2018 NDIA GROUND VEHICLE SYSTEMS ENGINEERING AND TECHNOLOGY SYMPOSIUM
POWER & MOBILITY (P&M) TECHNICAL SESSION
AUGUST 7-9, 2018 - NOVI, MICHIGAN

AUTOMATED OPTIMIZATION OF PHYSICS-BASED SENSOR MODELS FOR EMBEDDED CONTROL SYSTEMS

Dr. Tobias Gutjahr
ETAS Inc.
Ann Arbor, MI

Holger Kleinegraeber
ETAS GmbH
Stuttgart, Germany

Dr. Thomas Kruse
ETAS GmbH
Stuttgart, Germany

ABSTRACT

Modern electronic control units (ECUs) typically contain many physically based models represented by a complex structure of maps, curves and scalar parameters. The purpose of these models is to monitor or predict engine values that are normally measured by actual sensors. If the model structure is a good representation of the physical system and the parameters are well fitted, such a model can replace the sensor and serve as a virtual sensor to reduce the cost and complexity of the overall system. Virtual sensors are commonly used in the ECU for predicting engine torque, air pressure and flow, emissions, catalyst temperature, and exhaust gas temperatures. To ensure an optimal prediction quality of these models, their parameters need to be calibrated using real measurement data collected, e.g., in the vehicle or in the test cell. Due to the models' complexity and the high number of parameters, a manual calibration is very time consuming or even impossible. Instead, iterative multivariate optimization algorithms are more efficient. The optimization of the model parameters can be performed offline on a PC and doesn't require access to the physical target which helps to save time and costs throughout the entire calibration process.

This paper presents the implementation of an automated calibration procedure in a generic tool framework, ETAS ASCMO-MOCA, to enable a broad use in function calibration and virtual sensor development. The ECU model can be provided in the proprietary formats of MATLAB Simulink® and ETAS ASCET. A formula editor is also available to recreate the function in ASCMO-MOCA, in case the source model is not available. Recorded measurement data stimulates the model inputs for the optimization. The difference between the actual sensor values in the data set and the model output represents the optimizer's cost function. A gradient descent algorithm is used to find optimal calibration parameters while minimizing the deviation of the model prediction from the desired output behavior. For

practical reasons, additional constraints can be considered during the optimization for calibration curves and maps, such as smoothness factors or gradient limitations. Once the final optimized and verified model is available, it can be integrated into the ECU control strategy. Several examples from ECU function development are shown throughout the paper.

MOTIVATION

In the automotive industry, physics-based models are widely used as virtual sensors in electronic control units (ECUs) and as plant models in Model-in-the-Loop (MiL), Software-in-the-Loop (SiL) and Hardware-in-the-Loop (HiL) simulations. These models consist of many multidimensional calibration parameters combined in a complex but physically motivated structure. The process of properly calibrating the parameters has a strong impact on project targets like fuel consumption, emissions, drivability, and development costs. A major challenge is the constantly increasing complexity of these models, i.e. number of calibration parameters.

For example, Figure 1 shows a simplified implementation of an engine torque model in MATLAB/Simulink®. The model output is calculated based on the current values of engine speed, relative air mass, ignition timing, and the combination of four calibration labels. (Please refer to the application section in this paper regarding more detailed information about the purpose and meaning of the calibration parameters.) This relatively simple model contains a couple hundred individual calibration values considering multiple breakpoints within each calibration map and curve. A more realistic torque model likely has more than ten such complex calibration labels, which further increases the challenge of the overall calibration task. Therefore, a manual calibration is very time consuming and sometimes even impossible. To be more efficient, it makes sense to use automated optimization algorithms, where the optimization can be performed offline on a PC.

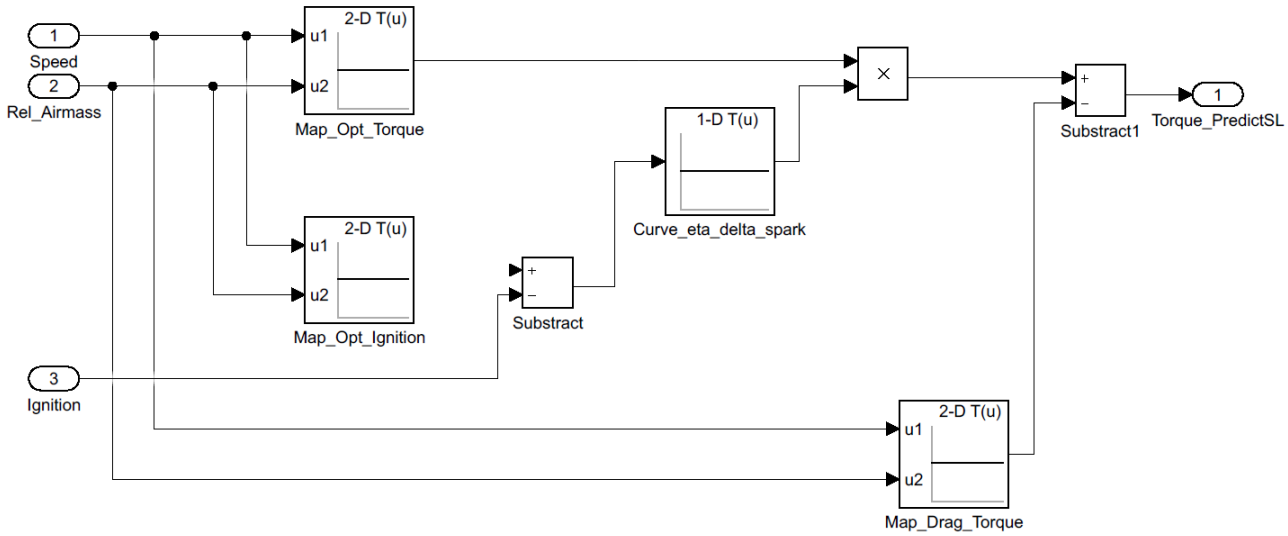

Figure 1: Simplified engine torque model. Torque is calculated based on the current values of engine speed, relative air mass, ignition timing, and the combination of four complex calibration labels.

Today, original equipment manufacturers (OEMs) and ECU suppliers often use specific in-house scripts or tools for the task of optimizing individual models. A generic tool framework for an automated model calibration is therefore presented in this paper. The user provides the function model and a set of recorded measurement data files with the desired output behavior. The tool then maps the recorded channels to the corresponding model inputs to perform simulation runs offline. An optimization algorithm iteratively adapts the calibration parameters and minimizes the deviation between recorded output and model prediction.

The benefit of such a virtual calibration approach is that access to the physical target, i.e. the vehicle or the ECU, is not required. This helps perform calibration tasks earlier in the ECU development cycle to significantly reduce the demand for real prototypes and to save development costs.

The remainder of this paper is organized as follows. In the next section, the core functionality of the automated calibration framework is described in detail, in particular, data import, supported model types and their optimization. The section thereafter discusses the application of the presented concept using a steady-state, transient, and closed-loop calibration example. The paper concludes with a brief summary.

AUTOMATED MODEL CALIBRATION

The tool suite ETAS ASCMO (Advanced Simulation for Calibration, Modeling and Optimization) provides model-based approaches for a broad range of calibration tasks to help with the increasing complexity of today's internal combustion engines [1, 2]. In addition to design of experiment (DoE) and data-driven modeling, ASCMO comes with powerful optimization algorithms, an interactive visualization, and prognosis features tailored to different calibration needs. The latest product extension, ASCMO-MOCA (Model Calibration), was designed for ECU function development and, in particular, the automatic calibration of physics-based ECU models or virtual sensors. Figure 2 shows the overall concept of ASCMO-MOCA consisting of the main modules: data import, model simulation, and optimization. An intuitive graphical user interface enables the user to perform the parameter optimization in a few easy steps.

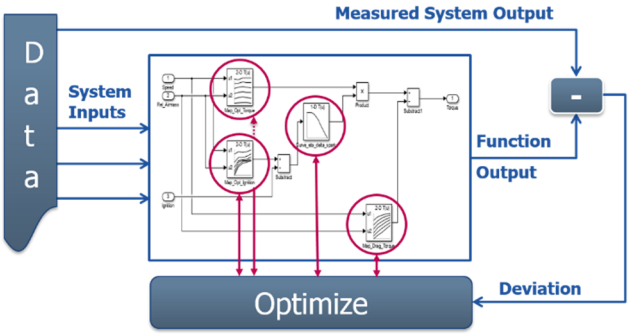

Figure 2: Overview of ASCMO-MOCA. Optimization of calibration parameters based on the model simulation against a recorded data set of desired output values.

Measurement Data

As a prerequisite for performing offline simulation and optimization, a recorded set of measurement data with the desired output behavior is needed from the real system. Depending on the particular application, this can be steady-state data from the engine test cell or transient data (with timestamp information) from vehicle test trips. In any case, the recorded data should cover all operating areas of the model, i.e. all input value combinations. Otherwise, the optimization result will only be valid for parts of the model's operating range. All common automotive measurement data file formats are supported: Measurement Data Format (MDF3 and MDF4), Excel, and comma separated values (CSV). The recorded channels from the measurement data are later mapped to the corresponding model

inputs for simulation. Thus, a signal for each model input needs to be present in the loaded data set. It is also possible to import multiple data sets from various test runs. The data can be separated into training data for optimizing the model parameters and test data for validating the model quality.

Model Formats

Besides measurement data, the ECU function model has to be provided. Currently, the proprietary formats of MATLAB/Simulink® and ETAS ASCET are supported. A formula editor is also available to recreate the function, in case the source model is not available. For example, Figure 3 shows the simplified engine torque model manually entered in the ASCMO-MOCA formula editor. It is worth noting that the formula representation provides the benefit of calculating the analytical gradients of the optimizer's cost function in each iteration step, rather than numerically as with the proprietary modeling formats. This generally leads to better performance, i.e. a reduced optimization time.

1	trqOpt[-] = MapOptTorque(Speed,Rel_Airmass)
2	ignOpt[-] = MapOptIgnition(Speed,Rel_Airmass)
3	deltaSpark[-] = ignOpt - Ignition
4	etaSpark[-] = CurveEtaDeltaSpark(deltaSpark)
5	product[-] = trqOpt * etaSpark
6	dragTorque[-] = MapDragTorque(Speed,Rel_Airmass)
7	TorquePredict[-] = product - dragTorque

Figure 3: Formula editor with simplified engine torque model.

Optimization

The sum of the squared residuals between the true sensor values y_k and the model output $\hat{y}_k(p)$ over the entire prediction horizon is used for optimization:

$$f(p) = \sum_{k=1}^{K} (\hat{y}_k(p) - y_k)^2 \qquad (1)$$

K represents the number of all sample points in the training data set, and p are the model's calibration parameters. In addition to the model error, a smoothness factor for the calibration curves and maps is considered in the optimization criterion. The goal is to avoid overfitting and to provide good extrapolation capabilities. A penalty term is therefore used to describe the roughness R of the calibration surface by the 2nd order derivatives of the map output z with respect to its grid point locations $\boldsymbol{x_n}$ and $\boldsymbol{y_m}$.

$$R = \frac{1}{N}\sum_{n=1}^{N-2}\frac{\partial^2 z}{\partial x_{x_n}^2} + \frac{1}{M}\sum_{m=1}^{M-2}\frac{\partial^2 z}{\partial y_{y_m}^2} \qquad (2)$$

N and M are the total number of grid points in direction of operating point axes 1 and 2, respectively. (Equation (2) is valid for a two-dimensional calibration map. In case of a calibration curve with only one operating axis, the second summand of equation (2) related to y_m can be neglected.) Based on equations (1) and (2), the overall cost function of the optimization task can be formulated as follows:

$$\min_p \left(\sum_{k=1}^{K} (\hat{y}_k(p) - y_k)^2 + \sum_{l=1}^{L} S_l R_l \right) \qquad (3)$$

$\boldsymbol{S_l}$ is an individual smoothing factor applied by the user for each of the $\boldsymbol{L}$ calibration maps and curves.

A gradient descent search algorithm [3] is used to solve the optimization problem, i.e. minimizing the model error while trying to find optimal values for the calibration labels p. The tool also supports a multi start of the optimization. This can help to avoid local minima in the cost function. Initial calibration values can be provided by the user in standard automotive file formats such as DCM [4] and CSV. The resulting calibrations after the

optimization are exportable into the same formats to use in other measurement and calibration tools, e.g. ETAS INCA.

APPLICATIONS

This section demonstrates the application of the proposed optimization procedure with three virtual sensor models. Different use cases are addressed including: steady-state input-output relationship (engine torque model), transient behavior (exhaust gas temperature model), and closed-loop system simulation (engine idle speed governor). The Root Mean Square Error (RMSE) is used in all examples to compare the model quality before and after the optimization:

$$RMSE = \sqrt{\frac{\sum_{k=1}^{K}(\hat{y}_k - y_k)^2}{K}} \qquad (4)$$

(In this equation, K is equal to the number of all sample points in the training data set. $\hat{y}_k$ is the model prediction and y_k represents the desired output behavior.)

Engine Torque Model

The engine torque model introduced in the beginning of the paper (Figure 1) is examined in this example. As mentioned, the model calculates torque based on the input values of engine speed, air mass and ignition timing. The model consists of three calibration maps and one calibration table. The purpose of the individual calibration labels is summarized in the following. Be aware that this is a simplified version of an engine torque model. A real-life model likely has more than ten calibration labels to consider additional influences such as a lambda value, variable valve timing, swirl flaps, etc.

- Map_Opt_Torque: Maximum inner torque from the combustion at the optimal ignition angle without any friction or pumping losses.
- Map_Opt_Ignition: Optimal ignition angle (from optimal combustion) depending on engine speed and air mass.
- Curve_eta_delta_spark: Reduction of inner torque in case of a deviation from the optimal ignition angle. The input to this curve is the deviation between the current ignition angle and the optimal ignition angle (the output of Map_Opt_Ignition).
- Map_Drag_Torque: To consider any friction and pumping losses, e.g. from unfired engine operation, depending on engine speed and air mass.

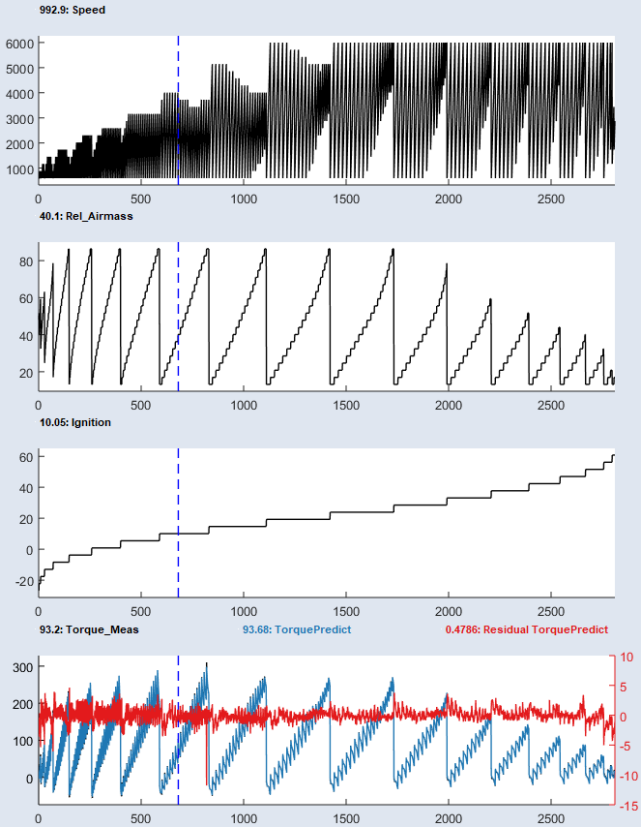

Figure 4: Time series plot in ASCMO-MOCA of the data used for optimizing the engine torque model: 1st graph – engine speed input, 2nd graph – relative air mass input, 3rd graph – ignition timing input, and 4th graph – desired engine torque (black) and optimized model output (blue). The red curve in graph 4 represents the remaining deviation between modeled and desired torque output after the optimization.

Figures 4 and 5 show the data set used in ASCMO-MOCA for optimizing the model parameters in a time series and scatter plot, respectively. The stimuli for the system inputs were based on a full factorial design plan (Figure 5) adhering to the physical constraints of the system. A total of 2812 steady-state sample points were collected in a test cell on an engine dynamometer.

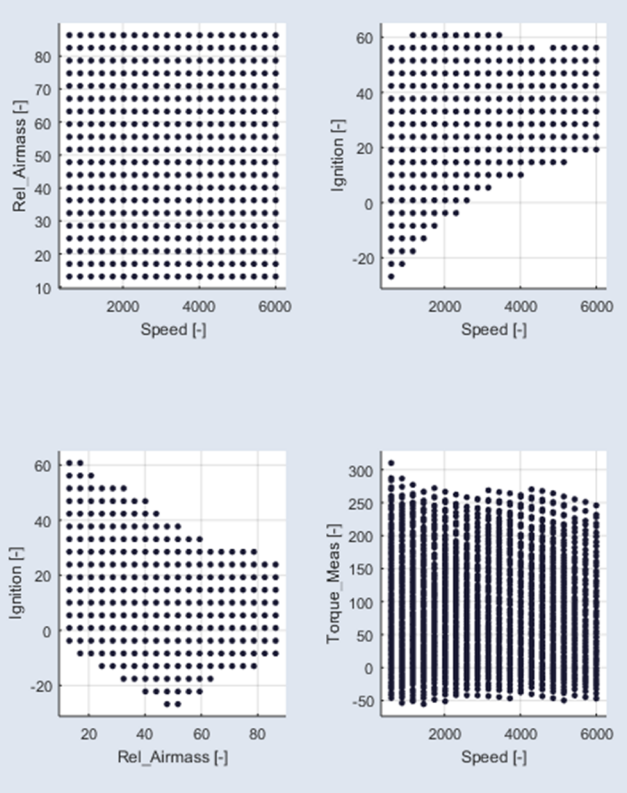

Figure 5: Scatter plot with the full factorial input design of the engine torque model data set.

The optimization results were as follows. The optimizer converged after 22 iterations with an RMSE of 1.83 Nm compared to an RMSE of 76.38 Nm before the optimization; see the histograms of the absolute model errors in Figure 6. Please note the different scales of the x-axes in Figure 6. The distribution of the post optimization histogram is narrower than the results before the optimization and aligns with the red signal (residual model deviation) in the bottom graph in Figure 4. The required computational time for the optimization strongly depends on the model complexity, i.e. the number of calibration labels, breakpoints within each label, and any additional constraints. The optimization for the 22 iterations in this example took about 1.5 minutes on a state-of-the-art engineering laptop. As mentioned above, there is a potential computation benefit if the model equation is entered as a formula expression due to the ability to calculate the cost function gradients analytically. For the same engine torque model provided as analytical formula (Figure 3), identical optimization results were achieved with a computation time of less than 2 seconds.

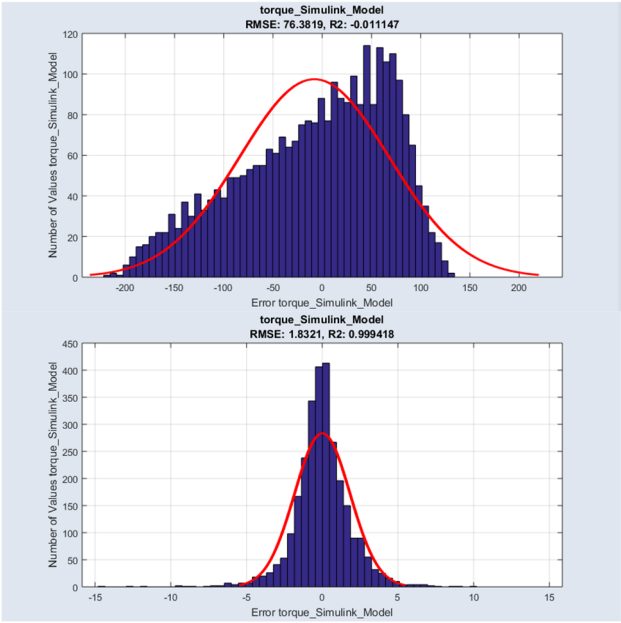

Figure 6: Absolute model error of the engine torque model before (top) and after (bottom) the optimization.

To visualize the changes in the model's calibration, Figure 7 compares the drag torque map before the optimization (grey) and after (colored). For the sake of simplicity, the other calibration labels are not shown. It is worth noting that the user can modify the calibration map manually via dragging the green dots at the

breakpoint locations and ASCMO-MOCA will immediately provide the updated model error caused by the changes.

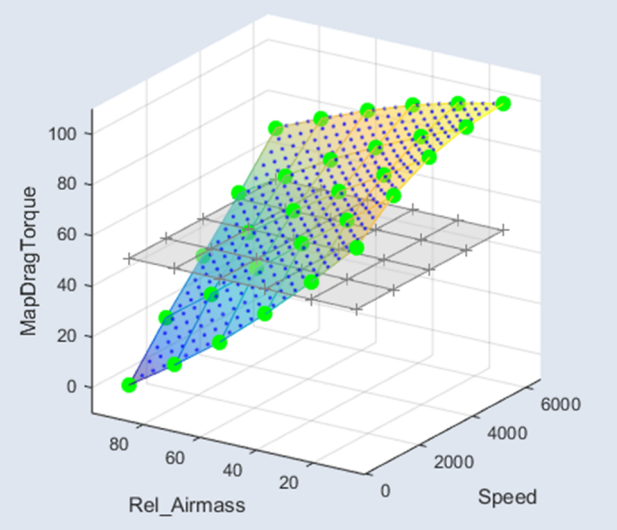

Figure 7: Drag torque map before (grey) and after the optimization (colored) in ASCMO-MOCA.

Exhaust Gas Temperature Model

This example shows the application of the optimization routine to a transient model, in particular, a simplified exhaust gas temperature model. The model calculates the temperature based on engine speed, injection quantity and air mass. The transient behavior of the model is introduced by a first-order delay element (PT1), as shown in Figure 8. The two calibration maps in this example are:

- Map_Exh_Temp_Steady: The steady-state temperature depending on engine speed and injection quantity.
- Curve_Time_Const: Time constant for changing the behavior of the PT1 transfer function depending on air mass.

Figure 9 summarizes the time series of the stimuli and the optimized model behavior. The data was recorded during an in-vehicle measurement campaign and consisted of 26482

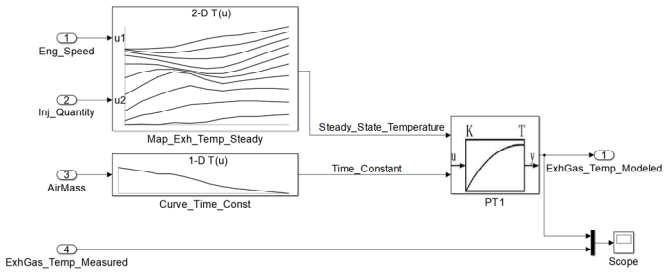

Figure 8: Simplified exhaust gas temperature model with two calibration labels and first-order transfer function.

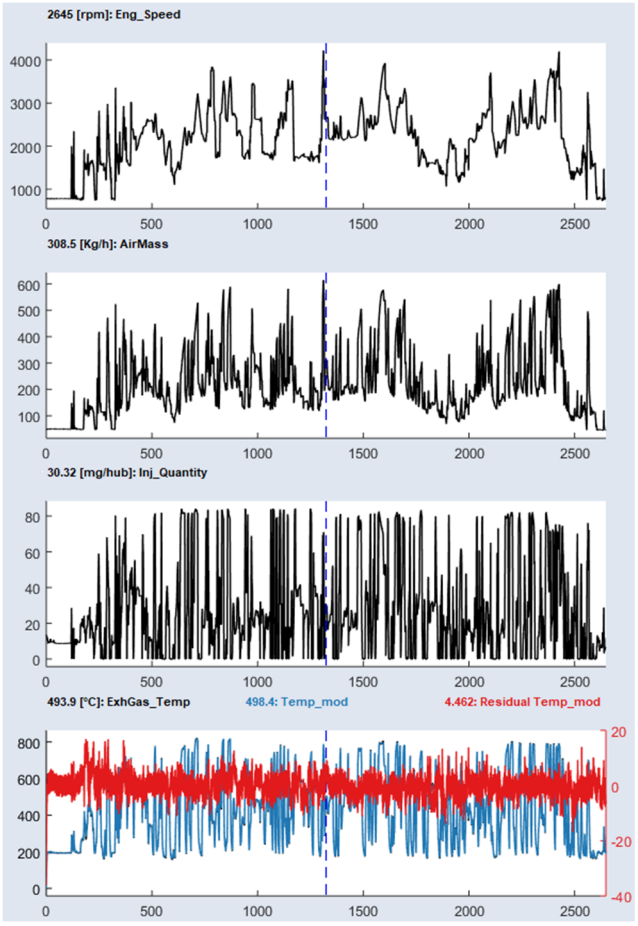

Figure 9: Time series plot with input stimuli (graphs 1 – 3) and output behavior (graph 4) of the exhaust gas temperature model. Graph 4 shows the desired temperature values (black), the optimized model output (blue), and the remaining model error (red).

samples captured at a sample rate of 10 Hz. The optimization was able to improve the RMSE from 197.57 °C down to 3.76 °C. The results for the two calibration labels (before and after the optimization) can be seen in Figures 10 and 11.

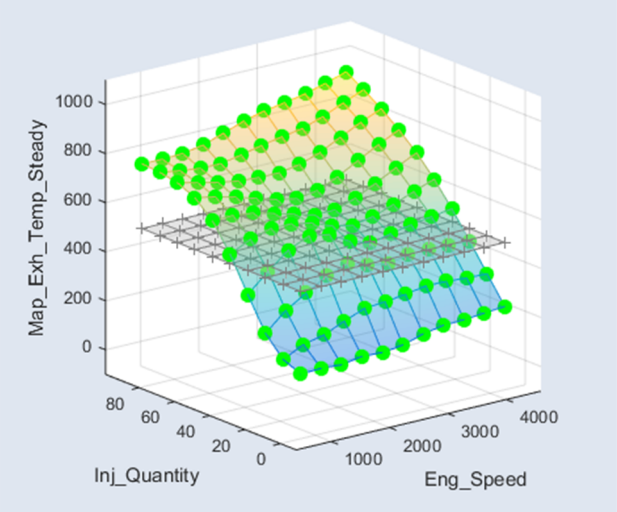

Figure 10: Steady-state exhaust temperature map before (grey) and after the optimization (colored).

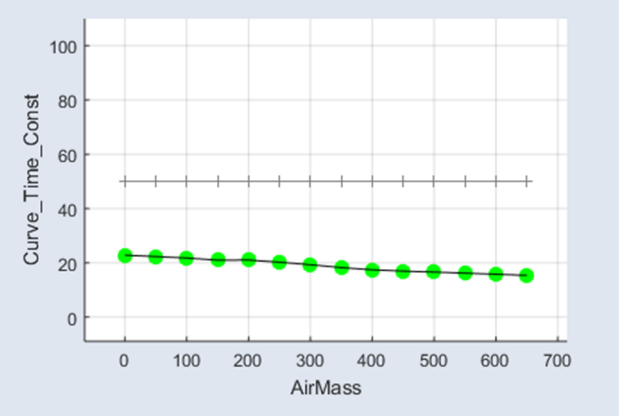

Figure 11: Time constant curve for the PT1 transfer function before (grey) and after the optimization (colored).

Engine Idle Speed Governor

The previous applications demonstrated the use case of optimizing the open-loop behavior of a function model. In contrast to this, the following example will focus on the optimization of a control algorithm in a closed-loop configuration connected to a plant model.

Figure 12 shows a simplified engine idle speed governor, where a proportional-integral (PI) controller is tasked to command a torque value to keep the engine speed at the desired rpm (revolutions per minute) level. The plant model of engine speed in this control loop is a transient data-driven model generated with ASCMO-DYNAMIC [5] and based on a representative set of in-vehicle measurements.

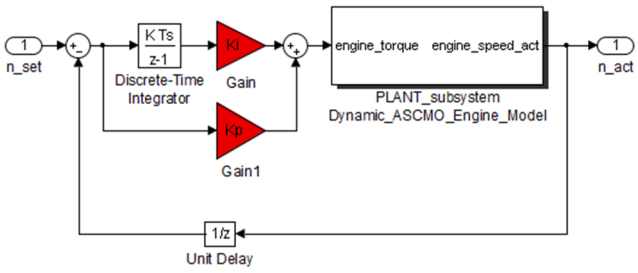

Figure 12: Engine idle speed governor control loop consisting of a discrete PI controller and transient engine speed model generated with ASCMO-DYNAMIC [5].

Compared to the previous examples, the idle controller optimization had far less computational requirements, since only two scalar labels (Kp and Ki) needed to be optimized. Figure 13 shows the ability of the calibrated controller to keep the engine speed at the desired value for the given profile with an RMSE of 25.02 rpm (Kp = 0.08 and Ki = 0.046). The RMSE before the optimization was 1115 rpm (Kp = 0 and Ki = 0). It can also be seen in Figure 13 that the calibrated controller causes overshoots in the closed-loop behavior. The user has the option to define additional constraints in the tool, e.g. for a maximum allowable deviation between set value and actual value. Depending on the threshold, this will result in a slower but smoother response of the controller.

Having the entire control loop available in such an offline calibration can help reduce the demand for hardware prototypes. New or modified control strategies can be calibrated and tested without actual measurement runs in the test cell or the vehicle.

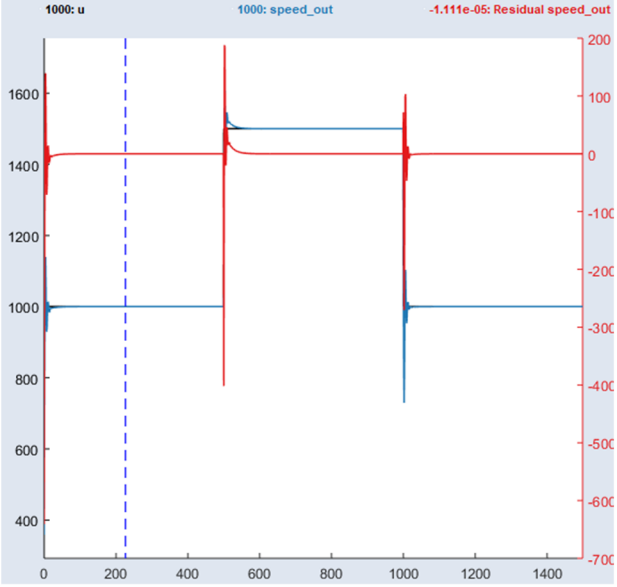

Figure 13: Response of engine rpm of the closed-loop control strategy to a given set value change: desired value (black), engine model response (blue), and deviation (red).

CONCLUSION

ASCMO-MOCA is a generic solution for an efficient optimization of calibration labels in virtual sensor models. A constrained non-linear optimization is used to match the model output to the desired sensor behavior and to find an optimal set of calibration values. The tool comes with an intuitive graphical user interface to load and analyze measurement data, import function models from different sources, define optimization tasks, and visualize and validate the optimization results. In practice, this can help reduce the required calibration time and the demand for hardware prototypes. The successful application of ASCMO-MOCA was shown in this paper on two ECU functions with steady-state and transient behavior. An optimization of a closed-loop simulation consisting of a control algorithm and plant model was also demonstrated. With the integration in an easy to use tool environment, model calibration is no longer restricted to modeling experts and can be made available to a wider audience of calibration engineers.

REFERENCES

[1]T. Kruse, S. Kurz, T. Lang, “Modern Statistical Modelling and Evolutionary Optimisation Methods for the Broad Use in ECU Calibration”, IFAC-AAC, Munich, Germany, 2010.

[2]T. Gutjahr, T. Kruse, T. Huber, “Advanced Modeling and Optimization for Virtual Calibration of Internal Combustion Engines“, NDIA Ground Vehicle Systems Engineering and Technology Symposium, Novi, MI, 2017.

[3] R. Fletcher, “Practical Methods of Optimi-zation“, 2nd Edition, Wiley, 2000.

[4]“DCM File Formats – Technical Note”, ETAS GmbH, 2012.

https://www.etas.com/download-center-files/products_ASCET_Software_Products/TechNote_DCM_File_Formats.pdf

[5]T. Kruse, T. Huber, H. Ulmer, T. Gutjahr, “New Approaches for the Data Driven Modelling of Dynamic Engine Behaviour”, 2nd Biennial Conference on Powertrain Mapping and Calibration, Bradford, UK, 2014.