Modellbasierte Online-Optimierung moderner Verbrennungsmotoren Teil 2: Grenzen des fahrbaren Suchraums

Model-Based Online Optimisation

of Modern Internal Combustion Engines
Part 2: Limits of the Feasible Search Space

The first part of this two-part article introduced the basis of the mbminimize algorithm, which was developed by the University of Tübingen in cooperation with the BMW Group for the model-based online optimisation of internal combustion engines. This second part describes its extensions regarding the handling of engine limits such as knocking. The main focus is on the construction of special mathematical models for these limits that make it possible to successively and controllably restrict the search space.

By Kosmas Knödler, Jan Poland, Thomas Fleischhauer, Alexander Mitterer, Stephan Ullmann and Andreas Zell

1 Introduction

In recent years, the number of adjustable engine parameters has been drastically increased in order to meet both the legal requirements and the customers' demands. For that reason, offline optimisation was established as a powerful method for the calibration of the engine control units. It performs a separation between test bed measurements and computer-aided data evaluation (see [1], [2] and [3] for this topic).

Nowadays, car manufacturers strive to achieve fully automated engine calibration directly at the test bed. Already existing online optimisation tools on the automotive market are CAMEO [5] and VEGA [6]. In the first part of this article [4], the mbminimize algorithm was introduced as an online optimisation algorithm. Its underlying idea is the use of information obtained by measurements for the further process. This procedure is known in the literature as Active Learning or Query. The information gain that can be obtained by the subsequent measurement is quantified by means of the actual objective function models. The points in the search space that maximize this so-called query criterion are measured next at the test bed.

In offline optimisation systems, engine limits are usually treated by models and the search space that is restricted as a result. Manual verification measurements at test beds guarantee that optimum candidates that violate certain limits are discovered, thus avoiding engine damage. The situation is quite different when the direct optimisation of the engine at the test bed in an online process is considered. Known procedures for the restriction of the search space appear to be both too restrictive and too little merged with the online optimisation algorithm.

This article describes the process of limit handling within the mbminimize algorithm. It consists of a controlled reaction to limit violations and robust limit modelling.

Figure 1 shows an example of a geometrically constrained search space. Using the measured violation points only, the pure star-shaped hull model is expanded by a controllable restrictivity. This increases the probability of finding optima close to the search space borders. It is known from experience that very good values for fuel consumption and exhaust emission are often obtained in such regions.

2 Classification of Limit Violations

The most important requirement for limit handling within an online optimisation process is the controllable reaction to detected limit violations. The engine must be driven back into secure regions of the search space by a robust adjustment strategy as fast as possible. In order to avoid further limit violations, it is necessary to identify the feasible search space as accurately as possible. As a result, the optimisation of the model for the objective function and the process of active learning become constraint optimisation problems. A continuous real valued minimization problem in *d* dimensions with a limited search space is defined by an objective function *f* and a search space $S \subset \Re^d$, which is limited by linear and non-linear functions, Eq. (1)

$$\begin{aligned} &\min_{x} f(\underline{x}), \quad f: S \rightarrow \Re, \quad \underline{x} \text{ a } f(\underline{x}) \\ &\underline{x}_{min} \leq \underline{x} \leq \underline{x}_{max} \\ &A \cdot \underline{x} \leq \underline{b}, \\ &c^{i}(\underline{x}) \leq 0, \quad i = 1, \ldots, n \end{aligned} \qquad \text{Eq. (1)}$$

The search space *S* is limited to a hypercube by the vectors $\underline{x}_{min}$ and $\underline{x}_{max}$ that constituted the physical ranges of the adjustable parameters. The matrix *A* together with the vector *b* define static linear limits, while the functions *c(x)* define non-linear limits. Examples of the objective function *f* are the fuel consumption that is to be minimized or the engine torque that is to be maximized. Linear limits represent functional dependencies of certain engine parameters. Well-known non-linear engine limits are engine knocking, the smooth running of the engine or the temperature of the exhaust gas. It is of special importance for the online optimisation of internal combustion engines that the shape of the functions *c(x)* is usually unknown before the process starts. In some cases, test bed engineers may have experience that might be transferred from earlier engine generations. Thus, these limits represent the greatest challenge to an online optimisation system.

The engine limits can be classified into three main classes: the static and the dynamic non-linear limits, whereby the latter are further split into hard and soft limits. The name dynamic suggests that the information on these limits must be dynamically increased during the process. The models used for these limits need to be refined continuously for the restriction of the search space. Hard and soft refer to the high or relatively low risk of possible damage to the engine.

Class 1: Static (Non-)Linear Engine Limits
Before the online optimisation process starts, there already exist static linear and non-linear limits of the search space that will not change during the process and which are well known. These limits can be integrated into the optimisation process without modelling.

Class 2: Hard Dynamic Engine Limits
Members of Class 2 limits restrict individual dimensions of the search space. The standard example is engine knocking, which is usually caused by too advanced ignition, thus a too large value for the ignition angle. The critical non-linear threshold can be set as a function of the remaining *d*-1 parameters, Eq. (2).

$$L : [0,1]^{d-1} \rightarrow [0,1] \qquad \text{Eq. (2)}$$

The original hypercube is limited by this function in either the positive or the negative direction in the d^{th} dimension (here, this is the dimension of the ignition angle). The function *c(x)* defined in Eq. (1) is given by Eq. (3).

$$c(\underline{x}) := x_d - L\left(x_1, \ldots, x_{d-1}\right) \qquad \text{Eq. (3)}$$

Consider the positive direction. If $c(\underline{x}^i)$ is greater than zero, then a limit violation is given at $\underline{x}^i$. Because of the small number of measured violation points, models for this class of limits run the risk of defining parts of the search space as non-feasible in an uncontrolled way (see Figure 1, left). In section 3.2, confidence models and hull models that show controllable restrictivity are described.

Class 3: Soft Dynamic Engine Limits
Soft dynamic engine limits are search space limits that are defined by thresholds for secondary objective functions, for example for the exhaust emissions. As in the primary objective function *f*, these functions usually depend on all engine parameters. Class 3 limits are described by a critical threshold L_0 and a function *L(x)* of all parameters *x*. In Eq. (4), there is a limit violation at $\underline{x}^i$.

$$c\left(\underline{x}^{i}\right) := L_0 - L\left(\underline{x}^{i}\right) > 0 \qquad \text{Eq. (4)}$$

The value of the function *L* is measured at each measuring point at the test bed. As a result, a relatively large amount of data is available to build a model. Further examples of Class 3 limits are the smooth running of the engine based on the variance of the indicated average pressure within the cylinder, and the exhaust gas temperature. As in the primary objective function, suitable models for this limit class are regression models. These are also discussed in section 3.2.

3 Influence of Limit Handling on the Online Optimisation

Functions of the mbminimize algorithm that have to be added and/or adapted are the limit handling itself and the query and optimisation functions. In the last two functions, the information about the feasible search space determined in the limit handling is used to select feasible measuring points for the limit models.

The mbminimize algorithm offers the possibility to work with a robust star-shaped straight adjustment strategy. Optionally, this strategy can be expanded by an optimisation of the adjusting path (see section 4) in order to move around already known limits. Only in the case of a new limit violation will the secure central point be used to stabilize the process. From the initial processing of the first design of experiments, the limit handling reacts to detected limit violations and computes models to define the non-feasible search space.

3.1 Limit Handling

The limit handling contains the limit monitoring module, in which signals from the test bed concerning limit violations are evaluated. An intermediate point that has been previously passed is supplied as a secure point and given back to the test bed automation system after a possible violation. The step back strategy is repeated if necessary in order to handle hysteresis effects.

The main goal is to identify the feasible search space that is as accurate as possible. A detected limit is therefore measured in two steps. First, a further approximation with a reduced step size takes place. If a limit violation occurs for the second time, the current point is defined as the boundary between the feasible and non-feasible region. This point is subsequently used to build a limit model. In the second step, the path planning, a 3-back-2-forward-strategy also with a small step size, is performed. Thus, a point that is sufficiently close to both the limit and the desired measuring point is measured.

3.2 Models for the Search Space Limits

As described in [4], the primary objective function is generally modelled by a regression model. The main focus lies on the global model quality and not on the accurate description of single measured function values. This also applies to soft engine limits of Class 3. Because of the much smaller number of measured points for Class 2 limits, the situation there is totally different. The small number of violation points must be taken into account by the model in order to avoid further limit violations within these regions. Thus, the models for the two classes must fulfil significantly different requirements. For Class 2 limits, confidence models such as RBF-max networks and confidence networks are considered. In addition, different hull models are suitable. Regression models such as neural networks [7] are used for modelling Class 3 limits.

3.2.1 Models for Class 2 Limits

Star-shaped Hulls:
Figure 1 shows the pure star-shaped hull model that precisely reproduces limit violations. However, large regions between the points of violation are rejected without control. This problem can be eliminated by adding a grid of points on the edge of the hypercube.

Confidence Models:
A limit to the search space such as engine knocking can be modelled using Eq. (3) and setting a function for the critical ignition angle depending on the remaining d-1 parameters. Confidence models are characterized by the fact that they are quite restrictive in regions that contain violation points, whereas they do not make a statement within ranges without data.

The name of the RBF-max network already indicates that it is based on an RBF network [7], and the activation function of the output neuron computes the maximum of its weighted inputs. On the left, **Figure 2** shows an example of an RBF-max network and on the right it shows the output of two RBF-max networks for limit violations $\underline{x}^i$.

Confidence networks can be formulated as a product of a regression model $h(x)$ for the search space limit and a confidence term $conf(x)$, Eq. (5).

$$g(\underline{x}) = conf\,(\underline{x}) \cdot h\,(\underline{x}) \qquad \text{Eq. (5)}$$

The confidence term has the property that its output values are close to one for points close to violation points $\underline{x}^i$ and close to zero for distant points. The left part of **Figure 3** shows the regression models, the middle part shows the confidence terms and the right part shows the resulting output of the confidence networks for two limits.

Both the RBF-max networks and the confidence networks restrict the search space in a more controllable way than the pure star-shaped hull shown in the left part of Figure 1. The restrictivity can be adapted in each case by parameters. However, RBF-max networks and confidence networks can define regions of the search space that cannot be reached on a straight line. With the aid of the optimised path planning described above, such points can be measured.

Hull Models:
Hull models are also suitable for the representation of hard limits. They work best in connection with a star-shaped straight-line adjustment strategy. Unlike confidence models, hull models are able to consider different Class 2 limits at the same time. Furthermore, they avoid the definition of regions that cannot be reached on straight lines. Hull models, such as the pure star-shaped hull in the left part of Figure 1, or convex hulls usually cut the search space in an uncontrolled way. A controllable restriction is also desirable here. **Figure 4**, left, shows a ”potato“ model that has been modified again by confidence terms in order to obtain the feasible search space on the right. Parameters of the confidence term allow the controllability of the restriction.

An edge and corner cutting procedure is shown in **Figure 5**. First of all, the facets are computed to cut the corners and edges of the hypercube. This leads to facets perpendicular to the connecting distance between the central point and the violation point. The dark grey areas in the left part show the non-feasible areas thus obtained. In order to make the procedure less restrictive, only those parts of the hyperplanes that lie close to a violation point are considered. This modification is shown in the right part of the figure.

3.2.2 Models for Class 3 Limits

In this section, models for soft engine limits such as the smooth running of the engine and the exhaust gas temperature are described. The true function is approximated by a regression model depending on all engine parameters. Polynomial models and neural networks ([7]) are used in particular. **Figure 6** shows an example of a model for a feed-forward network *net* with one hidden layer of neurons. The soft limit $L(x)$ in Eq. (4) is modelled by a neural network. If c(xi)>0 in Eq. (4) there is a limit violation in positive direction.

Figure 7, left, shows the shape of the model for a possible Class 3 limit and the corresponding threshold L_0, in which the function is approximated by a neural network that was trained with the measuring points marked in black. Regions of the search space in which the function runs above the threshold define the non-feasible range (black points in the right part of Figure 7).

The measured signals for soft limits usually change very slowly, which makes the corresponding limit handling more complicated. In addition, models for Class 3 limits are able to define search spaces with non-feasible islands in them, Figure 7, right. The use of optimised path planning as described above also helps in this situation.

4 Optimised Path Planning

As mentioned above and as shown in Figure 7, limit models can form feasible regions of the search space that cannot be reached on a straight line. In those situations, it might be desirable to modify the pure straight-line path planning in order to reach such regions. With the aid of classic optimisation algorithms for general problems of non-linear programming (SQP procedure), curved paths through the search space are computed. The current state of the limit modelling is used for this purpose. **Figure 8** shows two possible optimised paths that are finding a way through a limit landscape.

5 Summary

This article describes the process of limit handling integrated into the mbminimize algorithm. It consists in particular of a controlled reaction to limit violations and a robust limit modelling. The algorithm was examined with the aid of numerous test scenarios. Both the adjustment strategies – especially the optimisation of the path planning – and the limit models described here proved to be very efficient for the automated test bed application. Beyond that, the first attempts at implementing the real system were accomplished, and these are very promising.

This research was supported by the BMBF (grant no. 01 IB 805 A/1).

References

[1] Mitterer, A.; Zuber-Goos, F.: Modellgestützte Kennfeld-Optimierung – ein neuer Ansatz zur Steigerung der Effizienz in der Steuergeräte-applikation. In: ATZ 102 (2000), Nr. 3

[2] Schüler, M.; Hafner, M.; Isermann, R.: Einsatz schneller neuronaler Netze zur modellbasierten Optimierung von Verbrennungsmotoren, Teil 1: Modellbildung des Motor- und Abgasverhaltens. In: MTZ 61 (2000), Nr. 10

[3] M. Schüler, M.; Hafner, M.; Isermann, R.: Einsatz schneller neuronaler Netze zur modellbasierten Optimierung von Verbrennungsmotoren, Teil 2: Stationäre und dynamische Optimierung von Verbrauch und Emissionen. In: MTZ 61 (2000), Nr. 11

[4] Poland, J.; Knödler, K.; Zell, A.; Fleischhauer, Th.; Mitterer, A.; Ullmann, S: Modellbasierte Online-Optimierung moderner Verbrennungsmotoren, Teil 1: Aktives Lernen, In: MTZ 105 (2003), Nr. 5

[5] Gschweitl, K.; Pfluegl, H.; Fortuna, T.; Leithgoeb, R.: Steigerung der Effizienz in der modellbasierten Motorenapplikation durch die neue Cameo-Online-DoE-Toolbox. In: ATZ 103 (2001), Nr. 7/8

[6] Bredenbeck, J.: Statistische Versuchsplanung für die Online-Optimierung von Verbrennungs-motoren. In: MTZ 60 (1999), Nr. 11

[7] Zell, A.: Simulation Neuronaler Netze, Addison-Wesley, Bonn, 1994