# Development of algorithms A2.3.2

Following document summarizes interface between the algorithm and the QWTB toolbox [1] to which it will be integrated. Updated versions of the document will be present at the TracePQM project webpage sharepoint.

Version V0.5, 27.11.2017, Stanislav Mašláň.

## Input quantities

The format of the input quantities is given by the QWTB design. QWTB toolbox passes algorithms to the algorithm’s wrapper function as a structure containing substructures, one for each quantity. Each quantity structure may contain several items. First, the values ‘v’, associated uncertainty ‘u’, etc. for more details see documentation of the QWTB. Example of the input to the algorithms wrapper:

DI.Ts.v – value of sampling period

DI.Ts.u – uncertainty of sampling period

DI.y.v – input waveform data

DI.y.u – uncertainty of the input waveform data

…

Note the uncertainty **‘u’** may not be present!

Each algorithm may have any number of custom parameters that are entered by the user, such as window type, etc.

Each algorithm will automatically receive following quantities from the TWM system:

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Note** | **type** | **Description** |
| Ts |  | Real scalar | Sampling period in [Seconds]. |
| y or  u and i | 2 | Real column vector(s) | Sample data. For single input channel algorithm, such as THD, only one vector ‘y’ will be passed.  For multichannel algorithms, such as power, two vectors are passed, the voltage and current. Both ‘u’ and ‘i’ vectors have the same length.  The samples are in [Volts] as returned by the digitizer (no transducer scaling). |
| time\_shift |  | Real scalar | Timeshift between ‘u’ and ‘i’ channel in [Seconds]. Applies only for multichannel algorithms. |
| jitter |  | Real scalar | Sampling jitter value [Seconds]. |
| adc\_gain | 1 | 2D real matrix | 2D matrix of the absolute gain coefficients of the digitizer in [Vout/Vin]. I.e. value 1.001 means the sample data will be multiplied by 1.001 to get corrected value. Dependent on the frequency ‘adc\_gain\_f’ and amplitude ‘adc\_gain\_a’. |
| adc\_gain\_f | 1, 2 | Real column vector | Independent variable of the ‘adc\_gain’ containing nominal frequency in [Hertz], one item per row of ‘adc\_gain’. |
| adc\_gain\_a | 1, 2 | Real row vector | Independent variable of the ‘adc\_gain’ containing nominal amplitude in [Volts], one item per column of ‘adc\_gain’. |
| adc\_phi | 1 | 2D real matrix | 2D matrix of the absolute phase correction coefficients of the digitizer channel in [rad]. Value +12e-6 rad means the phase of harmonic component must be increased by 12e-6 rad. Note this is not interchannel phase correction! Dependent on the frequency ‘adc\_phi\_f’ and amplitude ‘adc\_phi\_a’. |
| adc\_phi\_f | 1, 2 | Real column vector | Independent variable of the ‘adc\_phi’ containing nominal frequency in [Hertz], one item per row of ‘adc\_phi’. |
| adc\_phi\_a | 1, 2 | Real row vector | Independent variable of the ‘adc\_phi’ containing nominal amplitude in [Volts], one item per column of ‘adc\_phi’. |
| tr\_gain | 1 | 2D real matrix | 2D matrix of the absolute gain coefficients of the transducer in [Vin/Vout] for dividers or [Ain/Vout] for shunt. Dependent on the frequency ‘tr\_gain\_f’ and amplitude ‘tr\_gain\_a’. |
| tr\_gain\_f | 1, 2 | Real column vector | Independent variable of the ‘tr\_gain’ containing nominal frequency in [Hertz], one item per row of ‘tr\_gain’. |
| tr\_gain\_a | 1, 2 | Real row vector | Independent variable of the ‘tr\_gain’ containing nominal rms value in [Volts] or [Ampers], one item per column of ‘tr\_gain’. |
| tr\_phi | 1 | 2D real matrix | 2D matrix of the absolute phase correction coefficients of the transducer in [rad]. Dependent on the frequency ‘tr\_phi\_f’ and amplitude ‘tr\_phi\_a’. |
| tr\_phi\_f | 1, 2 | Real column vector | Independent variable of the ‘tr\_phi’ containing nominal frequency in [Hertz], one item per row of ‘tr\_phi’. |
| tr\_phi\_a | 1, 2 | Real row vector | Independent variable of the ‘tr\_phi’ containing nominal rms value in [Volts] or [Ampers], one item per column of ‘tr\_phi’. |
| crosstalk\_re  crosstalk\_im | 1 | Real column vectors | Complex crosstalk coefficients expressing complex transfer from ‘u’ channel to ‘i’ channel defined as: crosstalk = i/u. Crosstalk in the opposite direction is assumed to be identical. The value is dependent on the frequency ‘crosstalk\_f’. |
| crosstalk\_f | 1, 2 | Real column vector | Independent variable of the ‘crosstalk’ containing nominal frequency in [Hertz], one item per row of ‘crosstalk’. |
| adc\_sfdr | 1, 2 | 2D real matrix | Spurious Free Dynamic Range coefficients of the digitizer channel [dBc]. The values are ratios of the fundamental amplitude to the highest spurious component, i.e. 100 dBc means highest spur is fundamental\_amplitude\*1e-5. The value is dependent on the fundamental frequency ‘adc\_sfdr\_f’ and amplitude ‘adc\_sfdr\_a’. |
| adc\_sfdr\_f | 1, 2 | Real column vector | Independent variable of the ‘adc\_sfdr’ containing frequency of the fundamental harmonic in [Hertz], one item per row of ‘adc\_sfdr’. |
| adc\_sfdr\_a | 1, 2 | Real row vector | Independent variable of the ‘adc\_sfdr’ containing amplitude of the fundamental harmonic in [Volts], one item per column of ‘adc\_sfdr’. |
| tr\_sfdr  tr\_sfdr\_f  tr\_sfdr\_a | 1, 2 |  | Spurious Free Dynamic Range coefficients of the transducer. Meaning is the same as for digitizer. |
| adc\_bits | 1, 2 | Integer scalar | Bit resolution of the ADC of the digitizer. |
| adc\_nrng | 1, 2 | Real scalar | Range of the digitizer channel in [Volts]. |
| lsb | 1, 2 | Real scalar | Value of the least significant bit of the ADC [Volts]. |

**Note 1):** The parameters are defined for each channel of the measurement system. For algorithms with single input **‘y’** the names of the parameters are as defined in the table above. For multichannel algorithms which have two inputs **‘u’** and **‘i’** the parameters will be combined with prefixes defining the channel. See following example for parameter naming rules:

|  |  |  |
| --- | --- | --- |
| **Parameter name** | **U channel parameter name** | **I channel parameter name** |
| adc\_gain | u\_adc\_gain | i\_adc\_gain |
| adc\_gain\_f | u\_adc\_gain\_f | i\_adc\_gain\_f |
| adc\_nrng | u\_adc\_nrng | i\_adc\_nrng |
| … | … | … |

**Note 2):** There parameters have no assigned uncertainty, just value **‘v’**.

Note if any correction is not available (not loaded to the TWM system), it will be still passed into the algorithm but with nominal value, such as 1.0 for gains, 0.0 for phase, etc.

Note that independent variables (**amplitude** and **frequency**) of the 1D or 2D dependencies in the input quantities table may differ for **each channel** and even for **gain** and **phase** of the same correction! The ranges and steps of the independent variables depend on the user correction data files. Each algorithm must check the range of each of the correction individually and somehow respond if the correction range does not cover the required range (throw and error, warning, etc.).

Note the 1D and 2D corrections which are dependent on the **frequency** or **amplitude** quantity may have one or both of the dependencies undefined! I.e. the corresponding dimension of the correction data **‘v’** and uncertainty **‘u’** matrices will have size of 1. In such case the algorithm shall assume the correction is not dependent on that quantity and apply the correction and its uncertainty in the whole range of **frequency**, **amplitude** or both. E.g.:

adc\_sfdr.v = [93]; means algorithm shall assume 93 dBc SFDR for all frequencies and amplitudes. adc\_sfdr.v = [93; 90]; adc\_sfdr\_f.v = [1e3; 1e4]; means to assume 93 dBc for frequency 1e3 Hz and 90 dBc for frequency 1e4.

Note the SFDR data are not meant as corrections. These are only for estimation of the uncertainty.

Note the **‘lsb’** parameter may not be present depending on the selected digitizer. If it is not available, the algorithm should use combination of the **‘adc\_bits’** and **‘adc\_nrng’** for estimation of the **‘lsb’**.

Note even if the algorithm will not implement **‘crosstalk’** correction, it should at least take it into account as an uncertainty for the uncertainty estimation.

## Output quantities

Algorithm may return any quantities: scalars, vectors or matrices. Naming of the output quantities is irrelevant. It will be translated by the QWTB toolbox.

If the algorithm calculates frequency **spectrum** in some intermediate phase of the calculation, it is preferred to return it as an output quantity together with its frequency scale so it can be displayed in the TWM software.

## Resources

[1] QWTB toolbox, www: <https://qwtb.github.io/qwtb/>