <z4>m.val***.
- ***Ri_ grad_ <z1>_ <z2>m.height*** Heights z_1 and z_2 [m].
- ***Ri_ grad_ <z1>_ <z2>m.flags*** The gradient Richardson Number inherits the quality codes from the wind speed and direction, and the virtual potential temperature data that were used.

3.6.12 Speed Gradient Richardson number

- ***Ri_ WS_ <z1>_ <z2>m.val*** Speed Richardson Number between heights z_1 and z_2 [-].
- ***Ri_ WS_ <z1>_ <z2>m.height*** Heights z_1 and z_2 [m].
- ***Ri_ WS_ <z1>_ <z2>m.flags*** The Richardson number inherits quality control flags from the individual speed and direction sensors that are used in this calculation.

3.6.13 Brunt-Vaisala Frequency

- ***BruntVaisala_ <z1>_ <z2>m.val*** Brunt-Vaisala Frequency (N) between heights $z1$ and $z2$ [s^{-1}].
- ***BruntVaisala_ <z1>_ <z2>height*** Heights $z1$ and $z2$ [m].
- ***BruntVaisala_ <z1>_ <z2>m.flags*** If the speed Richardson number Ri_S exceeds ± 10, the Brunt-Vaisala frequency is set to NaN and the flag **1007** is set.

3.6.14 Heat flux (sonic anemometers)

- ***wT_ Sonic_ <z>m_ mean.val*** The mean value of $w'T'_s$ at height z [m s^{-1} K].
- ***wT_ Sonic_ <z>m_ mean.flags*** The mean of $w'T'_s$ is only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded.
- ***Heat_ flux_ Sonic_ <z>m.val*** The heat flux Q at height z [W m^{-2}].
- ***Heat_ Flux_ Sonic_ <z>m.flags*** The heat flux is only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is recorded. The heat flux requires the air density ρ and so is only calculated if there is a thermistor temperature profile. Other quality control codes are inherited from the horizontal wind speed (Section 3.5.1).

3.6.15 Monin-Obukhov length

- ***MO_Length_Sonic_<z>m.val*** The Monin-Obukhov length L at height z [m].
- ***zover_MO_Length_Sonic_<z>m.val*** The ratio $\zeta = z/L$, which is the Monin-Obukhov length, normalized by the measurement height z [-].
- ***MO_Length_Sonic_<z>m.flags*** The Monin-Obukhov length L is only calculated if the number of data points exceeds the target percentage given in the configuration file by the parameter ***tower.sonicrotate***. If there is not enough data, the quality control code **1004** is set. Other quality control codes used are inherited from the horizontal wind speed (Section 3.5.1).
- ***zover_MO_Length_Sonic_<z>m.flags*** The ratio z/L is only calculated if L is calculated. If there is not enough data to calculate L, the quality control code **1004** is set. Other quality control codes used are inherited from the horizontal wind speed (Section 3.5.1). Quality control codes can also be inherited from the virtual potential temperature.

4 Low-frequency (10-minute) time-series data files

The ten-minute summary data described in Chapter 3 can be concatenated together to create a low-frequency time-series. As with the high-frequency data, this data may be available in a MATLAB format or as ASCII.

4.1 MATLAB files

A MATLAB data file will be created that contains a single structure ***all_data***. The fields in the structure will be the same as the fields in the 10-minute summary files (see Chapter 3), but will be arrays of the data collected in each 10-minute interval. For example:

- ***all_data.Raw_<Channel_name>_<z>m_mean***
  - ***..._mean.val*** Array of mean values during each 10-minute interval, after removing data outside of manufacturer's limits.
  - ***..._mean.date***. Array of serial date numbers of the start of the 10-minute interval.
  - ***..._mean.label***. Text string to use as a label for charts, etc.
  - ***..._mean.units***. Text string containing a LaTeX-formatted string describing the units, e.g. `'m s^{-1}'`.
  - ***..._mean.height***. The height z above ground [m].
  - ***..._mean.npoints***. Array of the number of points in each 10-minute interval.
  - ***..._mean.flags***. Cell array of QC codes.

There is another important variable in this file:

- ***all_data.version***
  - ***.version.val*** Array of the code version that was used to prepare the data from this time interval.

4.1.1 Example: checking the version of the data

Task: Check which version of the data processing software were used to create the data.

Find the unique values in the ***all_data.version*** array. The values in this array should be identical:

```
>> unique(all_data.version.val)
1.21
```

4.1.2 Example: plotting the time series of wind speed

Task: plot the wind speed at 80 m above ground against time.

First, start by checking the variables in the workspace, remembering that the data is delivered as a large structure:

```
>> who
Your variables are:

M4      tend    tstart
>> M4
```

Note the name of the variable for horizontal wind speed at 80 m above ground. Assume for this example that it is ***M4.Wind_Speed_Cup_80m***. Next, plot the value versus time:

```
figure
plot(M4.Wind_Speed_Cup_80m.date, M4.Wind_Speed_Cup_80m.val,'k-')
datetick('x')
xlabel('Date')
ylabel('Wind Speed [m/s]')
```

This plot includes all data, with no filtering for quality.

4.1.3 Example: plotting only good data

Continuing from the previous plot, we need to find data where the quality-control codes indicate good data. This means that the field, ***M4.Wind_Speed_Cup_80m.flag***, should be empty. Check the cells:

```
ipass = false(size(M4.Wind_Speed_Cup_80m.val));

for i = 1:numel(ipass)
    ipass(i) = isempty(M4.Wind_Speed_Cup_80m.flags{i});
end
```

Now, use the logical array to plot a subset of the data that pass the quality tests. This should be plotted on top of the plot from the previous example:

```
hold on
plot(M4.Wind_Speed_Cup_80m.date(ipass), ...
    M4.Wind_Speed_Cup_80m.val(ipass),...
    'go')
```

4.1.4 Example: finding a subset of data

Task: find all data where the wind speed is from a certain direction.

We need to filter all data to focus on winds from a range of directions. We'll do this first by finding the indices of the wind from the direction we want:

```
WD =M4.Wind_Direction_Vane_88m_mean.val;
idir = logical((WD > 135) | (WD < 75));
isubset = idir & ipass;
```

Next, we loop through all of the variables to find the data we want.

```
M4Fields = fieldnames(M4);

for fi = 1:numel(M4Fields)
    if isfield(M4.(M4Fields{fi}),'height')
        M4Subset.(M4Fields{fi}).label = M4.(M4Fields{fi}).label;
        M4Subset.(M4Fields{fi}).units = M4.(M4Fields{fi}).units;
        M4Subset.(M4Fields{fi}).height = M4.(M4Fields{fi}).height;
        M4Subset.(M4Fields{fi}).val = M4.(M4Fields{fi}).val(isubset);
        M4Subset.(M4Fields{fi}).npoints = M4.(M4Fields{fi}).npoints(isubset);
        M4Subset.(M4Fields{fi}).date = M4.(M4Fields{fi}).date(isubset);
        M4Subset.(M4Fields{fi}).flags = M4.(M4Fields{fi}).flags{isubset};
    end
end
```

And the structure ***M4Subset*** now contains all of the data that passes our selection criteria.

4.2 ASCII files

The ASCII concatenated 10-minute summary data file has 2 header lines and then data lines.

The header files are:

1. Variable names
2. Units

The data lines are comma-separated, fixed width fields. Each row includes the values for a different 10-minute interval. Each column is a different variable. For a description of the variables that are included, see the description of the data products in Chapter 3. Quality-control codes are also used, but have been simplified in the text file compared to the MATLAB data. Quality-control codes are included in the variable, $<$***var_ name***$>$***_ QC***. The following codes are used:

- **1 = pass**
- **0 = flag**
- **-1 = fail**

Missing data are given the value -999. The reader is requested to confirm the data format (e.g. integer or n-digit precision) before filtering to remove missing data.