Optipus is an open-source software and is fully available on GitHub.

It is written in Python 3.7.9 and uses several free available Python packages.

The main task is implementing and automating the Model-Based Experimental Design (Automated DoE). Therefore, the program needs to find suitable experiments, communicate with an existing system, scale and transform measured data, and design and validate models. Moreover, a graphical user interface (GUI), proper logging, and export/import possibilities are required.

Main program design

The main program design implements a simple state machine that executes different programs/tasks within an environment of managed error and logging handling. Therefore the main functionality of the Automated DoE can be programmed within small tasks/programs without dealing with concrete error handling.

This state machine and the main functionality are placed in its own thread, which is controlled via a local web-server. Hence it is also possible to implement the GUI as a Website and use the standard web technologies to realize the GUI.

Automated DoE

The automation of the Model-Based Experimental Design (Automated DoE) is implemented within 6 Tasks/States, which are executed in the correct order from the mentioned superior state machine.

In the first state ("InitDoE"), the program is initialized, and so everything is reset, and all variables are defined. Thereby, basic objects (like the "factorSet") are directly adopted from the interface/GUI. Moreover, a new directory in the logging-Folder is created, where all essential information of the run is stored.

The logic for finding new experiments is implemented in the following state ("FindNewExperiments"). Thereby the python package "pyDOE2" is used, which provides a variety of functions to create designs for any number of factors. The default logic uses the 2-level Full-Factorial ("ff2n") function to get a full factorial design for the number of defined factors.

Keep in mind that those experiments are not returned all at once. Instead, they are returned in sets to establish several design iterations. Additionally, after the full factorial design sets with face center points are returned. That face center points result in more design iterations with the possibility to gain better and more accurate models.

Ensuing, the state ("ExecuteExperiments") implements the functionality that the experiments are conducted. The interface and logic to an existing PLC are implemented in the class XamControl and used in this state. The PLC forwards the factor/experiments values and returns the measurement values to the program. In this state, all experiments within one design iteration are conducted after proceeding to the next state.

The conducted experiments are evaluated in the fourth state ("EvaluateExperiments"). The data gets scaled to a unit range from -1 to 1. Moreover, the data is transformed into a more standard deviated data set.

Afterward, the prepared data is extended by linear and quadratic factors. Starting with the extended data set, a model is created using (Multi-Linear-Regression) MLR. Thereby the functionalities from the python package "statsmodel" are used. (An alternative package would be "scipy".) Using the information of significance and weights of model coefficients, the implementation starts to remove non-significant terms. Thereby, squared terms are removed before linear terms, and linear terms are removed before a single factor. For each model, statistical scores like R^2 and Q^2 are stored. So it is possible that the best overall model can be filtered with respect to a definable objective/logic. The default filtering logic tries to maximize Q^2, minimize the number of model coefficients, and avoid substantial R^2 drops.

The best model can be used to predict the response for the conducted experiments and compare them against the measured ones. Thereby it is possible to detect possible outliers. Outliers could result from measurement uncertainties and are handled within the following state ("HandleOutliers").

Here experiments that show a notable difference between prediction and observation are re-conducted. Depending on the results, one can decide to adjust or remove the measurement.

The final state ("StopDoE") implements the final clean-up and is reached after all experiments are conducted. If there are still experiments, the program starts a new design iteration in the state ("FindNewExperiments"). Additional, a stop is conceivable if the best model in the design iteration shows no/less performance increase compared to older ones. In this final state, the best overall model is determined. Out of the best models from each design iteration, the program filters for the best overall model with the highest Q2 score. (Alternatively, the objective can be changed to any other score/logic.)

Optimization

Besides the automation of the Model-Based Experimental Design (Automated DoE), the factors are optimized to the maximum response using the best overall model. Here the functionalities for bounded optimization from the Python package "scipy" are used.

After optimization, an optional robustness test is possible. Thereby the initial factor bounds are adjusted around the optimum, and the Automated DoE is started again. The logging/results are stored within a subfolder in the same folder as the initial run.

GUI

A local web-server is implemented to realize a Graphical User Interface (GUI). Thus it is possible to use the standard web technologies (HTML5, CSS3, JS, ...) for the GUI itself. The GUI is reachable with any browser under the address/link: "http://localhost:8080/" and can also be accessed simultaneously. In general five important subpages are provided.

In the "defines"-section, the user ser can define the factors (names, bounds, units, symbols) and specify the folder paths for interacting with the existing PLC. Noteworthy is, that one can also define network paths, allowing that the PLC is located at a different place but within the same network. Moreover adjustments for a possible UPC/UA implementation/alternative could be adopted in this section.

In the "import"-section, one can import exported experiments or entire runs. The format can be found in section 23, allowing to import measurements from any other system. The imported measurements are used if the same experiments are requested instead of conducting the same experiment again.

The section "Automated DoE" allows controlling the Automated DoE process. One can start/stop and pause/resume the process. Moreover, detailed status information is printed in this section, and export is possible (If the process is pausing).

In the "Result"-section, one can directly/live evaluate and investigate current design models and intermediate results. Also, the investigation of previous runs is possible within this section.

In the last section ("About"), some minor information (Version) can be found.