

# User Guide: RTDD Log Viewer

Anisa Noor Corina

Norwegian University of Science and Technology  
Department of Petroleum Engineering and Applied Geophysics  
February 9, 2017

# Preface

This user guide is created to help users to get familiar with the RTDD log viewer program and application. The developers expect that users can improve the program by reporting bugs and updating the worksheets; **List Canonical Mnemonic.xlsx** and **Unit Conversion.xlsx**. The version 1.0 of RTDD log viewer was developed by Anisa N Corina, Isak Swahn, Sigve Hovda, and Pål Skalle.

The program is also a combination with SeisLab 3.01, copyright© 2010, Eike Rietsch, copyright © 2005, Primož Cermelj, copyright © 2016, The MathWorks, Inc., copyright © 2016, MathWorks. All rights reserved.

# 1. Introduction

This document contains information regarding the RTDD Log Viewer. RTDD Log Viewer is a program containing source codes that can be compiled to run the application to read real-time drilling data (RTDD). The goal of this application is to provide a user interface enabling user to load and visualize RTDD logging data.

## 1.1 Organization of the manual

The user manual consists of four main sections: *introduction*, *system requirements*, *getting started* and *user interface basis*.

1. **Introduction** provides information of the system and its purpose in general terms.
2. **System requirements** provides a general overview of the system requirement to compile and run RTDD Log Viewer.
3. **Getting started** explains how to prepare the RTDD Log Viewer program before compiling and prepare a template for plotting the curves.
4. **RTDD log viewer application** explains the components and features of the viewer application so users getting familiar and enable to run the software quickly.
5. **Examples of plotting templates** shows some examples of well log plots based on plotting template.

In the appendix, user can find the additional information:

1. Appendix A: Parsing process
2. Appendix B: Standard format of LAS file
3. Appendix C: Standard format of CSV file

#### 4. Appendix D: Standard format of ASC file

## 2. System requirements

The required tool to compile and run this software is MATLAB. The recommended MATLAB version is R2016a, but *any compatibility issues may encountered in older version*. The system requirement of MATLAB R2016a can be found in [http://se.mathworks.com/support/sysreq/current\\_release/](http://se.mathworks.com/support/sysreq/current_release/) and for the older version in [http://se.mathworks.com/support/sysreq/previous\\_releases.html](http://se.mathworks.com/support/sysreq/previous_releases.html). The requirements are also provided in operating systems Windows, Mac, and Linux. The users are expected to have a basic knowledge of coding in MATLAB to get familiar with the program.

## 3. Getting started

### 3.1 Download the RTDD log viewer

The RTDD Log Viewer program can be downloaded in server `\\iipstudio.iiv.ntnu.no\drilldb$`. The RTDD Log Viewer directory is composed of several directories:

- `log_viewer`: Source code of RTDD log viewer.
- `examples`: Examples of log data and some templates to plot the track.

### 3.2 Compiling RTDD log viewer

The interface of RTDD Log Viewer is contained in MATLAB code `main_log.m` and MATLAB figure `main_log.fig` which located in directory `log_viewer`. The RTDD log viewer application is opened by compiling the `main_log.m` in MATLAB. To compile the RTDD log viewer:

1. Open `main_log.m`
2. Press run button or F5
3. The user interface of RTDD log viewer will appear in a new window

### 3.3 Preparing the plotting template

In order to show the log curves in the application, user is required to write source codes determining the input of plotting template. Some plotting templates are ready to be used in directory `example`. A plotting template must contain information of the log tracks and log curves which are written as input: `inputTrack` and `inputCurve`.

### 3.3.1 Writing input for inputTrack

`inputTrack` is a variable with class of `struct`<sup>1</sup>. This variable contains all the information regarding the track properties: name, plotted curves, and the scale of the axes. All these information are defined in fields of `inputTrack` structure,

`inputTrack.Name` specifies the log track to be plotted and the track name.

There is no limitation of the number of the tracks to be plotted. The input can be written as: `inputTrack.Name = {'<track #1>', '<track #2>', ...}`

`inputTrack.Curves` contains the canonical name of curves which is assigned in each particular track. The canonical name is the default name of log curves that acknowledged by the RTDD program. This is constructed due to a large number of curves mnemonic name which are varies within each service company. The list of canonical and mnemonic name are saved in a worksheet `List Canonical Mnemonic.xlsx` in directory `log_viewer`. There can be multiple curves plotted in one track up until 5 curves. The input can be written as:

`inputTrack.Curves= {'<can.name #1>', '<can.name #2>'},  
{'<can.name #1>'...}`

`inputTrack.XScale` contains the scale of values along the x-axes. The input must be in 'linear' or 'log'. The 'linear' input set the curves in the corresponding track to be plotted in a linear x-scale, while 'log' input plots the curve in a logarithmic x-scale. The input can be written as: `inputTrack.Curves= {'linear'|'log', ...}`

**Note:** It is not suggested to assign log curves that are supposed to be plotted in logarithmic scale (logarithmic log curves) and linear log curves together in one track. For example, resistivity log is suggested to be plotted in separate logarithmic track.

An example of input of `inputTrack` below shows that there are 3 tracks to be plotted, Track 1, Track 2, and Track 3. Track 1 has 2 curves, hookload and WOB, while each Track 2 and Track 3 has 1 curve, SPP and BPOS curve consecutively.

---

<sup>1</sup><http://se.mathworks.com/help/matlab/ref/struct.html>

```
inputTrack = struct(...
    'Name',{'Track 1','Track 2','Track 3'},...
    'Curves',{{'HKLD','WOB'},...
                {'SPP'},...
                {'BPOS'}}},...
    'XScale',{'linear','linear','linear'});
```



**Note:** The program can be forced to acknowledge curve which is not listed in the `List Canonical Mnemonic.xlsx`. As an example, if user want to plot `STICK_RT`, which is not listed in the worksheet, the user can set `'STICK_RT'` in `inputTrack.Curves`. Then, the program will be forced to acknowledge `STICK_RT` curve in the log file and plot it. However, it is recommended to add the mnemonic name which is not listed into the worksheet.

### 3.3.2 Writing input for inputCurves

`inputCurves` is an input variable with class of `struct` which contains information of the curves properties to be plotted. The properties that can be set include curve line width, log curve range value, curve line color, and curve units. All these information are defined in fields of `inputCurves` structure,

`inputCurves.Name` contains the canonical name of the curves based on the curve name in input `inputTrack`. This input **must be written as the first field**. The input can be written:

```
inputCurves.Name = {'<canonical name #1>','<canonical
name #2>',...}
```

`inputCurves.Unit` contains the unit of the curves that user desired. By changing the curve unit, the curve unit from the log file will be converted to the new input. If the user set the input with `'auto'`, the curve unit from the log file will be converted to the default unit. The default unit of log curves can be found in worksheet `List Canonical Mnemonic.xlsx` and the list of units are collected in worksheet `Unit Conversion.xlsx`. The input can be written:

```
inputCurves.Unit = {'<unit in string>','auto',...}
```

**Note:** It is suggested for user to write the unit input following the list in worksheet `Unit Conversion.xlsx` because the conversion process is case sensitive.



`inputCurves.Min` contains the minimum value of the curves. If this variable is set to 'auto', the program will automatically set the default value. The input can be written:

```
inputCurves.Min = {<numeric value>|'auto',...}
```

`inputCurves.Max` contains the maximum value of the curves. If this variable is set to 'auto', the program will automatically set the default value. The input can be written:

```
inputCurves.Max = {<numeric value>|'auto',...}
```

`inputCurves.LineWidth` contains the line width value. If this variable is set to 'auto', the curves are plotted with 0.5 points. The input can be written:

```
inputCurves.LineWidth = {<numeric value>|'auto',...}
```

`inputCurves.LineColor` contains the line color of the curves. The input must be in RGB triplet, a three-element row vector whose elements specify the intensities of the red, green, and blue components of the color. The intensities must be in the range [0,1]. If this variable is set to 'auto', the curves are plotted following the default colors.

```
inputCurves.LineColor = {<RGB triplet>|'auto',...}
```

These curve properties are optional, if user don't want to change the default curve properties, this input variable can be set as an empty matrix, `inputCurves=[]`. User can decide which curve properties to be changed. An example of `inputCurves` is shown below. This input follows the example of input `inputTracks` in Chapter 3.3.1. The input in example shows that user change the properties of curves HKLD and SPP without changing the properties of curves WOB and BPOS.

```
inputCurves = struct('Name',{ 'HKLD', 'SPP'},...  
    'Unit',{ 'KGFM', 'bar'},...  
    'Min',{ 'auto',0},...  
    'Max',{200000, 'auto'},...  
    'LineWidth',{ 'auto',0.8},...  
    'LineColor',{[0.000, 0.447, 0.741],[0.800,  
        0.000, 0.200]});
```



**Suggestion:** If user only want to change only several properties, example **Min** and **Max** properties of the curves, user can remove the other unnecessary fields. Example:

```
inputCurves = struct('Name',{ 'HKLD' , 'SPP' },...  
    'Min',{ 'auto' ,0,},...  
    'Max',{200000, 'auto' });
```

## 4. RTDD log viewer application: user interface basis

### 4.1 RTDD log viewer application user interface

After compiling the source code, RTDD log viewer application will appear in a window. The components and terms of the interface is shown in Fig. 4.1.

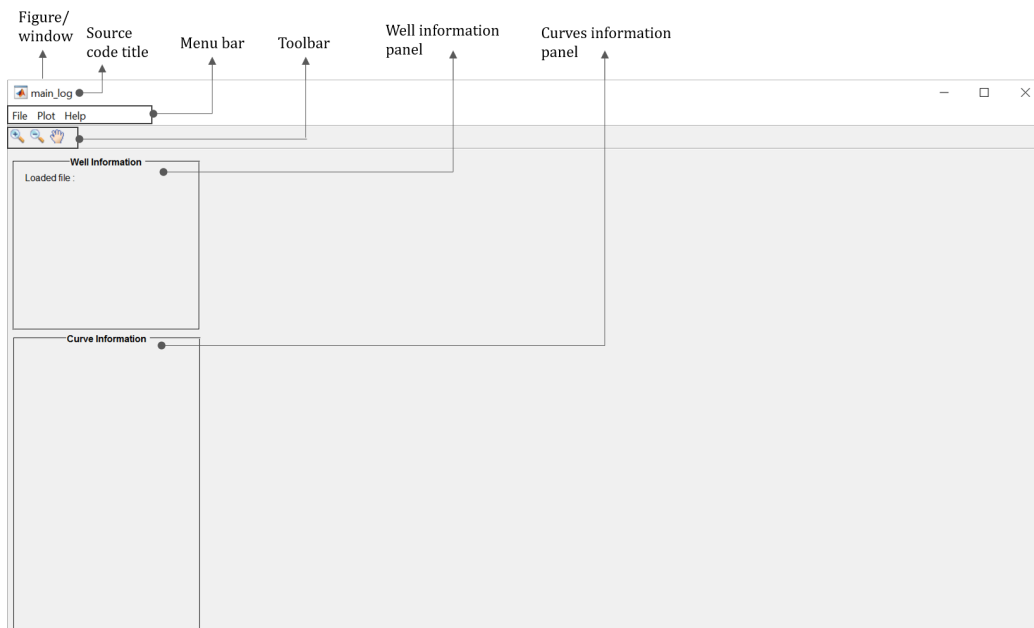


Figure 4.1: Full window of log viewer application

### 4.1.1 Menu bar

The RTDD log viewer application contains a pull-down menu bar so user can access the features of the application. The menu bar contains:

**File** menu provides an option to load the preferred files.

**Plot** menu provides an option to plot the loaded file based on the selected template.

**Help** menu provides option to open the user manual or show the information of the application.

### 4.1.2 Toolbar



**Pan tool** activates mouse-based panning for any axes in the current figure.



**Zoom in tool** activates interactive zooming into the point beneath the mouse for any axes in the figure.



**Zoom out tool** activates interactive zooming out from the point beneath the mouse for any axes in the figure.

### 4.1.3 Well information panel

The well information panel provides information of the log file that successfully loaded into the application and information of the well. The information will appear after user load the file. The information contains:

- Loaded file name
- Well name
- Well field
- Well location
- Country
- Service: the service company which perform the logging operation
- Start date: the date and time of logging operation executed
- Stop date: the date and time of logging operation ended

Well Information	
Loaded file : well_RT.mat	
<b>Name</b>	6608/10-K-3 H
<b>Field</b>	Norne
<b>Location</b>	Norwegian North Sea
<b>Country</b>	Norway
<b>Service</b>	Slumberger D&M
<b>Start</b>	06-Sep-06 15:47:03
<b>Stop</b>	08-Sep-06 10:59:53

Figure 4.2: Well information panel after load the log file

Curve Information	
<b>time</b>	s
<b>rop5</b>	M/HR
<b>incl_cont_rt</b>	degrees
<b>gr_cdr_rt</b>	API units
<b>bvel</b>	m/s
<b>bpos</b>	m
<b>hkld</b>	KKGF
<b>swob</b>	KKGF
<b>dept</b>	m
<b>tqa</b>	KMDN
<b>rpm</b>	RPM
<b>crpm_rt</b>	RPM
<b>stick_rt</b>	RPM
<b>trpm_rt</b>	RPM
<b>tflo</b>	LPM
<b>cdr_atmp_rt</b>	DEGC

Figure 4.3: Curve information panel after load the log file

#### 4.1.4 Curves information panel

This panel shows the mnemonic names and units of all of the logging curves contained in the log file. This panel will be shown once the file loaded.

## 4.2 Load file into RTDD log viewer

There are 3 types of log file that can be loaded into the program:

- \*.las (LAS 2.0 and 3.0)
- \*.csv
- \*.asc
- \*.mat

To load the log file into the application,

1. In the menu bar, choose **File** > **Load file...**
2. A new window will show up and choose the desired log file (\*.las/\*.csv/\*.mat)
3. Click **OK** to load.

During loading process, the file is also parsed by the program. The details of parsing process can be found in Appendix A. In order to avoid errors during parsing, it is very important to check the log file according to the standard format. All the standardized formats for the log file are discussed in Appendix B, Appendix C, and Appendix D.

## 4.3 Save the parsed file

The output from parsing can be saved in .mat file. To save the file:

1. In the menu bar, choose **File** > **Save file...**
2. A new window will show up and select the destination folder of the file
3. Click **Save** to save the file.

## 4.4 Plot log curves in RTDD log viewer

To plot the curves from the loaded file in the application:

1. In the menu bar, choose **Plot** > **Load template...**
2. Select the plotting template (see Chapter 3.3 for information of preparing plotting template)
3. Click **OK** to plot.

## 4.5 Navigating around RTDD log viewer

This section provides information of terms and features in RTDD log viewer application after plotting the curves. The components of the applications after the curves plotted are shown in Figure 4.4.

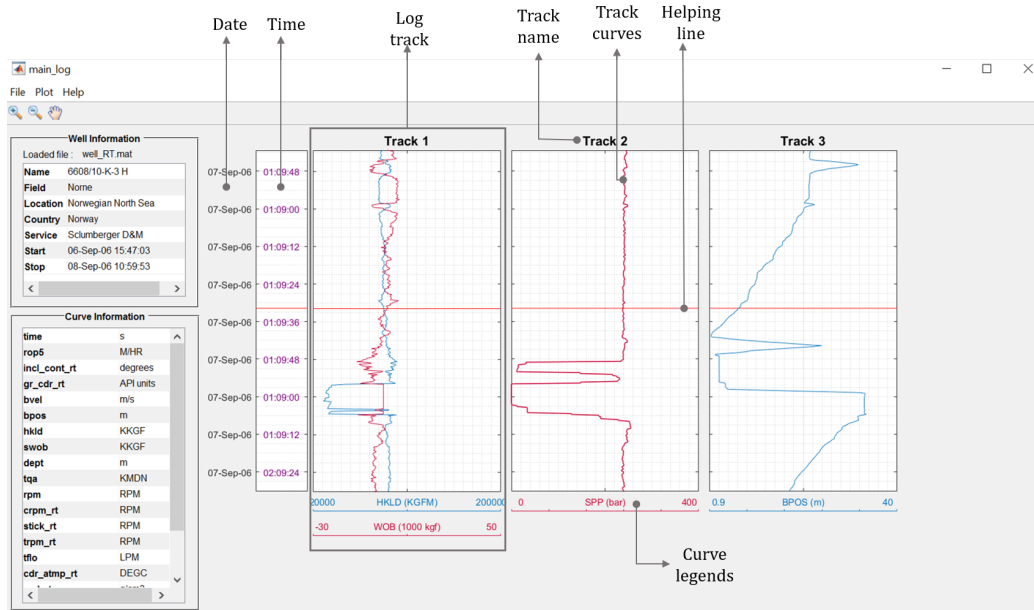


Figure 4.4: RTDD log viewer after curves plotted

**Date and time** shows the date and time which are the curves' reference.

**Log track** is described as one component which consists of track name, track curves, and track legends.

**Track name** shows the tracks' name which is based on the user input.

**Track curves** shows the plotted curves which is based on the user input.

**Track legends** shows the names, units, and minimum and maximum value of plotted curves.

Some of the features in RTDD log viewer application.

1. Helping line
2. Hide curves feature

3. Interactive data cursor
4. Track description
5. Zoom in and out
6. Pan

#### 4.5.1 Helping line

Helping line is a horizontal line which stretched over date and time axes and all log tracks axes. This helping line is moving interactively as the user move the cursor over the log track axes. The purpose of helping line is to improve reading the curves, especially during comparing curves value. The helping line has red color as can be seen in Figure 4.4.

#### 4.5.2 Hide curves feature

This feature provides a functionality to hide the curve(s) plotted in a log track. To hide the curve(s),

1. Hover the cursor over the legend of the selected curve
2. Click the legend to hide the curve
3. Click the legend again to unhide the curve



**Attention:** This feature can not be processed while *zoom or pan mode* is active.



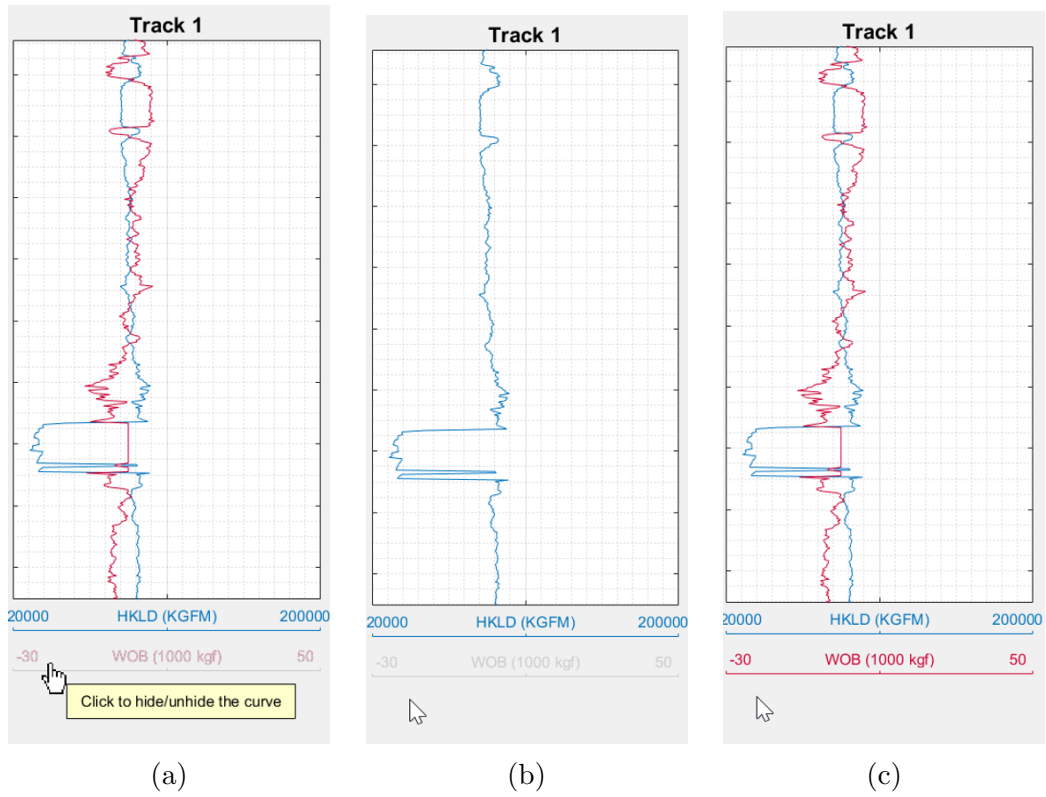


Figure 4.5: Hide curves feature: (a) Hover cursor over legend, (b) the curves hidden and legend turns to grey color, and (c) unhide the curve by clicking legend again

### 4.5.3 Interactive data cursor

This feature provides a functionality to show a data tip. A data tip is a small box that appears when user move the cursor over the line curve and floats within an axes. The data tip display the values of the curve at data cursor location, including the date and time.

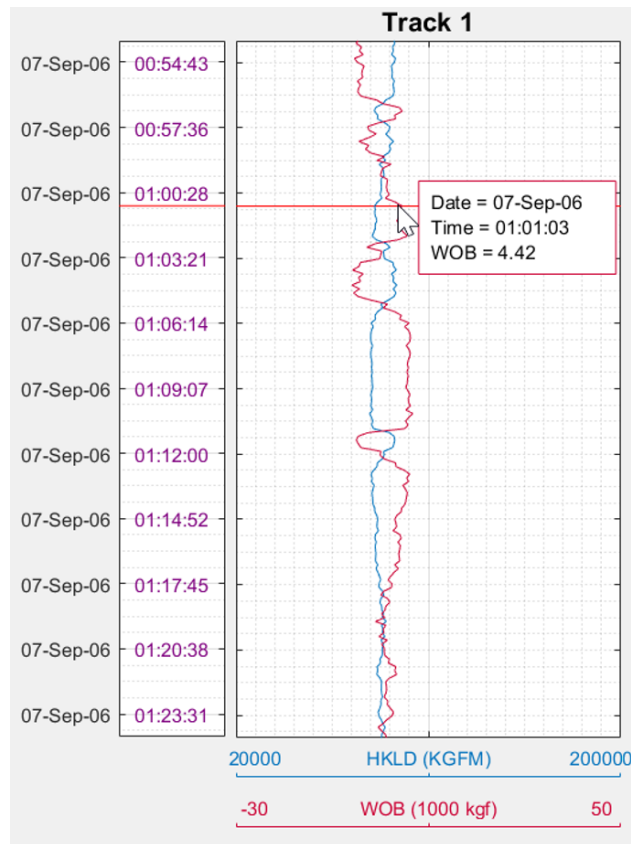


Figure 4.6: Interactive data cursor and data tip



**Attention:** The data tip is not appear when *zoom or pan mode* is active.

#### 4.5.4 Track description

Track description feature provides a description of the curves plotted within the selected track. To view the track description,

1. Hover the cursor over the track name
2. Click the track name. Then, a new window will appear
3. Click close to close the track description.

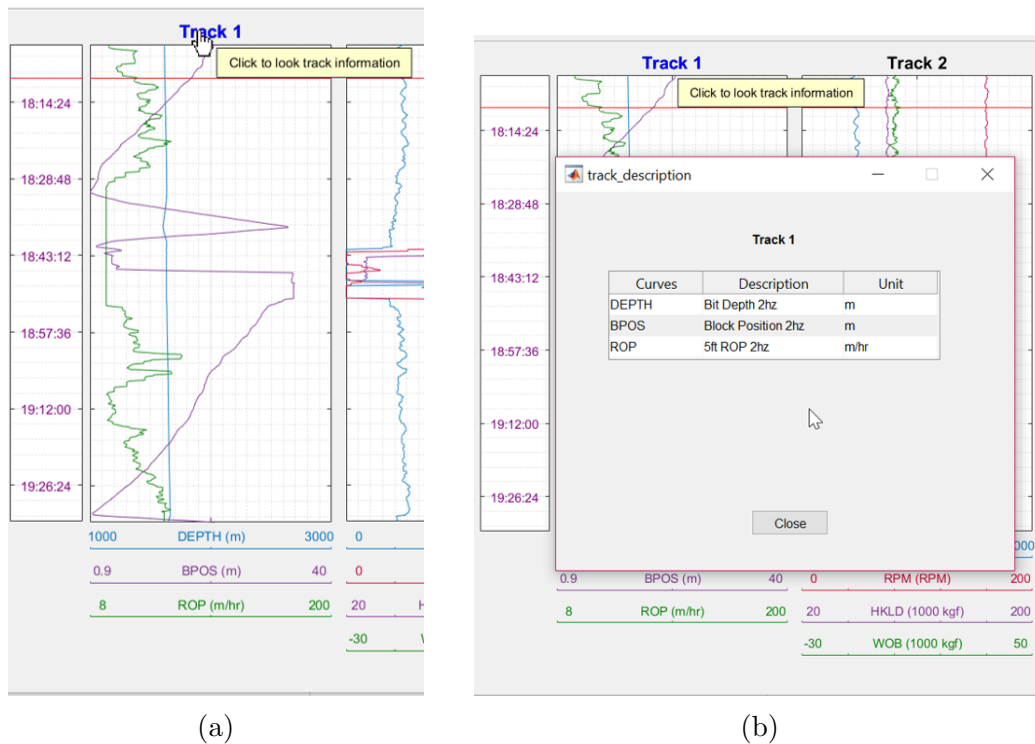


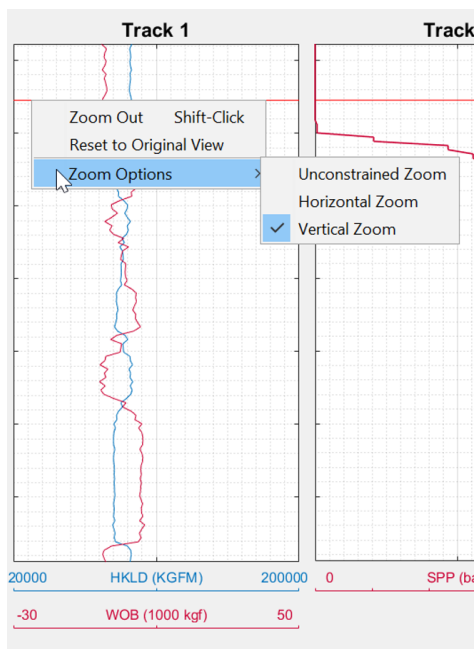
Figure 4.7: Track description features: (a) Click the track name and (b) new window showing track description will appear

#### 4.5.5 Zoom in and out

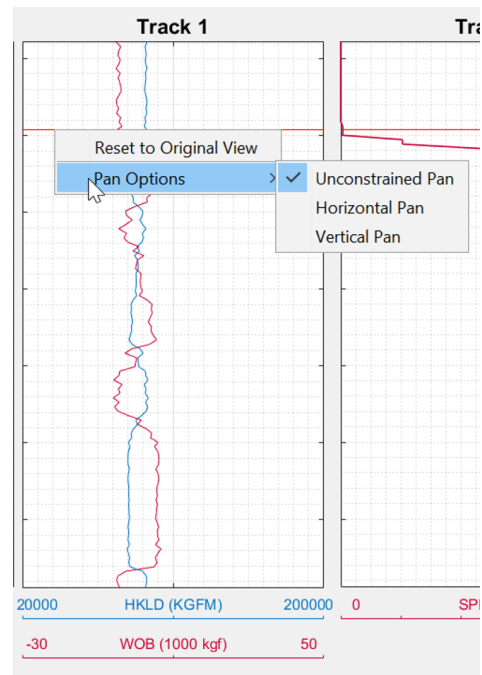
Zoom in/out features can be activated by clicking the zoom in/out icon in toolbar. During the zoom in/out mode is active, user can found some options of zoom in/out functionality from the pop-up menu which appears from right-click operation on the axes. In this pop-up menu, user can select zooming motion — horizontal, vertical, and both direction — and reset to the original view.

#### 4.5.6 Pan

Pan feature can be activated by clicking the pan icon in the toolbar. During the pan mode is active, user can found some options of pan functionality from the pop-up menu which appears from right-click operation on the axes. In this pop-up menu, user can select panning motion — horizontal, vertical, and both direction — and reset to the original view.



(a) Options in the pop-up menu during the active zoom in/out mode



(b) Options in the pop-up menu during the active pan mode

Figure 4.8: (a) Zoom in/out and (b) pan options

## 5. Examples of plotting templates

The example of real time drilling log data and plotting templates are provided in the `example`. This chapter shows the results from those templates.

### 5.1 Track template #1

The plotting template is

```
inputTrack = struct(...
    'Name',      {'Track 1','Track 2','Track 3',...
                  'Track 4','Track 5','Track 6'},...
    'Curves',   {{ 'DEPTH','INCL','DBTM'},...
                  { 'BPOS','ROP','BVEL'},...
                  { 'TORQ','RPM','HKLD','WOB'},...
                  { 'SPP','TFLO'},...
                  { 'ECD','DHAT'},...
                  { 'GR'}},...
    'XScale',    {'linear','linear','linear',...
                  'linear','linear','linear'});
inputCurves = struct('Name',{'GR','BPOS','ROP'},...
                     'Min',{0,0,0},...
                     'Max',{200,'auto','auto'});
```

This template has 6 tracks:

1. Track 1 : Depth, inclination, and DBTM (bit depth measured)
2. Track 2 : BPOS (block position) ROP (rate of penetration), and BVEL (block velocity)
3. Track 3 : Torque, RPM, hookload, and WOB (weight on bit)
4. Track 4 : SPP (stand pipe pressure) and flow rate

5. Track 5 : ECD (equivalent circulating density) and DHAT (downhole annular temperature)
6. Track 6 : Gamma ray

The result from this plotting template is shown in figure below

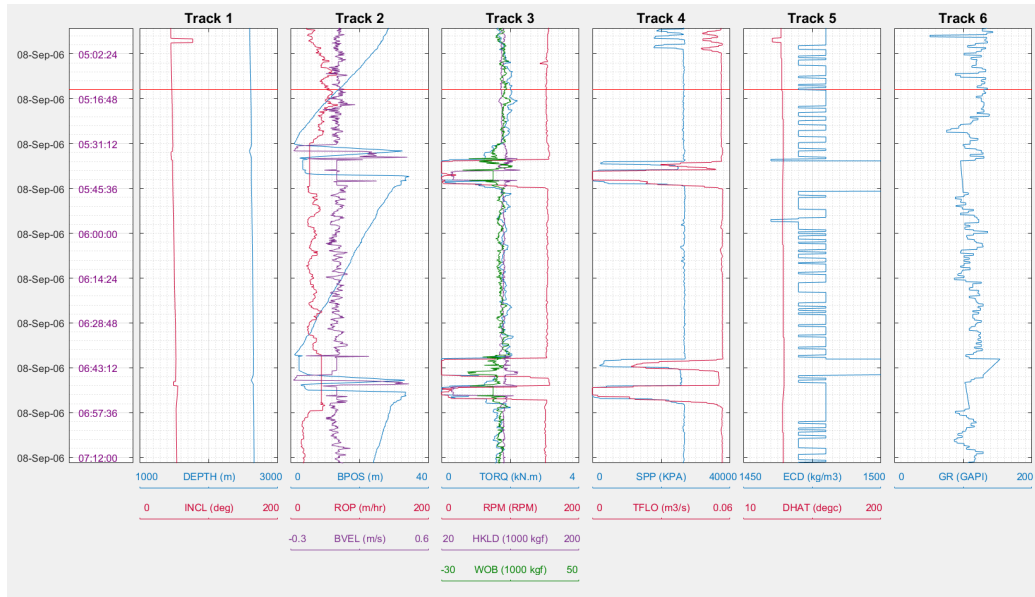


Figure 5.1: Curves plotted from template #1

## 5.2 Track template #2

The plotting template is

```
inputTrack = struct(...
    'Name',      {'Track 1','Track 2','Track 3',...
                  'Track 4'},...
    'Curves',   {{ 'DEPTH','DBTM','BPOS','ROP'},...
                  { 'TORQ','RPM','HKLD','WOB'},...
                  { 'SPP','TFLO'},...
                  { 'GR'}},...
    'XScale',    {'linear','linear','linear',...
                  'linear'});
inputCurves = struct('Name',{ 'BPOS','ROP'},...
                     'Min',{0,0});
```

This template has 4 tracks:

1. Track 1 : Depth, DBTM, BPOS, and ROP
2. Track 2 : Torque, RPM, hookload, and WOB
3. Track 3 : SPP and flow rate
4. Track 4 : Gamma Ray

The result from this plotting template is shown in figure below

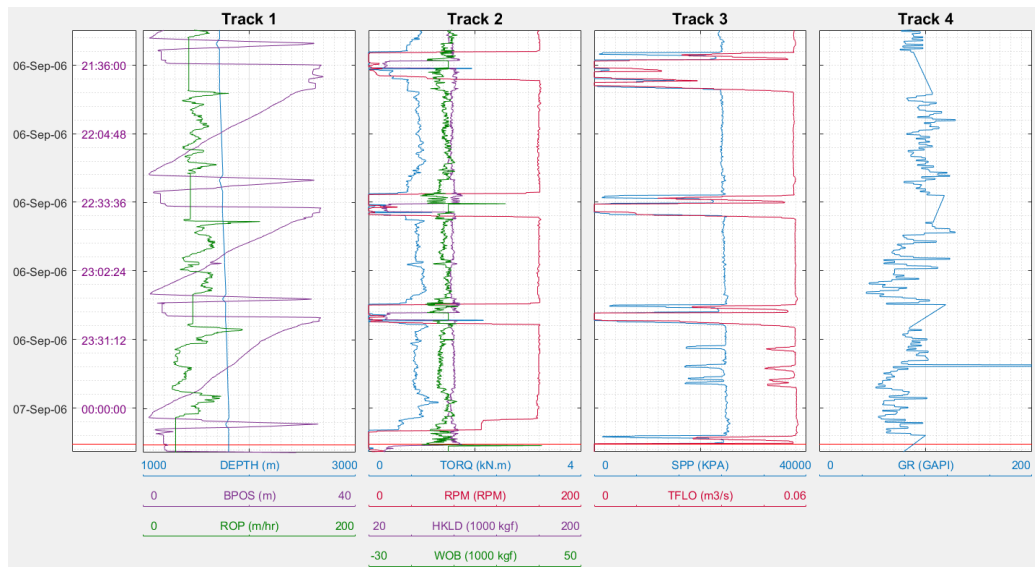


Figure 5.2: Curves plotted from template #2

## 5.3 Track template #3

The plotting template is

```
inputTrack = struct(...
    'Name',      {'Track 1','Track 2','Track 3',...
                  'Track 4'},...
    'Curves',   {{ 'DEPTH','DBTM','BPOS'},...
                  { 'ROP','BVEL'},...
                  { 'TORQ','RPM'},...
                  { 'HKLD','WOB'}},...
    'XScale',    {'linear','linear','linear',...
                  'linear'});
inputCurves = struct('Name',{ 'BPOS','ROP'},...
                     'Min',{0,0});
```

This template has 4 tracks:

1. Track 1 : Depth, DBTM, and BPOS
2. Track 2 : ROP and BVEL
3. Track 3 : Torque and RPM
4. Track 4 : Hookload and WOB

The result from this plotting template is shown in figure below

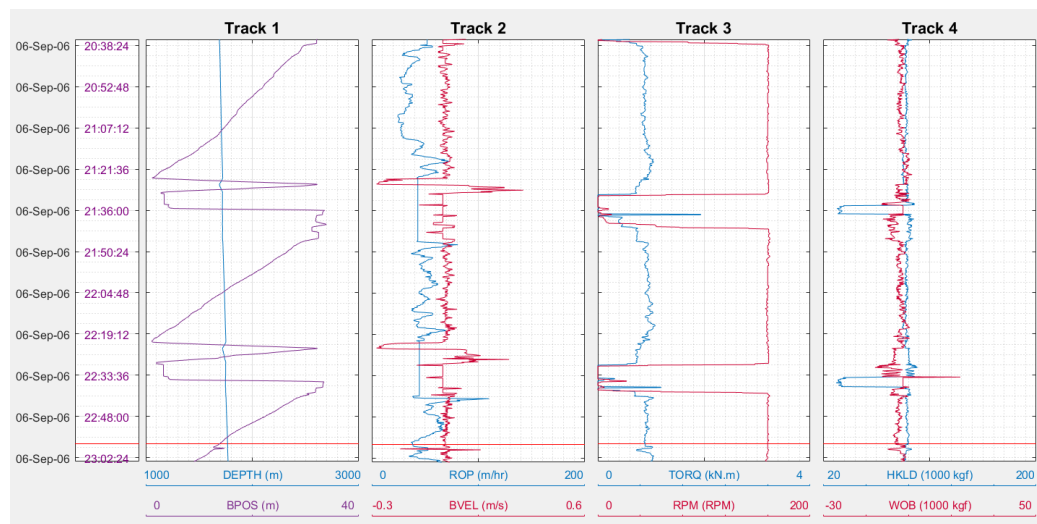


Figure 5.3: Curves plotted from template #3



# Appendices

## A. Parsing process

This chapter discusses a brief details about parsing process within the program. Parsing refers to process to analyze and extract the component parts of raw log file. The parsing program is linked to the log viewer program, but user can also run the parsing program separately. These programs are:

- `read_asc_file`, a parsing program for ASC-file. This program can be found in directory `lib`
- `read_csv_file`, a parsing program for CSV-file. This program can be found in directory `lib`
- `read_las_file`, a parsing program for LAS-file. This program can be found in directory `lib > S4M > Geophysics_3.0`

### A.1 The output of parsing

All types of log file will be processed in the parsing program to return *a standard output* which is a MATLAB `struct` class type. This output *must* contain fields:

1. `curve_info`, refers to the curve information. This field is a class of `cell` which contains the curve mnemonic names, units, and description in a successive row.
2. `curves`, refers to the curves data. This field is a class of `double`. Each column represent the data value of each curve contained in the log file. The time (and date) string in CSV and ASC file will be converted in a float number based on ISO standard.
3. `first`, refers to the first depth (or time) in the file
4. `last`, refers to the last depth (or time) in the file
5. `step`, represents the actual difference between every successive index value

6. `units`, refers to the unit of depth (or time) reference
7. `null`, refers to null values
8. `wellname`, refers to the well name
9. `date`, refers to date logged

The following components may or may not be present in the output depends on the availability of these information in the log file

1. `date_time` refers to string of date and time from CSV and ASC file
2. `company` refers to company name
3. `field` refers to the well field
4. `location` refers to the well location
5. `country` refers to the well location (country)
6. `service` refers to logging company

## **A.2 Editing and saving the output into MAT-file**

The output from parsing program enables the user to edit the log file easier. For example, if a user want to create a new data curve by combining the existed curves, the user can make a data value easily by editing the field **curves**.

Another advantage is that the output can also be saved into a MAT-file, a MATLAB file that contains a MATLAB variable. Saving can be performed from the RTDD log viewer application (see Chapter 4.3) or separately by compiling the program.

## B. Standard format of LAS file

LAS is short for Log ASCII standard which holds file specification for various versions of LAS. The specification of LAS file version 2.0 can be found in [http://www.cwls.org/wp-content/uploads/2014/09/LAS\\_20\\_Update\\_Jan2014.pdf](http://www.cwls.org/wp-content/uploads/2014/09/LAS_20_Update_Jan2014.pdf) and version 3.0 in [http://www.cwls.org/wp-content/uploads/2014/09/LAS\\_3\\_File\\_Structure.pdf](http://www.cwls.org/wp-content/uploads/2014/09/LAS_3_File_Structure.pdf).

### B.1 Troubleshoot for non-standardized LAS file

The developers noticed that some of the LAS files do not follow the specifications, giving errors during parsing and/or plotting the well log. Within this section, developers listed the most common issues causing errors.

#### B.1.1 Check the well information under $\sim$ W section of raw .LAS file

The problem may arise from:

- Different units of **STRT**, **STOP**, and **STEP**. The units of these three information must match. In a real-time drilling data, the unit of these information can be in SEC or S.
- The time index value of **STRT** and/or **STOP** is in string format. The supported format is a floating number, while format {dd/mm/yy} and/or {hh:mm:ss} format is not supported. More detailed information of time-index value: Chapter B.1.3.
- Missing **DATE** information. It is recommended to provide the date information to avoid error during reading time-index value.
- Make sure each line of well information is delimited with a break or a new line.

### B.1.2 Check the ASCII log data under ~A section of raw .LAS file

Check the time-reference data. The parsing program only support time-reference data in floating numbers and does not support time data in string format, such as in {dd/mm/yy} or {hh:mm:ss}. More detailed information of time index-value format: Chapter B.1.3.

### B.1.3 Time-index value format

The time index is the most crucial information within the real-time log file, thus the formats must be correct. All of the time-index value information must be written in a floating number format, while the string format is not supported. In order to get a floating number of time, usually the time information is encoded based on the standard format. There are wide ranges of standard format on timestamp, but the most formats used in real-time drilling data are UNIX time, GPS epoch time, ISO calendar.

The log viewer program is designed based on proleptic ISO calendar format, but still can process other time-index format. If time-index value is based on UNIX time and GPS epoch time, user must provide the **DATE** under the well information section, or else the program will give an error. Other standard formats beside UNIX time, GPS epoch time, and ISO calendar are not supported within the program. An example of GPS epoch time-index well log can be found in well 30/3 - A-5 B.

### B.1.4 Troubleshoot for time index value in string format

The developer created an additional program to fix the .LAS file of which time-index values are in string format. This additional program must be run separately because it does not linked in the log viewer program. The program is `fix_las_time_date.m` and can be found in `lib` directory.

Below is the example of non-standardized LAS file with wrong format of time and date. The highlighted parts indicates parts that are not following the standard format and required to be fixed.

#### Well Information Block

#MNEM.UNIT	Data Type	:Information
STRT. <span style="background-color: yellow;">    </span>	<span style="background-color: yellow;">09-AUG-2006 13:13:05</span>	: START INDEX
STOP. <span style="background-color: yellow;">    </span>	<span style="background-color: yellow;">09-AUG-2006 13:14:25</span>	: STOP INDEX
STEP. <span style="background-color: yellow;">SEC</span>	10.0000	: STEP
NULL.	-999.25	: NULL VALUE
COMP.	Conoco Phillips	: COMPANY
WELL.	2/4X-04B	: WELL
FLD.	Ekofisk	: FIELD
LOC.		: LOCATION
PROV.		: PROVINCE
CNTY.		: COUNTY
STAT.		: STATE
CTRY.	Norway	: COUNTRY
SRVC.	Schlumberger	: SERVICE COMPANY
DATE.	09-AUG-2006 13:13:05	: LOG DATE
UWI.		: UNIQUE WELL ID
API.		: API NUMBER

#### Curve Information Block

#

#MNEM.UNIT		: Curve Description
<span style="background-color: yellow;">TIME</span>	<span style="background-color: yellow;">.HHMMSS</span>	:
<span style="background-color: yellow;">DATE</span>	<span style="background-color: yellow;">.D</span>	:
DEPT	.FT	: Bit Depth 2hz
BVEL	.FT/S	: Block Velocity 2hz
BPOS	.FT	: Block Position 2hz
ROP5	.F/HR	: 5ft ROP 2hz

#

#### Parameter Information Block

#MNEM.UNIT Value:Description

```

#
# Other Information Block
#

  A TIME      DATE      DEPT    BVEL    BPOS    ROP5
13:13:05     09-Aug-06    11966.73  0.00    89.07   -999.25
13:13:15     09-Aug-06    11966.73  2493.07  89.07   -999.25
13:13:25     09-Aug-06    11966.73  0.00    89.07   -999.25
13:13:35     09-Aug-06    11966.73  0.00    89.07   -999.25
13:13:45     09-Aug-06    11966.73  0.00    89.07   -999.25
13:13:55     09-Aug-06    11966.73  0.00    89.07   -999.25
13:14:05     09-Aug-06    11966.73  0.00    89.07   -999.25
13:14:15     09-Aug-06    11966.73  0.00    89.07   -999.25
13:14:25     09-Aug-06    11966.73  0.00    89.07   -999.25

```

By correction from `fix_las_time_date.m`, the result will look like following result.

#### Well Information Block

#MNEM.UNIT	Data Type	:Information
STRT.S	0	: START INDEX
STOP.S	80	: STOP INDEX
STEP.S	10.0000	: STEP
NULL.	-999.25	: NULL VALUE
COMP.	Conoco Phillips	: COMPANY
WELL.	2/4X-04B	: WELL
FLD.	Ekofisk	: FIELD
LOC.		: LOCATION
PROV.		: PROVINCE
CNTY.		: COUNTY
STAT.		: STATE
CTRY.	Norway	: COUNTRY
SRVC.	Schlumberger	: SERVICE COMPANY
DATE.	09-AUG-2006 13:13:05	: LOG DATE
UWI.		: UNIQUE WELL ID
API.		: API NUMBER

#### Curve Information Block

#

#MNE	MUNIT	Curve Description
ETIM	.S	: Elapsed Time
DEPT	.FT	: Bit Depth 2hz
BVEL	.FT/S	: Block Velocity 2hz
BPOS	.FT	: Block Position 2hz
ROP5	.F/HR	: 5ft ROP 2hz

#

Parameter Information Block

#MNE UNIT Value:Description

#

# Other Information Block

#

A TIME	DEPT	BVEL	BPOS	ROP5
0	11966.73	0.00	89.07	-999.25
10	11966.73	2493.07	89.07	-999.25
20	11966.73	0.00	89.07	-999.25
30	11966.73	0.00	89.07	-999.25
40	11966.73	0.00	89.07	-999.25
50	11966.73	0.00	89.07	-999.25
60	11966.73	0.00	89.07	-999.25
70	11966.73	0.00	89.07	-999.25
80	11966.73	0.00	89.07	-999.25



### C. Standard format of CSV file

This chapter discusses the standard format for CSV-file to be able parsed by parsing program. CSV file is a *comma-separated value* file. In general. CSV-file has two main sections, they are curve information and log data section.

```

Time (Time), ACTC (N/A), DCHM (N/A), DRTM (N/A), GASA (N/A), G_CO2 (N/A), G_IC5
(N/A), G_TotP (N/A), HKLX (tonne), MFOA (l/min), MWLN (N/A), RSDX (N/A), SPM3
(spm), TDH (N/A), TQX (N/A), BPOS (m), DCHR (N/A), DRTV (N/A), G_C1 (N/A), G_H2SA
(N/A), G_NC4 (N/A), G_TotPM (N/A), HPHA (N/A), MFOP (N/A), MWTI (N/A), RSUX
(N/A), SPBA (bar), TFLQ (N/A), TSTK (N/A), WOBX (tonne), CHKP (N/A), DCHV (N/A), DVER
(N/A), G_C2 (N/A), G_H2SX (N/A), G_NC5 (N/A), HHPA (N/A), HPSA (N/A), MSP1 (N/A), ROP
(m/h), SPM1 (spm), SWOB (N/A), TNUM (N/A), TVA (N/A), DBTM (m), DMEA (m), FVOC
(N/A), G_C3 (N/A), G_IC4 (N/A), G_TotIL (N/A), HKLD (tonne), MDOA (N/A), MTOA
(N/A), RPM (rpm), SPM2 (spm), TCHR (N/A), TQA (kN.m), TVCA (N/A) 2004-12-25
13:01:01,2,-999.25,5737,-999.25,32,-999.25,0.27,-999.25,-999.25,-
999.25,-999.25,-999.25,-24.99,-999.25,-999.25,-999.25,2822.29,-999.25,-
21,-999.25,0.27,53407.75,-999.25,1.57,-999.25,-999.25,-999.25,-999.25,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-21,-999.25,0.1,21.9,-999.25,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-
999.25,0.01,-999.25,1.57,52.95,-999.25,-999.25,-999.25,-999.25,-999.25,-
2004-12-25 13:12:21,2,-999.25,5738,-999.25,32,-999.25,0.44,-999.25,-
999.25,-999.25,-999.25,-999.25,-24.99,-999.25,-999.25,-999.25,2822.28,-
999.25,-21,-999.25,0.44,53540.2,-999.25,1.57,-999.25,-999.25,-999.25,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-21,-999.25,0.05,21.94,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-
999.252004-12-25 13:16:55,2,-999.25,5739,-999.25,32,-999.25,0.56,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-24.98,-999.25,-999.25,-
999.25,2822.27,-999.25,-21,-999.25,0.56,53592.4,-999.25,1.57,-999.25,-
999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-
999.25,0.02,21.96,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-999.25,-

```


Figure C.1: Example of CSV-file and the sections

### C.1 Well information section

Log file with .csv file does not have well information section. However some parameters are determined automatically within the parsing program in accordance with the parsing output.

## C.2 Curve information section

The curve information is one line which located at the *first* line of the file and contains the mnemonic name and unit of the curves. The curve unit is written next to each curve mnemonic name and it must be begin and ended with parentheses ((.)). The delimiter of each curve information is comma (,). The curve information must be started with **TIME**.

 **Attention:** It is *important* to provide the unit of **TIME** curve. If there is no **TIME** unit provided, the time string is assumed following format **yyyy-mm-dd HH:MM:SS** by default.

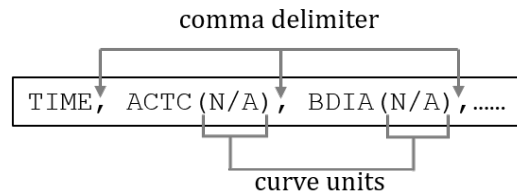


Figure C.2: Example of CSV-file curve information and delimiters

## C.3 Log data section

In each line, the value of each curve is delimited by comma. Unless the **TIME** data, the rest of log data *must be in float format*.

## C.4 Example of standardized CSV-file

The example of standardized CSV-file to be able parsed by the program is shown below.

```
Time(yyyy-mm-dd HH:MM:SS),ACTC(N/A),DCHM(N/A),DRTM(N/A),GASA  
(N/A),G_CO2(N/A),G_IC5(N/A),G_TotP(N/A),HKLX(tonne),MFOA(l/m  
in)  
2004-12-25 13:01:01,2,-999.25,5737,-999.25,32,-999.25,0.27,-  
999.25,-999.252004-12-25 13:12:21,2,-999.25,5738,-999.25,32,  
-999.25,0.44,-999.25,-999.252004-12-25 13:16:55,2,-999.25,57  
39,-999.25,32,-999.25,0.56,-999.25,-999.252004-12-25 13:37:4  
3,2,-999.25,5741,-999.25,32,-999.25,0.62,-999.25,-999.25200  
4-12-25 14:15:36,3,5735,-999.25,-999.25,-999.25,-999.25,-999  
.25,-999.25
```

## D. Standard format of ASC-file

The supported format of ASC-file which can be parsed by the programs is discussed within this chapter. ASC-file contains ASCII text format and has 3 sections: well information, curve information, and log data section.

Well information	OPERATOR: STATOIL WELL : 33/9-C-33 WELLBORE: 33/9-C-33 AT3 2526-P FIELD : STATEJORD PLATFORM: STATEJORD C COUNTRY : NORWAY
Curve mnemonic	"DATE", "TIME", "LOGTIME", "ACTC", "BDIA", "DBTM", "DBTV", "DMEA", "DVER", "RSU", "RSD", "ROP", "BPOS", "HKL", "WOB", "TRQ", "RPMA", "RPMB", "BROT", "BDTI", "TBR", "SPP", "CEPP", "WHP", "KLP", "CHP", "CCVL", "CFO", "CFI", "CDI", "CDO", "CTVL", "TVA", "TPVT", "ETPT", "MFO", "MFI", "MDO", "MDI", "MTO", "MTI", "ECDB", "ECDM", "GAS"
Curve units	dd-mm-yy, hh:mm:ss, s, , IN, M, M, M, M, M/S, M/S, M/HR, M, TON, TON, KNM, RPM, RPM, HR, HR, REV, BAR, BAR, BAR, BAR, BAR, M3, LPM, LPM, G/C3, G/C3, M3, M3, M3, M3, %, LPM, G/C3, G/C3, DEGC, DEGC, G/C3, G/C3, %
Log data	"03-Feb-06", " 18:00:00", 823456800, -9999, -9999, 7099.5728515624996, -9999, 7099.57373046875, 5393.799771, -9999, 0.1491970494389534, 4.8316221237182617, 10.608583450317383, 93.365982055664063, 2.7101669311523437, 20888.446130048673, 61.53044509578357, 61.541478726154125, -9999, -9999, -9999, 305.58145751953128, 0.8812619864940644, -9999, -0.19152350872755053, -0.83781866431236274, -9999, -9999, 0, -9999, -9999, 1119.9832519531251, 105.54001159667969, 2.6094852447509767, -9999, 59.631559753417967, 1717.8865234375, -0.75240004062652588, 1.5959184527397157, 54.19200057983403, 28.568001556396553, 1.679634000000001, 1.7038400173187256, 5.8297562599182132E-2 "03-Feb-06", " 18:00:05", 823456805, -9999, -9999, 7099.5798828124998, -9999, 7099.58251953125, 5393.805265, -9999, 0.1491970494389534, 5.1147618293762207, 10.601447296142577, 93.276779174804688, 2.7993698120117187, 20764.539662992098, 61.409915158045493, 61.54313201594897, -9999, -9999, -9999, 305.71358337402347, 0.88723745346069338, -9999, -0.18961726427078249, -0.83998104333877566, -9999, -9999, 0, -9999, -

Figure D.1: Example of ASC-file and the sections

### D.1 Well information section

The well information section is located at the first line of ASC-file. This section is not mandatory and may filled with information similar to well information section of LAS-file. The delimiter of the well information is colon.

In case the ASC-file contains well information, the information can contain:

1. WELL : <value/name>, refers to well name
2. COMP : <value/name>, refers to company name
3. FIELD : <value/name>, refers to the field name
4. COUNTRY : <value/name>, refers to country of the well location
5. STRT : <value/name>, refers to the start time
6. STOP : <value/name>, refers to the stop time
7. STEP : <value/name>, refers to the actual difference between every successive index value
8. NULL : <value/name>, refers to null values
9. DATE : <value/name>, refers to the date of logging operation

The last five well information will be calculated or determined automatically by the parsing program in case these information are not provided in the file.

## D.2 Curve information section

Both curves mnemonic names and units must be begin and ended by quotation mark("). The delimiter between each curve mnemonic name and unit is comma (.). The curve units are written directly one line after the mnemonic names line. The curve information must contain and begin with **DATE** and **TIME**.

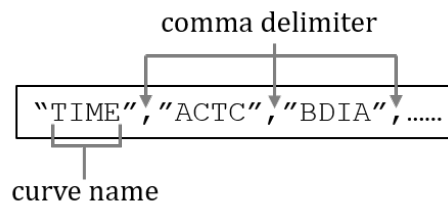


Figure D.2: Example of ASC-file curve mnemonic names



**Attention:** It is *important* to provide the unit of **DATE** and **TIME** curve. If there are no **DATE** and **TIME** unit provided, the time string is assumed following format **dd-mmm-yy HH:MM:SS** by default.

## D.3 Log data section

In each line, the value of each curve is delimited by comma. Unless the **DATE** and **TIME** data, the rest of log data *must be in float format*.

## D.4 Example of standardized ASC-file

The example of standardized ASC-file to be able parsed by the program is shown below.

```
WELL : 33/XX-X-XX
COMP : COMPX
FIELD : FIELD X
COUNTRY : NORWAY
```

```
    "DATE", "TIME", "LOGTIME", "ACTC", "BDIA", "DBTM", "DBTV", "DMEA",
    "DVER", "RSU"
dd-mmm-yy, hh:mm:ss, s, , IN, M, M, M, M, M/S
"01-Jan-07", "17:00:00", 852138000, -9999, -9999, 4516.958984375
, -9999, 4595.0146484375, 1191.05388900000003, -9999"01-Jan-07",
"17:00:05", 852138005, -9999, -9999, 4516.958984375, -9999, 4595.
0146484375, 1191.05388900000003, -9999"01-Jan-07", "17:00:10", 8
52138010, -9999, -9999, 4516.958984375, -9999, 4595.0146484375, 1
191.05388900000003, -9999"01-Jan-07", "17:00:15", 852138015, -99
99, -9999, 4516.958984375, -9999, 4595.0146484375, 1191.05388900
00003, -9999"01-Jan-07", "17:00:20", 852138020, -9999, -9999, 451
6.958984375, -9999, 4595.0146484375, 1191.05388900000003, -9999
```