Met4FoF-redundancy

Release 0.1

GK

Jun 14, 2021

GETTING STARTED:

Met4FoF-redundancy is a software package containing software tools that can be used to analyze measurement data which contain redundancy. The main part of the development has been performed at VSL (Netherlands), with help of the partner institutes NPL (UK), PTB (Germany) and UCAM (UK).

This software has been developed as part of the joint European Research Project EMPIR 17IND12 Met4FoF "Metrology for the Factory of the Future".

A scientific publication explaining the ideas behind this software can be found in [Kok01]. Related work can be found in [Kok02].

The package is written in Python 3. The source code can be found at GitHub.

CHAPTER

ONE

GETTING STARTED

Please download the Python code on GitHub and use the functions that may be useful for you.

Usage of the module `MFred` is relatively straightforward. Note that all top level functions have their own *test* functions illustrating their usage. Please refer to other sections in this documentation for more information.

Usage of the module `agentMFred` requires the agentMet4FoF framework and in particular the metrological agents. See this Read the Docs page for information on how to install and use the agentMet4FoF framework.

CHAPTER

TWO

MFRED: PYTHON FUNCTIONS FOR BEST ESTIMATE CALCULATION IN A SENSOR NETWORK IN THE PRESENCE OF REDUNDANCY

In this section some method for analysing redundant measurement data is presented. *Redundancy* means that there is more than one way to derive the value of the measurand *Y* from the values of the sensor data *X_i*. Following main cases are considered in the module:

1. Redundant measurement of the measurand *Y* by independent sensors directly measuring *Y*
2. Redundant measurement of the measurand *Y* by correlated sensors directly measuring *Y*
3. Redundant measurement of the measurand *Y* by correlated sensors *X_i* indirectly measuring *Y*, with a linear relationship $\mathbf{y} = \mathbf{a} + A * \mathbf{x}$ between the vector $\mathbf{x}$ of sensor values and the vector $\mathbf{y}$ containing the various (redundant) estimates of the measurand *Y*, where $\mathbf{a}$ is a vector and *A* a matrix both of appropriate size.

Details of the different modules are presented in the next sections.

2.1 Details of the main module `redundancy1`

The module `redundancy1` implements methods for analysing redundant estimates provided by redundant measurement data from a sensor network.

The main functions included in the file *redundancy1.py* are:

1. ***calc_consistent_estimates_no_corr()*: Calculation of *n_rows* of best estimates for *n_rows* of sets of** independent estimates with associated standard uncertainty.
2. ***calc_best_estimate()*: Calculation of the best estimate for a given set of estimates with associated uncertainty** matrix.
3. *calc_lcs()*: Calculation of the largest subset of consistent estimates of a measurand.
4. ***calc_lcss()*: Calculation of the largest subset of sensor values that yield consistent estimates of a measurand** linked to the sensor values by a linear system of equations.

The scientific publication giving more information on this topic is:

> G. Kok and P. Harris, "Uncertainty Evaluation for Metrologically Redundant Industrial Sensor Networks," 2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT, Roma, Italy, 2020, pp. 84-88, doi: 10.1109/MetroInd4.0IoT48571.2020.9138297.

exception Met4FoF_redundancy.MFred.redundancy1.**AllColumnsZeroSum**
Custom exception to handle the case when all columns have zero sum

exception Met4FoF_redundancy.MFred.redundancy1.**ColumnNotZeroError**
Custom exception to handle the case when a redundant column has not been reduced to zero

exception Met4FoF_redundancy.MFred.redundancy1.**SensorsNotLinearlyIndependentError**
Custom exception to handle the case when sensor results are not linearly independent

exception Met4FoF_redundancy.MFred.redundancy1.**SystemMatrixNotReducibleError**
Custom exception to handle the case when the system matrix *A* is not reducible

Met4FoF_redundancy.MFred.redundancy1.**calc_best_est_lin_sys**(*a_arr*, *a_arr2d*, *x_arr*, *vx_arr2d*, *problim*)
Function to calculate the best estimate of a linear system **y** = **a** + A * **x** and determines if the inputs are consistent in view of *problim*.

Parameters

- **a_arr** (*np.ndarray of shape (n_estimates)*) – vector **a** of linear system **y** = **a** + A * **x**
- **a_arr2d** (*np.ndarray of shape (n_estimates, n_sensors)*) – matrix A of linear system **y** = **a** + A * **x**
- **x_arr** (*np.ndarray of shape (n_sensors)*) – vector with sensor values vector **x** of linear system **y** = **a** + A * **x**
- **vx_arr2d** (*np.ndarray of shape (n_sensors, n_sensors)*) – uncertainty matrix associated with vector x_arr
- **problim** (*float*) – probability limit used for consistency evaluation. Typically 0.95.

Returns

- **isconsist** (*bool*) – indicator whether provided estimates are consistent in view of *problim*
- **ybest** (*float*) – best estimate
- **uybest** (*float*) – standard uncertainty of best estimate
- **chi2obs** (*float*) – observed chi-squared value

Met4FoF_redundancy.MFred.redundancy1.**calc_best_estimate**(*y_arr*, *vy_arr2d*, *problim*)
Calculate the best estimate for a set of estimates with associated uncertainty matrix, and determine if the set of estimates are consistent using a provided limit probability.

Parameters

- **y_arr** (*np.ndarray of shape (n)*) – vector of estimates of a measurand Y
- **vy_arr2d** (*np.ndarray of shape (n, n)*) – uncertainty matrix associated with y_arr
- **problim** (*float*) – probability limit used for assessing the consistency of the estimates. Typically, problim equals 0.95.

Returns

- **isconsist** (*bool*) – indicator whether provided estimates are consistent in view of *problim*
- **ybest** (*float*) – best estimate of measurand
- **uybest** (*float*) – uncertainty associated with *ybest*
- **chi2obs** (*float*) – observed value of chi-squared, used for consistency evaluation

Met4FoF_redundancy.MFred.redundancy1.**calc_consistent_estimates_no_corr**(*y_arr2d*, *uy_arr2d*, *prob_lim*)
Calculation of consistent estimate for n_sets of estimates y_ij (contained in y_arr2d) of a quantity Y, where each set contains n_estims estimates. The uncertainties are assumed to be independent and given in uy_arr2d. The consistency test is using limit probability limit prob_lim. For each set of estimates, the best estimate, uncertainty, observed chi-2 value and a flag if the provided estimates were consistent given the model are given as output.

Parameters

- **y_arr2d** (*np.ndarray of size (n_rows, n_estimates)*) – each row contains m=n_estimates independent estimates of a measurand
- **uy_arr2d** (*np.ndarray of size (n_rows, n_estimates)*) – each row contains the standard uncertainty u(y_ij) of y_ij = y_arr2d[i,j]
- **prob_lim** (*limit probability used in consistency test. Typically 0.95.*) –

Returns

- **isconsist_arr** (*bool array of shape (n_rows)*) – indicates for each row if the n_estimates are consistent or not
- **ybest_arr** (*np.ndarray of shape (n_rows)*) – contains the best estimate for each row of individual estimates
- **uybest_arr** (*np.ndarray of shape (n_rows)*) – contains the uncertainty associated with each best estimate for each row of *y_arr2d*
- **chi2obs_arr** (*observed chi-squared value for each row*)

Met4FoF_redundancy.MFred.redundancy1.**calc_lcs**(*y_arr*, *vy_arr2d*, *problim*)

Function to calculate the best estimate of a measurand based on individual estimates of the measurand with associated uncertainty matrix.

Parameters

- **y_arr** (*np.ndarray of shape (n)*) – vector with estimates of the measurand
- **vy_arr2d** (*np.ndarray of shape (n, n)*) – uncertainty matrix of the vector y_arr
- **problim** (*float*) – limit probability used in the consistency evaluation. Typically 0.95.

Met4FoF_redundancy.MFred.redundancy1.**calc_lcss**(*a_arr*, *a_arr2d*, *x_arr*, *vx_arr2d*, *problim*)

Calculation of the largest consistent subset of sensor values and the implied best estimate.

Parameters

- **x_arr** (*np.ndarray of shape (n_sensors)*) –
- **vx_arr2d** (*np.ndarray of shape (n_sensors, n_sensors)*) –
- **a_arr** (*np.ndarray of shape (n_estimates)*) –
- **a_arr2d** (*np.ndarray of shape (n_estimates, n_sensors)*) –
- **problim** (*float*) –
- **a_arr** – vector **a** of linear system **y** = **a** + A * **x**
- **a_arr2d** – matrix A of linear system **y** = **a** + A * **x**
- **x_arr** – vector with sensor values vector **x** of linear system **y** = **a** + A * **x**
- **vx_arr2d** – uncertainty matrix associated with vector x_arr
- **problim** – probability limit used for consistency evaluation. Typically 0.95.

Returns

- **isconsist** (*bool*) – indicator whether provided estimates are consistent in view of *problim*
- **ybest** (*float*) – best estimate
- **uybest** (*float*) – standard uncertainty of best estimate

- **chi2obs** (*float*) – observed chi-squared value

Met4FoF_redundancy.MFred.redundancy1.**print_output_cbe**(*isconsist_arr*, *ybest_arr*, *uybest_arr*, *chi2obs_arr*)

Function to print the full output of calc_best_estimate.

Parameters

- **isconsist_arr** (*bool array of shape (n_rows)*) – indicates for each row if the n_estimates are consistent or not
- **ybest_arr** (*np.ndarray of shape (n_rows)*) – contains the best estimate for each row of individual estimates
- **uybest_arr** (*np.ndarray of shape (n_rows)*) – contains the uncertainty associated with each best estimate for each row of *y_arr2d*
- **chi2obs_arr** (*observed chi-squared value for each row*) –

Met4FoF_redundancy.MFred.redundancy1.**print_output_lcs**(*n_sols*, *ybest*, *uybest*, *chi2obs*, *indkeep*, *y_arr*)

Method to print the output of the method *calc_lcs()*.

Parameters

- **n_sols** (*int*) – number of best solutions
- **ybest** (*float or np.ndarray of shape (n_sols)*) – best estimate or vector of best estimates
- **uybest** (*float or np.ndarray of shape (n_sols)*) – standard uncertainty of best estimate or vector with standard uncertainty of best estimates
- **chi2obs** (*float*) – observed chi-squared value of all best solutions
- **indkeep** (*np.ndarary of shape (n) or (n_sols, n)*) – indices of retained estimates of y_arr for the calculation of the best estimate ybest
- **y_arr** (*np.ndarray of shape (n)*) – individual estimates of measurand

Met4FoF_redundancy.MFred.redundancy1.**print_output_single**(*isconsist*, *ybest*, *uybest*, *chi2obs*)

Function to print the output of a single row of the calculate_best_estimate function.

Parameters

- **isconsist** (*bool*) – Indicates if provided estimates were consistent
- **ybest** (*float*) – best estimate
- **uybest** (*float*) – uncertainty of best estimate
- **chi2obs** (*float*) – observed value of chi-squared

2.2 Details of the test module test1

The module test_MFred.py calls all test functions which are implemented in the module redundancy1. These test functions are:

- *test_calc_consistent_estimates_no_corr()*
- *test_calc_best_estimate()*
- *test_calc_lcs()*
- *test_calc_lcss()*

test_redundancy.test_MFred.**test_calc_best_estimate**()
Test function for calc_best_estimate.

test_redundancy.test_MFred.**test_calc_consistent_estimates_no_corr**()
Test function for calc_consistent_estimates_no_corr(), implementing two test cases.

test_redundancy.test_MFred.**test_calc_lcs**()
Test function for `calc_lcs()`. Implements 4 test cases.

test_redundancy.test_MFred.**test_calc_lcss**()
Test function for method `calc_lcss()`. Implements 4 test cases.

CHAPTER

THREE

AGENTMFRED: PYTHON SOFTWARE AGENTS FOR PROCESSING REDUNDANT MEASUREMENT DATA

Some of the methods of the module `MFred` have been incorporated into a Redundancy Agent that can be used in the Met4FoF agent framework. The Redundancy Agent can be found in `redundancyAgents1`. It uses of a metrological datastream which can be found in `metrological_streams_v2`. The usage of the Redundancy Agent is illustrated with two examples contained in two tutorials.

In tutorial `redundancyAgents_tutorial_1` four independent signals are generated and the Redundancy Agent calculates the best estimate with associated uncertainty, respecting the input uncertainties, and rejecting sensor values that may be erroneous. In this case the sensors directly measure the measurand.

In tutorial `redundancyAgents_tutorial_2` a single signal containing redundant, correlated information is analyzed, and the best estimate with associated uncertainty, respecting all provided input uncertainties, and rejecting sensor values that may be erroneous. In this case the sensors do not directly measure the measurand, but the measurand is linked to the sensor values by means of four linear equations. The fact that there are four equations and not just one is the cause of the redundancy.

Details of the different modules are presented in the next sections.

3.1 Details of the module `metrological_streams_v2`

This module contains the definition of a metrological datastream that can be used to generate data for usage in the agentMET4FOF framework.

class Met4FoF_redundancy.agentMFred.metrological_streams_v2.**MetrologicalMultiWaveGenerator**(*sfreq: int = 500, freq_arr: numpy.array = array([50]), ampl_arr: numpy.array = array([1]), phase_ini_arr: numpy.array = array([0]), intercept: float = 0, device_id: str = 'DataGenerator', time_name: str = 'time', time_unit: str = 's', quantity_names: Union[str, Tuple[str, ...]] = ('Length', 'Mass'), quantity_units: Union[str, Tuple[str, ...]] = ('m', 'kg'),*

are returned.

Parameters

- **sfreq** (*float*) – sampling frequency which determines the time step when next_sample is called.
- **intercept** (*float*) – constant intercept of the signal
- **freq_arr** (*np.ndarray of float*) – array with frequencies of components included in the signal
- **ampl_arr** (*np.ndarray of float*) – array with amplitudes of components included in the signal
- **phase_ini_arr** (*np.ndarray of float*) – array with initial phases of components included in the signal

3.2 Details of the module redundancyAgents1

This module defines a Redundancy Agent that can be used in the agentMET4FOF framework. It has two main data processing types: - lcs: best estimate calculation using Largest Consistent Subset method - lcss: best estimate calculation using Largest Consistent Subset of Sensor values method

class Met4FoF_redundancy.agentMFred.redundancyAgents1.**MetrologicalMultiWaveGeneratorAgent**(*name=''*, *host=None*, *serializer=None*, *transport=None*, *attributes=None*, *backend='osbrain'*, *mesa_model=No*

An agent streaming a signal composed of various sine and cosine components. Takes samples from the MultiWaveGenerator and pushes them sample by sample (or in batches) to connected agents via its output channel.

agent_loop()

Model the agent's behaviour On state *Running* the agent will extract sample by sample the input data streams content and push it via invoking AgentMET4FOF.send_output().

init_parameters(*signal: Met4FoF_redundancy.agentMFred.metrological_streams_v2.MetrologicalMultiWaveGenerator = <Met4FoF_redundancy.agentMFred.metrological_streams_v2.MetrologicalMultiWaveGenerator object>*, ***kwargs*)

Initialize the input data Initialize the input data stream as an instance of the MultiWaveGenerator class

Parameters **signal** (*Signal*) – the underlying signal for the generator

class Met4FoF_redundancy.agentMFred.redundancyAgents1.**RedundancyAgent**(*name=''*, *host=None*, *serializer=None*, *transport=None*, *attributes=None*, *backend='osbrain'*, *mesa_model=None*)

This is the main Redundancy Agent class. Main calculation types are `lcs()` and `lcss()`, as defined in the module `redundancy1`.

agent_loop()

Model the agent's behaviour On state *Running* the agent will extract sample by sample the input data streams content and push it via invoking `AgentMET4FOF.send_output()`.

init_parameters(*input_data_maxlen=25*, *output_data_maxlen=25*)

Initialize the input data stream as an instance of the `MultiWaveGenerator` class.

Parameters **signal** (*Signal*) – the underlying signal for the generator

init_parameters1(*calc_type*, *sensor_key_list*, *n_pr*, *problim*)

Parameters used for both methods `lcs()` and `lcss()`.

Parameters

- **calc_type** (*str*) – calculation type: 'lcs' or 'lcss'
- **sensor_key_list** (*list of strings*) – list containing the names of the sensors that should feed data to the Redundancy Agent
- **n_pr** (*integer*) – size of the batch of data that is handled at once by the Redundancy Agent
- **problim** (*float*) – limit probability used for conistency evaluation

init_parameters2(*fsam*, *f1*, *f2*, *ampl_ratio*, *phi1*, *phi2*)

Additional parameters used for this particular example in combination with the `lcss()` method. It provides the prior knowledge needed to make the information contained in the data redundant. This method sets up the vector **a** and matrix *A* for the system **y** = **a** + *A* * **x**.

Parameters

- **fsam** (*float*) – sampling frequency
- **f1** (*float*) – first frequency of interest in signal
- **f2** (*float*) – second frequency of interest in signal
- **ampl_ratio** (*float*) – ratio of the amplitudes of the two frequency components
- **phi1** (*float*) – initial phase of first frequency component
- **phi2** (*float*) – initial phase of second frequency component

on_received_message(*message*)

Handles incoming data from 'default' channels. Stores 'default' data into an internal buffer

Parameters **message** (*dict*) – Only acceptable channel value is 'default'.

3.3 Details of the module redundancyAgents_tutorial_1

Example 1 of using a Redundancy Agent. Four signals are generated and data is supplied to the Redundancy Agent. The Redundancy Agent calculates the best consistent estimate taking into account the supplied uncertainties.

Met4FoF_redundancy_tutorials.redundancyAgents_tutorial_1.**demonstrate_redundancy_agent_four_signals**()
At the start of the main module all important parameters are defined. Then the agents are defined and the network is started. The network and the calculated results can be monitored in a browser at the address http://127.0.0.1:8050/.

3.4 Details of the module redundancyAgents_tutorial_2

Example 2 of using a Redundancy Agent. A single signal is generated and supplied to the Redundancy Agent. The Redundancy Agent uses the redundancy in the data vector together with some prior knowledge in order to calculate the best consistent estimate taking into account the supplied uncertainties.

Met4FoF_redundancy_tutorials.redundancyAgents_tutorial_2.**demonstrate_redundancy_agent_onesignal**()
At the start of the main module all important parameters are defined. Then the agents are defined and the network is started. The network and the calculated results can be monitored in a browser at the address http://127.0.0.1:8050/.

CHAPTER

FOUR

INDICES AND TABLES

- genindex
- modindex
- search

CHAPTER

FIVE

REFERENCES

BIBLIOGRAPHY

[Kok01] G. Kok and P. Harris, "Uncertainty Evaluation for Metrologically Redundant Industrial Sensor Networks," 2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT, Roma, Italy, 2020, pp. 84-88, doi: https://dx.doi.org/10.1109/MetroInd4.0IoT48571.2020.9138297.

[Kok02] G. Kok and P. Harris, "Quantifying Metrological Redundancy in an Industry 4.0 Environment," 2020 IEEE International Workshop on Metrology for Industry 4.0 & IoT, Roma, Italy, 2020, pp. 464-468, doi: https://dx.doi.org/10.1109/MetroInd4.0IoT48571.2020.9138235.

PYTHON MODULE INDEX

m

t

INDEX

S

T