New Features in ibaAnalyzer 6.11.0

Authors: R. Kolesnik, C. Reinbrecht, T. Seitz, M. Verschaeve

Date: 20 November 2018

iba AG • Königswarterstr. 44 • D-90762 Fürth • Germany • www.iba-ag.com

Table of contents

1 Unicode support

1.1 Short notes about Unicode standard

Unicode is a computing industry standard for the consistent encoding, representation, and handling of text expressed in *most of the world's writing systems*. The most recent version, Unicode 11.0, contains a repertoire of *137,439 characters* covering 146 modern and historic scripts, as well as multiple symbol sets and emoji. (https://en.wikipedia.org/wiki/Unicode)

Simply speaking, a Unicode string can contain symbols from multiple languages altogether, for example, “UήCödę Ющ 내ㄴÜä” contains German, Russian, Korean, Arabic and other letters.

UTF-8, UTF-16 and other Unicode encodings

Unicode can be implemented by different *character encodings*. The Unicode standard defines *UTF-8* (from one to four bytes per character)**,** *UTF-16* (from two to four bytes per character), and UTF-32 (always four bytes per character), and also several other encodings are in use.

- *UTF-8* is currently the most popular **Unicode** encoding in the world. For example, it’s the standard encoding for OPC UA and Comtrade. Also it’s the de facto standard in the web sites, Linux and Android
- *UTF-16* is the standard Unicode encoding for Microsoft Windows and NTFS file system.

Both standards coexist simultaneously nowadays, so **i**baAnalyzer has to support *both* of them in some way – each for its particular purpose. In most cases there is no necessity for the user to think about encodings though.

1.2 Unicode Folder names and File names

Folder names

The Non-Unicode version of ibaAnalyzer was unable to browse folders if their names contained some symbols not belonging to currently selected ANSI code page. If the path to some file (analysis file, data file, sql file, etc.) contained such symbols on any folder level then it was not possible to open that file.
Unicode version of ibaAnalyzer is free of abovementioned disadvantage and it has a *full support of Unicode paths*. So, you can give your folders any names mixing all possible languages if necessary.

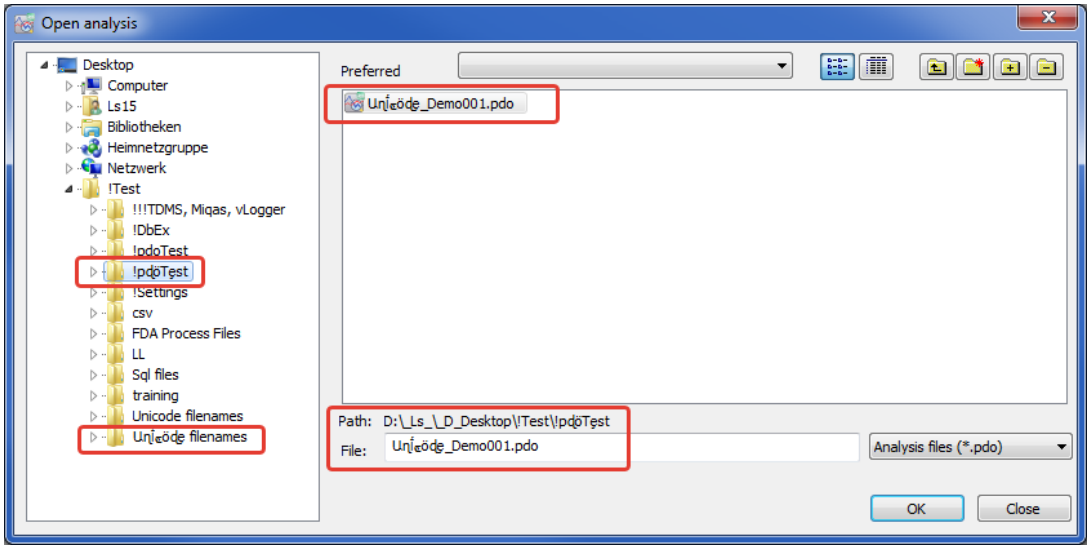
Filenames of analyses and data files

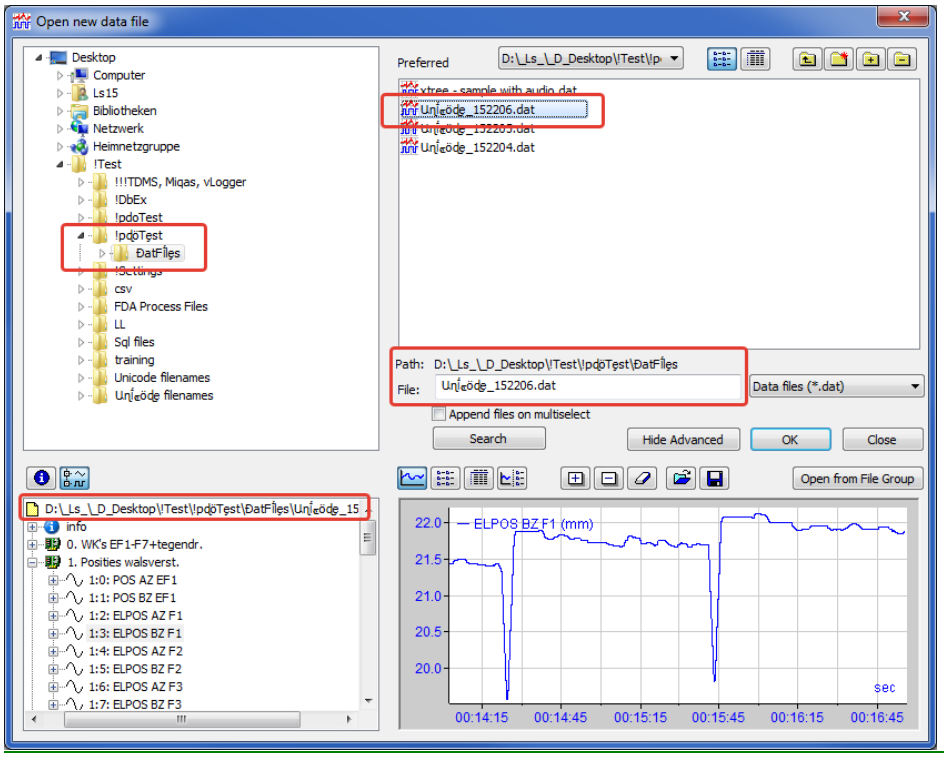
You can use full range of Unicode charset for a filename when you save your analysis or for a filename when you perform export or extract. Note, though, that it will not be possible to open such a file in older version of ibaAnalyzer.

With Unicode version of ibaAnalyzer you can open files with any names.

The same is true for other files ibaAnalyzer can work with. You can use Unicode filenames for:

- *.**ini** files when you export or import preferences
- *.**zip** files when you export or import settings
- *.**sql** files when you export or import SQL queries
- Global macros and global filters (which are actually the *.**mcr** and *.**fil** files located in %AppData%\Roaming\iba\ibaAnalyzer\)

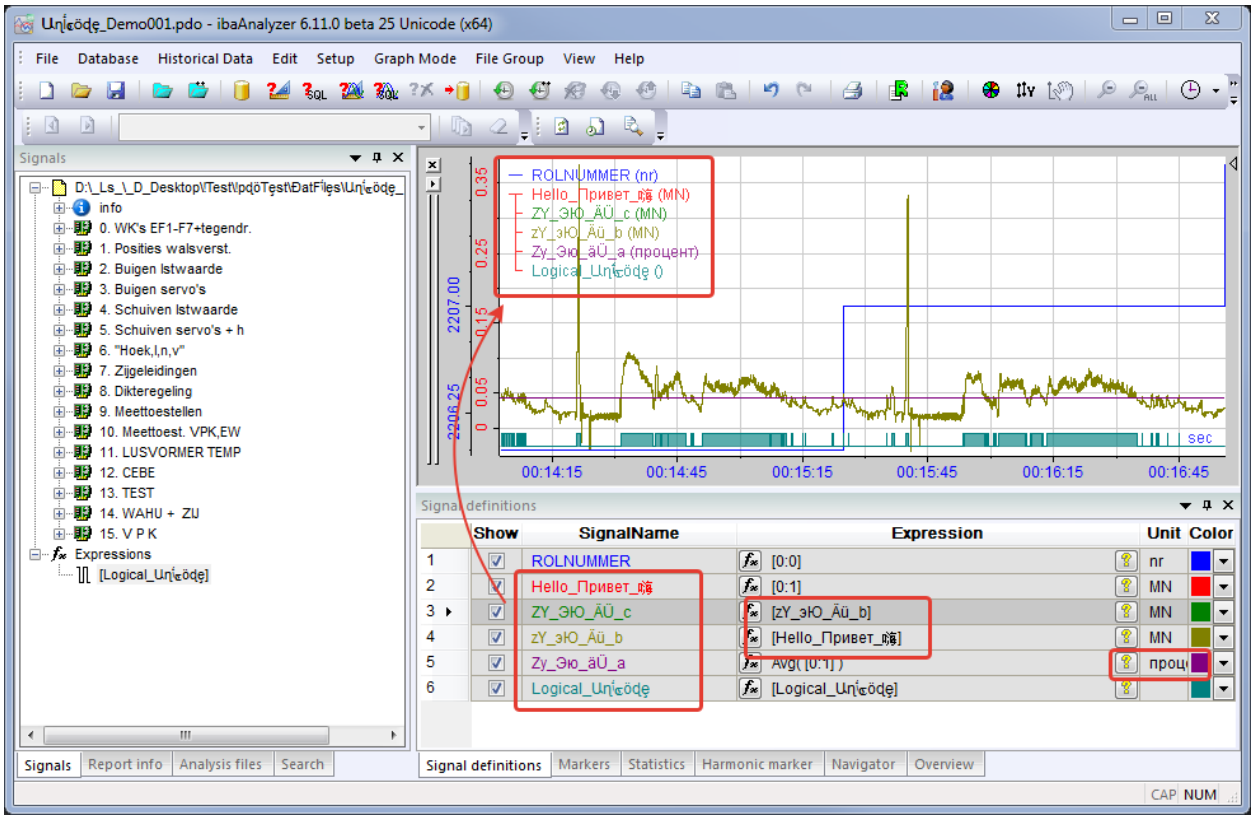




1.3 Signals and expressions

In the non-Unicode version of ibaAnalyzer a set of symbols available for using in signal names was limited to the ANSI code page selected in Windows Control Panel. Moreover, if some analysis was tried to be opened on the OS with a different code page settings, then signal names could look different (e.g. German umlauts could be replaced with Cyrillic letters or vice versa).

In the Unicode version of ibaAnalyzer you can use *any desired language* (or a mixture of any languages) when you type names, units and comments of signals. Analysis appearance does not depend anymore on the code page selected in Windows Control Panel, and all texts will look the same on different PCs.

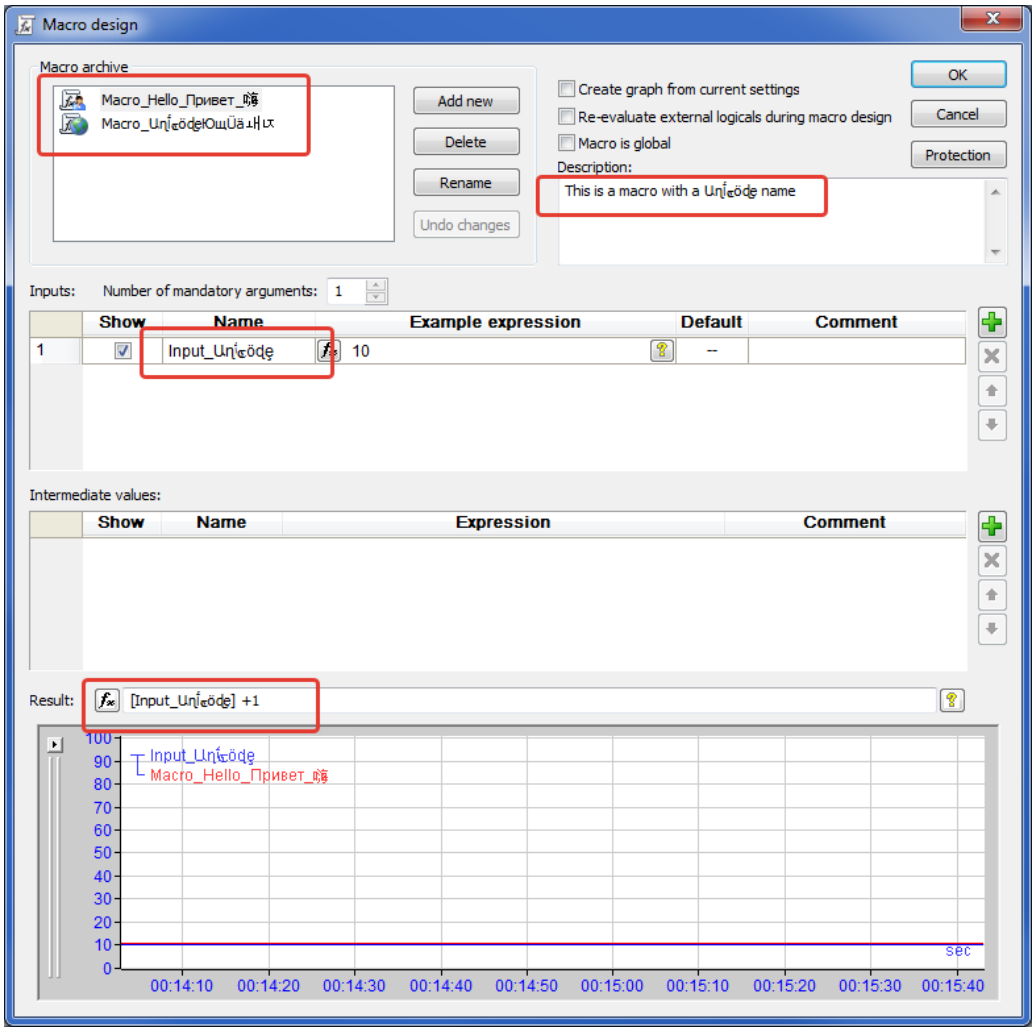


Case-insensitive Intellisense

When you edit an expression ibaAnalyzer can show a hint when you start typing a signal name, expression name, macro, filter or formula. Intellisense has case-insensitive behavior, i.e. if you start typing *“abc”*, then all of the following words will be a match: “*ABCDE*”, “*aBcEf*”, “*Abcgh*”. Case-insensivity is guaranteed for ASCII-symbols and for you current language – that is for symbols belonging to the locale currently selected in Windows Control Panel. Case-insensivity is not guaranteed for other languages.

1.4 Macros, Filters, Logical signals

You can use *any desired language* when you work with Macros – this concerns their names, description, argument names, etc.

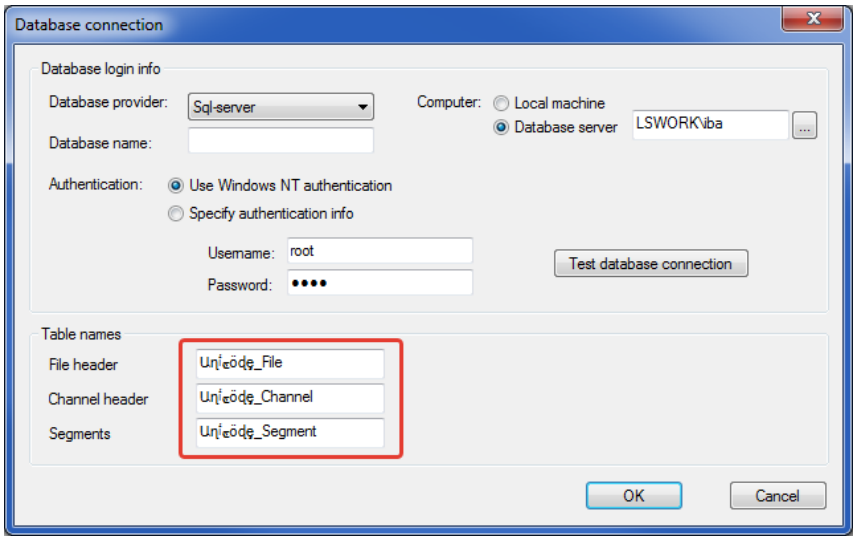


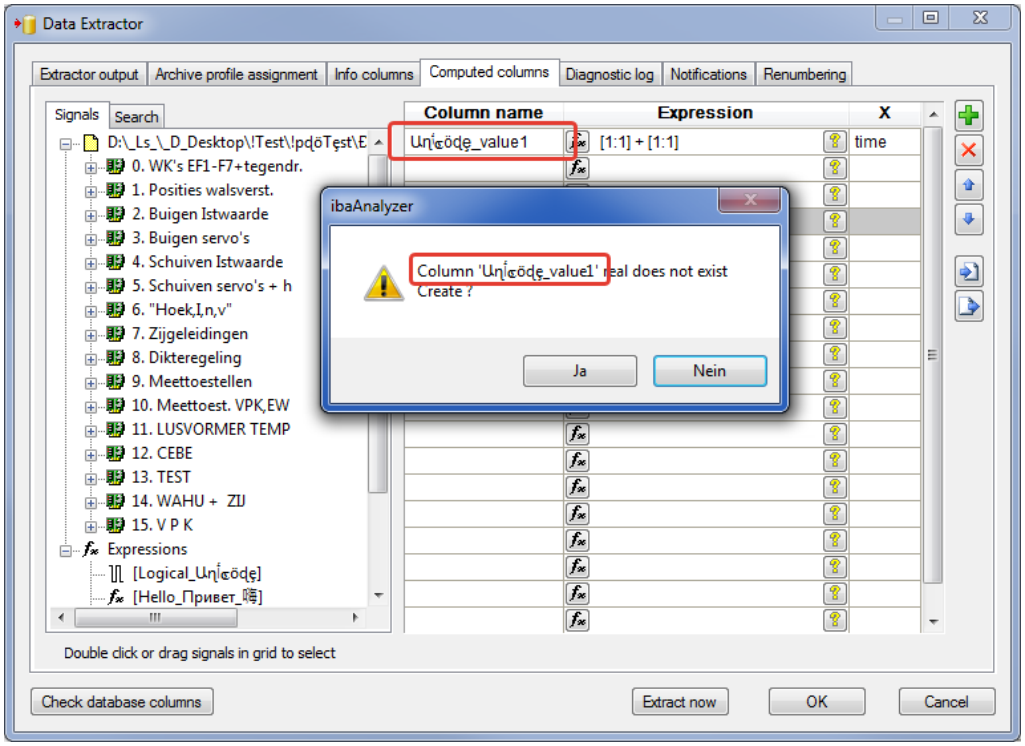
The same is applicable to Filters and Logical signals.

1.5 Interface to Databases

The situation with usage of Unicode strings when dealing with databases is somewhat more difficult. Though, the Unicode version of ibaAnalyzer itself has a full support of all Unicode symbols, this can happen that some problems can appear when dealing with some *DBMS's* even if they are also declared to have Unicode support. Some compatibility problems can also appear on the level of DBMS's ODBC driver.

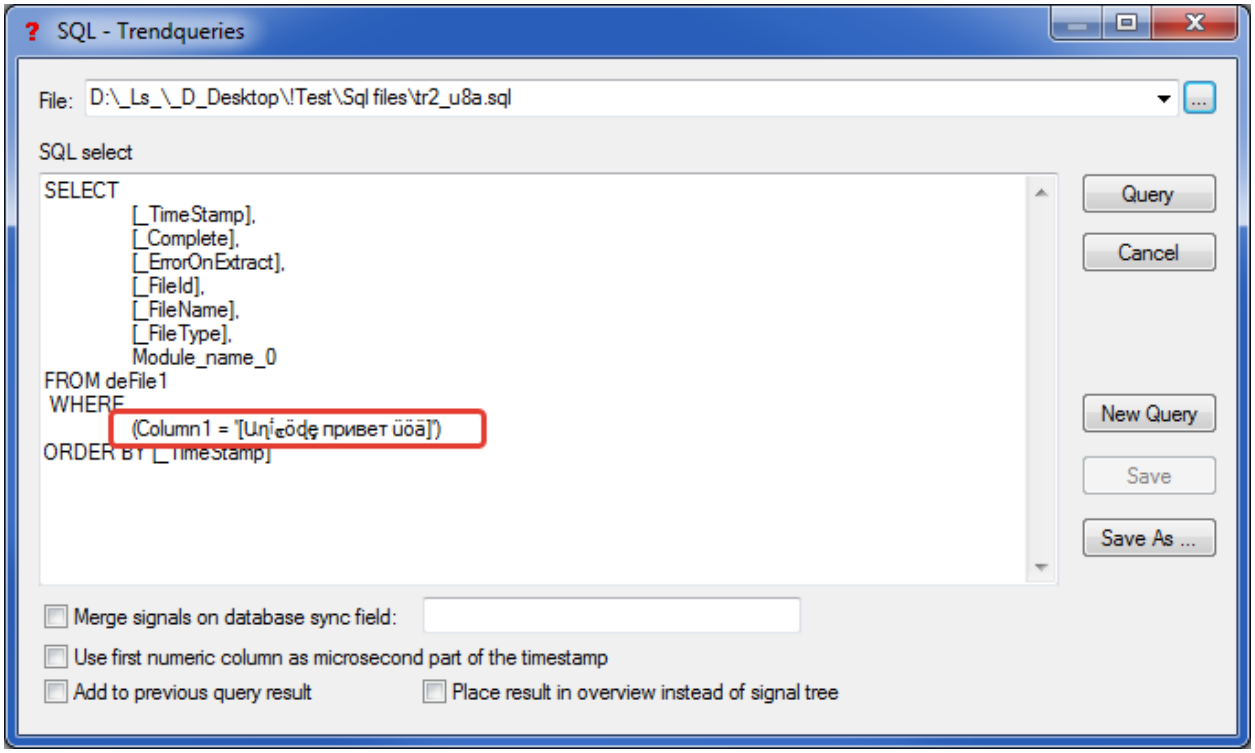
Technically it is allowed to enter any characters for *Table names* and *Column names*. But **you are STRONGLY ADVISED to use ASCII-ONLY names for this**! Even if *Creation of default tables* and *Checking of database columns* succeed, you may encounter some problems on further steps.





Sql import/export

Though, most probably you will use ASCII-only names for DB tables and columns, your SQL requests still may contain some Unicode symbols if you use respective string values (e.g. for comparison operator).

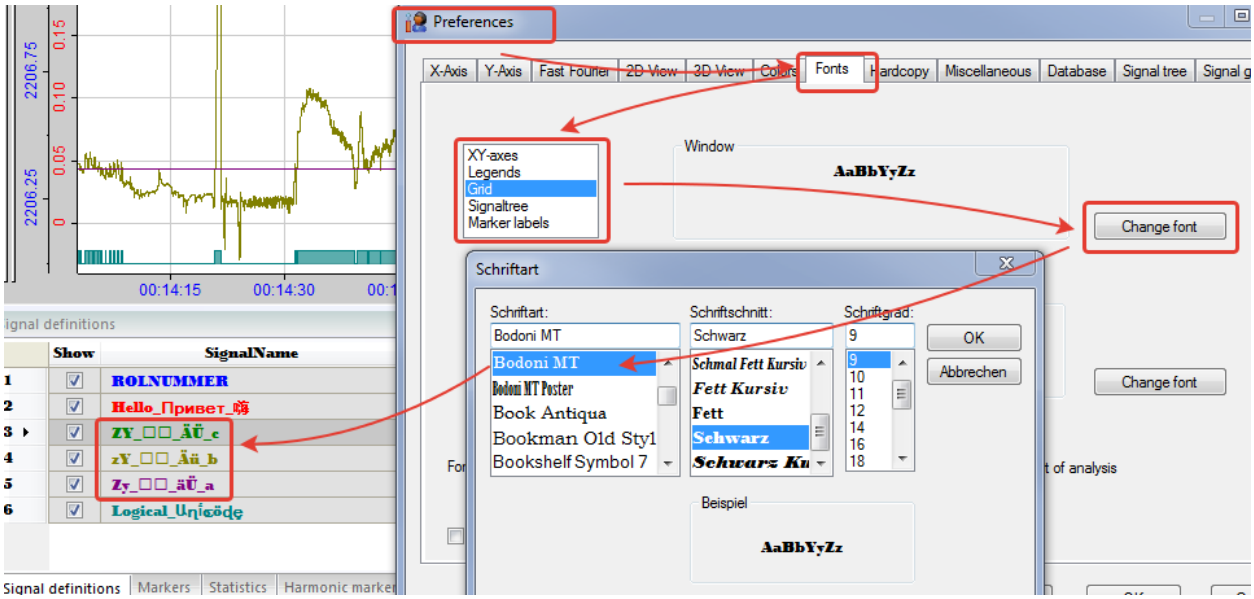


In the Unicode version of ibaAnalyzer format of *.sql files is UTF-8 with BOM, so all multilingual information is stored properly.

1.6 Fonts

Unicode standard contains more than 137 thousands of symbols, though most of *popular fonts do NOT cover all of these symbols*.

This means that it can happen that you type or paste some valid Unicode character which is not included into the font you are currently using. In this case this symbol will be displayed as (most likely) an empty rectangle. This is not a malfunction of ibaAnalyzer, but just a characteristic of fonts and font rendering mechanism. To solve such problem you may try to change the font in *Preferences -> Fonts*. For example, if you have some symbols incorrectly displayed in expressions window, try changing the font for the Grid. Change fonts for other GUI elements if necessary.



Anyway, even if some symbol is still not rendered correctly (displayed as an empty box) using some font, this does *not influence the functionality* in general. For example, if you have a problematic symbol in the signal name, you can still display such signal, reference it in expressions, drag-and-drop it, export it, use it for report generator, etc.

1.7 Compatibility

Backward compatibility

The Unicode version of ibaAnalyzer has full backward compatibility to all older data formats. This means that Unicode version of ibaAnalyzer can open (and process correctly) any file that was created by the old non-Unicode version of ibaAnalyzer (*.pdo, *.ini, *.zip, *.sql, etc.)

Forward compatibility

We tried to maximize *forward compatibility* as much as possible; though, obviously, sometimes it's not possible even theoretically.

The simple general rule is as follows. If you are not going to use old versions of ibaAnalyzer, you are encouraged to use all the benefits of Unicode.

On the other hand, if you are going to open your data regularly with old versions of ibaAnalyzer then try to avoid using anything but ASCII symbols for file names, signal names and anything else.

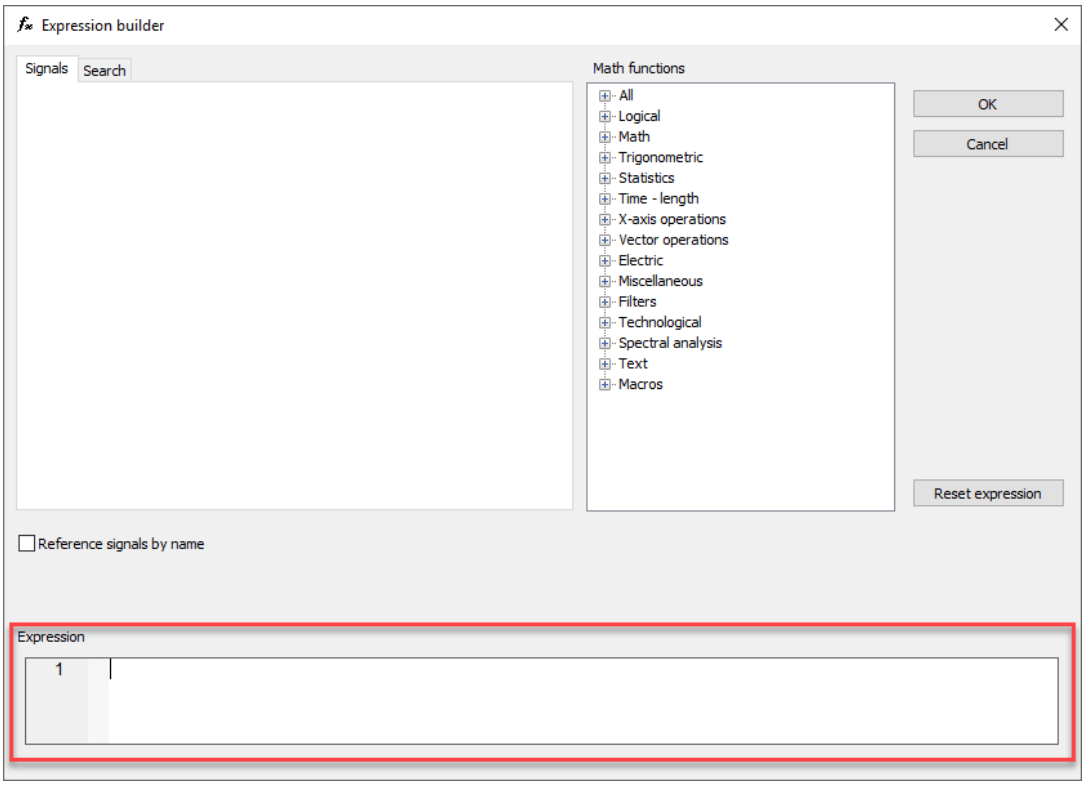
Generally, the old non-Unicode ibaAnalyzer will be able to recognize all ASCII symbols, all symbols from the current ANSI code page (selected in Windows control panel) and some other Unicode symbols that can be mapped to a valid symbol of current ANSI code page. Other symbols will be replaced with "?". Let's call such symbols 'incompatible symbols' for simplicity.

The table below describes typical situations that you can encounter if you try to handle ‘new’ files using an ‘old’ version of ibaAnalyzer.

Situation	Behavior of the non-Unicode ibaAnalyzer	Workaround, comments
A file is located in the path that contains some incompatible symbols	The file will not even be visible in ibaAnalyzer’s *Open file* window	Rename folders or move the file in such a way, that it has a path containing compatible symbols only
Filename has incompatible symbols	File will be visible, but its filename will be truncated to ANSI and you won’t be able to open it	Rename the file
Analysis has a signal inside that has incompatible symbols in its name, units, comments, etc.	Signal name (unit, comment, etc) will be truncated to ANSI. In all the other ways analysis will be fully functional.	Not needed
You are trying to load preferences from ‘new’ *.**ini** file.	File will be opened and processed successfully, though all strings inside will be truncated to ANSI.	Not needed. ‘old’ *.ini file has ANSI encoding. new *.ini file has “UTF-16 with BOM” encoding.
You are trying to load settings from ‘new’ *.**zip** file.	File will be opened and processed successfully, though all strings inside will be truncated to ANSI.	Not needed. Format itself was not changed. There is an *.xml file inside a *.zip file. *.xml file has UTF8 encoding in both ‘old’ and ‘new’ versions. The incompatibility appears only because of necessity to convert UTF8 to ANSI.
You are trying to open ‘new’ *.**sql** file.	File contents will be significantly corrupted.	Please convert the file by hand from ‘new’ UTF8 format to ‘old’ ANSI format, before trying to open it with the old ibaAnalyzer.
Some Global Macros and Global Filters are not visible	Note that Global Macros and Global Filters are stored as files. Non-Unicode version of ibaAnalyzer can fail to find them if you give non-ASCII names to them	Locate the *.mcr and *.fil files by hand and rename them to compatible names.

2 Multi-line input

In the current version of ibaAnalyzer the input of expressions over multiple lines and the auto formatting of arbitrary expressions is supported in the expression editor. Note that the behavior in the Signal grid is not changed and completely functions as before. The changes regard the Expression editor only.



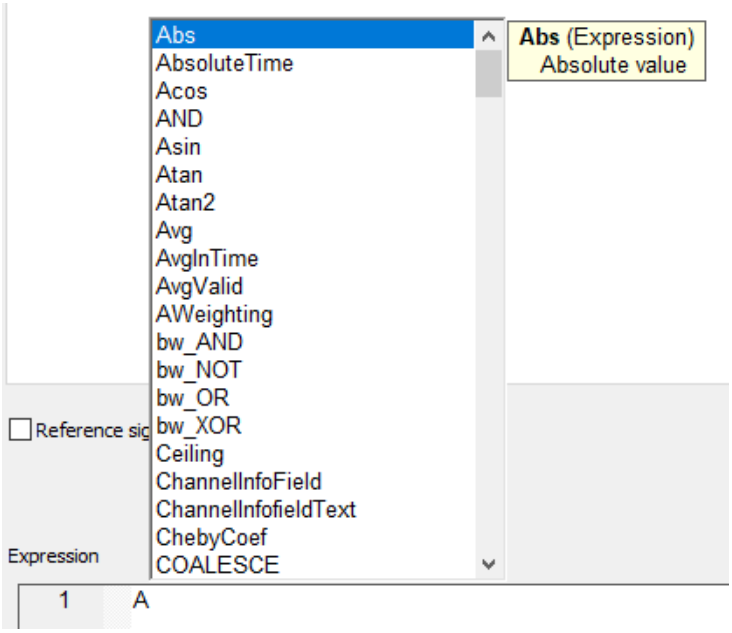
2.1 Typing

While typing in the expression window, ibaAnalyzer automatically detects *keywords* and *highlights* them using different colors. The following objects are recognized and individually formatted

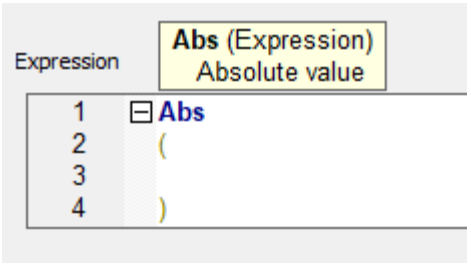
- Functions, like e.g. AvgInTime
- Signals, like e.g. [my-signal]
- Strings, like e.g. “text”
- Numbers

Further, brackets will be added automatically and the *indentation level is adjusted*.

Additionally, the usual *auto-complete* functionality is provided which enables fast typing and function selection.



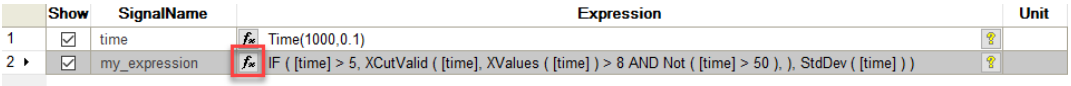
As usual, *tooltips* are displayed for every function in the dropdown menu and also for functions in the expression depending on the mouse courser position.



Note that complete freely configurable input (“new lines wherever you want”) is in principle possible, however it is *not stored and overwritten* by an automatic formatting. The auto-format is either triggered by entering a right round bracket “)” or by closing with “OK” and opening again.

2.2 Auto-formatting

The multi-line input has an *auto-format* functionality. This means, that all expressions are *automatically displayed over multiple lines* in the expression editor. An advantage of this pre-set auto format is that any pdo-file can be opened and all expression are automatically structured in the expression editor.

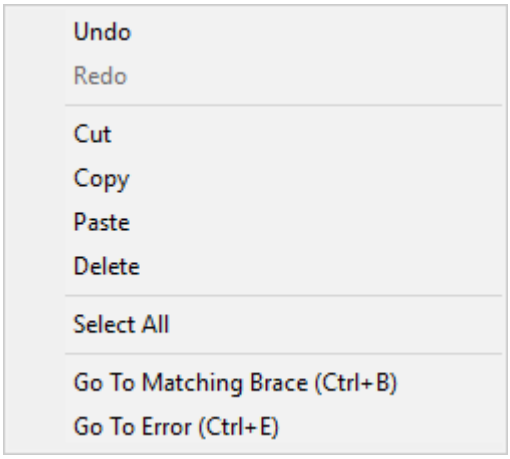


```
IF
(
    [time]
    >
    5,
    XCutValid
    (
        [time],
        XValues
        (
            [time]
        )
        >
        8
        AND
        Not
        (
            [time]
            >
            50
        ),
    ),
    StdDev
    (
        [time]
    )
)
```

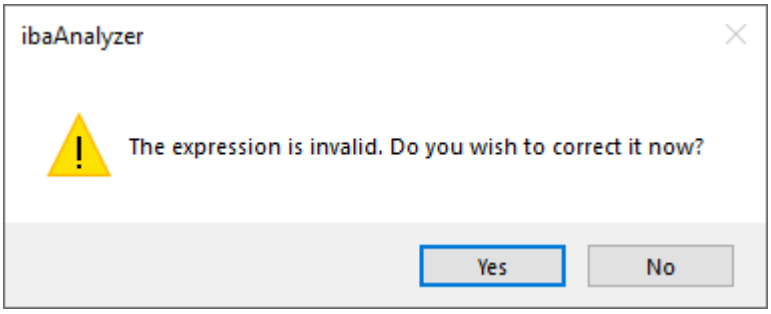
2.3 Error detection

For expressions which are not interpretable by ibaAnalyzer an automatic *error detection* is implemented. Functionality is as follows:

When opening an erroneous expression, the editor does an auto-format and the cursor automatically jumps to the *first position* where the expression could *not be parsed* any longer. The same behavior can be forced by pressing **Ctrl+E** or via the context menu in the multi-line input.



If an erroneous expression is entered in multi-line mode, the user is asked when pressing OK if he wants to correct the expression.



When pressing **No**, nothing happens and the expression editor is closed. When pressing **Yes**, the expression is again auto-formatted and the cursor jumps to the first position where the expression could not be parsed any longer.

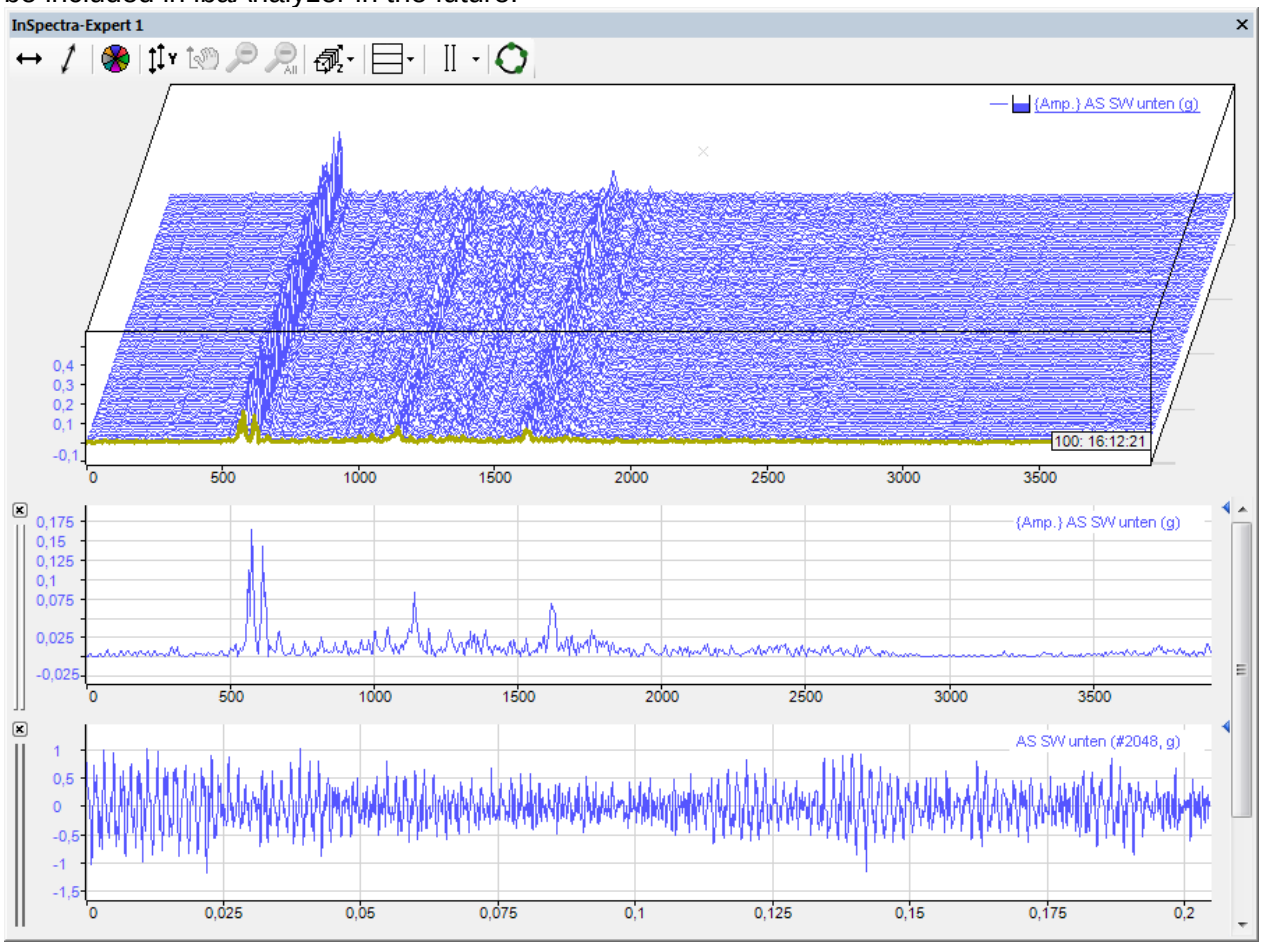
2.4 Further functionality

- Bracket jumping: Press **Ctrl + B** to jump between matching brackets. This functionality is also provided via the context menu.
- Bracket highlighting: Brackets with missing counterpart are indicated by a *red underline*
- Font size can be adjusted by pressing **Ctrl** and using the *mouse wheel*

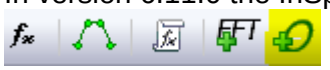
3 ibaAnalyzer-InSpectra

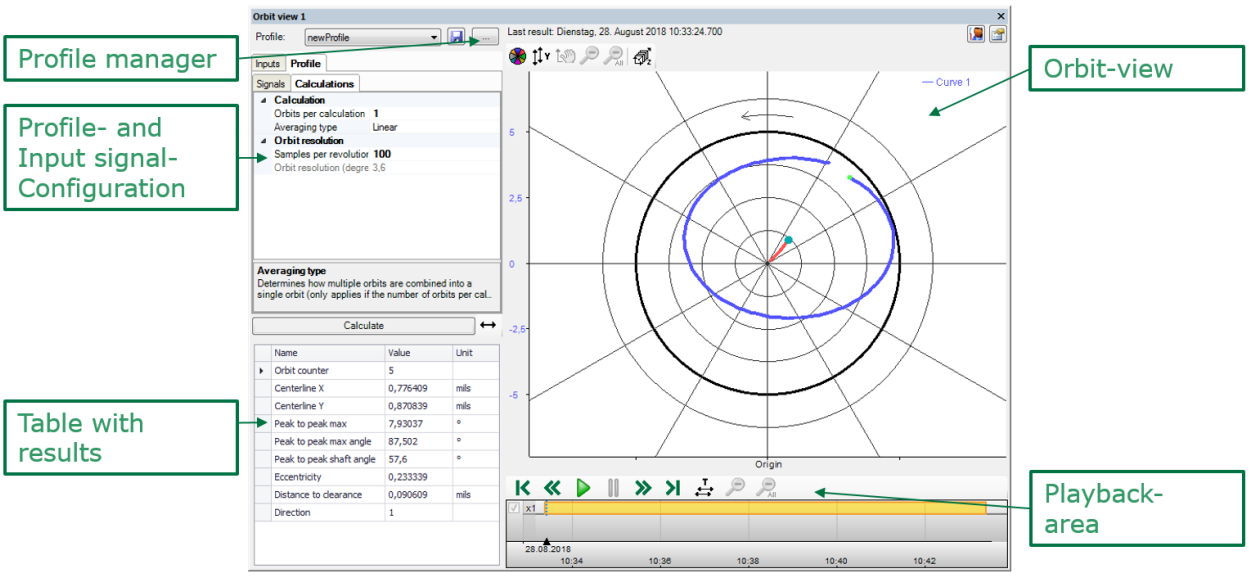
3.1 InSpectra-Expert

The FFT-view has been replaced by „InSpectra-Expert“. 
The complete functionality of the FFT-view in ibaPDA has been migrated to version 6.11.0 of ibaAnalyzer. Further, all functionalities of the previous FFT-view in ibaAnalyzer are included. Note that the InSpectra-Expert functionalities have not been migrated from ibaPDA yet, and will be included in ibaAnalyzer in the future.



3.2 InSpectra-Orbit

In version 6.11.0 the InSpectra-Orbit-Module from ibaPDA was added to ibaAnalyzer.

The new concept contains exactly the view from ibaPDA including all functionalities. Inputs and profiles can be configured on the left side of the view. The settings and options are the same as in the online InSpectra in ibaPDA and profiles are exchangeable with ibaPDA. Results and characteristic values calculated by the module based on the actual configuration are shown in the table below. For navigating through the data, a playback area is located under the view.



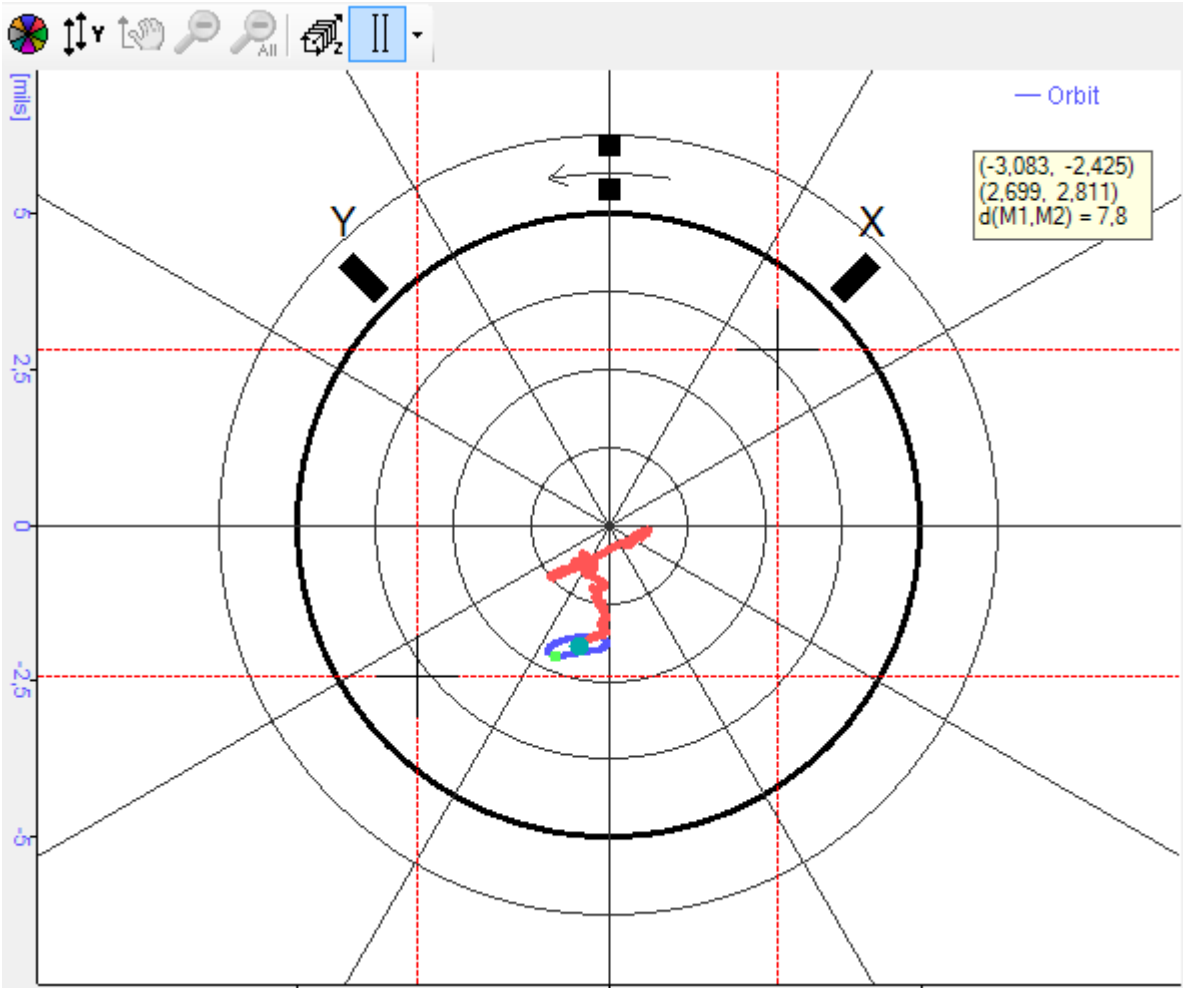
With ibaInSpectra-Orbit for instance the shaft movement inside a sleeve bearing can be monitored, analyzed and visualized. The calculated results are:

- **Orbit counter**
  Number of orbits since start of acquisition
- **X, Y**
  X- and Y-component of the shaft position, provided by proximity probes (actual value, compensated)
- **Centerline X, Y**
  Average X- and Y-position over the orbits of one calculation
- **Peak to peak…**
  - **max**: Max. distance between the points of all orbits of a calculation (=Sppmax)
  - **max angle**: Angle of the shaft position at Sppmax in the X/Y coordinates system
  - **max shaft angle**: Rotational angle of the shaft at Sppmax
- **Eccentricity**
  Values between 0 (=good) and 1 (= bad)
- **Distance to clearance**
  given in mils or µm (sensor units)

The visualization contains:

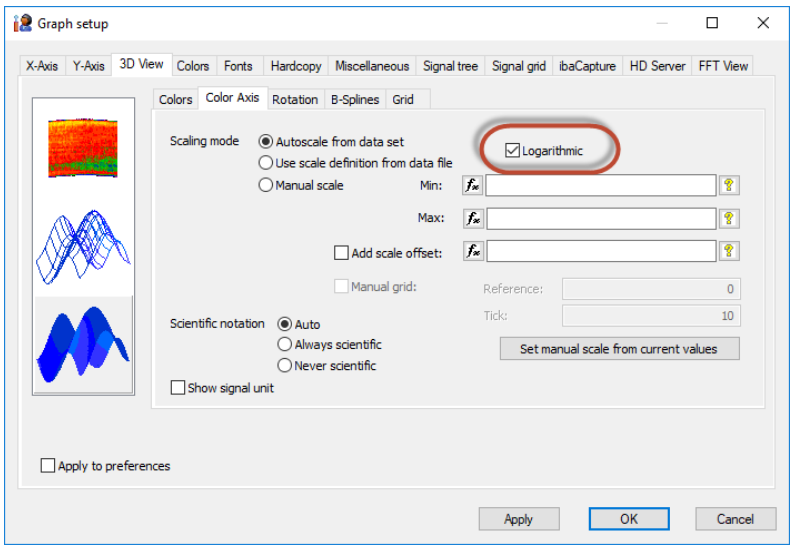
- Orbit (blue)
- Centerline (red)
- Key phasor (green)
- Clearance circle (black)
- Speed steps
- Sensor positions
- Direction of rotation
- Two marker
- Legend with marker positions

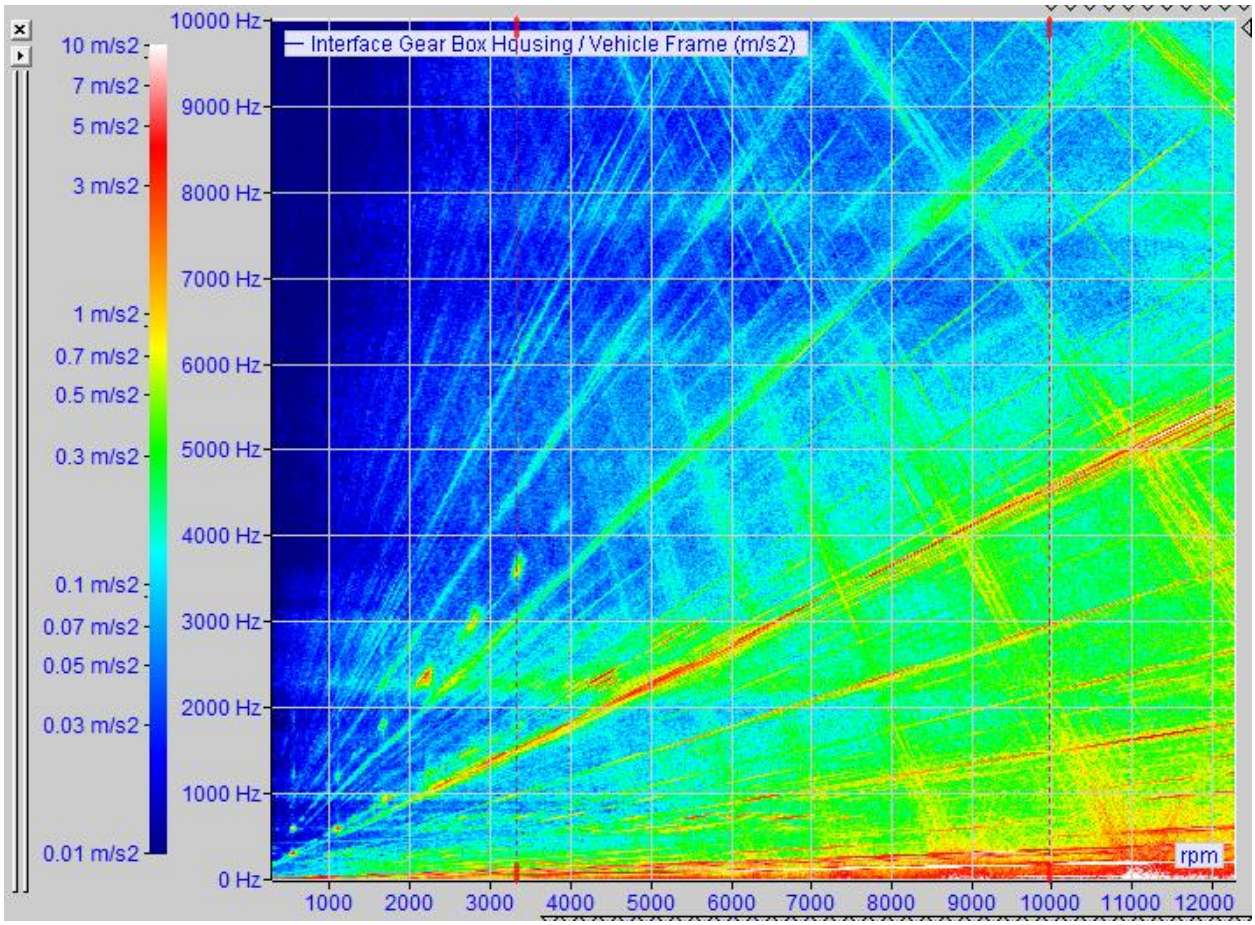
Further information on InSpectra-Orbit can be achieved with the manual or via ibaSupport.


[mils]
5
2,5
0
-2,5
-5
Y
X
— Orbit
(-3,083, -2,425)
(2,699, 2,811)
d(M1,M2) = 7,8

4 Logarithmic scaling

Logarithmic scaling of the value axis is 2D contour plots and 3D plots are possible since version 6.10.3 of ibaAnalyzer. In previous versions of ibaAnalyzer, this option was present (inherited from Y-axis properties) but the scaling was not implemented and hence not applied.

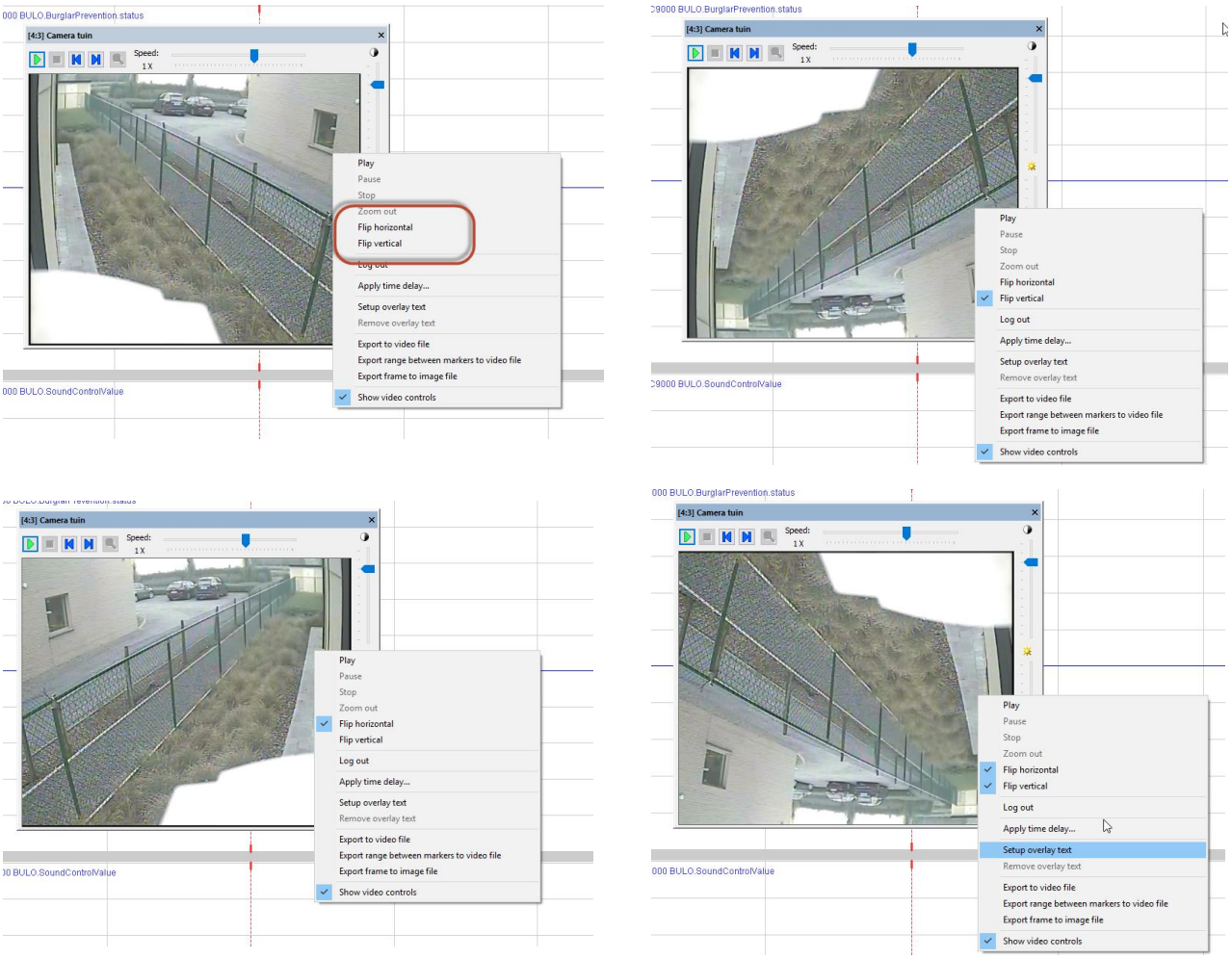




5 ibaCapture modifications

5.1 Ability to flip the image vertically or horizontally

If version 4.2.2 or more recent of the ibaCapture player is installed, one can select to flip the camera image horizontally or vertically by right clicking on the video window and selecting the option from the context menu. This can also be disabled again from the context menu.

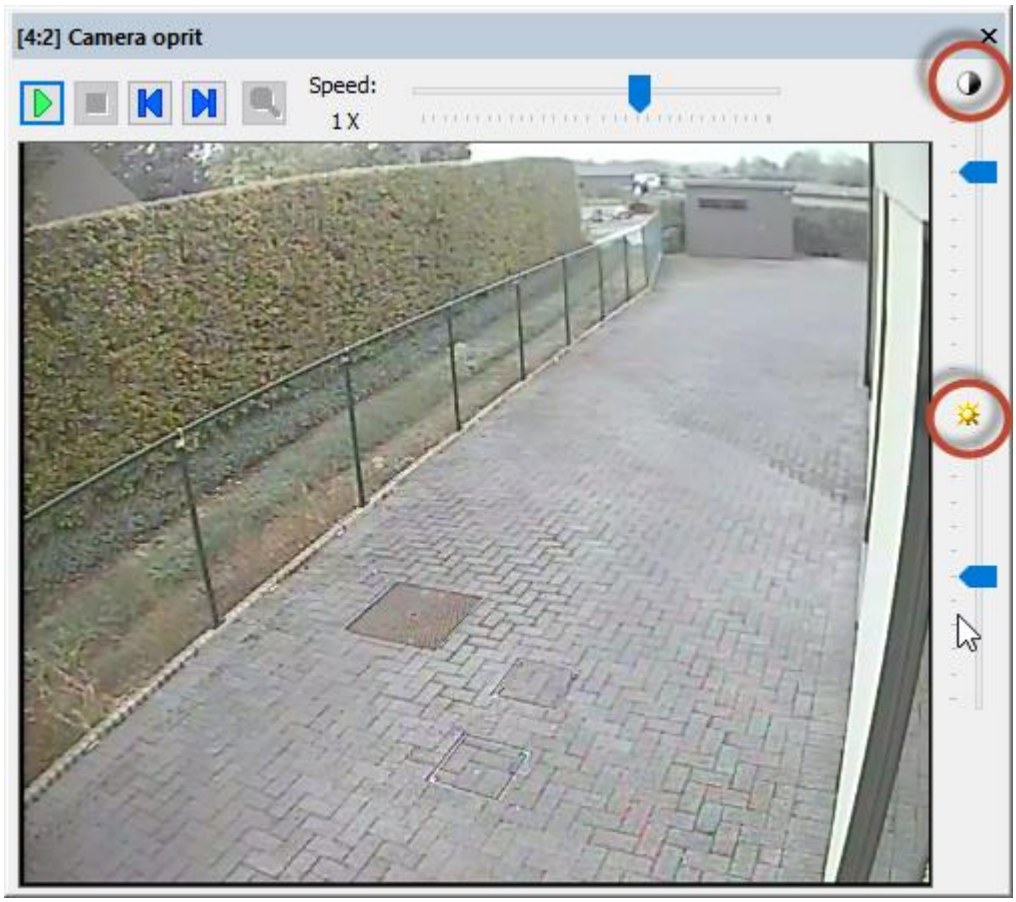


5.2 Ability to stream embedded video immediately from the .dat file.

If version 4.2.2 or more recent of the ibaCapture player is installed, when one loads a .dat file which has embedded video files in it (embedded by ibaAnalyzer extract or export), any videos will load immediately without the embedded videos being unpacked first. The .dat file itself with an appropriate offset is immediately streamed by the player.

5.3 Ability to reset brightness and contrast to their default values.

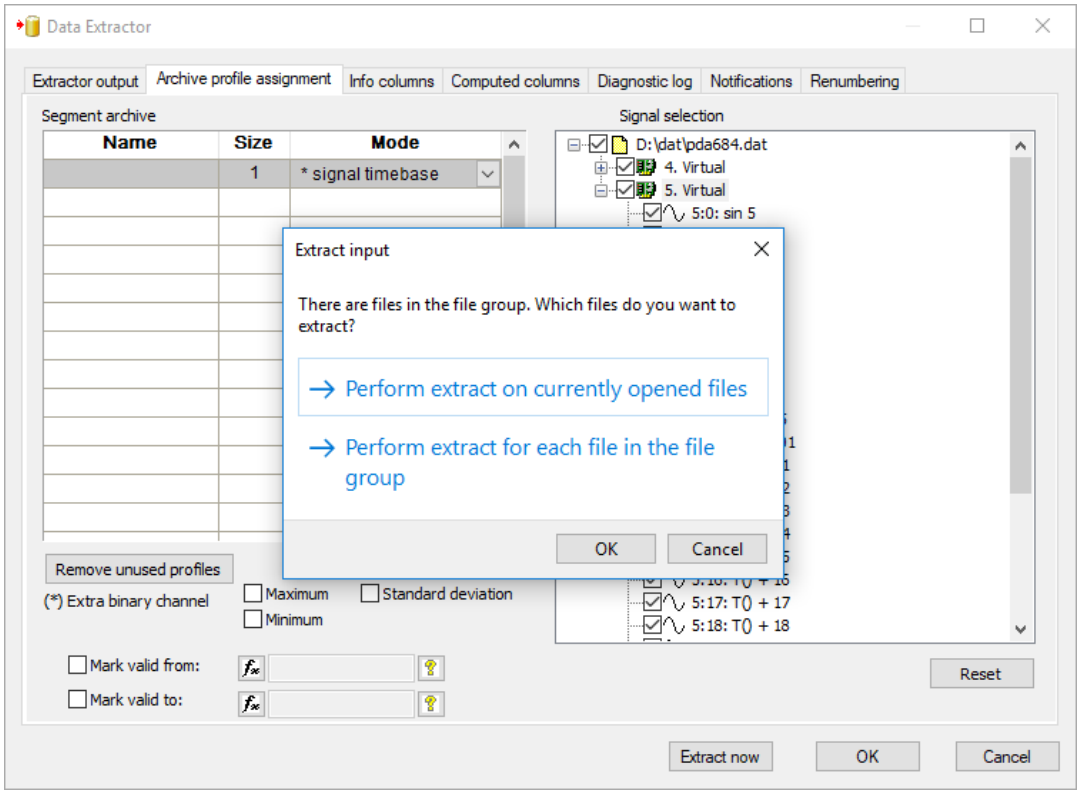
By clicking the respective icon on top of the slider, the contrast or brightness is reset to its default value.



6 Export / extract

6.1 Batch extracting of file group entries.

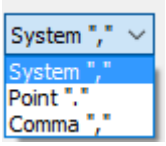
When performing an extract operation interactively when there are entries in the file group, one will be asked if one wishes to perform the extract on the currently opened `.dat` file or on the entries in the file group.



We envision that this functionality will be especially useful for extracting multiple entries resulting from an HD server query.

6.2 Decimal character selection

When extracting or exporting data to the CSV text file format, one now has the option to specify the decimal character used in the values.



The available options are:

- Use the character determined by the Windows regional settings, this is the default and was the behavior of ibaAnalyzer prior to version 6.10.4.
- Use a dot '.'
- Use a comma ','

The functionality is available for:

- Extracting to CSV file.
- Exporting to CSV file.
- Selecting to export a graph to clipboard or CSV file from the context menu (accessible by right clicking on a graph)