Safe Active Learning of a High Pressure Fuel Supply System

Mark Schillinger[1] Benedikt Ortelt[1] Benjamin Hartmann[1]
Jens Schreiter[2] Mona Meister[2] Duy Nguyen-Tuong[2]
Oliver Nelles[3]

[1]Bosch Engineering GmbH, Germany, mark.schillinger@de.bosch.com
[2]Robert Bosch GmbH, Germany
[3]Automatic Control, Mechatronics, Department of Mechanical Engineering, University of Siegen, Germany

Abstract

When modeling technical systems as black-box models, it is crucial to obtain as much and as informative measurement data as possible in the shortest time while employing safety constraints. Methods for an optimized online generation of measurement data are discussed in the field of Active Learning. Safe Active Learning combines the optimization of the query strategy regarding model quality with an exploration scheme in order to maintain user-defined safety constraints. In this paper, the authors apply an approach for Safe Active Learning based on Gaussian process models (GP models) to the high pressure fuel supply system of a gasoline engine. For this purpose, several enhancements of the algorithm are necessary. An online optimization of the GP models' hyperparameters is implemented, where special measures are taken to avoid a safety-relevant overestimation. A proper risk function is chosen and the trajectory to the sample points is taken into account regarding the estimation of the samples feasibility. The algorithm is evaluated in simulation and at a test vehicle.

Keywords: machine learning, system identification, active learning, Gaussian process models, automotive applications

1 Introduction

For calibration purposes, models are used in order to speed up the calibration process, reduce the risk of damages of the system and reduce the time the real system needs to be available. For example, these models can be used in Hardware-in-the-Loop or Software-in-the-Loop environments, or even for automatic model based controller tuning. These models can either be constructed exploiting physical principles (white-box modeling), using data-based modeling techniques (black-box modeling) or a combination of the former (gray-box modeling). White-box models are often hard to generate, as the physical principles and parameters are frequently very complex, hard to model or unknown. Black-box and gray-box modeling overcomes these disadvantages, but strongly depends on informative measurement data. One approach for gathering this measurement data while keeping safety constraints and estimating a black-box model is presented in this paper. We apply the approach to the high pressure fuel supply system (HPFS system) of a gasoline engine.

Active Learning is a subfield of machine learning. Its main idea is to employ a learning algorithm, which chooses queries to be labeled on its own. Labeling data for model learning is often costly, for example due to necessary manpower or expensive test bench time. Thus, it is beneficial to optimize the queries to be labeled in order to minimize the amount of necessary data to achieve a sufficient model quality. In (Settles, 2009) an overview about the Active Learning topic and the corresponding literature is provided.

When taking measurements of a technical system, it is essential to guaranty the integrity of the system. Especially in the case of open-loop measurements, critical system states can occur when choosing improper input signals. For example, when measuring the HPFS system, the maximum allowed rail pressure may not be exceeded.

In many cases, an automation system will avoid threshold violations. Nonetheless, even the attempt to measure unsafe input signals yields side effects: high stress on the test subject, wasted measurement time, and the risk of emergency shutdowns during automated measurement runs. Thus, it is beneficial to avoid critical system inputs in advance.

One possibility to combine the goals of Active Learning, i. e. learning the best model possible using the smallest amount of data, with the goal of avoiding unsafe input queries is presented in (Schreiter et al., 2015). There, a Safe Active Learning algorithm (SAL algorithm) based on GP models is introduced, which finds a set of input samples maximizing the model's entropy, provided that an estimated safety measure is satisfied.

In this paper, the algorithm proposed in (Schreiter et al., 2015) is translated to the real-world application of the HPFS system of a gasoline engine. Compared to the theoretical investigations in (Schreiter et al., 2015), the algorithm is modified in order to meet the requirements of our system. To the best of the authors' knowledge, this is the first time the algorithm is evaluated at a real-world system.

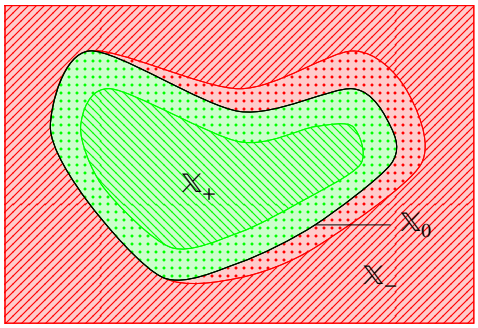

Figure 1. Partition of the input space $\mathbb{X}$ into a safe explorable area $\mathbb{X}_+$ and an unsafe region $\mathbb{X}_-$ separated by the unknown decision boundary $\mathbb{X}_0$. The figure is taken from (Schreiter et al., 2015). There, a discriminative function is learned over the dotted area for recognizing whether the exploration becomes risky.

Table 1. Overview of the models used in SAL

	regression model	classifier
model output	f	g
training data	$y(\boldsymbol{x}_i)$	$h(\boldsymbol{x}_i)$ or $c(\boldsymbol{x}_i)$
SAL's goal	max. model's entropy	satisfy safety constraints

This contribution is organized as follows: First, the fundamentals of the SAL algorithm and the HPFS system are presented. In Section 3, we describe our enhancements of the algorithm and the necessary design decisions. Subsequently, the evaluation in simulation and at a test vehicle is shown.

2 Fundamentals

In this section, we will commemorate the fundamentals of SAL as introduced in (Schreiter et al., 2015) and describe the HPFS system considered in this paper.

2.1 Safe Active Learning

The key goals of SAL according to (Schreiter et al., 2015) are:

1. approximate the system based on sampled data as informative as possible,
2. use a limited budget of measured points, and
3. ensure that critical regions of the considered system are avoided during the measurement process.

A compact and connected input space $\mathbb{X} \subset \mathbb{R}^d$ is defined that is divided into two subspaces $\mathbb{X}_+$ and $\mathbb{X}_-$ in which the system is safe or unsafe, respectively (compare Figure 1). The latter should be avoided with a probability higher than the user defined threshold δ.

In this input space, two GP models are learned. One of the models is a regression of the system output y. For this model, a usual GP is used, which is trained using noisy observations $y = f + \varepsilon$, $\varepsilon \sim \mathcal{N}(0, \sigma^2)$. The second model is a problem specific classifier, used to estimate the boundary $\mathbb{X}_0$. It learns a discriminative function $g : \mathbb{X} \to \mathbb{R}$, mapped to the unit interval to describe the class likelihood for each point. This model is trained with class labels $c(\boldsymbol{x}_i) \in \{-1, +1\}$ or discriminative function values $h(\boldsymbol{x}_i) \in (-1, 1)$, depending on the location of the measured point $\boldsymbol{x}_i$ in $\mathbb{X}$ (compare Figure 1). Thereby it is possible either to only use information whether a point was feasible or not, or to use more detailed data about the grade of its feasibility. Here, the discriminative function value is given by $h = g + \zeta$, where $\zeta \sim \mathcal{N}(0, \tau^2)$ specifies the noise. The mixed kind of training data results in a non-Gaussian model likelihood, hence a Laplace approximation is required, to calculate the model's posterior.

Both GP models use zero mean centered Gaussian priors and squared exponential covariance functions. A major prerequisite is that the hyperparameters of both models need to be known in advance. These are the signal magnitude $\sigma_\bullet^2$, the length-scales $\boldsymbol{\lambda}_\bullet \in \mathbb{R}^d$, and the noise variance σ^2 or τ^2. They are summarized in $\boldsymbol{\theta}_\bullet$. The dot $\bullet$ is a placeholder for the regression and discriminative model, specified by index f or g subsequently. An overview of the two models is given in Table 1.

In SAL, the next measurement point $\boldsymbol{x}_{i+1}$ is selected based on a differential entropy criterion. Thus, $\boldsymbol{x}_{i+1}$ is chosen such that the entropy of the regression model f increases as much as possible. Namely, the variance at $\boldsymbol{x}_{i+1}$ is maximized. In order to ensure safety, the optimization is constrained with a safety criterion depending on the predicted probability of failure from the discriminative model. For more details, we refer to (Schreiter et al., 2015).

2.2 The high pressure fuel supply system

The main components of a HPFS system are the high pressure rail, the high pressure fuel pump, and the ECU (Engine Control Unit; compare Figure 2). The pump is actuated by the crankshaft of the engine. A demand control valve in the pump allows to control the delivered volume per stroke. A pressure-relief valve is also included in the pump, but should never open if possible. Hence, we want to limit the maximum pressure during the whole measurement process. The pump transports the fuel to the rail, which contains the pressure sensor. From there, it is injected into the combustion chambers. See (Robert Bosch GmbH, 2005) for more details.

The system has three inputs and one output. The engine speed ($nmot$) affects the number of strokes per minute of the pump and the engine's fuel consumption. The fuel pump actuation (MSV) gives the fuel volume which is transported with every stroke of the pump. It is applied by opening and closing the demand control valve accordingly during one stroke of the pump. The injection time is a variable calculated by the ECU, which sums up the opening times of the single injectors and is, thus, related to the discharge of fuel from the rail.

A notable difference to the systems considered in

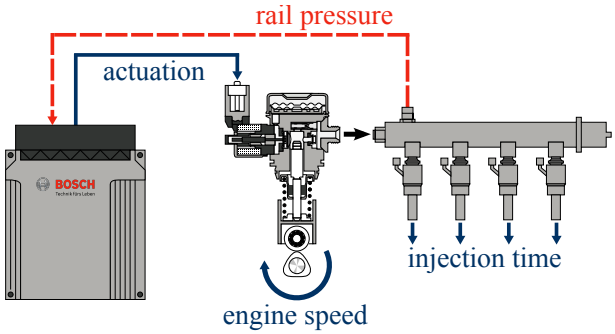

Figure 2. Sketch of the HPFS system's main components, inputs (continuous lines), and output (dashed line). The figure is taken from (Tietze et al., 2014).

(Schreiter et al., 2015) is that the input signals cannot be set directly, but have to be changed continuously. This becomes clear using the example of the engine speed, which cannot change immediately. Another reason for changing the input signals slowly is the system's dynamic behavior. Too fast actions could lead to pressure overshoots which can damage the engine.

If we move from each measurement point on the shortest path to the next one, we would have to assume that $\mathbb{X}_+$ is convex. However, experiments show that this assumption does not hold in the case of the HPFS system. Thus, we always make the detour via a global safe point (GSP), which weakens the precondition on $\mathbb{X}_+$ to star-convexity. This is a sufficient approximation of its real shape if the GSP is chosen correctly. When predicting the feasibility of an input point, the trajectory from the GSP to that point needs to be considered as well.

During the measurement procedure, the injection time is not varied manually but set by the ECU. The permissible times depend on many factors and a wrong choice could extinguish the combustion or even damage components. The engine load would have a major influence on the HPFS system via the injection time, but is omitted to prevent the necessity of a vehicle test bench. In principle, both variables could be additionally considered in the SAL algorithm without major changes of the method.

3 Design and Implementation

The system considered in this paper is defined as follows: the d-dimensional input $\boldsymbol{x} \in \mathbb{X}$ results in a one-dimensional output $y \in \mathbb{R}$ to be modeled. As the input space is divided in a safe and unsafe subspace (compare Section 2.1), we are only interested in measuring the system output $y : \mathbb{X}_+ \to \mathbb{R}$. The safe and unsafe subspace must be distinguishable by supervising the d_z-dimensional additional output $\boldsymbol{z} \in \mathbb{R}^{d_z}$. We assume that all outputs are only observable within $\mathbb{X}_+$. Outside of this subspace, the system cannot be operated safely and thus, no steady state measurements can be taken.

In order to apply the SAL algorithm presented in (Schreiter et al., 2015) to the HPFS system of a real car, three main issues have to be solved:

- learning the hyperparameters $\boldsymbol{\theta}_f$ of the regression model and $\boldsymbol{\theta}_g$ of the discriminative model,
- defining the risk function $\tilde{h} : \boldsymbol{z} \to [-1, 1]$ based on the supervised system output, which is, in combination with measured data, used to calculate the discriminative function value $h = \tilde{h}(\boldsymbol{z})$, and
- implementing the assessment of the feasibility of the trajectory to the next sample.

In the progress of solving these issues, several changes of the algorithm become necessary. This includes:

- implementing an online estimation of the hyperparameters, which requires carefully chosen limits,
- finding a heuristic for an initial set of hyperparameters, and
- replacing the problem specific classifier applied in (Schreiter et al., 2015), which can be trained with labels as well as discriminative function values, by a usual GP regression model.

In the following, these issues are discussed.

3.1 Training of the hyperparameters

In (Schreiter et al., 2015) it is assumed that the hyperparameters of the GP models are given in advance. Thus, when modeling a system from scratch, the hyperparameters need to be determined before the SAL algorithm can be started. The hyperparameters are usually learned by maximizing the marginal likelihood, as shown in (Rasmussen and Williams, 2006). Therefore, already observed data is required. If we assume that we only know one starting point in $\mathbb{X}_+$ in advance, we have almost no idea where the system is in safe operation, so that it is not possible to safely generate measurement data to estimate the hyperparameters.

If we have expert knowledge, which gives us a sufficiently large subspace of $\mathbb{X}_+$, we could generate and measure a space-filling design of experiment (DoE) in this subspace. With the generated data, we could estimate the hyperparameters and subsequently start the SAL algorithm. In Section 4.1, we benchmark this approach against the online hyperparameter training we will develop in the following. A representative subspace of $\mathbb{X}_+$ and a suitably chosen number of predefined measurement points are necessary to obtain a good model with little measurement data. This contrasts the goals of SAL to model the system with little initial knowledge about $\mathbb{X}_+$ and to be content with little measurement data. In industrial practice, a system of similar complexity as the HPFS system would be modeled using about 25 measurement points. Thus, the necessity of previous hyperparameter estimation is a major drawback of the original SAL algorithm.

To overcome this drawback, we estimate the hyperparameters during the SAL. We have to be very careful doing so, as falsely estimated hyperparameters can lead to an overestimation of $\mathbb{X}_+$ and hence to samples in $\mathbb{X}_-$. The estimation is especially error-prone if only a small number of samples is available yet. To reduce this risk, we limit the classifiers length-scales λ_g during the first optimization steps.

According to (Rasmussen and Williams, 2006), the characteristic length-scales λ of a Gaussian process with squared exponential kernel can be interpreted as the distances one has to move in the input space, before the function can change significantly. Thus, large length-scales λ_g of the classifier will result in a fast exploration, because the SAL algorithm assumes little changes in the discriminative function. We do not want the length-scales of the classifier to become too large, as this gives raise to overestimate $\mathbb{X}_+$. To prevent this, λ_g is limited to $\frac{\Delta x}{4}$ until 10 points have been measured and to $\frac{\Delta x}{2}$ until 20 measurement samples have been acquired, where $\boldsymbol{\Delta x} \in \mathbb{R}^d$ is the extent of $\mathbb{X}$ in each input dimension. During the first 5 steps, λ_g is kept constant at $\frac{\Delta x}{4}$, to ensure a good conditioning of the hyperparameter optimization problem. With increasing number of available samples, the estimation of the hyperparameters improves and the limits can be relaxed.

This setup was chosen heuristically and shows good results in the practical application. A motivation for this choice, though no formal derivation, can be given using the expected number of level-zero upcrossings of g in the one-dimensional case. According to (Rasmussen and Williams, 2006), the mean number of level-zero upcrossings in the unit interval for a GP with squared exponential kernel is

$$\mathbb{E}(N_0) = \frac{1}{2\pi\lambda}. \tag{1}$$

As we assume $\mathbb{X}_+$ to be compact and connected and, if $\mathbb{X}_-$ is not empty, we expect the discriminative function h to have $\frac{2}{3}$ level-zero upcrossings in $\mathbb{X}$ on average (compare Figure 3). If we scale $\mathbb{X}$ to the unit interval, this results in a length-scale of

$$\lambda = \frac{\Delta x}{2\pi\mathbb{E}(N_0)} = \frac{\Delta x}{2\pi\frac{2}{3}} \approx \frac{\Delta x}{4}. \tag{2}$$

The other initial hyperparameters of the classifier were set to $\sigma_g^2 = 1$ and $\tau^2 = 0.01$, based on the amplitude and the expected (low) noise level of the discriminative function. Poorly estimated hyperparameters of the regression model f have less severe consequences as those of the classifier g, as they do not result in samples outside $\mathbb{X}_+$. They will lead to a wrong density of the measurement samples' distribution, but this can be corrected by inserting additional samples, once the hyperparameters are well known. Furthermore, the limitations on the length-scales of the classifier will restrict the exploration speed and subsequently enforce a minimum sample density. Thus, no limits on

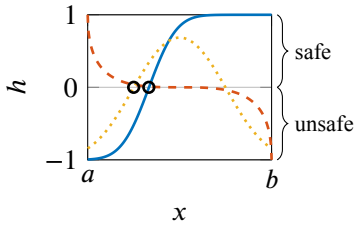

Figure 3. Three representative examples for a discriminative function $h(x)$ without noise in 1D. In this case, $\mathbb{X} = [a, b]$ and $\mathbb{X}_+ = \{x \in \mathbb{X} : h(x) > 0\}$. As one can see, two of the three examples have one zero-level upcrossing, marked with circles, the last one has none.

regression hyperparameters were enforced. The initial hyperparameters until 5 points have been sampled, were chosen as $\lambda_f = \frac{\Delta x}{4}$, $\sigma_f^2 = \left(\frac{\Delta y}{2}\right)^2$, and $\sigma^2 = \left(\frac{\Delta y}{200}\right)^2$, based on similar considerations as in the classifier case. Here, Δy denotes the expected range of the output signal y.

3.2 The discriminative model

Reference (Schreiter et al., 2015) assumes that there is only limited information about the discriminative function when measuring deep within $\mathbb{X}_+$. In contrast, the supervised output z of the HPFS system can be observed within the whole space $\mathbb{X}_+$. Thus, it is reasonable always to use the discriminative function value instead of class labels for learning the discriminative model. Another reason for doing so is hyperparameter training. Using the original algorithm and starting deep inside $\mathbb{X}_+$, it is likely that for the first samples only positive labels are drawn. It is not possible to learn correct hyperparameters from these labels using the standard training method, as shown in (Xiao et al., 2015). Therefore, the hyperparameters of the discriminative function would be estimated wrongly, which may again result in an overestimation of $\mathbb{X}_+$.

Using labels in $\mathbb{X}_-$ has some drawbacks, too. As a label only determines the discriminative function to be positive or negative in a certain point, not to have a specific value, the training algorithm can gain more information from discriminative values than from labels only. Furthermore, the regression model is not updated at all when no output y can be measured. Thus, the optima of the unconstrained part of the optimization problem described in Section 2.1 will not change if a labeled sample from $\mathbb{X}_-$ is added. In combination with the not too strong influence on the discriminative model, this may result in the next sample being generated very close to the preceding sample, as simulations showed.

For these reasons, we use a standard GP regression for the discriminative function instead of the problem specific classifier from (Schreiter et al., 2015). This has the additional advantage that no Laplace approximation for calculating the posterior is necessary. If a sample inside $\mathbb{X}_-$ is drawn, the measurement automation tries to measures at a point near the border, but inside of $\mathbb{X}_+$ instead. At this position, y as well as $\boldsymbol{z}$ can be measured. Since the original sample is usually near the boundary, the replacement sample is not far away.

3.3 The risk function

The risk function is used to encode the supervision of the systems's outputs $\boldsymbol{z}$ into a scalar discriminative function value $\tilde{h}$. This function value should be in the interval $(-1, 1)$, where -1 describes the least permissibility, 1 the highest, and 0 represents the boundary between the allowed and disallowed region.

The shape of the discriminative function h has an influence on the exploring behavior of the SAL algorithm. Comparing the continuous and the dashed line in Figure 3, one can see that the continuous line has a rather well defined zero-crossing, whereas the dashed line has only a small gradient near zero. While the former will result in a faster exploration, but with an increased risk of false positive samples in case of wrongly estimated hyperparameters, the latter will yield a slower exploration, as the discriminative model has to become very certain before samples near the boundary are queried.

By defining the risk function $\tilde{h}(\boldsymbol{z})$ accordingly, we can alter the discriminative function $h(\boldsymbol{x})$ to be learned. A proper choice could further improve the learning and exploring behavior of the algorithm. Unfortunately, $\tilde{h}$ has to be defined before starting the SAL, when the dependency of $\boldsymbol{z}$ regarding $\boldsymbol{x}$ is still unknown. Perhaps, an online optimization of $\tilde{h}$ would be possible, but this is beyond the scope of this contribution. Instead, we define $\tilde{h}$ as a linear function of $\boldsymbol{z}$. In case of the HPFS system with $\boldsymbol{z} = p$ this yields

$$\tilde{h} = 1 - \frac{p(\boldsymbol{x})}{p_{\max}} \tag{3}$$

where p is the current and $p_{\max}$ the highest allowed rail pressure.

3.4 The path to the next sample

As described in Section 2.2, we need to include the path from the GSP to the next sample in the estimation of its feasibility. Therefore, we require the constraint of the optimization problem described in Section 2.1 to be fulfilled not only at the sample point $\boldsymbol{x}_{i+1}$, but also at a number of waypoints between the GSP and $\boldsymbol{x}_{i+1}$. Even though this is only an approximation for the path's feasibility, experiments showed good results, given a sufficiently smooth discriminative function. Furthermore, it can be calculated quite fast and simple. The major drawback of this approach is that the derivative of the probability cannot be calculated analytically anymore, and thus is not available for the optimization algorithm. This results in a slightly reduced rate of convergence of the constrained optimization problem.

3.5 The algorithm

The SAL algorithm is implemented in MATLAB. Listing 1 describes the program flow in pseudocode.

Listing 1. Pseudocode for the SAL algorithm.

```
require initial measurement data D_{m_0}
    containing m_0 ≥ 1 samples, initial
    hyperparameters θ_f and θ_g, desired
    sample size n, desired safety δ
train models f and g using D_{m_0}
for i = m_0 + 1, ..., n do
        get x_i from optimization (compare
            Section 2.1)
        measure y_i and z_i and add them to
            D_{i-1} to get D_i
        calculate h(x_i) = h̃(z_i) and add it to
            D_i
        if i is large enough (compare
            Section 3.1) then
                optimize hyperparameters of
                     f and g using D_i while
                     keeping maximal
                    length-scales, where
                    applicable
        end if
        train models f and g using D_i
end for
```

4 Evaluation

4.1 Evaluation in simulation

In the first step, the SAL algorithm is evaluated using a simulated HPFS system with additive zero-mean Gaussian noise. For this purpose, a data-based GP model of the system was generated using measurement data acquired with a space-filling DoE and the ODCM algorithm presented in (Hartmann et al., 2016). ODCM allows to skip some samples based on their estimated feasibility, after a number of points in $\mathbb{X}_-$ has been sampled.

To assess the quality of the classifier, several criteria can be used. In the simulation case, we assess the classifier on a set of 10 000 test points with space-filling distribution. For the interpretation of the results, it is beneficial to consider the relative measures sensitivity and specificity. They describe the fraction of correctly classified test points in $\mathbb{X}_+$ and $\mathbb{X}_-$, respectively.

In order to benchmark the quality of the regression model, we use the root mean square error (RMSE) on m test data samples $y_{\mathrm{t},k}$ and the corresponding model outputs $f_{\mathrm{t},k}$, $k \in \{1, \ldots, m\}$.

In Table 2, a comparison of the SAL algorithm with pre- and online estimated hyperparameters is shown. In the first case, 5 to 20 initial points are sampled in a space covering about 20 % of $\mathbb{X}_+$. This region has to be defined by an expert in advance. In our case it spans about 50 % of the engine speed range and 30 % of the fuel pump actuation range. These points are used for learning the hyperparameters, which are not altered during the following SAL phase. In total, 25 points are sampled and used for modeling. At the end, the hyperparameters are optimized again using all available training data. In case of SAL without predetermined hyperparameters, these are learned online using the

Table 2. Comparison of SAL with pre- and online-learned hyperparameters in simulation after 25 samples

	pre-learned			online-learned
Initial points	5	12	20	0
RMSE	0.5900	0.0383	0.1340	0.0386
Sensitivity	0.7845	0.9928	0.9785	0.9941
Specificity	0.9647	1	1	1
Samples in $\mathbb{X}_-$	14.7	1.2	0.1	0.8

approach described in Section 3.1. All values are averaged over ten runs of the algorithm.

As one can see, 5 initial points are not enough for proper hyperparameter optimization. This leads to a disadvantageous placement of the samples, many samples in $\mathbb{X}_-$, and, subsequently, to a bad model with high RMSE. For the initial hyperparameter estimation, 12 points are a better choice, since this leads to the lowest RMSE of the final model and a small number of samples in $\mathbb{X}_-$. Nonetheless, we can outperform this variant regarding the number of unwanted samples with our online hyperparameter learning approach. The number of unwanted samples can be further reduced using 20 initial points. The remaining 5 points after the initialization are not enough to explore the whole $\mathbb{X}_+$ though, which results in a worse RMSE and reduced sensitivity compared to 12 initial points and the online learning approach.

Despite that, the variants using initial points show another unwanted property, which is not obvious from the data in Table 2: The first points during the SAL phase are often sampled far away from the initial space. In case the initial space is representative for the whole input space, this might indicate a well trained discriminative model which is able to extrapolate. This holds true in case of the HPFS system. Nonetheless, this step induces a high risk of samples in $\mathbb{X}_-$ if the initial space is not exactly representative and the discriminative function increases faster than estimated on the outside. In the online learning case, we do not observe such behavior, but a more steady exploration.

The sensitivity can be seen as a measure for the coverage of $\mathbb{X}_+$. One can see the same pattern as with the other quality parameters: The online learning approach performs best, closely followed by the 12 initial points-case.

With the right number of initial points, using pre-estimated hyperparameters performs almost equally well as the online hyperparameter learning variant. Nevertheless, it shows several drawbacks: The need to define a safe subspace in advance using expert knowledge, the right number of initial points that is hard to choose, and the large steps outside the initial space once the SAL algorithm is started. All of these drawbacks can be overcome using the online hyperparameter learning approach.

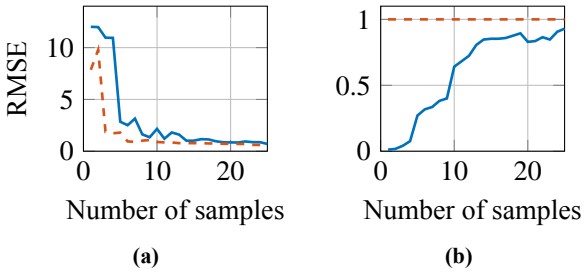

Figure 4. (a) shows the RMSE using SAL (continuous line) and space-filling plans (dashed line) at a test vehicle. The RMSE was calculated using a set of 60 space-fillingly distributed test points in $\mathbb{X}_+$. After 25 measured samples, the SAL approach obtains a RMSE of 0.7045, while a space-filling DoE results in 0.6225. (b) displays sensitivity (continuous line) and specificity (dashed line) using SAL at the test vehicle. The measures were calculated using a set of 129 space-filling test points in $\mathbb{X}$. After 25 samples, the sensitivity reaches 0.9294 and does not change considerably with a further increasing number of samples.

4.2 Evaluation at a test vehicle

After successful test runs in simulation, we apply the SAL algorithm with online hyperparameter estimation to the HPFS system of a real test vehicle. For this purpose, we use a car with 1.4 L four-cylinder gasoline engine. We implemented an automation in MATLAB, which is able to read and write labels from and to the ECU via ETAS INCA MIP, ETAS INCA, and an ETK (an ECU interface).

In Figure 4a, a comparison of the RMSEs achieved by SAL and space-filling designs over the number of training points is given. Our approach performs comparably to the space-filling DoEs. This seems acceptable, as the space-filling DoEs do not consider any limits in the input space and cannot be conducted easily this way. More measurement points do not result in a major improvement of the RMSE. The measured points in input space are shown in Figure 5.

Figure 4b shows sensitivity and specificity for each step of the SAL algorithm averaged over two runs. As one can see, the sensitivity rises rather continuously until most of $\mathbb{X}_+$ is explored. The specificity stays at its maximum value of 1 for the whole time. Note that the number of test points is much smaller compared to the simulation case, which leads to a decreased resolution of sensitivity and specificity. In average over two runs of the SAL algorithm, 0.5 points in $\mathbb{X}_-$ are sampled. As all unwanted points are sampled near the boundary, only little risk for the engine arises from them.

5 Conclusions

The test runs show that the introduced variant of the SAL approach manages to obtain a good model of the HPFS system while correctly estimating the limits of the drivable region $\mathbb{X}_+$. Only very few points are sampled in $\mathbb{X}_-$ and those which are, lie near the boundary. The resulting

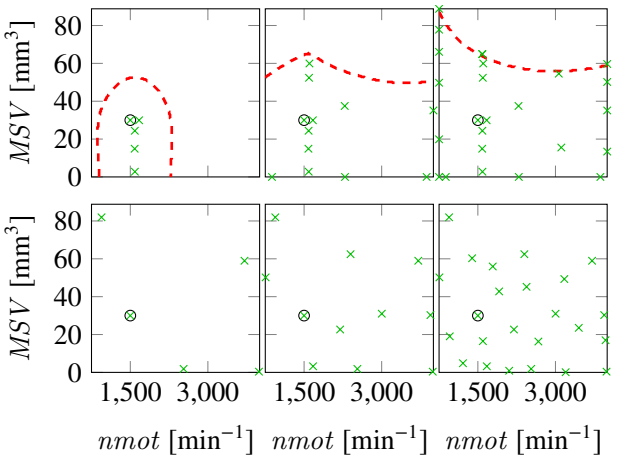

Figure 5. Measured points (crosses) in case of SAL (upper plots) and space-filling DoEs (lower plots). The algorithmic steps and plans are shown for 5, 12, and 25 points, respectively. The GSP is indicated by a circle. In the SAL case, the current estimated boundary $\mathbb{X}_0$ is plotted as dashed line.

model is almost as good as a model learned from space-filling distributed data. It must be pointed out that the latter does not comply with safety constraints or implement an exploration scheme, in contrast to the SAL approach.

References

B. Hartmann, E. Kloppenburg, P. Heuser, and R. Diener. Online-methods for engine test bed measurements considering engine limits. In *16th Stuttgart International Symposium*, Wiesbaden, 2016. Springer Fachmedien. doi:10.1007/978-3-658-13255-2_92.

C. E. Rasmussen and C. K. I. Williams. *Gaussian processes for machine learning*. The MIT Press, 2006. ISBN 026218253X.

Robert Bosch GmbH, editor. *Ottomotor-Management. Systeme und Komponenten.* Friedrich Vieweg & Sohn Verlag, 3rd edition, 2005. ISBN 3-8348-0037-6.

J. Schreiter, D. Nguyen-Tuong, M. Eberts, B. Bischoff, H. Markert, and M. Toussaint. Safe exploration for active learning with gaussian processes. In *Machine Learning and Knowledge Discovery in Databases*, pages 133–149. Springer, 2015.

B. Settles. Active learning literature survey. Computer Sciences Technical Report 1648, University of Wisconsin–Madison, 2009.

N. Tietze, U. Konigorski, C. Fleck, and D. Nguyen-Tuong. Model-based calibration of engine controller using automated transient design of experiment. In *14th Stuttgart International Symposium*, Wiesbaden, 2014. Springer Fachmedien. doi:10.1007/978-3-658-05130-3_111.

Y. Xiao, H. Wang, and W. Xu. Hyperparameter selection for gaussian process one-class classification. *Neural Networks and Learning Systems, IEEE Transactions on*, 26(9):2182–2187, 2015. ISSN 2162-237X. doi:10.1109/TNNLS.2014.2363457.