# New Features in ibaAnalyzer 6.5.0

Author: Igor Jacques, Michael Verschaeve

Date: 30 September 2014

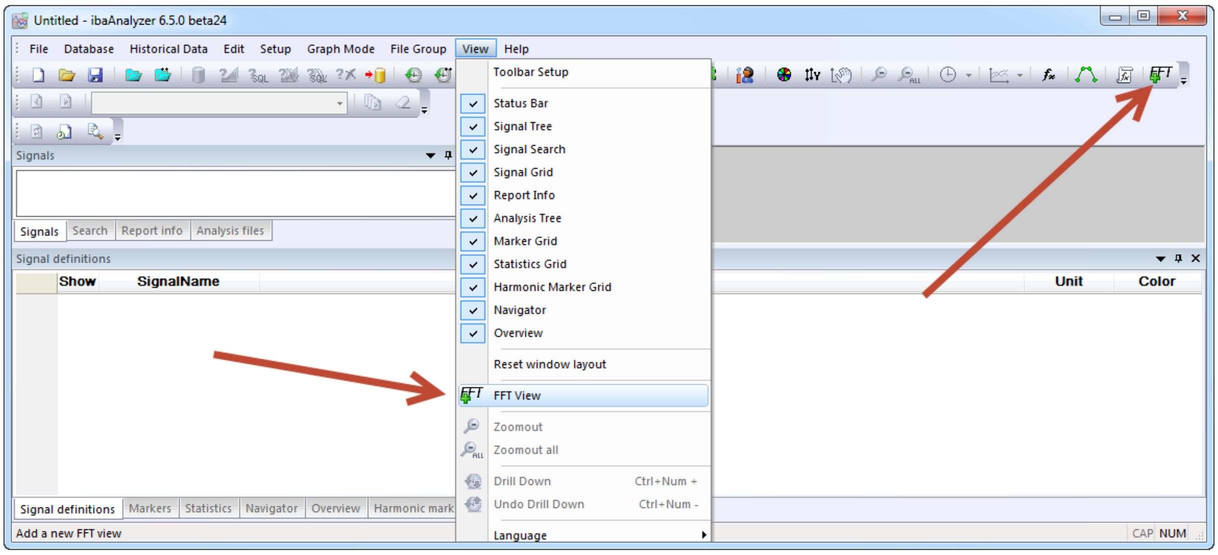
# Table of contents

# 1 FFT integration

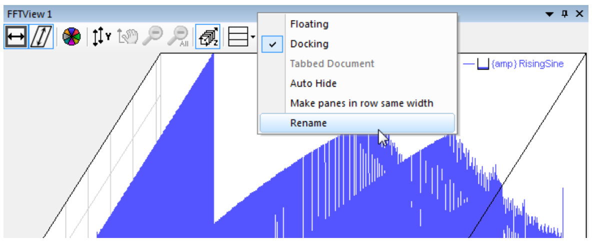
The ibaPda FFT View was integrated into ibaAnalyzer to facilitate offline frequency analysis.

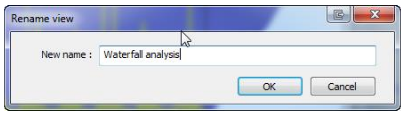
## 1.1 General

For each sensor signal that needs to be analyzed, one can create a dockable “FFT view window” via the toolbar or the View menu:


Untitled - ibaAnalyzer 6.5.0 beta24
File Database Historical Data Edit Setup Graph Mode File Group View Help
Toolbar Setup
Status Bar
Signal Tree
Signal Search
Signal Grid
Report Info
Analysis Tree
Marker Grid
Statistics Grid
Harmonic Marker Grid
Navigator
Overview
Reset window layout
FFT View
Zoomout
Zoomout all
Drill Down Ctrl+Num +
Undo Drill Down Ctrl+Num -
Language
Signals Search Report info Analysis files
Signal definitions
Show SignalName Unit Color
Signal definitions Markers Statistics Navigator Overview Harmonic mark
Add a new FFT view
CAP NUM

After creation, the FFT view will have a title bar labeled “FFTView” followed by an index. This name can be changed by right-clicking the title bar and selecting “rename” from the context menu:

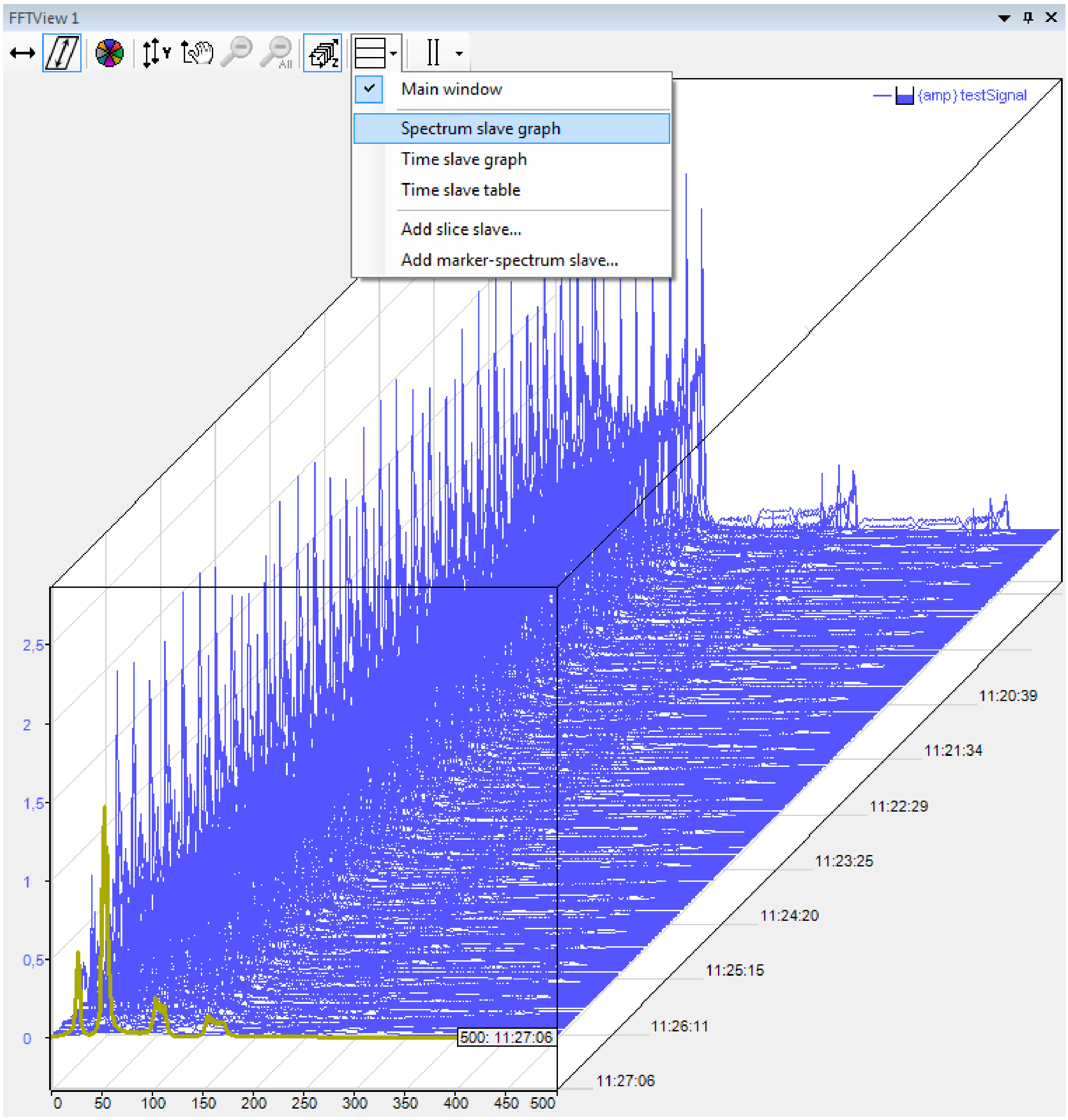




To analyze a signal, one can either

- Drag an analog signal from the signal tree in the FFT view
- Drag an analog signal from an ibaAnalyzer graph into the FFT view

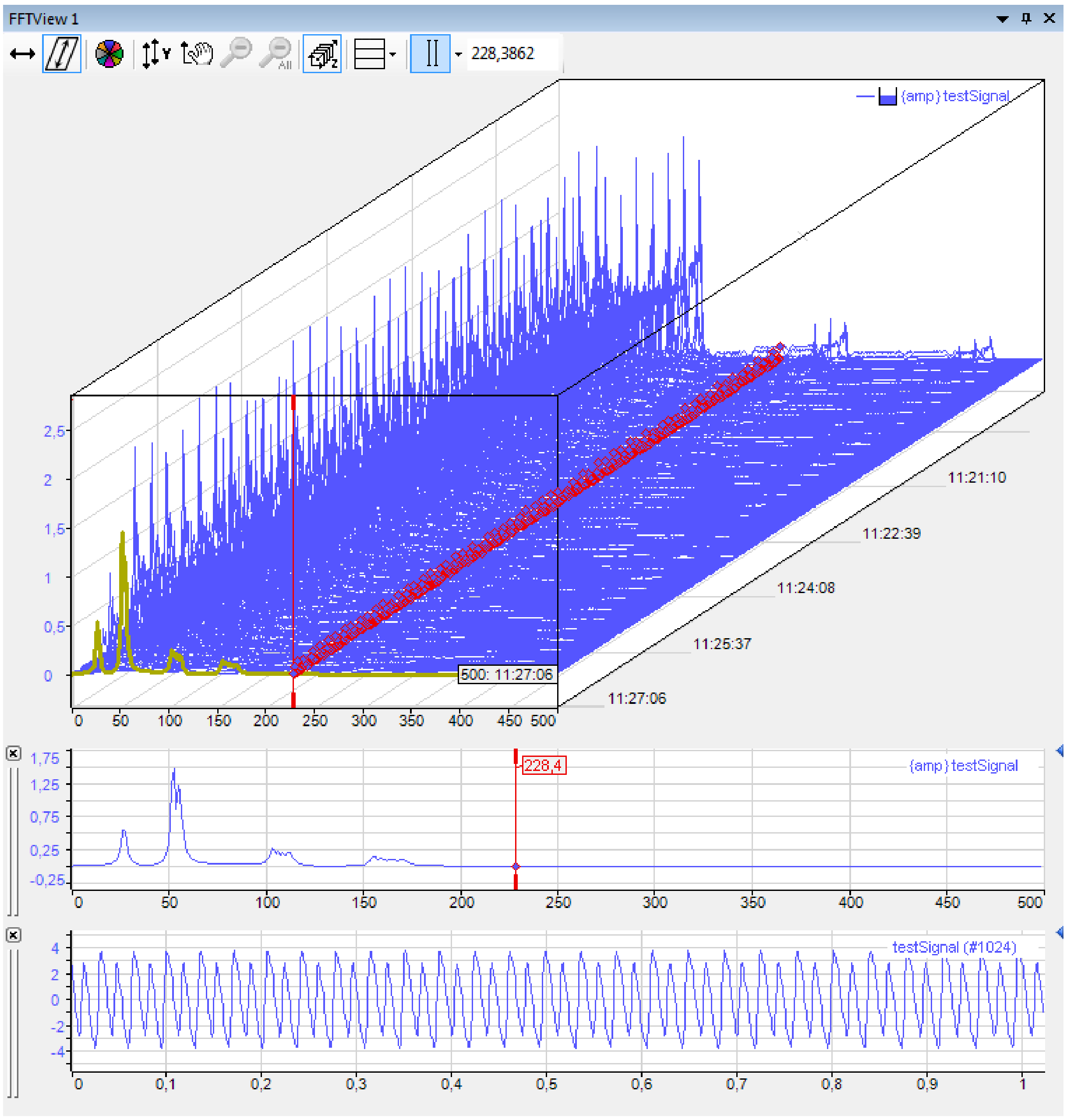
After that, one will see the waterfall of the signal. In the example below, you see a waterfall plot, with 500 planes. Each plane corresponds to one FFT. One can change the selected plane by scrolling.



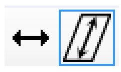
To make a deeper analysis of one plane, one can enable slave windows:

- Spectrum slave graph: displays the spectrum of the plane selected in the waterfall

- Time slave graph: displays the time signal based on which the FFT was calculated



The waterfall of the FFT view will not always display the complete .dat file. At the left of the toolbar in the screenshot, there are two important buttons:

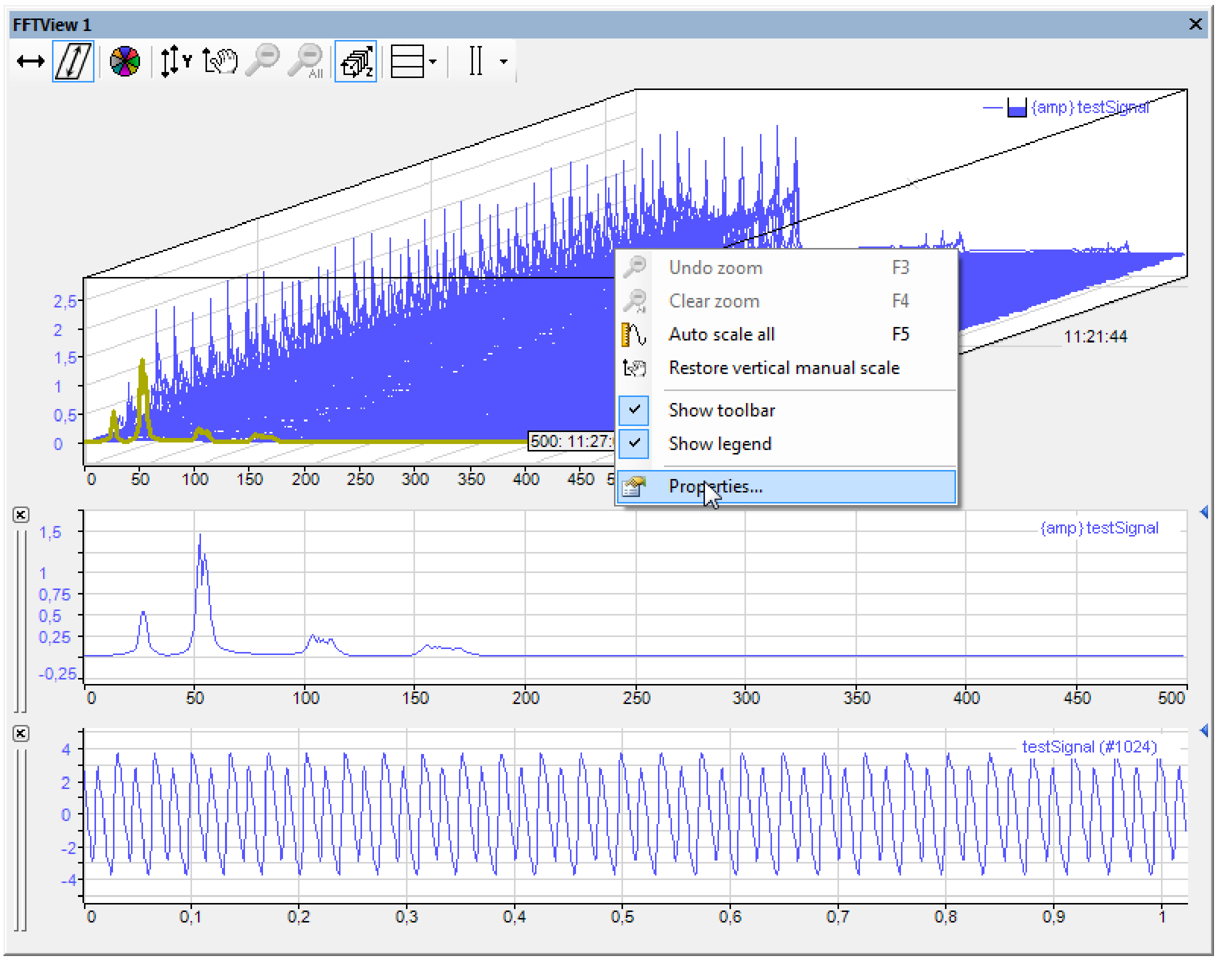


If the first button is checked, the FFT view will only analyze the data that is in the **ibaAnalyzer zoom area**. You can easily change the zoom area via the ibaAnalyzer navigator window. The FFT view does not use zero padding. The remaining data is not used, nor displayed.

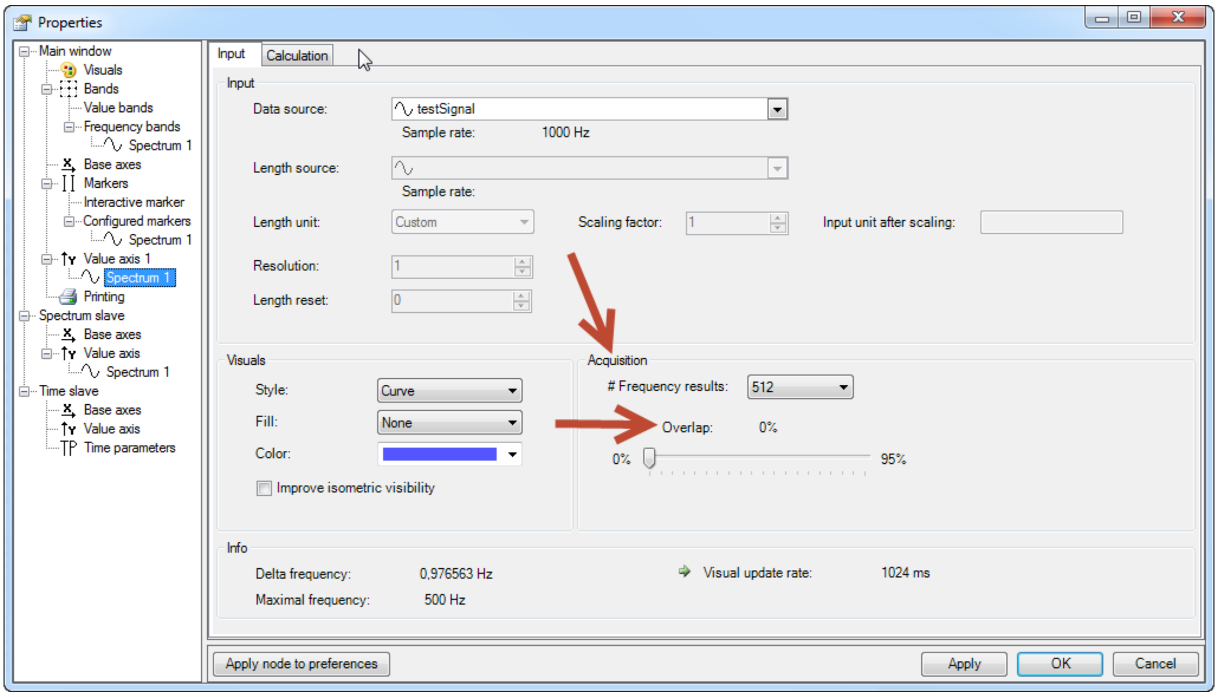
If the second button is checked, the **number of planes** in the FFT view is **adapted automatically**, matching the size of the dat file. In case the number of samples for one FFT is very high (e.g. >1000000), the number of planes is limited automatically, thereby limiting the memory use. In case of 1048576 samples per plane, the number of planes is limited to 25.

## 1.2 Calculation settings

The calculation settings can be changed via the context menu of the FFT view.



If you click on the "Spectrum 1" node in the tree on the left, you can change the mathematical details of the FFT. The most important setting is the number of Frequency results. If the option "Mechanical notation" in the base axes node is off, then the number of frequency results is half the number of samples used for one FFT. You can also specify an overlap percentage.



Input

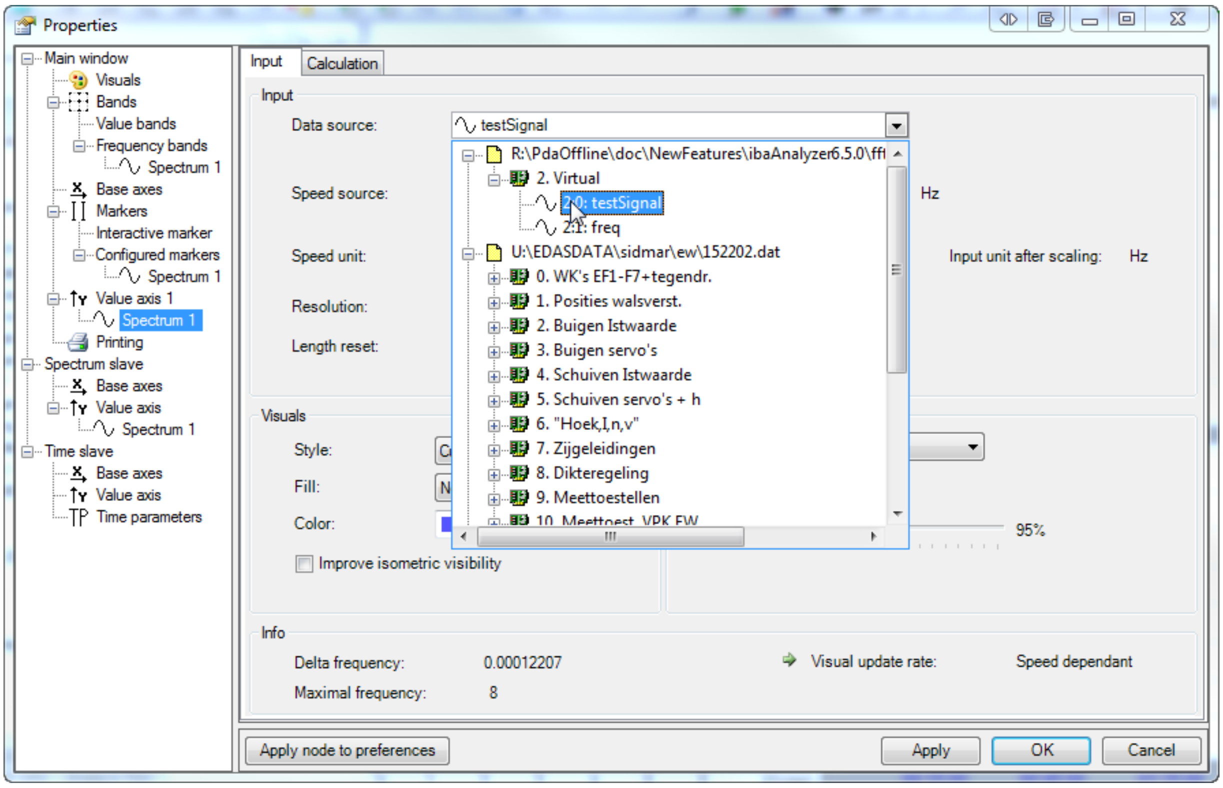
Data source: testSignal
Sample rate: 1000 Hz
Length source:
Sample rate:
Length unit: Custom
Scaling factor: 1
Input unit after scaling:
Resolution: 1
Length reset: 0

Visuals
Style: Curve
Fill: None
Color:
Improve isometric visibility

Acquisition
# Frequency results: 512
Overlap: 0%
0% 95%

Info
Delta frequency: 0,976563 Hz
Maximal frequency: 500 Hz
Visual update rate: 1024 ms

The data sources can be chosen or changed from an ibaAnalyzer signal tree in the dropdown box.



Input

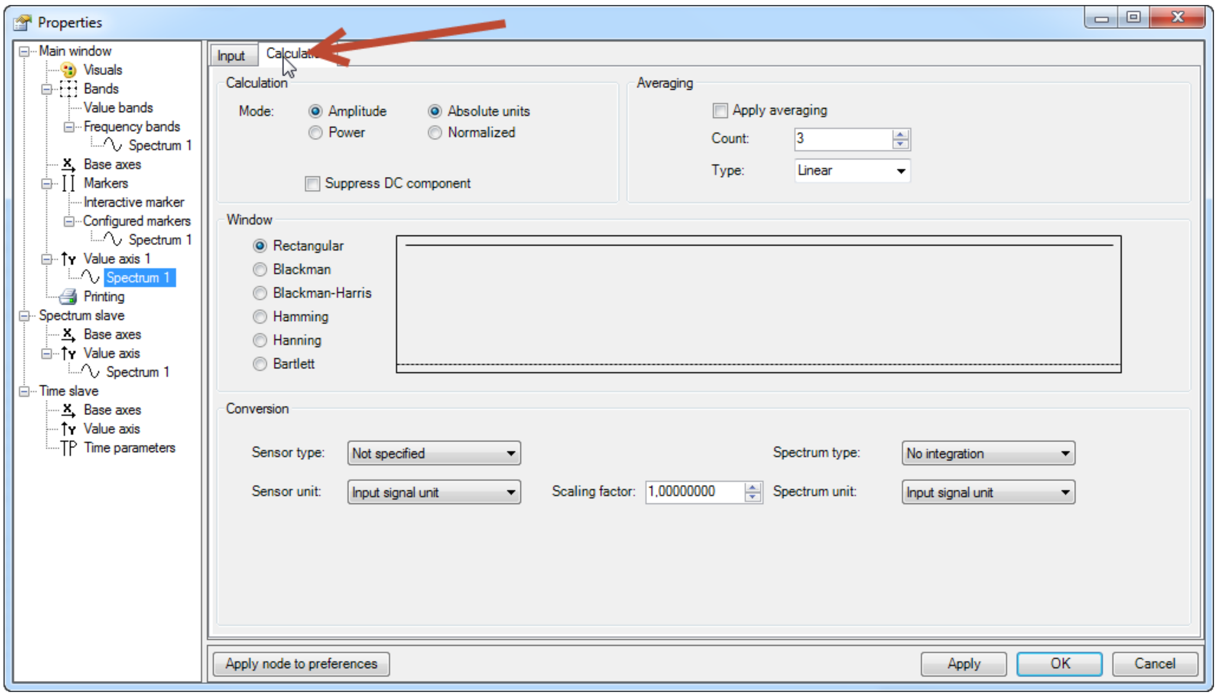
Data source: testSignal
R:\PdaOffline\doc\NewFeatures\ibaAnalyzer6.5.0\ffi
2. Virtual
2.0: testSignal
2.1: freq
U:\EDASDATA\sidmar\ew\152202.dat
0. WK's EF1-F7+tegendr.
1. Posities walsverst.
2. Buigen Istwaarde
3. Buigen servo's
4. Schuiven Istwaarde
5. Schuiven servo's + h
6. "Hoek,I,n,v"
7. Zijgeleidingen
8. Dikteregeling
9. Meettoestellen
10. Meettoest. VPK EW

Speed source:
Hz
Speed unit:
Input unit after scaling: Hz
Resolution:
Length reset:

Visuals
Style:
Fill:
Color:
Improve isometric visibility
95%

Info
Delta frequency: 0.00012207
Maximal frequency: 8
Visual update rate: Speed dependant

More settings can be set in the Calculation TAB:


Properties

- Main window
  - Visuals
  - Bands
    - Value bands
    - Frequency bands
      - Spectrum 1
  - Base axes
  - Markers
    - Interactive marker
    - Configured markers
      - Spectrum 1
  - Value axis 1
    - Spectrum 1
  - Printing
- Spectrum slave
  - Base axes
  - Value axis
    - Spectrum 1
- Time slave
  - Base axes
  - Value axis
  - Time parameters

Input | Calculation

Calculation
Mode: [✓ Amplitude] Power | [✓ Absolute units] Normalized
☐ Suppress DC component

Averaging
☐ Apply averaging
Count: 3
Type: Linear

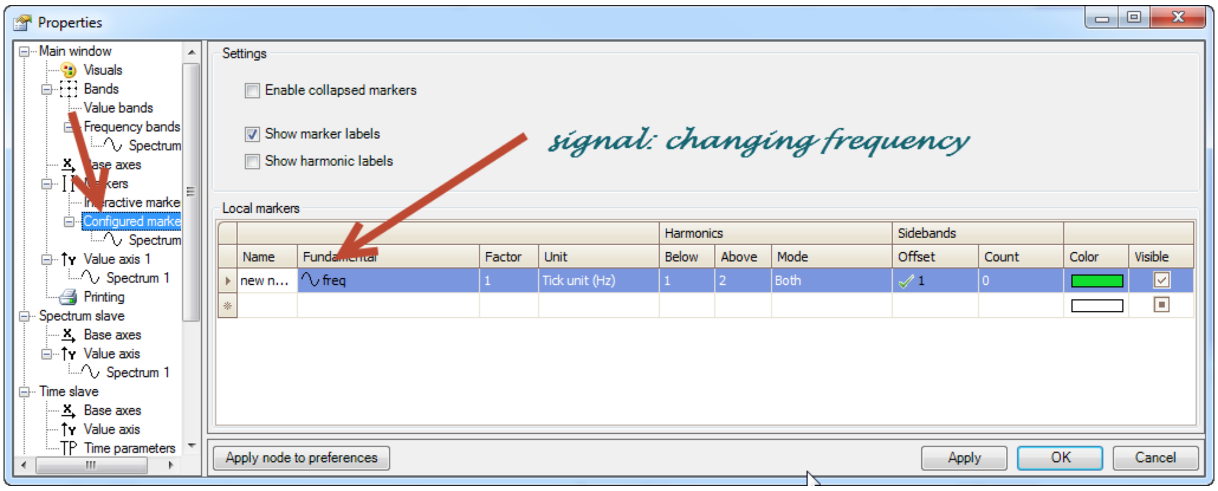
Window
[✓ Rectangular] Blackman, Blackman-Harris, Hamming, Hanning, Bartlett

Conversion
Sensor type: Not specified
Sensor unit: Input signal unit
Scaling factor: 1,00000000
Spectrum type: No integration
Spectrum unit: Input signal unit

There is support for power/amplitude, normalization, scaling, averaging, windowing, integration/differentiation and units.

## 1.3 Slice slave

Next to the spectrum and the time slave, the FFT view contains two additional types of slaves. The slice slave is one of them. One can use it to study the spectrum value at a moving frequency. First, you have to define a marker in the waterfall and attach this marker to a signal:



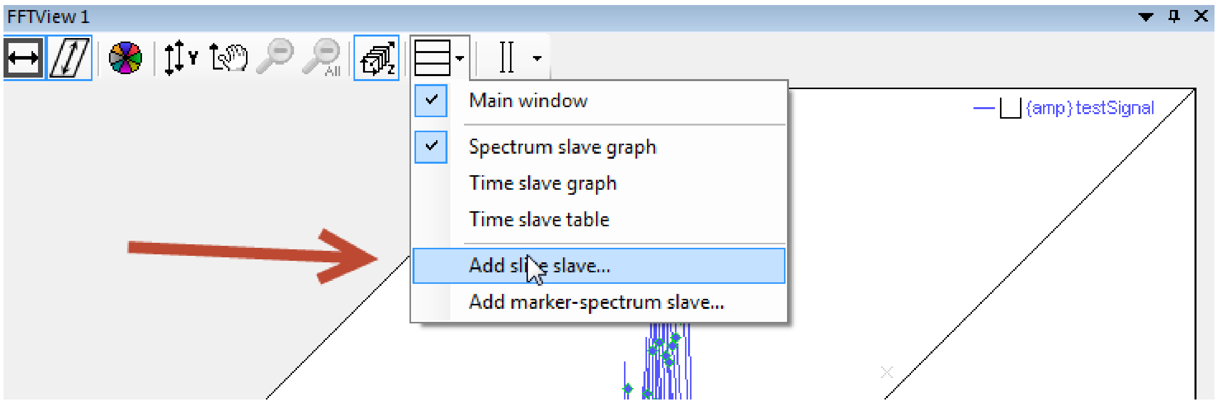
Settings

- [ ] Enable collapsed markers
- [✓] Show marker labels
- [ ] Show harmonic labels

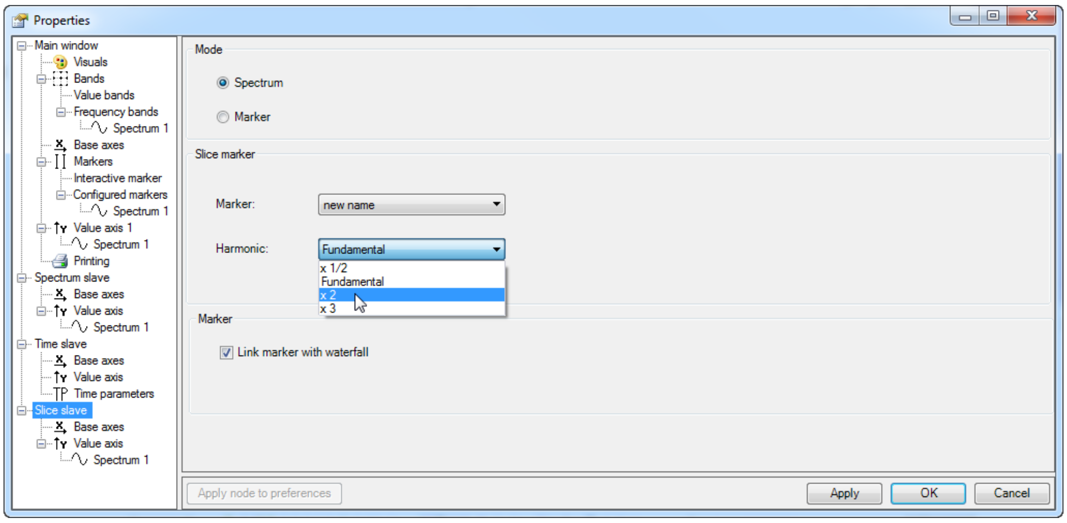
Local markers

| | | | | Harmonics | | | Sidebands | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| Name | Fundamental | Factor | Unit | Below | Above | Mode | Offset | Count | Color | Visible |
| new n... | freq | 1 | Tick unit (Hz) | 1 | 2 | Both | ✓ 1 | 0 | 🟩 | ✓ |
| | | | | | | | | | ⬜ | |

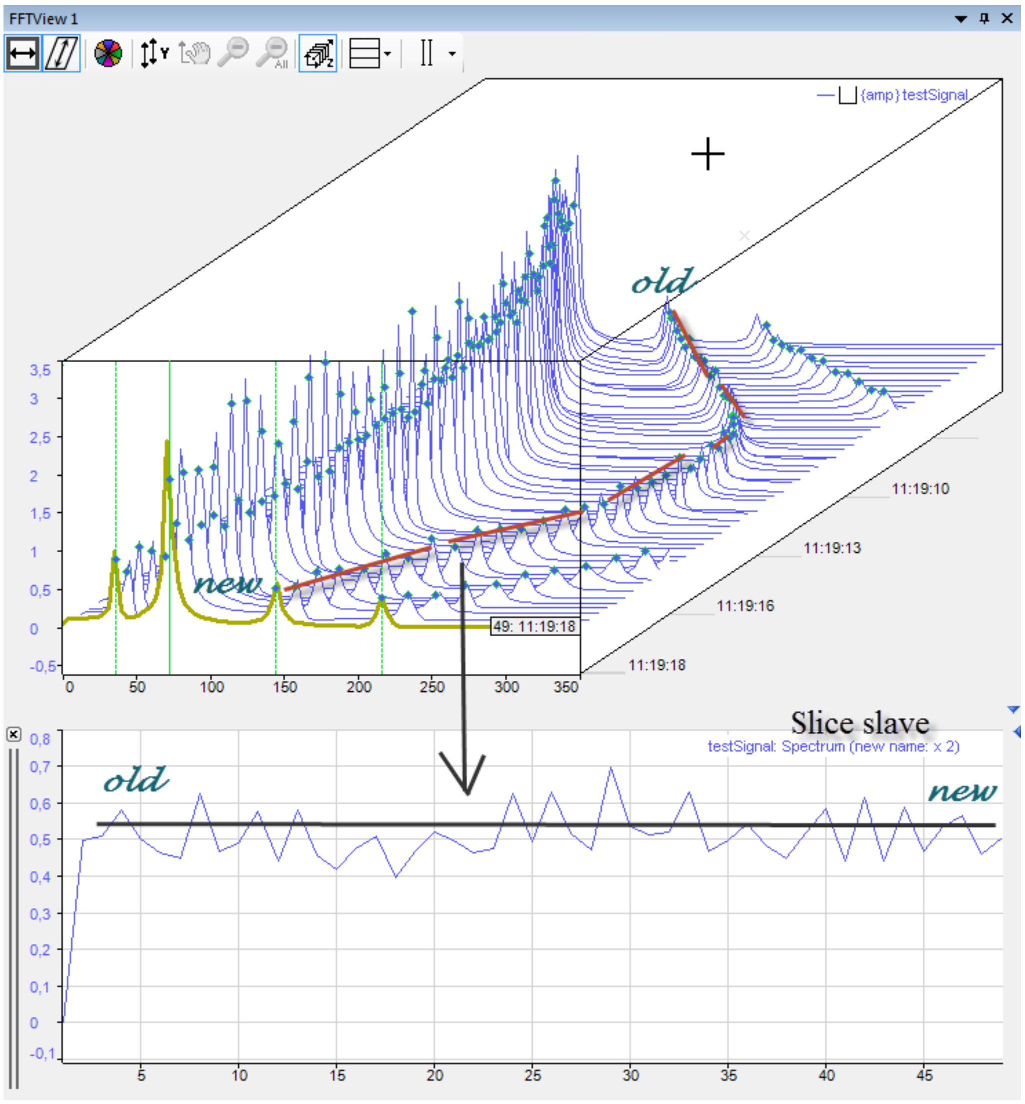
Then you can add a slice slave window. If you click to add the window, the property window will appear automatically.



You choose the marker and the harmonic for the slice slave:

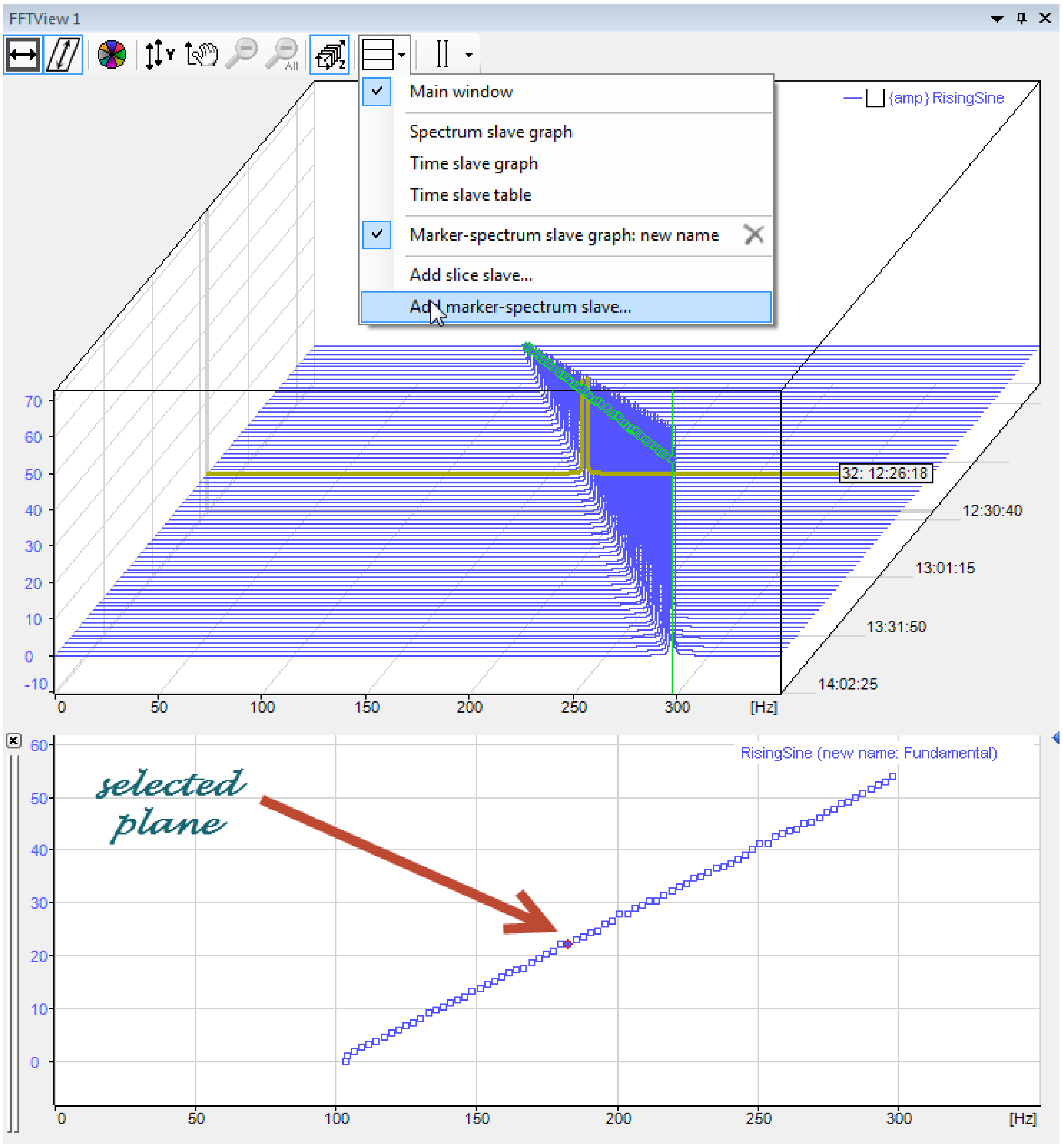


We have chosen to link the slice slave to the second harmonic of the marker named “new name”.

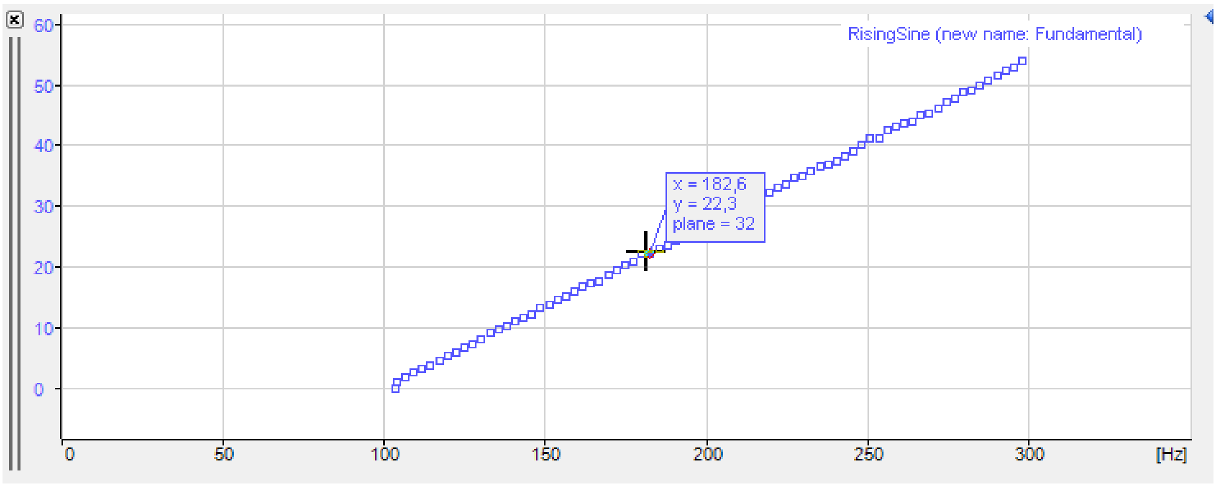


## 1.4 Marker-spectrum slave

A marker-spectrum slave can plot “the height of the spectrum at a dynamic frequency”, against this dynamic frequency. One can link this marker-spectrum slave graph to any marker present in the waterfall. In the screenshot below, the bottom marker-spectrum slave is linked to the green marker.



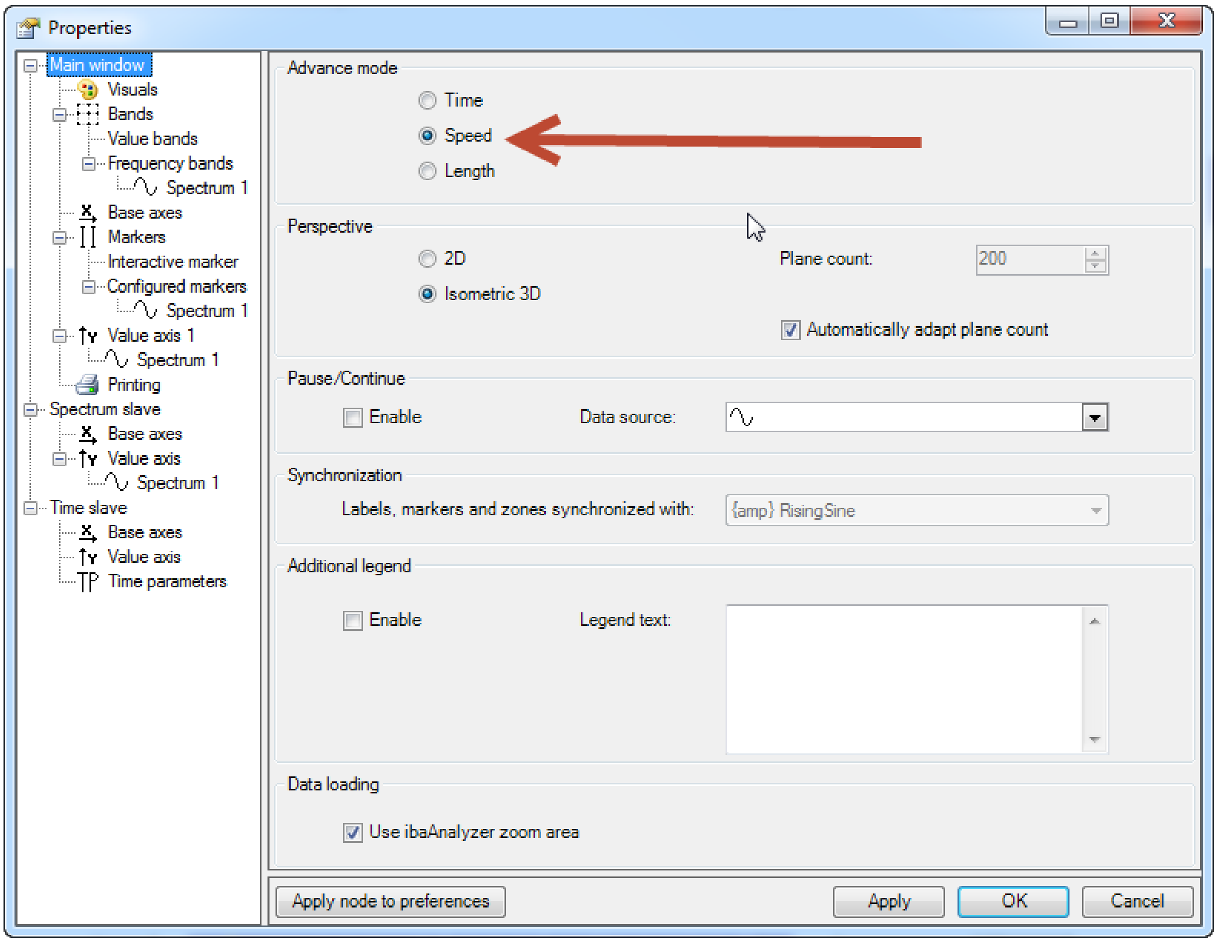
One can hoover the selected dot in the marker-spectrum slave, to see a tooltip providing the number of the plane (here it is 32).



## 1.5 Order calculations (Order spectrum analysis)

The FFT view supports FFT calculations in order. An order spectrum is like a normal FFT spectrum, but it is normalized against one specific parameter of the process, e.g. the speed at which an object is moving through a production line or the rotational speed of a motor.

Two modes are available: speed and length mode. Here, we shortly discuss an example in speed mode.



Advance mode

Time [✓ Speed] Length

Perspective

2D [✓ Isometric 3D]

Plane count: 200

[✓] Automatically adapt plane count

Pause/Continue

[ ] Enable Data source:

Synchronization

Labels, markers and zones synchronized with: {amp} RisingSine

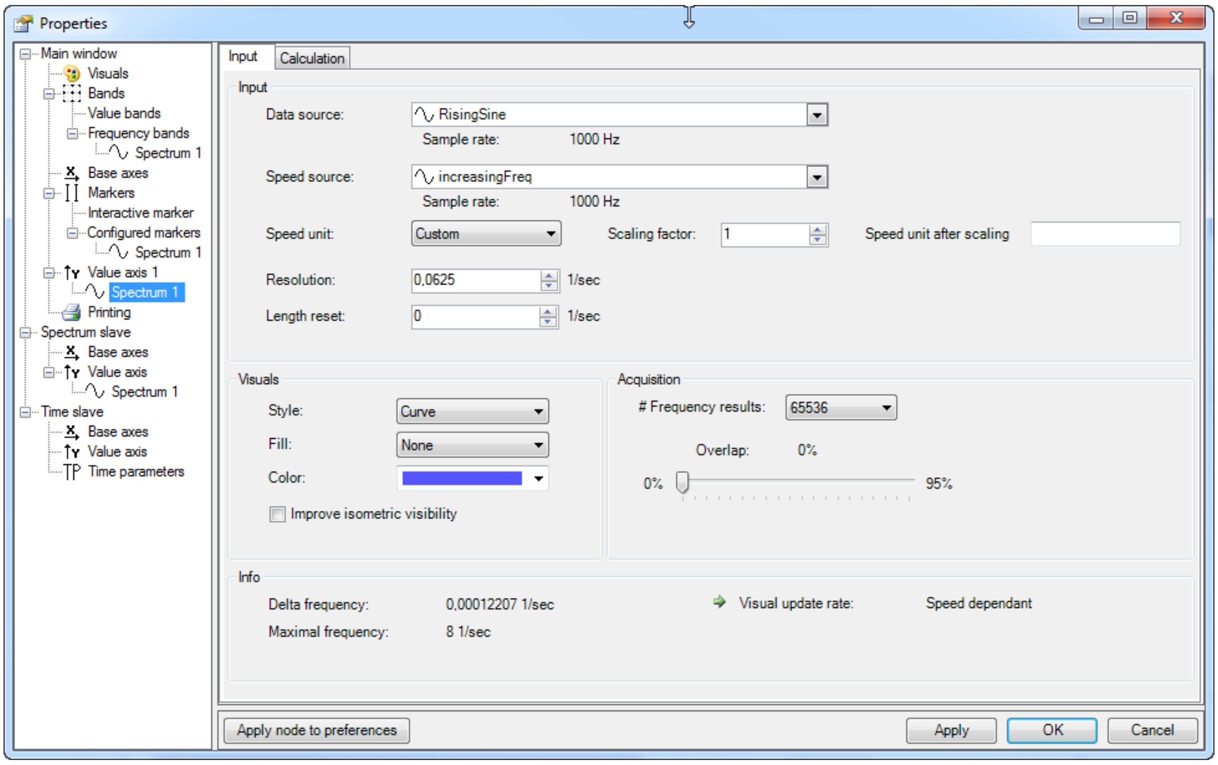
Additional legend

[ ] Enable Legend text:

Data loading

[✓] Use ibaAnalyzer zoom area

The mathematical details of the order calculation can be set in the following dialog:


Properties

Main window
- Visuals
- Bands
  - Value bands
  - Frequency bands
    - Spectrum 1
- Base axes
- Markers
  - Interactive marker
  - Configured markers
    - Spectrum 1
- Value axis 1
  - Spectrum 1
- Printing

Spectrum slave
- Base axes
- Value axis
  - Spectrum 1

Time slave
- Base axes
- Value axis
- Time parameters

Input | Calculation

Input
Data source: RisingSine
Sample rate: 1000 Hz
Speed source: increasingFreq
Sample rate: 1000 Hz
Speed unit: Custom
Scaling factor: 1
Speed unit after scaling
Resolution: 0,0625 1/sec
Length reset: 0 1/sec

Visuals
Style: Curve
Fill: None
Color:
Improve isometric visibility

Acquisition
# Frequency results: 65536
Overlap: 0%
0% 95%

Info
Delta frequency: 0,00012207 1/sec
Maximal frequency: 8 1/sec
Visual update rate: Speed dependant

Apply node to preferences
Apply OK Cancel

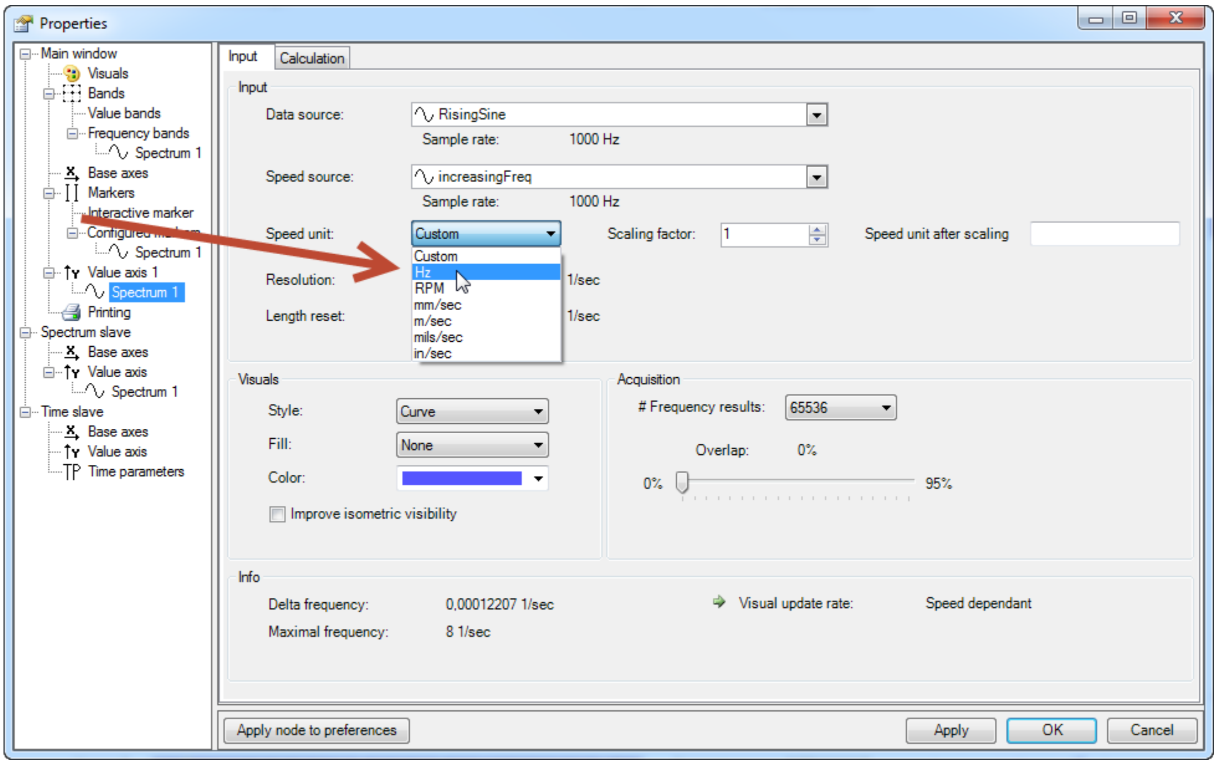
The most important settings are:

- The number of frequency results
  - In this example, this number is half the number of samples for one FFT.
  - In case the option “mechanical notation” is on, this number is the number of samples for one FFT divided by 2.56.
- The speed/length source. This signal can be selected from the ibaAnalyzer signal tree.
- The speed/length unit
- The resolution (simplified explanation):
  - Each time that the integral of the speed signal has increased with this amount, the vibration signal is sampled once.
  - In case the mode is not speed but length, the same applies, but without integral.

Determining the ideal settings for an order calculation is work for an expert. Therefore, in this document, we only discuss a simple example. Suppose you have

- a vibration signal
- a speed signal in Hz, i.e. number of revolutions per second

The speed unit should be set to Hz:



Input

Data source: RisingSine
Sample rate: 1000 Hz

Speed source: increasingFreq
Sample rate: 1000 Hz

Speed unit: Custom
Scaling factor: 1
Speed unit after scaling

Resolution: 1/sec
Length reset: 1/sec

Visuals

Style: Curve
Fill: None
Color:
Improve isometric visibility

Acquisition

# Frequency results: 65536
Overlap: 0%
0% 95%

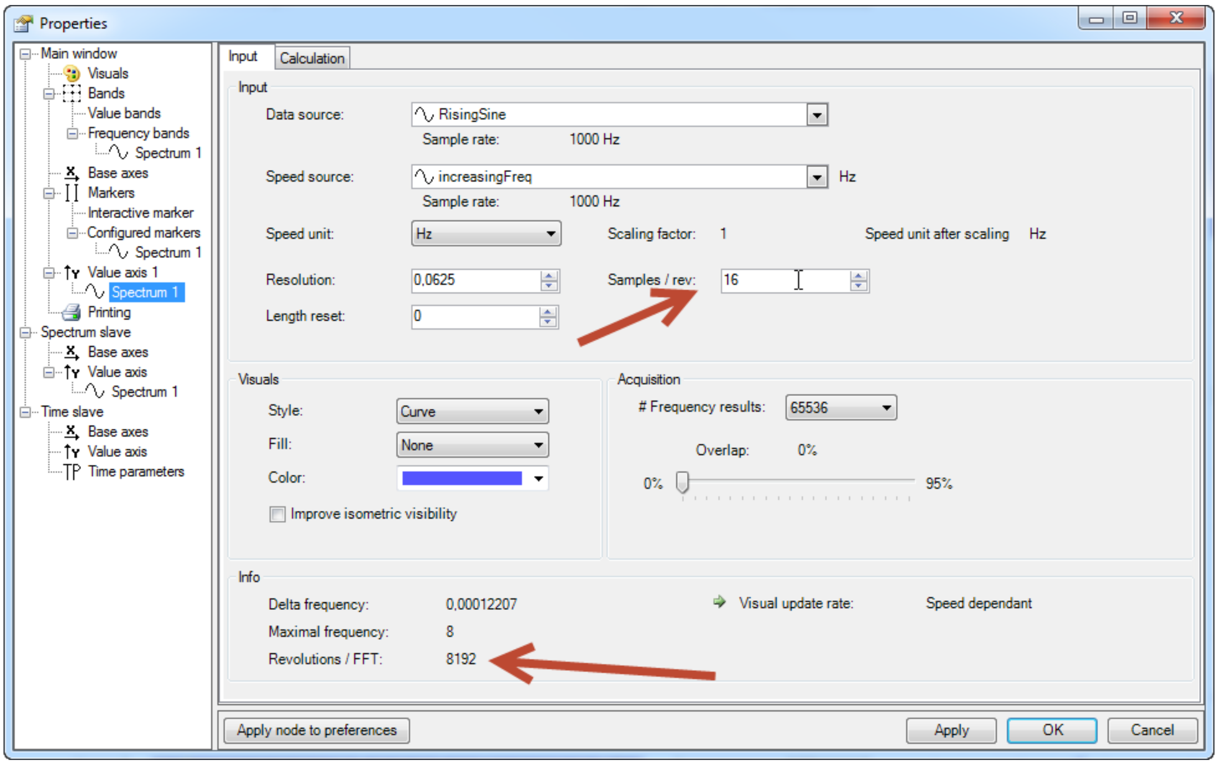
Info

Delta frequency: 0,00012207 1/sec
Maximal frequency: 8 1/sec
Visual update rate: Speed dependant

In case the speed unit is changed to Hz (or RPM), another text box appears which allows specifying the number samples per revolution. The resolution setting is updated automatically (the resolution is the inverse of the number samples per revolution).

At the bottom, the number of revolutions per FFT is displayed. This latter number depends on the number of frequency results and the number of samples per revolution:

- ➔ (65336 * 2) / 16 = 8192 revolutions per FFT



Input

| | | | |
|---|---|---|---|
| Data source: | RisingSine | | |
| Sample rate: | 1000 Hz | | |
| Speed source: | increasingFreq Hz | | |
| Sample rate: | 1000 Hz | | |
| Speed unit: | Hz | Scaling factor: 1 | Speed unit after scaling Hz |
| Resolution: | 0,0625 | Samples / rev: | 16 |
| Length reset: | 0 | | |

Visuals

| | |
|---|---|
| Style: | Curve |
| Fill: | None |
| Color: | 🟦 |

☐ Improve isometric visibility

Acquisition

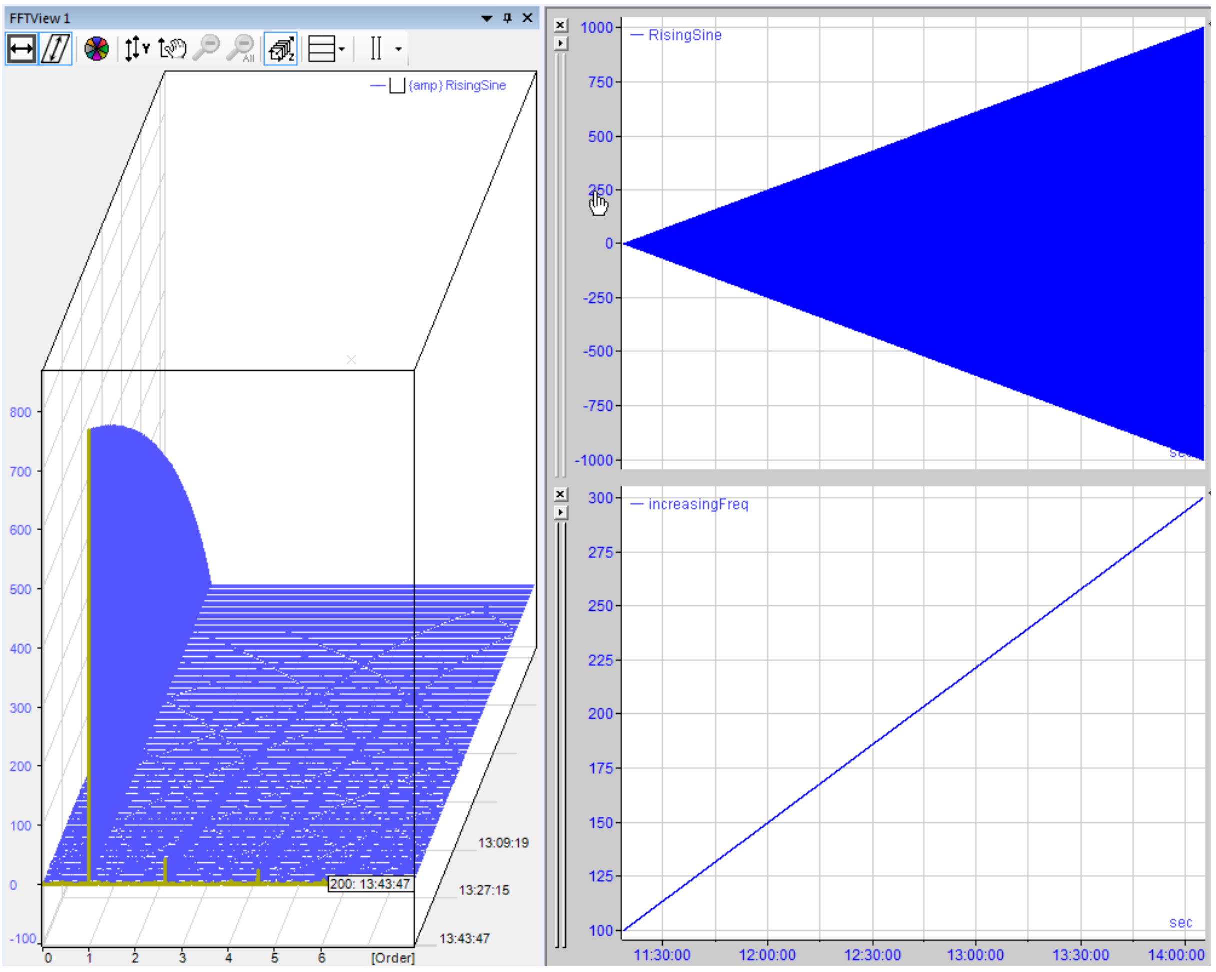
| | |
|---|---|
| # Frequency results: | 65536 |
| Overlap: | 0% |

0% 95%

Info

| | | | |
|---|---|---|---|
| Delta frequency: | 0,00012207 | Visual update rate: | Speed dependant |
| Maximal frequency: | 8 | | |
| Revolutions / FFT: | 8192 | | |

Below, you see the result of this order calculation:



On the top right, you see the graph of “RisingSine”, the vibration signal. On the bottom right, you see the graph of “increasingFreq”, the frequency of the sine wave.

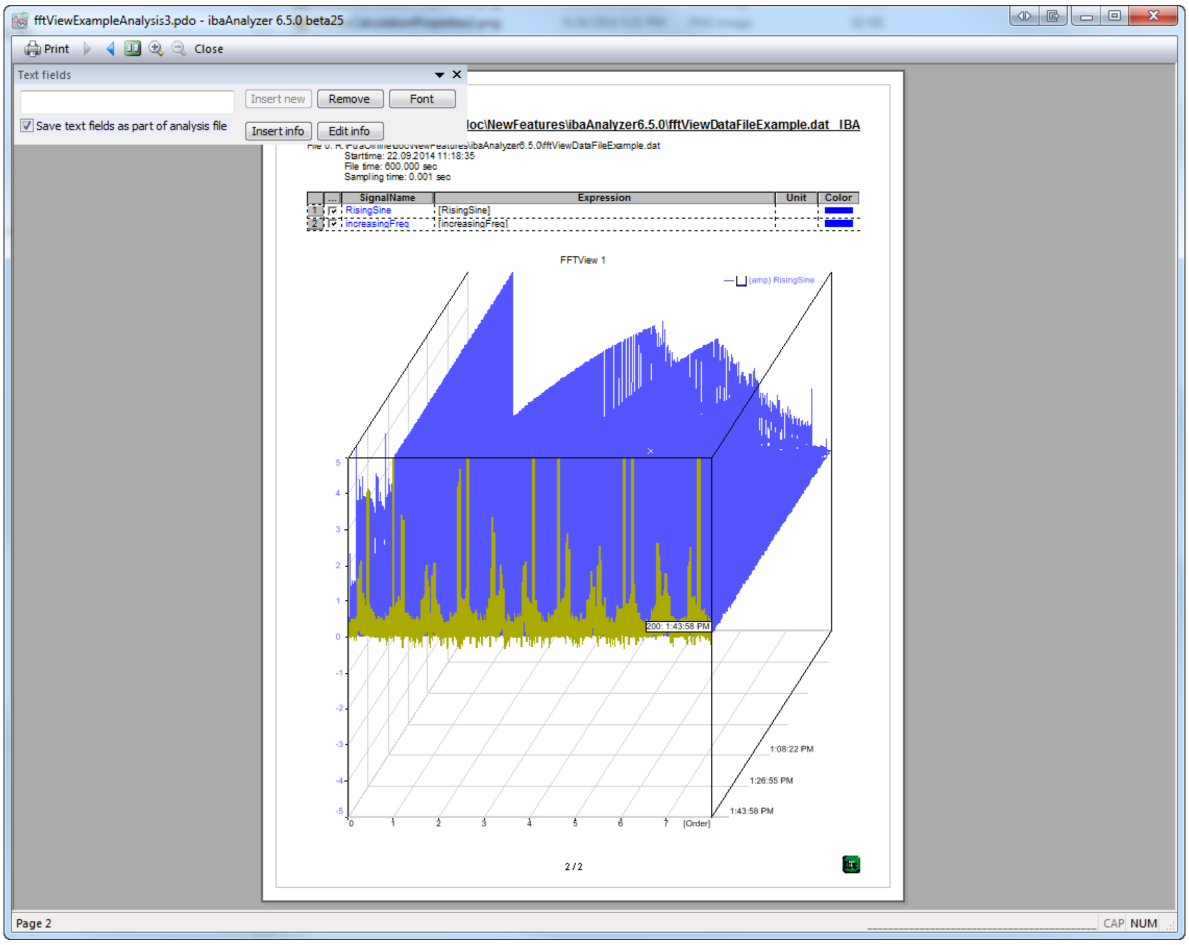
In the FFT view**, you see that the order calculation has cancelled out each change in speed. Because of the technique of order calculation, the spectrum still has a very high resolution**; the delta frequency is about 0.0001220, as you can see at the bottom of the property window.

This example was just a simplified case. Our software supports a wide range of settings to perform online and offline frequency analysis. Note that order calculations are a very complex matter; please contact iba support for more information.

## 1.6 Report generation and printing

### 1.6.1 Printing

The default ibaAnalyzer print functionality (*Menu → File → Print…*) and print preview (*Menu → File → Print Preview* ), will also depict the FFT views after the signal grid and after the ibaCapture views (if any).



loc\NewFeatures\ibaAnalyzer6.5.0\fftViewDataFileExample.dat IBA

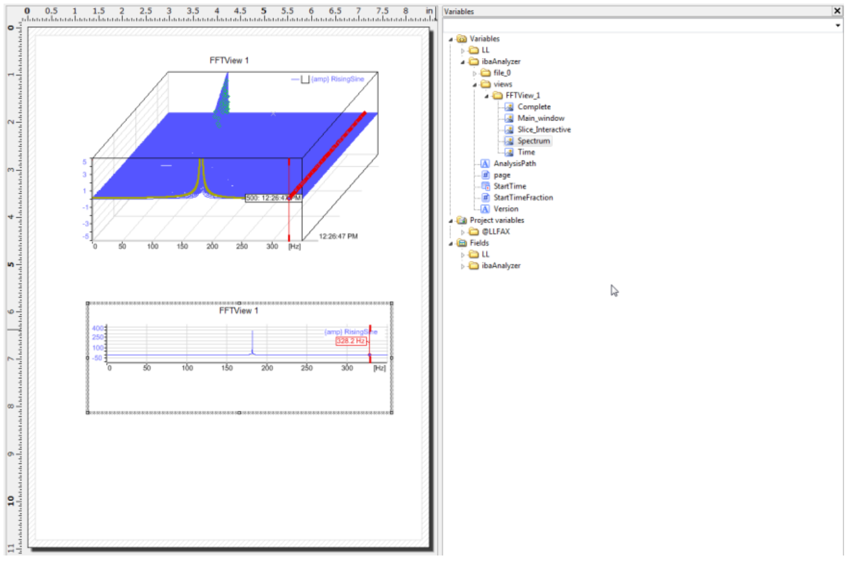
Starttime: 22.09.2014 11:18:35
File time: 600.000 sec
Sampling time: 0.001 sec

|  | ... | SignalName | Expression | Unit | Color |
|---|---|---|---|---|---|
| 1 | ✓ | RisingSine | [RisingSine] |  | 🟦 |
| 2 | ✓ | increasingFreq | [increasingFreq] |  | 🟦 |

The views are however not customizable in the default print, for this we recommend using the more extensive report generator.

### 1.6.2 Report generation

Similar as for ibaAnalyzer graphs and ibaCapture views, the FFT views can be dragged into the designer in order to include them in a report.



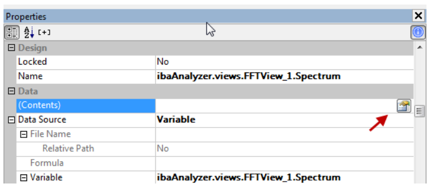
The objects are located in the variables tree under

*Variables → ibaAnalyzer → views*

A sub tree for each FFT view is present; the tree has the following objects:

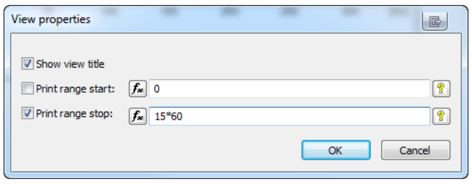
- <u>Complete</u>: this object depicts the entire FFT view including the main window and the slaves, if one does not allocate enough room on the page for this object, this is indicated by depicting a scroll-bar.
- <u>Main_window</u>: this object depicts only the main window of the FFT view.
- This is followed by an object for each slave window…

After having dragged an object onto the page in the designer, one can customize it by double clicking on it or by going to *(Contents)* and clicking the customize button in the properties.



Properties

| | |
|---|---|
| ⊟ Design | |
| Locked | No |
| Name | **ibaAnalyzer.views.FFTView_1.Spectrum** |
| ⊟ Data | |
| (Contents) | |
| ⊟ Data Source | **Variable** |
| ⊟ File Name | |
| Relative Path | No |
| Formula | |
| ⊟ Variable | **ibaAnalyzer.views.FFTView_1.Spectrum** |



View properties

Show view title: ✓

Print range start: 0

Print range stop: ✓ 15*60

The options currently available are

- Show view title: One can uncheck this option if the object should be depicted without a title (useful for objects representing slave windows).
- Print range start: Start of the time range used to calculate the FFTs depicted in the report.
  One can check this option and specify an ibaAnalyzer expression if one desires an alternative time range start. If this option is not checked, the start of the time range equals that of the FFT view in the main ibaAnalyzer window which is, depending on the mode of the FFT view, the start of the zoomed area or the start of the entire ibaAnalyzer time range.
- Print range stop: End of the time range used to calculate the FFTs depicted in the report. One can check this option and specify an ibaAnalyzer expression if one desires an alternative time range end. If this option is not checked the end of the time range equals that of the FFT view in the main ibaAnalyzer window which is, depending on the mode of the FFT view, the end of the zoomed in area or the end of the entire ibaAnalyzer time range.

## 1.7 Miscellaneous features

Next to features already discussed, the FFT view in ibaAnalyzer provides:

- Customizable additional legends
- Full support of markers: dynamic and static, harmonics, sidebands, different visualization settings…
- Full support for labels: dynamic and static labels
- Support for color zones
  - Value zones: other colors depending on the height of the spectrum
  - Delta zones: other colors depending on the frequency range
- Many zoom and scaling settings
- …

# 2 New functions

## 2.1 SetYatX function

The *SetYatX* function allows one to generate a copy of an existing expression with one sample altered or inserted.

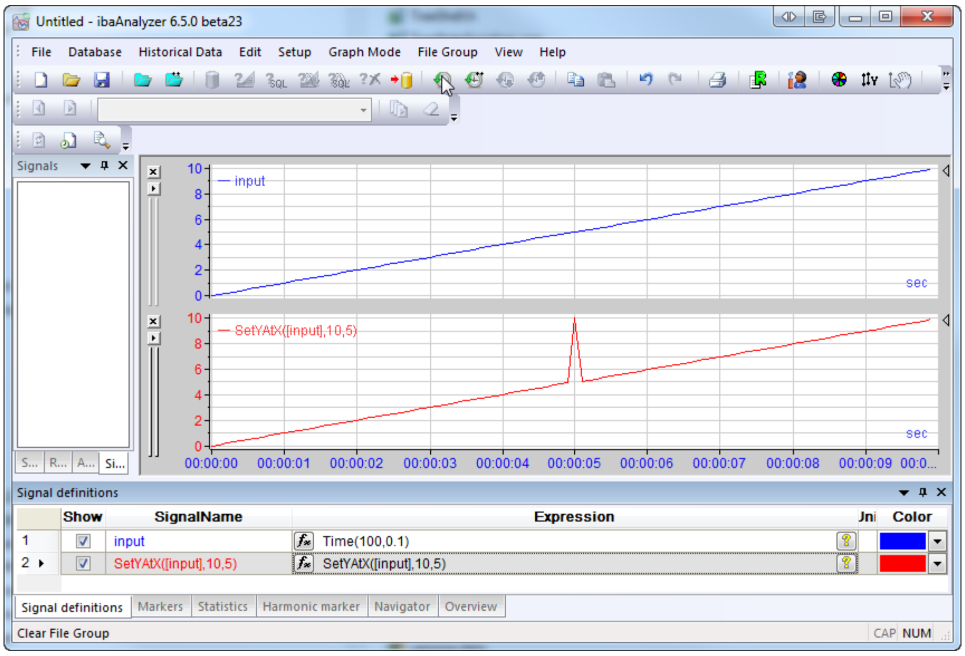
The function takes the following arguments:

- Expression: Signal to alter.
- Value: The new value of the sample that will be inserted or will alter an existing sample. This parameter is expected to be a constant; if the provided expression is not constant, its values will be averaged before being used in the function.
- X: The location to insert a sample into. This parameter is expected to be a constant; if the provided expression is not constant, its values will be averaged before being used in the function.

The function will behave differently depending on whether the *Expression* input parameter is equidistantly sampled or not. If equidistantly then:

- If the *X* parameter is smaller than the offset of *Expression*, *Expression* will be returned unaltered.
- If the *X* parameter is exactly equal to the location of the last sample of *Expression* plus one sample interval, *Expression* will be returned with the additional sample added to the end. This is useful to extend an existing signal by one sample. Note that to determine the last sample of *Expression* plus one sample interval, the *XSize* function can be used.
- In all other cases, the sample at the exact *X* location or if no such sample exist, the sample immediately to the left of *X* in *Expression,* is altered to take on the new *Value*.

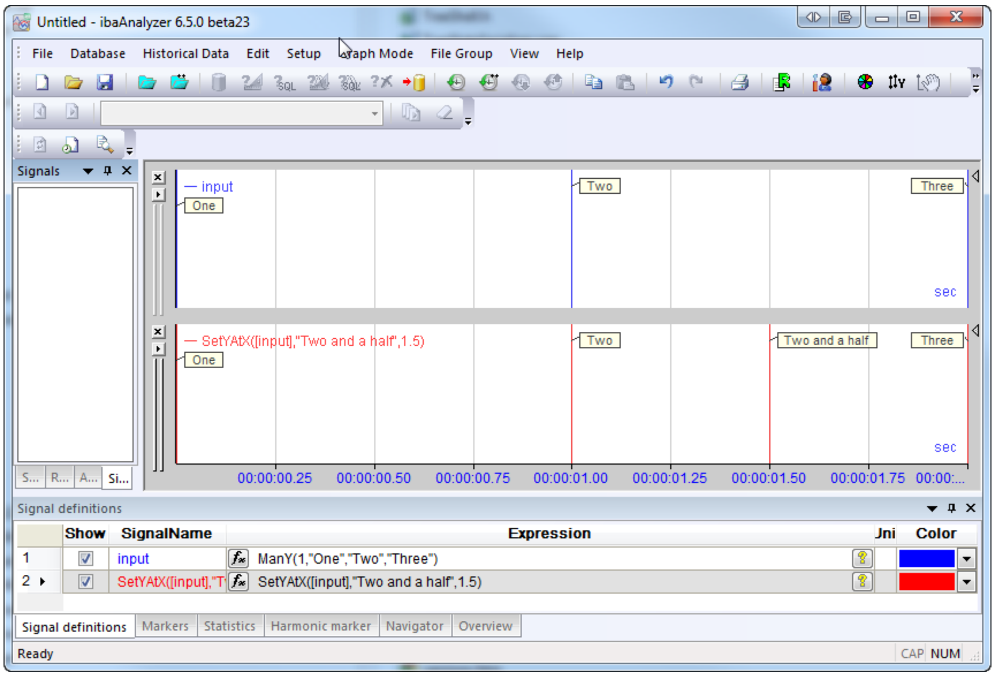
If *Expression* is not equidistantly sampled, the function will replace a sample if X is incident with the location of an existing sample in *Expression*, otherwise it will insert a new sample.


Untitled - ibaAnalyzer 6.5.0 beta23
File Database Historical Data Edit Setup Graph Mode File Group View Help
Signals
input
SetYAtX([input],10,5)
sec
00:00:00 00:00:01 00:00:02 00:00:03 00:00:04 00:00:05 00:00:06 00:00:07 00:00:08 00:00:09
Signal definitions Markers Statistics Harmonic marker Navigator Overview
Clear File Group
CAP NUM

Signal definitions

| | Show | SignalName | Expression | Jni | Color |
|---|---|---|---|---|---|
| 1 | ✓ | input | Time(100,0.1) | | 🟦 |
| 2 | ✓ | SetYAtX([input],10,5) | SetYAtX([input],10,5) | | 🟥 |

Note that the function can also be used to insert a new text in a text sample:


Untitled - ibaAnalyzer 6.5.0 beta23
File Database Historical Data Edit Setup Graph Mode File Group View Help
Signals
input
One
Two
Three
sec
SetYAtX([input],"Two and a half",1.5)
One
Two
Two and a half
Three
sec
00:00:00.25 00:00:00.50 00:00:00.75 00:00:01.00 00:00:01.25 00:00:01.50 00:00:01.75 00:00:...
Signal definitions
Signal definitions Markers Statistics Harmonic marker Navigator Overview
Ready CAP NUM

| | Show | SignalName | Expression | Jni | Color |
|---|---|---|---|---|---|
| 1 | ✓ | input | ManY(1,"One","Two","Three") | | 🟦 |
| 2 | ✓ | SetYAtX([input],"T | SetYAtX([input],"Two and a half",1.5) | | 🟥 |

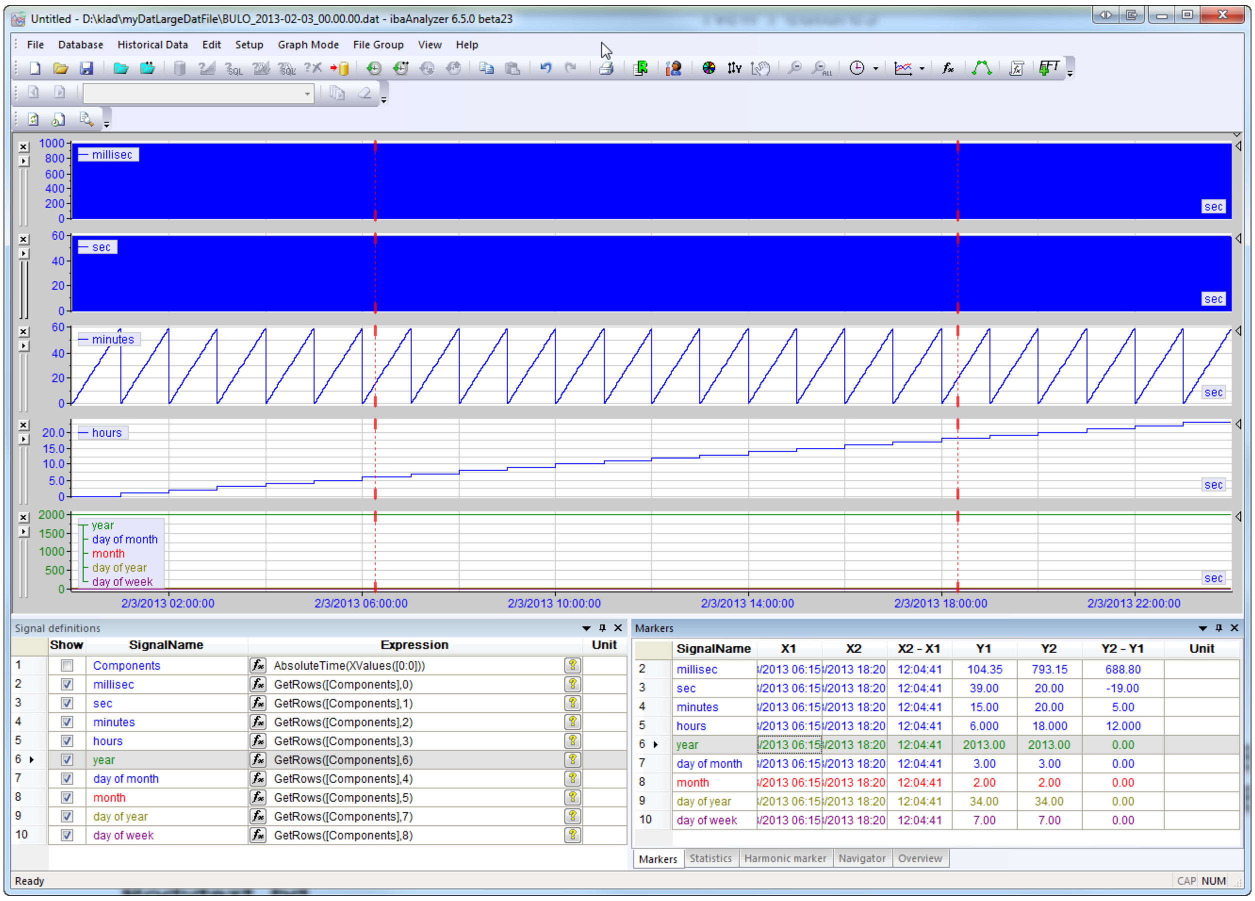
## 2.2 AbsoluteTime function

This function allows one to convert ibaAnalyzer relative times (e.g. retrieved by *XFirst*, *XLast* or *XValues* functions) to absolute time. The absolute time is returned as a vector of components, the desired specific time component can then be selected with the *GetRows* function.

The function takes the following arguments:

- Time: The relative time to convert. This can be a constant (e.g. retrieved by the *XFirst* or *XLast* functions) or a varying signal (e.g. retrieved by the *XValues* function). In the former case, a vector with constants is returned. In the latter case, a vector with varying signals is returned. If *Time* is non-equidistantly sampled, the result is non-equidistantly sampled as well with a sample reported corresponding with every sample in *Time*.

- DoSync: If FALSE, *Time* is taken to be relative to the time-axis start time. If TRUE, the time is taken relative to the start time of the first file the *Time* expression is dependent on. This parameter has no effect if the "Synchronize on recording time" mode for the Time-axis is not active as both start times are identical in this case. The time expression can depend on signals from different files; if the first such file is a chain of appended files, the start times of all the files in the chain are taken into account. This parameter is optional and can be omitted, in which case it is FALSE by default.



Signal definitions

| | Show | SignalName | Expression | Unit |
|---|---|---|---|---|
| 1 | ☐ | Components | AbsoluteTime(XValues([0:0])) | |
| 2 | ✓ | millisec | GetRows([Components],0) | |
| 3 | ✓ | sec | GetRows([Components],1) | |
| 4 | ✓ | minutes | GetRows([Components],2) | |
| 5 | ✓ | hours | GetRows([Components],3) | |
| 6 | ✓ | year | GetRows([Components],6) | |
| 7 | ✓ | day of month | GetRows([Components],4) | |
| 8 | ✓ | month | GetRows([Components],5) | |
| 9 | ✓ | day of year | GetRows([Components],7) | |
| 10 | ✓ | day of week | GetRows([Components],8) | |

Markers

| | SignalName | X1 | X2 | X2 - X1 | Y1 | Y2 | Y2 - Y1 | Unit |
|---|---|---|---|---|---|---|---|---|
| 2 | millisec | /2013 06:15 | /2013 18:20 | 12:04:41 | 104.35 | 793.15 | 688.80 | |
| 3 | sec | /2013 06:15 | /2013 18:20 | 12:04:41 | 39.00 | 20.00 | -19.00 | |
| 4 | minutes | /2013 06:15 | /2013 18:20 | 12:04:41 | 15.00 | 20.00 | 5.00 | |
| 5 | hours | /2013 06:15 | /2013 18:20 | 12:04:41 | 6.000 | 18.000 | 12.000 | |
| 6 | year | /2013 06:15 | /2013 18:20 | 12:04:41 | 2013.00 | 2013.00 | 0.00 | |
| 7 | day of month | /2013 06:15 | /2013 18:20 | 12:04:41 | 3.00 | 3.00 | 0.00 | |
| 8 | month | /2013 06:15 | /2013 18:20 | 12:04:41 | 2.00 | 2.00 | 0.00 | |
| 9 | day of year | /2013 06:15 | /2013 18:20 | 12:04:41 | 34.00 | 34.00 | 0.00 | |
| 10 | day of week | /2013 06:15 | /2013 18:20 | 12:04:41 | 7.00 | 7.00 | 0.00 | |

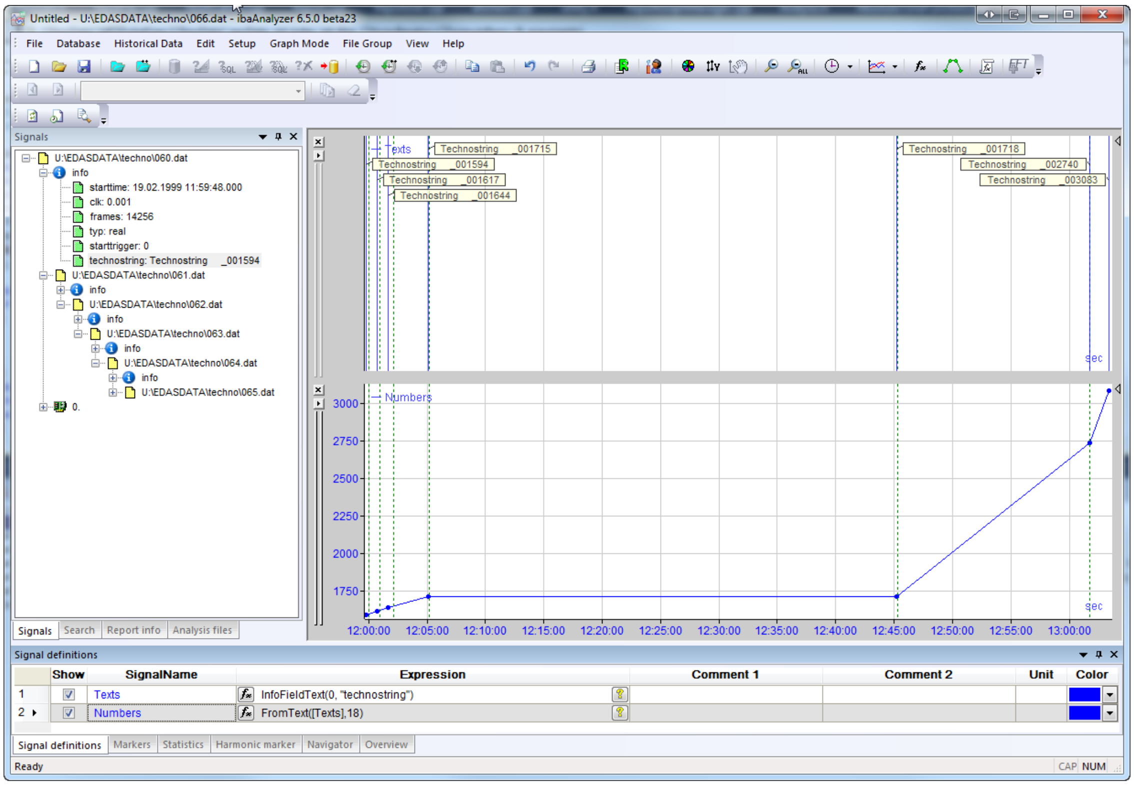
The function returns a vector of time components, the components are:

- Row 0: The milliseconds component.
- Row 1: The seconds component.
- Row 2: The minutes component.
- Row 3: The hours component.
- Row 4: The day of the month.
- Row 5: The month component.
- Row 6: The year component.
- Row 7: The day of the year.
- Row 8: The day of the week, Monday being 1 and Sunday being 7.

## 2.3 FromText function

The function FromText converts a text channel to a numeric channel. The function takes the following arguments:

- Text: The text channel to convert.
- Begin: Start index in the string to start interpreting the string as a number. It should be zero-indexed, i.e. specify 0 to indicate the start of the string. If this parameter is omitted, 0 will be used as default value.
- End: The index of the last character to be used to interpret the string as a number. Further characters will be ignored. If this parameter is omitted, the entire remainder of the string will be used.



U:\EDASDATA\techno\060.dat
- info
  - starttime: 19.02.1999 11:59:48.000
  - clk: 0.001
  - frames: 14256
  - typ: real
  - starttrigger: 0
  - technostring: Technostring _001594
  - U:\EDASDATA\techno\061.dat
    - info
    - U:\EDASDATA\techno\062.dat
      - info
      - U:\EDASDATA\techno\063.dat
        - info
        - U:\EDASDATA\techno\064.dat
          - info
          - U:\EDASDATA\techno\065.dat
- 0.

Signal definitions

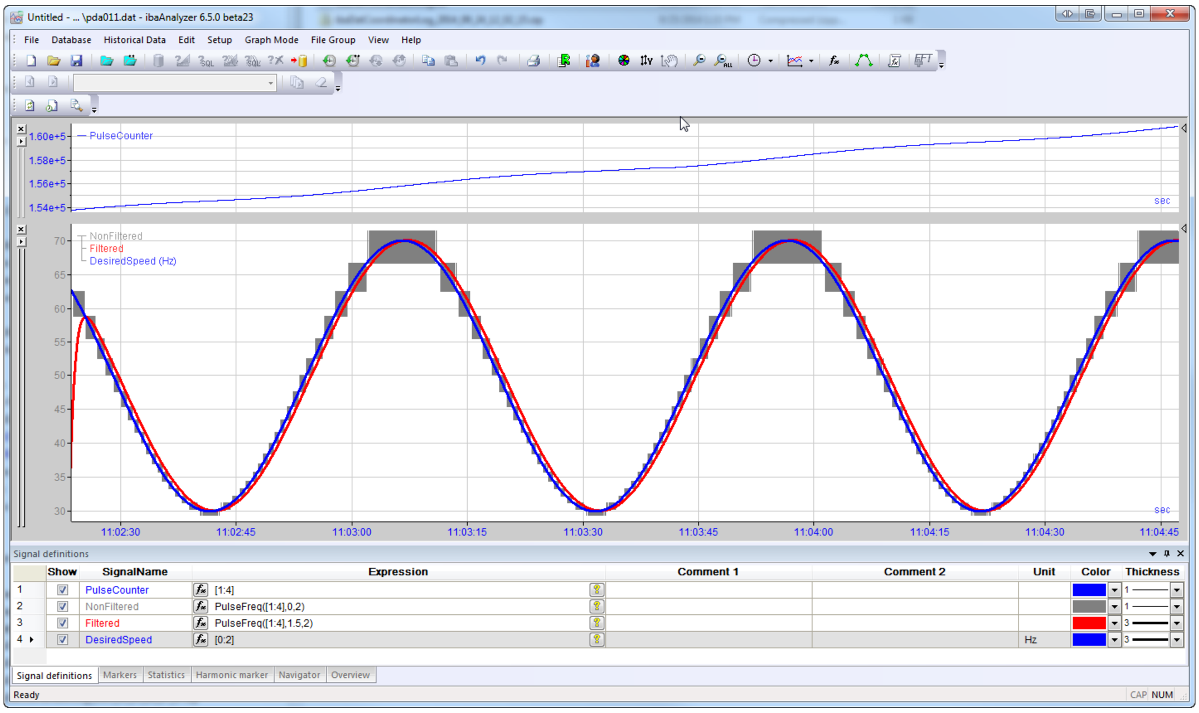
| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | Texts | InfoFieldText(0, "technostring") | | | | 🟦 |
| 2 | ✓ | Numbers | FromText([Texts],18) | | | | 🟦 |

## 2.4 PulseFreq function

Given a pulse or pulse counter signal, this function estimates the frequency of the pulses in Hz. The function takes the following arguments

- Expression: The pulse or pulse counter.
- Omega: If different from zero, a filter is applied to the result; this is identical as if the *Lp* function would have been applied with the same *Omega* value. This parameter is optional; if omitted, no filter is applied.
- EdgeType: One can specify if the signal is a pulse counter or a pulse. In case of a pulse, this parameter also indicates if the rising edge, the lower edge or both should be used to estimate the frequency. This parameter can take on the following values:
  - -1: Assume a pulse and use the falling edges to estimate the frequency.
  - 0 : Assume a pulse and use both the falling and rising edge to estimate the frequency.
  - 1 : Assume a pulse and use the rising edges to estimate the frequency.
  - 2 : Assume a pulse counter.

  This parameter is optional; if omitted, a pulse counter is assumed.



| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color | Thickness |
|---|---|---|---|---|---|---|---|---|
| 1 | ✓ | PulseCounter | [1:4] | | | | 🟦 | 1 |
| 2 | ✓ | NonFiltered | PulseFreq([1:4],0,2) | | | | ⬜ | 1 |
| 3 | ✓ | Filtered | PulseFreq([1:4],1.5,2) | | | | 🟥 | 3 |
| 4 | ✓ | DesiredSpeed | [0:2] | | | Hz | 🟦 | 3 |

## 2.5 Interpolation functions for vectors

Note, these functions were implemented in 6.4.1, but they haven't been documented in a new features document until now.

### 2.5.1 VectorLSQPolyCoef

To recapitulate, the ibaAnalyzer *LSQPolyCoef* function calculates for a given channel X and Y and a degree, the coefficients of the polynomial determined by the method of least squares so that the polynomial maps X onto Y as close as possible. The function returns a vector of constants where the first row is the constant of the polynomial; the second row is the coefficient of the linear term of the polynomial, the third row is the coefficient of the quadratic term and so on.

*VectorLSQPolyCoef* makes a least squares approximation as well, but this time, the input is a *vector* of signals and the least squares method is applied for every sample in time, while the X and Y sets are gotten in the cross-profile direction. The function takes the following arguments:

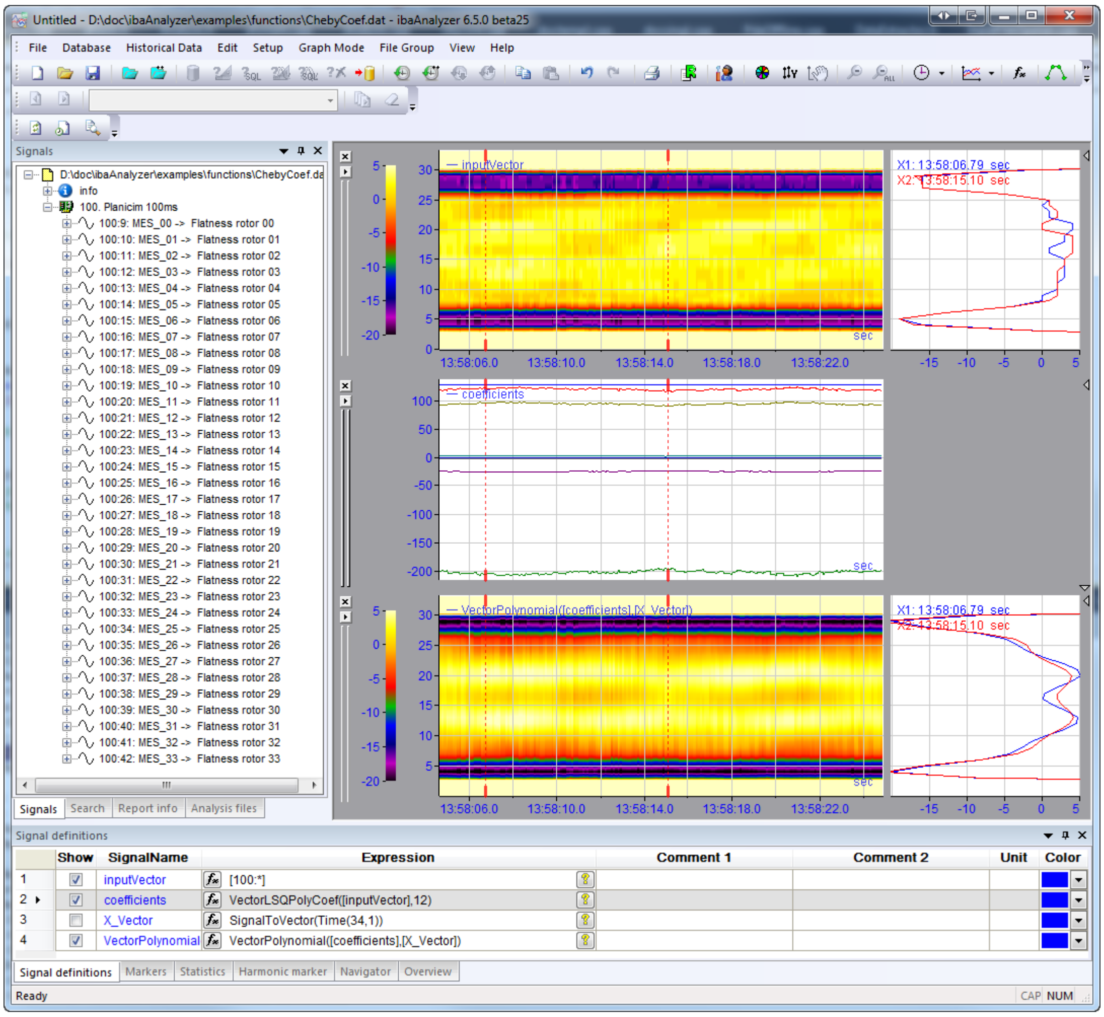
- Vector: The vector to compute the coefficients for. For each sample in time, the Y set will be taken from the values of the vector. The X set will be simply the indices of the rows in the vector (e.g.: 0, 1, 2, 3,…) unless the input vector was produced in the *logicals* dialog with a *Zone offset* or *Zone widths* specified or the *SetZoneWidths* function was used to produce the input vector.
- Degree: The degree of the polynomial to calculate, the number of coefficients returned in the result vector for each sample in time will be one larger than this.

The function returns a vector with the coefficients of the polynomial. Contrary to the *LSQPolyCoef* function, the coefficients in the vector will vary over time if the input vector varies.

### 2.5.2 VectorPolynomial

Given the coefficient vector returned from *VectorLSQPolyCoef,* the *VectorPolynomial* function can be used to construct a visual polynomial approximation of the profile used in *VectorLSQPolyCoef.* The function takes the following arguments:

- Coefs: The coefficient vector generated by *VectorLSQPolyCoef*.
- Vector: A vector that will be used to determine where to evaluate the polynomials given by *Coefs*. Again, the X-set will be simply the indices of the rows in the vector (e.g.: 0, 1, 2, 3,…), unless the input vector was produced in the logicals dialog with a *Zone offset* or *Zone widths* specified or the *SetZoneWidths* function was used to produce the input vector. The actual values of the signal in the vector will be ignored. Easiest is to simply provide the input vector used in the *VectorLSQPolyCoef* function, but if one wishes to interpolate or extrapolate, an alternative vector can be specified (see the following screenshot for an alternative)



Untitled - D:\doc\ibaAnalyzer\examples\functions\ChebyCoef.dat - ibaAnalyzer 6.5.0 beta25

D:\doc\ibaAnalyzer\examples\functions\ChebyCoef.da
- info
- 100. Planicim 100ms
  - 100:9: MES_00 -> Flatness rotor 00
  - 100:10: MES_01 -> Flatness rotor 01
  - 100:11: MES_02 -> Flatness rotor 02
  - 100:12: MES_03 -> Flatness rotor 03
  - 100:13: MES_04 -> Flatness rotor 04
  - 100:14: MES_05 -> Flatness rotor 05
  - 100:15: MES_06 -> Flatness rotor 06
  - 100:16: MES_07 -> Flatness rotor 07
  - 100:17: MES_08 -> Flatness rotor 08
  - 100:18: MES_09 -> Flatness rotor 09
  - 100:19: MES_10 -> Flatness rotor 10
  - 100:20: MES_11 -> Flatness rotor 11
  - 100:21: MES_12 -> Flatness rotor 12
  - 100:22: MES_13 -> Flatness rotor 13
  - 100:23: MES_14 -> Flatness rotor 14
  - 100:24: MES_15 -> Flatness rotor 15
  - 100:25: MES_16 -> Flatness rotor 16
  - 100:26: MES_17 -> Flatness rotor 17
  - 100:27: MES_18 -> Flatness rotor 18
  - 100:28: MES_19 -> Flatness rotor 19
  - 100:29: MES_20 -> Flatness rotor 20
  - 100:30: MES_21 -> Flatness rotor 21
  - 100:31: MES_22 -> Flatness rotor 22
  - 100:32: MES_23 -> Flatness rotor 23
  - 100:33: MES_24 -> Flatness rotor 24
  - 100:34: MES_25 -> Flatness rotor 25
  - 100:35: MES_26 -> Flatness rotor 26
  - 100:36: MES_27 -> Flatness rotor 27
  - 100:37: MES_28 -> Flatness rotor 28
  - 100:38: MES_29 -> Flatness rotor 29
  - 100:39: MES_30 -> Flatness rotor 30
  - 100:40: MES_31 -> Flatness rotor 31
  - 100:41: MES_32 -> Flatness rotor 32
  - 100:42: MES_33 -> Flatness rotor 33

| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | inputVector | [100:*] | | | | 🟦 |
| 2 | ✓ | coefficients | VectorLSQPolyCoef([inputVector],12) | | | | 🟦 |
| 3 | | X_Vector | SignalToVector(Time(34,1)) | | | | 🟦 |
| 4 | ✓ | VectorPolynomial | VectorPolynomial([coefficients],[X_Vector]) | | | | 🟦 |

## 2.6 Ranges in signal ID's

Note, this functionality was implemented in 6.4.2, but it hasn't been documented in a new features document until now.

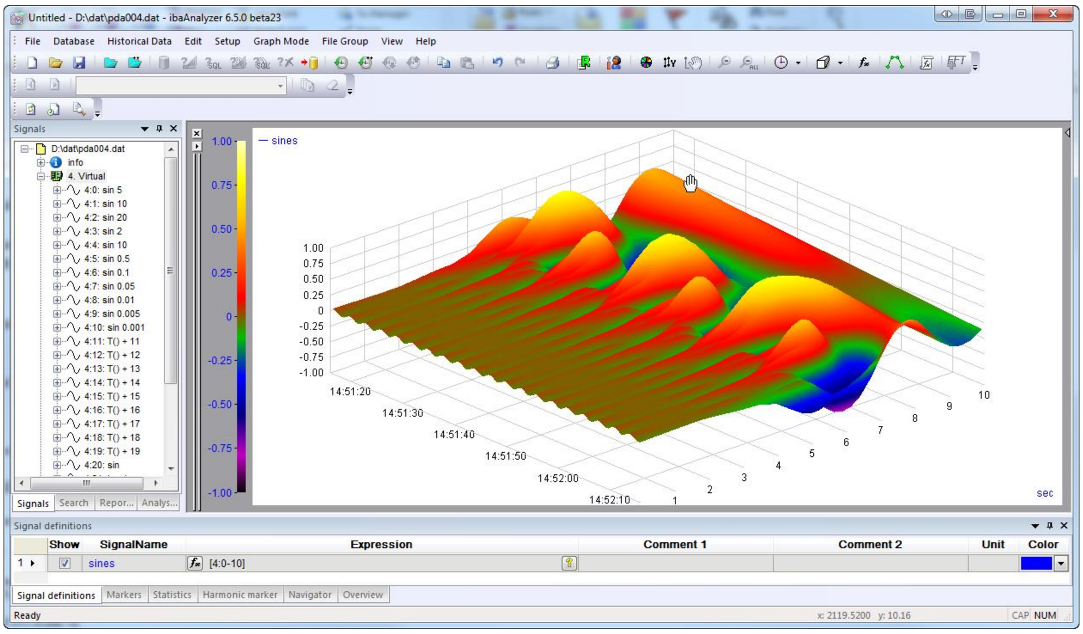
In an ibaAnalyzer expression, when specifying a channel by numeric ID, you can replace any part of the ID by a range. IbaAnalyzer will automatically create a vector in such case. This eliminates the need to go to the "Logical signals" dialogs in many cases. You can specify an asterisk ('*') to indicate all available channels or a range by specifying two zero-based indices separated by a dash ('-'). Some examples:

- `[0:*]` : All analog signals from the first module.
- `[0.1-3]` : The second, third and fourth digital signal of the first module.
- `[*:1]` : For each module, the second analog signal.
- `[*_2:0-2]` : For each file opened, the first three analog signals of the third module.
- `[1_*:0:0]` : From the second opened file (QDR), for each measuring location, the first analog signal of the first module.

Note that you can combine vectors created in this manner with the *MakeVector* function, e.g.:

```
MakeVector([0:0-3],[0:5])
```

creates a vector of 5 signals consisting of the first four analog signals of the first module and the sixth signal.



| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 ▸ | ✓ | sines | [4:0-10] | | | | 🟦 |

Signal definitions Markers Statistics Harmonic marker Navigator Overview
Ready
x: 2119.5200 y: 10.16
CAP NUM

## 3 TDMS files

If ibaAnalyzer has the E-dat license activated, one can import files in the National Instruments TDMS data format (files with extension *.tdm* or *.tdms*).

The channel groups of such a file are shown as modules, the respective channels of a channel group are located under the respective module. File properties are presented as infofields and channel properties as channel infofields. Currently, the channel group properties are not reported as there is no analogous concept in ibaAnalyzer.

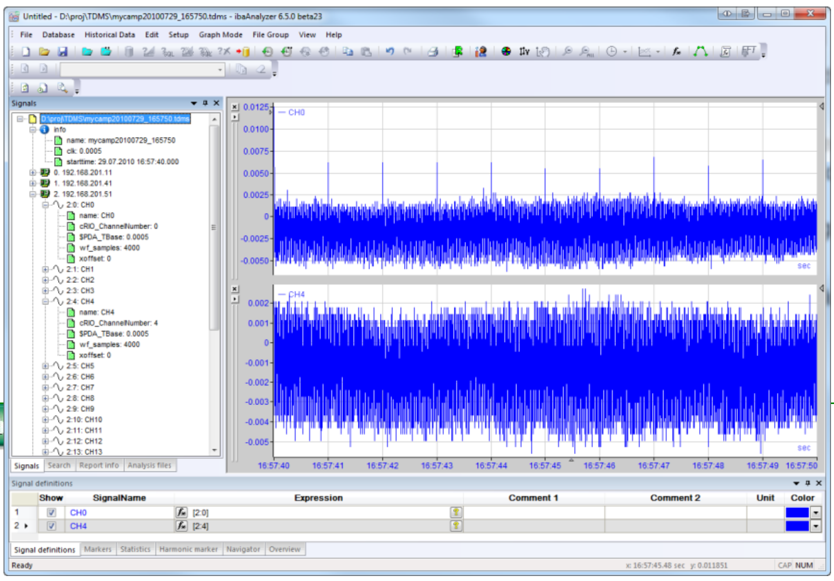
Determining the timestamps of the channel samples currently happens on the file start time property (datetime) and the *wf_increment*, *wf_start_offset* and *wf_start_time* channel properties. Furthermore, the following channel properties will be used:

- name: Name given to the channel.
- unit_string: If present, unit given to the channel data.
- description: If present, used as first comment of the channel.
- wf_xname: If present, the value of this channel will be used to represent the channel in an alternative physical domain. The following values are accepted:
    - length: Length based signals, in the ibaAnalyzer selected default length unit.
    - frequency: Frequency based in Hertz.
    - invlength: Inverse length in 1 over the ibaAnalyzer selected default length unit.

  If the parameter is not present or its value is not supported, the channel is presented as time based data in seconds.

To our knowledge, there is no unique way in the TDMS data file format for a channel to indicate that the data stored in a channel should be interpreted as a digital signal. However, ibaAnalyzer will interpret the channel as digital if one of the following criteria is met:

- The channel property “ *iba_digital* ” is present, is of type unsigned 8-bit integer and has a value different from zero.
- The channel property “ *unit_string* ” is present and has the value “ *bit* ”.
- The name of the channel ends with “ *[bit]* ”.


Untitled - D:\proj\TDMS\mycamp20100729_165750.tdms - ibaAnalyzer 6.5.0 beta23

```
D:\proj\TDMS\mycamp20100729_165750.tdms
  info
    name: mycamp20100729_165750
    clk: 0.0005
    starttime: 29.07.2010 16:57:40.000
  0. 192.168.201.11
  1. 192.168.201.41
  2. 192.168.201.51
    2:0: CH0
      name: CH0
      cRIO_ChannelNumber: 0
      $PDA_TBase: 0.0005
      wf_samples: 4000
      xoffset: 0
    2:1: CH1
    2:2: CH2
    2:3: CH3
    2:4: CH4
      name: CH4
      cRIO_ChannelNumber: 4
      $PDA_TBase: 0.0005
      wf_samples: 4000
      xoffset: 0
    2:5: CH5
    2:6: CH6
    2:7: CH7
    2:8: CH8
    2:9: CH9
    2:10: CH10
    2:11: CH11
    2:12: CH12
    2:13: CH13
```

Signal definitions

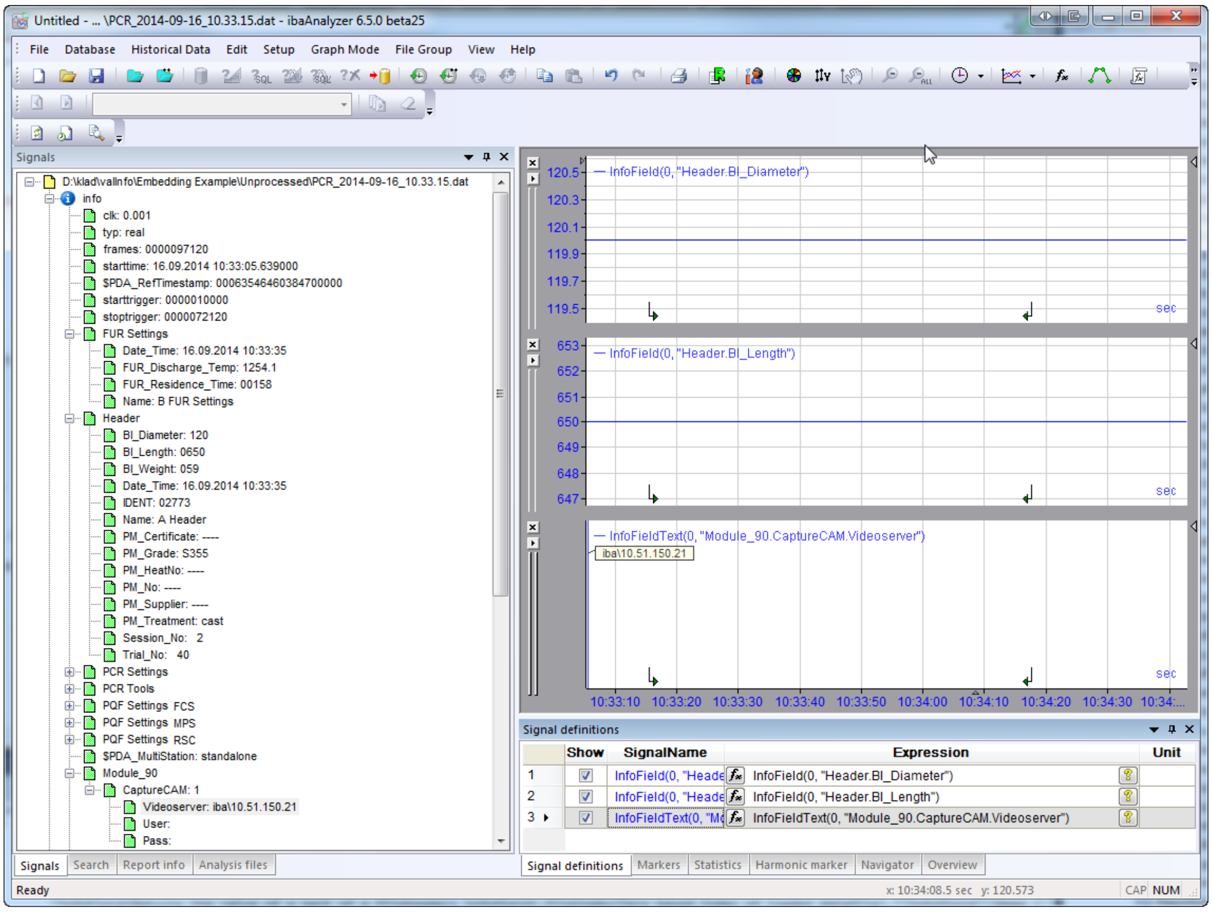
| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | CH0 | [2:0] | | | | 🟦 |
| 2 | ✓ | CH4 | [2:4] | | | | 🟦 |

Ready x: 16:57:45.48 sec y: 0.011851 CAP NUM

# 4 Signal tree and search dialog

## 4.1 Infofields subtrees.

If an infofield has one or more dots (“.”) in its name, these dots will be used as separators to split up the infofield name in parts. The infofield will be shown without the dots in a tree structure with the nodes corresponding to the parts. If one or more infofield names have the same prefix up to the last dot, they will be shown as sibling nodes. The *InfoField* and *InfoFieldText functions* still require one to specify all dots (the leaf nodes can still be double-clicked for this purpose).


Untitled - ... \PCR_2014-09-16_10.33.15.dat - ibaAnalyzer 6.5.0 beta25
File
Database
Historical Data
Edit
Setup
Graph Mode
File Group
View
Help
Signals
InfoField(0, "Header.BI_Diameter")
InfoField(0, "Header.BI_Length")
InfoFieldText(0, "Module_90.CaptureCAM.Videoserver")
iba\10.51.150.21
sec
Signals
Search
Report info
Analysis files
Signal definitions
Markers
Statistics
Harmonic marker
Navigator
Overview
Ready
x: 10:34:08.5 sec y: 120.573
CAP NUM

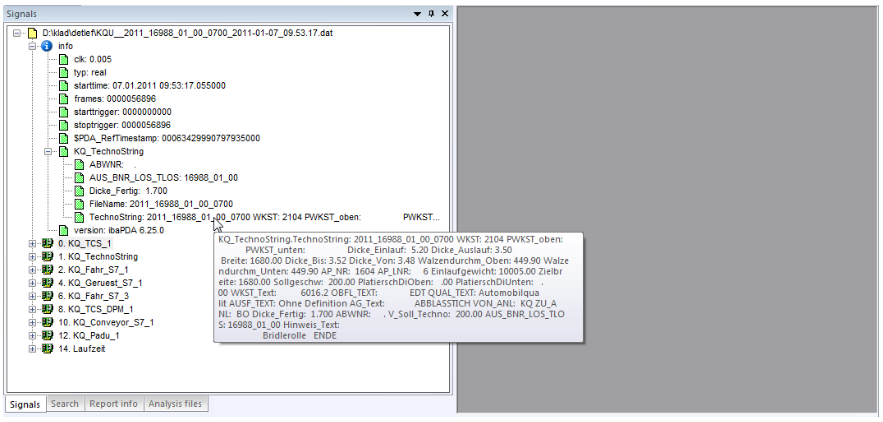
```
D:\klad\valInfo\Embedding Example\Unprocessed\PCR_2014-09-16_10.33.15.dat
  info
    clk: 0.001
    typ: real
    frames: 0000097120
    starttime: 16.09.2014 10:33:05.639000
    $PDA_RefTimestamp: 00063546460384700000
    starttrigger: 0000010000
    stoptrigger: 0000072120
    FUR Settings
      Date_Time: 16.09.2014 10:33:35
      FUR_Discharge_Temp: 1254.1
      FUR_Residence_Time: 00158
      Name: B FUR Settings
    Header
      BI_Diameter: 120
      BI_Length: 0650
      BI_Weight: 059
      Date_Time: 16.09.2014 10:33:35
      IDENT: 02773
      Name: A Header
      PM_Certificate: ----
      PM_Grade: S355
      PM_HeatNo: ----
      PM_No: ----
      PM_Supplier: ----
      PM_Treatment: cast
      Session_No:   2
      Trial_No:   40
    PCR Settings
    PCR Tools
    PQF Settings FCS
    PQF Settings MPS
    PQF Settings RSC
    $PDA_MultiStation: standalone
    Module_90
      CaptureCAM: 1
        Videoserver: iba\10.51.150.21
        User:
        Pass:
```

| | Show | SignalName | Expression | Unit |
|---|---|---|---|---|
| 1 | ✓ | InfoField(0, "Heade | InfoField(0, "Header.BI_Diameter") | |
| 2 | ✓ | InfoField(0, "Heade | InfoField(0, "Header.BI_Length") | |
| 3 | ✓ | InfoFieldText(0, "Mc | InfoFieldText(0, "Module_90.CaptureCAM.Videoserver") | |

This enhancement is purely aesthetical and does not affect any other ibaAnalyzer functionality.

## 4.2 Tooltip for long infofields in signal tree.

For performance reasons, the total length of the infofield name and the concatenated value cannot exceed 80 characters before it is cut off when shown in the signal tree. However, in the current version of ibaAnalyzer, a tooltip is shown. It shows the entire infofield if one hovers over it.

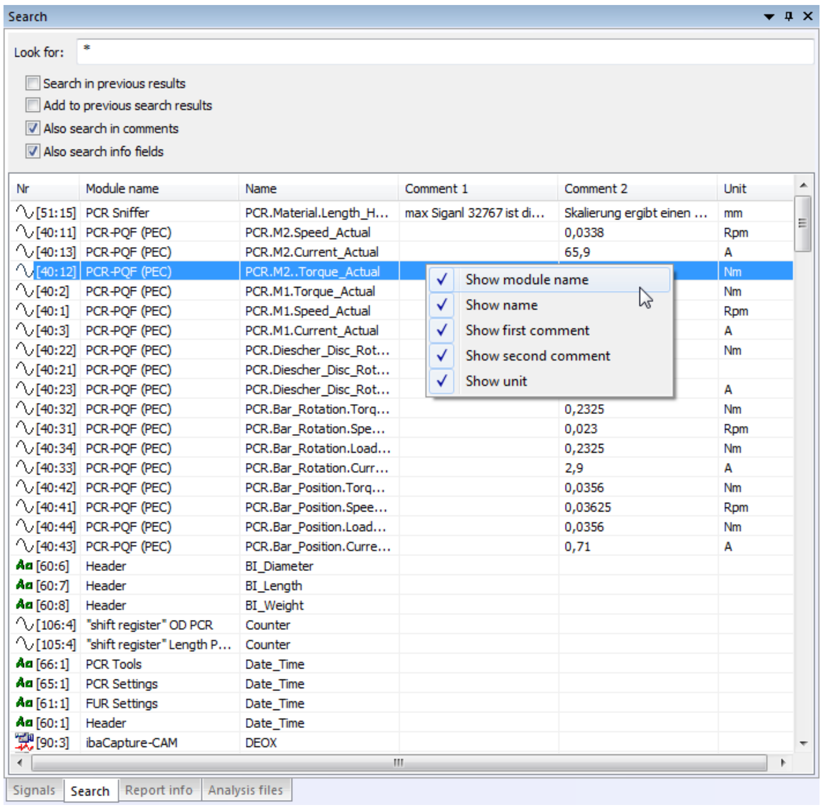


```
D:\klad\detlef\KQU__2011_16988_01_00_0700_2011-01-07_09.53.17.dat
  info
    clk: 0.005
    typ: real
    starttime: 07.01.2011 09:53:17.055000
    frames: 0000056896
    starttrigger: 0000000000
    stoptrigger: 0000056896
    $PDA_RefTimestamp: 00063429990797935000
    KQ_TechnoString
      ABWNR:   .
      AUS_BNR_LOS_TLOS: 16988_01_00
      Dicke_Fertig:  1.700
      FileName: 2011_16988_01_00_0700
      TechnoString: 2011_16988_01_00_0700 WKST: 2104 PWKST_oben:          PWKST...
    version: ibaPDA 6.25.0
  0. KQ_TCS_1
  1. KQ_TechnoString
  2. KQ_Fahr_S7_1
  4. KQ_Geruest_S7_1
  6. KQ_Fahr_S7_3
  8. KQ_TCS_DPM_1
  10. KQ_Conveyor_S7_1
  12. KQ_Padu_1
  14. Laufzeit
```

```
KQ_TechnoString.TechnoString: 2011_16988_01_00_0700 WKST: 2104 PWKST_oben:
        PWKST_unten:              Dicke_Einlauf:  5.20 Dicke_Auslauf: 3.50
 Breite: 1680.00 Dicke_Bis: 3.52 Dicke_Von: 3.48 Walzendurchm_Oben: 449.90 Walze
ndurchm_Unten: 449.90 AP_NR:  1604 AP_LNR:     6 Einlaufgewicht: 10005.00 Zielbr
eite: 1680.00 Sollgeschw:  200.00 PlatierschDiOben:   .00 PlatierschDiUnten:   .
00 WKST_Text:        6016.2 OBFL_TEXT:           EDT QUAL_TEXT: Automobilqua
lit AUSF_TEXT: Ohne Definition AG_Text:          ABBLASSTICH VON_ANL:  KQ ZU_A
NL:  BO Dicke_Fertig:  1.700 ABWNR:     . V_Soll_Techno:  200.00 AUS_BNR_LOS_TLO
S: 16988_01_00 Hinweis_Text:
              Bridlerolle   ENDE
```

## 4.3 Additional module column in search dialog

It is possible to add a column in the search dialog that depicts the name of the module to which the signals belong. Also, the signal name and unit can now be hidden (previously only comments could be hidden or shown). To show or hide columns, one can right-click on the grid and check or uncheck the entries in the context menu.



Look for: *

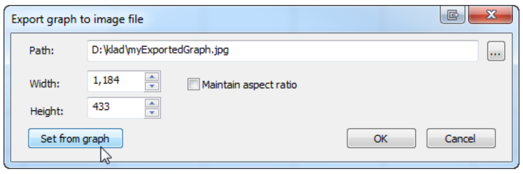
- [ ] Search in previous results
- [ ] Add to previous search results
- [x] Also search in comments
- [x] Also search info fields

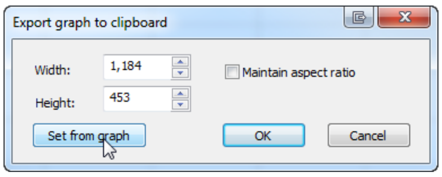
| Nr | Module name | Name | Comment 1 | Comment 2 | Unit |
|---|---|---|---|---|---|
| [51:15] | PCR Sniffer | PCR.Material.Length_H... | max Siganl 32767 ist di... | Skalierung ergibt einen ... | mm |
| [40:11] | PCR-PQF (PEC) | PCR.M2.Speed_Actual |  | 0,0338 | Rpm |
| [40:13] | PCR-PQF (PEC) | PCR.M2.Current_Actual |  | 65,9 | A |
| [40:12] | PCR-PQF (PEC) | PCR.M2..Torque_Actual |  |  | Nm |
| [40:2] | PCR-PQF (PEC) | PCR.M1.Torque_Actual |  |  | Nm |
| [40:1] | PCR-PQF (PEC) | PCR.M1.Speed_Actual |  |  | Rpm |
| [40:3] | PCR-PQF (PEC) | PCR.M1.Current_Actual |  |  | A |
| [40:22] | PCR-PQF (PEC) | PCR.Diescher_Disc_Rot... |  |  | Nm |
| [40:21] | PCR-PQF (PEC) | PCR.Diescher_Disc_Rot... |  |  |  |
| [40:23] | PCR-PQF (PEC) | PCR.Diescher_Disc_Rot... |  |  | A |
| [40:32] | PCR-PQF (PEC) | PCR.Bar_Rotation.Torq... |  | 0,2325 | Nm |
| [40:31] | PCR-PQF (PEC) | PCR.Bar_Rotation.Spe... |  | 0,023 | Rpm |
| [40:34] | PCR-PQF (PEC) | PCR.Bar_Rotation.Load... |  | 0,2325 | Nm |
| [40:33] | PCR-PQF (PEC) | PCR.Bar_Rotation.Curr... |  | 2,9 | A |
| [40:42] | PCR-PQF (PEC) | PCR.Bar_Position.Torq... |  | 0,0356 | Nm |
| [40:41] | PCR-PQF (PEC) | PCR.Bar_Position.Spee... |  | 0,03625 | Rpm |
| [40:44] | PCR-PQF (PEC) | PCR.Bar_Position.Load... |  | 0,0356 | Nm |
| [40:43] | PCR-PQF (PEC) | PCR.Bar_Position.Curre... |  | 0,71 | A |
| [60:6] | Header | BI_Diameter |  |  |  |
| [60:7] | Header | BI_Length |  |  |  |
| [60:8] | Header | BI_Weight |  |  |  |
| [106:4] | "shift register" OD PCR | Counter |  |  |  |
| [105:4] | "shift register" Length P... | Counter |  |  |  |
| [66:1] | PCR Tools | Date_Time |  |  |  |
| [65:1] | PCR Settings | Date_Time |  |  |  |
| [61:1] | FUR Settings | Date_Time |  |  |  |
| [60:1] | Header | Date_Time |  |  |  |
| [90:3] | ibaCapture-CAM | DEOX |  |  |  |

# 5 Export

## 5.1 Persistent sizes for exported graphs

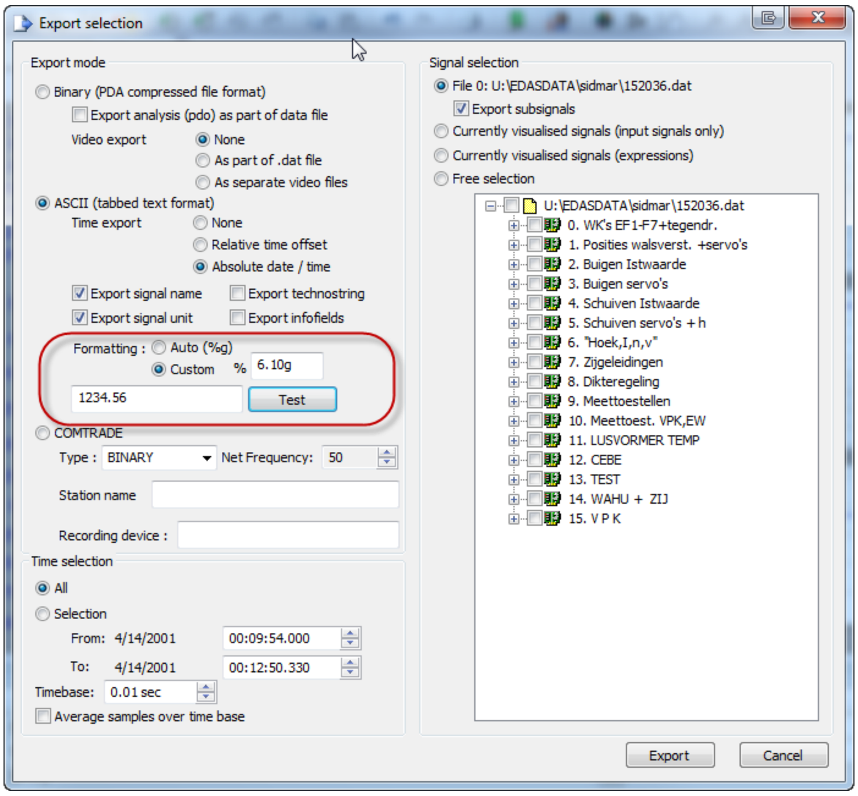
When exporting graphs as an image to a file or the clipboard, the specified width and height are now persistent rather than taken from the graph. A button is present to reset the width and height to the dimensions of the selected graph.


Export graph to image file
Path: D:\klad\myExportedGraph.jpg
Width: 1,184
Height: 433
Maintain aspect ratio


Export graph to clipboard
Width: 1,184
Height: 453
Maintain aspect ratio

## 5.2 Export numerical format

The options in the extract dialog to specify the number formatting for extracting to ASCII file are now also present in the file export dialog.



Export selection

Export mode

- ( ) Binary (PDA compressed file format)
  - [ ] Export analysis (pdo) as part of data file
  - Video export: (●) None ( ) As part of .dat file ( ) As separate video files
- (●) ASCII (tabbed text format)
  - Time export: ( ) None ( ) Relative time offset (●) Absolute date / time
  - [✓] Export signal name [ ] Export technostring
  - [✓] Export signal unit [ ] Export infofields
  - Formatting : ( ) Auto (%g) (●) Custom % 6.10g
  - 1234.56 Test
- ( ) COMTRADE
  - Type : BINARY Net Frequency: 50
  - Station name
  - Recording device :

Time selection

- (●) All
- ( ) Selection
  - From: 4/14/2001 00:09:54.000
  - To: 4/14/2001 00:12:50.330
- Timebase: 0.01 sec
- [ ] Average samples over time base

Signal selection

- (●) File 0: U:\EDASDATA\sidmar\152036.dat
  - [✓] Export subsignals
- ( ) Currently visualised signals (input signals only)
- ( ) Currently visualised signals (expressions)
- ( ) Free selection

U:\EDASDATA\sidmar\152036.dat
- 0. WK's EF1-F7+tegendr.
- 1. Posities walsverst. +servo's
- 2. Buigen Istwaarde
- 3. Buigen servo's
- 4. Schuiven Istwaarde
- 5. Schuiven servo's +h
- 6. "Hoek,I,n,v"
- 7. Zijgeleidingen
- 8. Dikteregeling
- 9. Meettoestellen
- 10. Meettoest. VPK,EW
- 11. LUSVORMER TEMP
- 12. CEBE
- 13. TEST
- 14. WAHU + ZIJ
- 15. V P K

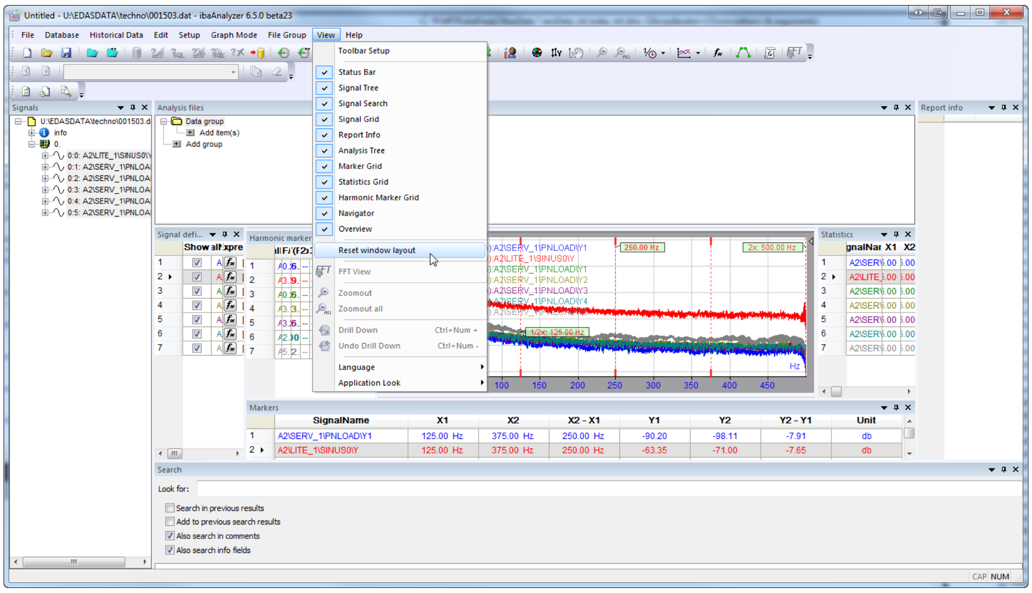
Export Cancel

## 6 View

### 6.1 Reset dockable window layout

The layout of dockable windows (signal tree, signal grid, marker grids, etc…) can be reset to the default locations that ibaAnalyzer uses when it is installed for the first time.

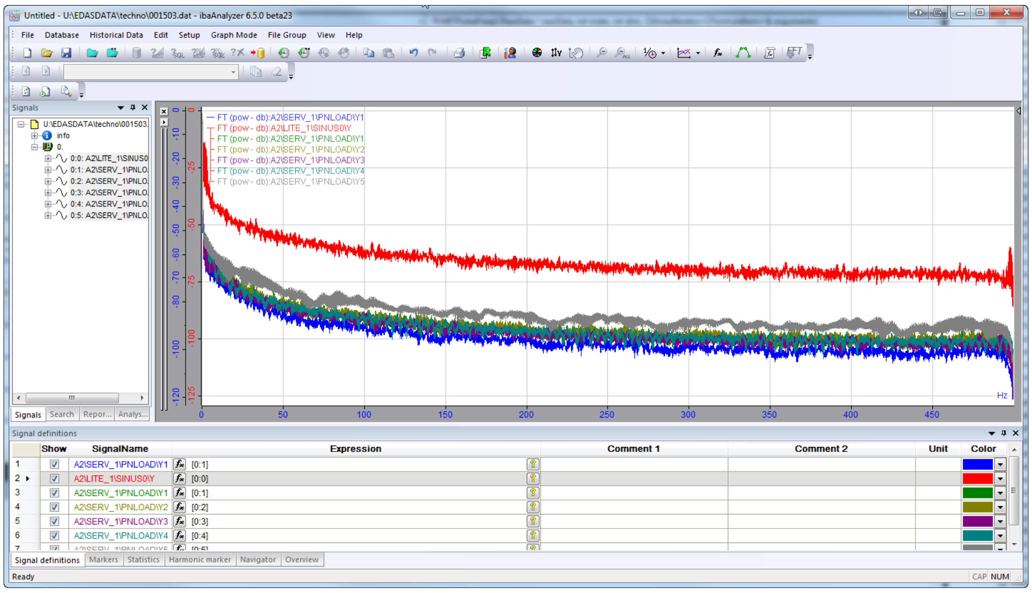
*Menu → View → Reset window Layout*



Markers

| | SignalName | X1 | X2 | X2 - X1 | Y1 | Y2 | Y2 - Y1 | Unit |
|---|---|---|---|---|---|---|---|---|
| 1 | A2\SERV_1\PNLOAD\Y1 | 125.00 Hz | 375.00 Hz | 250.00 Hz | -90.20 | -98.11 | -7.91 | db |
| 2 | A2\LITE_1\SINUS0\Y | 125.00 Hz | 375.00 Hz | 250.00 Hz | -63.35 | -71.00 | -7.65 | db |

- [ ] Search in previous results
- [ ] Add to previous search results
- [✓] Also search in comments
- [✓] Also search info fields


Untitled - U:\EDASDATA\techno\001503.dat - ibaAnalyzer 6.5.0 beta23
File
Database
Historical Data
Edit
Setup
Graph Mode
File Group
View
Help
Signals
U:\EDASDATA\techno\001503.
info
0.
0:0: A2\LITE_1\SINUS0
0:1: A2\SERV_1\PNLO.
0:2: A2\SERV_1\PNLO.
0:3: A2\SERV_1\PNLO.
0:4: A2\SERV_1\PNLO.
0:5: A2\SERV_1\PNLO.
Signals
Search
Repor...
Analys...
FT (pow - db):A2\SERV_1\PNLOAD\Y1
FT (pow - db):A2\LITE_1\SINUS0\Y
FT (pow - db):A2\SERV_1\PNLOAD\Y1
FT (pow - db):A2\SERV_1\PNLOAD\Y2
FT (pow - db):A2\SERV_1\PNLOAD\Y3
FT (pow - db):A2\SERV_1\PNLOAD\Y4
FT (pow - db):A2\SERV_1\PNLOAD\Y5
Hz
Signal definitions
Markers
Statistics
Harmonic marker
Navigator
Overview
Ready
CAP NUM

Signal definitions

| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | A2\SERV_1\PNLOAD\Y1 | [0:1] | | | | 🟦 |
| 2 | ✓ | A2\LITE_1\SINUS0\Y | [0:0] | | | | 🟥 |
| 3 | ✓ | A2\SERV_1\PNLOAD\Y1 | [0:1] | | | | 🟩 |
| 4 | ✓ | A2\SERV_1\PNLOAD\Y2 | [0:2] | | | | 🟨 |
| 5 | ✓ | A2\SERV_1\PNLOAD\Y3 | [0:3] | | | | 🟪 |
| 6 | ✓ | A2\SERV_1\PNLOAD\Y4 | [0:4] | | | | 🟦 |
| 7 | ✓ | A2\SERV_1\PNLOAD\Y5 | [0:5] | | | | ⬜ |

# 7 Miscellaneous features

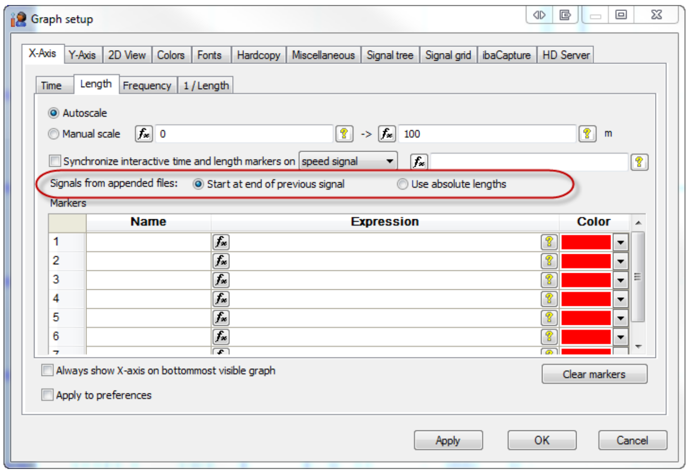
## 7.1 Treatment of signal offsets for length based signals

Length based signals generated by ibaPDA currently cannot have an x-offset different from 0. However, when extracted through ibaAnalyzer and derived from a computation, an x-offset can be introduced. When selecting such a signal from a chain of appended .dat files, the offsets introduce a gap; i.e. the offset is interpreted relatively to the end of the signal of the previous file.

In the current version of ibaAnalyzer, you can select that the offset should be interpreted as absolute; i.e. the signal of any file in the chain shall start at exactly the specified offset regardless of any other files in the chain. If the offset is smaller than the length of the signal in the previous file (including any offset of that signal), it might however be partially or entirely overlapped by that signal. To activate this option:

*Graph Setup* or *Preferences* → *X-Axis* → Select *"Length"* tab → *Signal from appended files* → Select the *"Use absolute lengths"* radio button.

To deactivate the option, select "*Start at end of previous signal*", this option is the default.


Graph setup

X-Axis | Y-Axis | 2D View | Colors | Fonts | Hardcopy | Miscellaneous | Signal tree | Signal grid | ibaCapture | HD Server

Time | Length | Frequency | 1 / Length

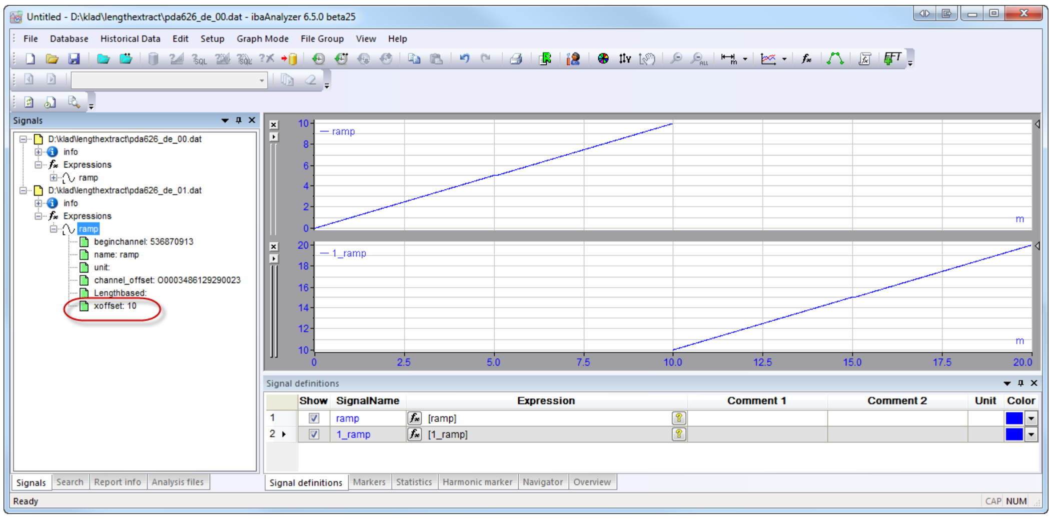
[✓ Autoscale]
Manual scale 0 -> 100 m
Synchronize interactive time and length markers on speed signal
Signals from appended files: [✓ Start at end of previous signal] Use absolute lengths

Markers

| | Name | Expression | Color |
|---|---|---|---|
| 1 | | | 🟥 |
| 2 | | | 🟥 |
| 3 | | | 🟥 |
| 4 | | | 🟥 |
| 5 | | | 🟥 |
| 6 | | | 🟥 |

Always show X-axis on bottommost visible graph
Apply to preferences

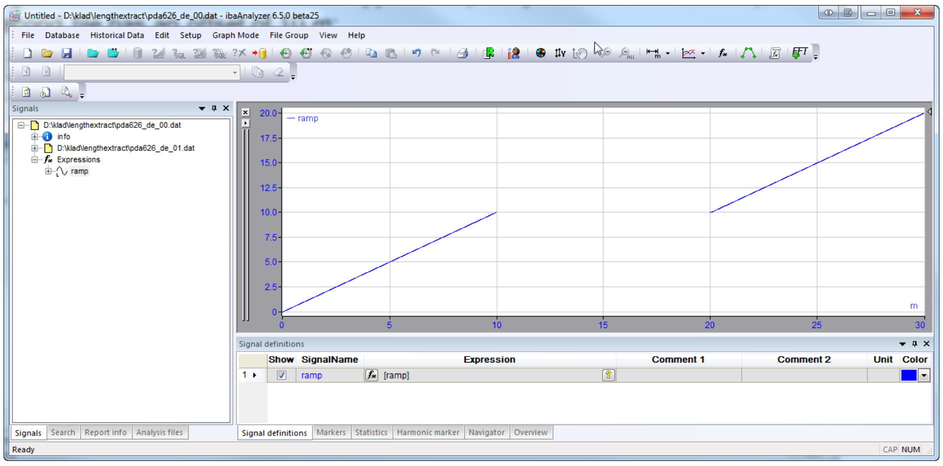
E.g.: two files "Added" rather than appended (the option has no influence), the signal in the second file has an offset of 10 m:



D:\klad\lengthextract\pda626_de_00.dat
info
Expressions
ramp
D:\klad\lengthextract\pda626_de_01.dat
info
Expressions
ramp
beginchannel: 536870913
name: ramp
unit:
channel_offset: O0003486129290023
Lengthbased:
xoffset: 10

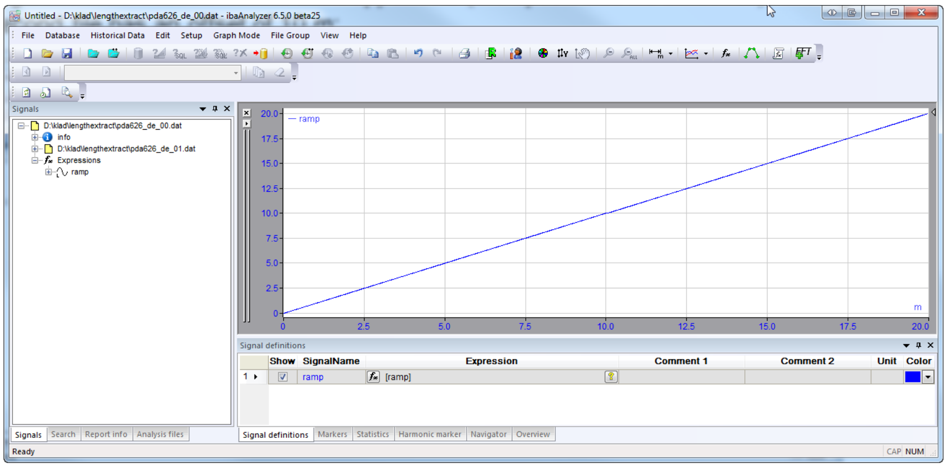
| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | ramp | [ramp] | | | | 🟦 |
| 2 | ✓ | 1_ramp | [1_ramp] | | | | 🟦 |

The same two files appended with the default option “*Start at end of previous signal*”; the offset of the second signal introduces a gap of 10 m:



| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | ramp | [ramp] | | | | 🟦 |

The same two files appended with the new option “*Use absolute lengths*”; the gap has disappeared as the second signal now starts at exact offset 10:

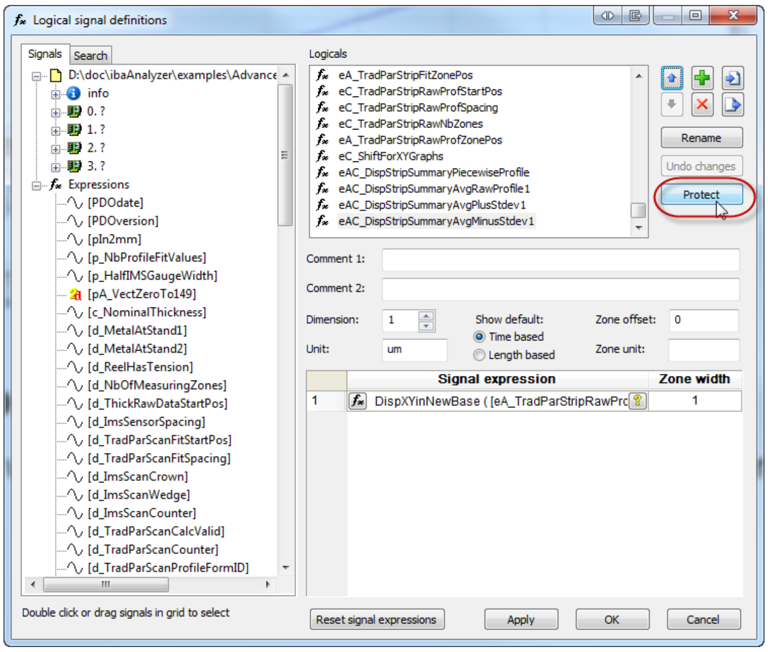


| | Show | SignalName | Expression | Comment 1 | Comment 2 | Unit | Color |
|---|---|---|---|---|---|---|---|
| 1 | ✓ | ramp | [ramp] | | | | 🟦 |

## 7.2 Password protection for logicals

Note, this functionality was implemented in 6.4.2, but it hasn't been documented in a new features document until now.

In the *logicals* dialog, there is a *Protect* button, here you can specify a new or alter an existing password.

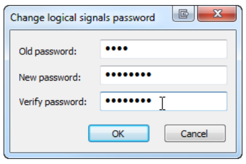


D:\doc\ibaAnalyzer\examples\Advance
info
0. ?
1. ?
2. ?
3. ?
Expressions
[PDOdate]
[PDOversion]
[pIn2mm]
[p_NbProfileFitValues]
[p_HalfIMSGaugeWidth]
[pA_VectZeroTo149]
[c_NominalThickness]
[d_MetalAtStand1]
[d_MetalAtStand2]
[d_ReelHasTension]
[d_NbOfMeasuringZones]
[d_ThickRawDataStartPos]
[d_ImsSensorSpacing]
[d_TradParScanFitStartPos]
[d_TradParScanFitSpacing]
[d_ImsScanCrown]
[d_ImsScanWedge]
[d_ImsScanCounter]
[d_TradParScanCalcValid]
[d_TradParScanCounter]
[d_TradParScanProfileFormID]

eA_TradParStripFitZonePos
eC_TradParStripRawProfStartPos
eC_TradParStripRawProfSpacing
eC_TradParStripRawNbZones
eA_TradParStripRawProfZonePos
eC_ShiftForXYGraphs
eAC_DispStripSummaryPiecewiseProfile
eAC_DispStripSummaryAvgRawProfile1
eAC_DispStripSummaryAvgPlusStdev1
eAC_DispStripSummaryAvgMinusStdev1

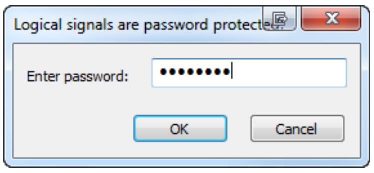
Comment 1:
Comment 2:
Dimension: 1
Show default: [✓ Time based] Length based
Zone offset: 0
Unit: um
Zone unit:

| | Signal expression | Zone width |
|---|---|---|
| 1 | DispXYinNewBase ( [eA_TradParStripRawPrc | 1 |



Old password: ••••
New password: ••••••••
Verify password: ••••••••

After having saved the analysis, the next time one loads the analysis and tries to access the *logicals* through the *logicals* dialog, a password dialog will pop up:



Enter password: ••••••••

If one can not specify the correct password, access to the *logicals* dialog will be denied.

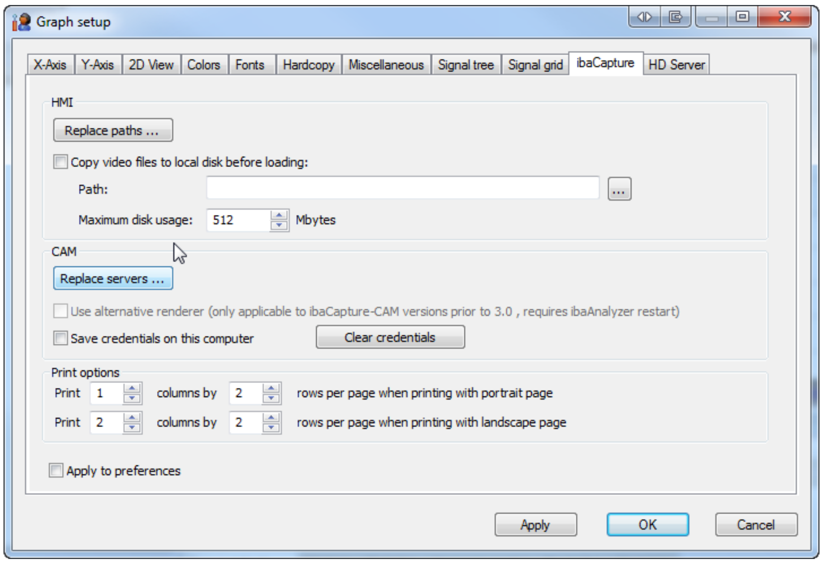
Note that this not prevent one from viewing the *logicals* in graphs or using the *logicals* in expressions specified in the signal grid, only the expressions used to compose the *logicals* are protected. Once a correct password is specified, the password will not be asked until the analysis is reloaded and one tries to access the *logicals* dialog again.

## 7.3 Path and server replacements for ibaCapture-CAM and HMI

Note, this functionality was implemented in 6.4.2 but it hasn't been documented in a new features document until now.

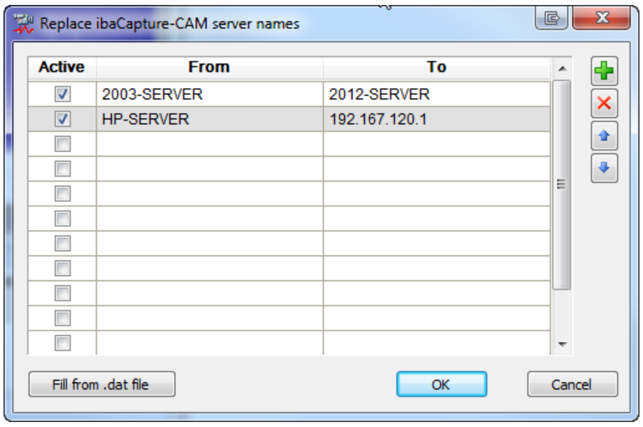
In previous versions of ibaAnalyzer, it was possible to specify a server replacement for ibaCapture-CAM in the case the original server had migrated since the recording of the .dat file or was only accessible through IP-address. Likewise, it was possible to specify a path replacement for ibaCapture-HMI if the *.fbr* files had moved. This was however limited as only one replacement could be specified, which is problematic if the .dat file contained ibaCapture modules from several locations or several .dat files were opened.

In the current version of ibaAnalyzer, the ibaCapture settings dialog (Preferences or Setup) has a button labeled *"Replace paths"* in the HMI section and in the CAM section, it has a button labeled *"Replace servers".*



Copy video files to local disk before loading:
Path:
Maximum disk usage: 512 Mbytes
Use alternative renderer (only applicable to ibaCapture-CAM versions prior to 3.0 , requires ibaAnalyzer restart)
Save credentials on this computer
Print 1 columns by 2 rows per page when printing with portrait page
Print 2 columns by 2 rows per page when printing with landscape page
Apply to preferences

Clicking the buttons causes an additional dialog to pop up:



| Active | From | To |
|---|---|---|
| ☑ | 2003-SERVER | 2012-SERVER |
| ☑ | HP-SERVER | 192.167.120.1 |
| ☐ | | |
| ☐ | | |
| ☐ | | |
| ☐ | | |
| ☐ | | |
| ☐ | | |
| ☐ | | |
| ☐ | | |
| ☐ | | |

The dialog consists of the following elements:

- A grid where one can specify the replacements. The following columns are present:
    - Active: Uncheck this column to disable an entry.
    - From: The source string to replace (server in the case of ibaCapture-CAM or a full or partial path in the case of ibaCapture-HMI). This is the server or path that is expected to be encountered in the opened .dat files.
    - To: The replacement string. This is the server (ibaCapture-CAM) or replacement (partial) path (ibaCapture-HMI) to where the actual video data is located.
    - The standard buttons to insert, remove or move entries up or down.
    - A button labeled “Fill from .dat file” that will fill in the “From” entries as ibaAnalyzer encounters them in any currently opened .dat files.
    - Ok button to confirm the replacements and return to the setup dialog.
    - Cancel button to discard the changes and return to the setup dialog.

The order of replacements is important as the first entries will be tried first in the case of multiple (partial) matches.